Using Geometric Computation for Characterizing and Visualizing Geological Structures

by

Po Kong Lai

A thesis submitted to the Faculty of Graduate Studies and Postdoctoral Affairs
in partial fulfilment of the requirements for the degree of

Master of Computer Science

in

Computer Science

Ottawa-Carleton Institute for Computer Science

School of Computer Science

Carleton University

Ottawa, Ontario, Canada

Copyright ©2013

Po Kong Lai

Abstract

We present several image processing methods which extract additional geological information from 3 dimensional (3D) laser images of rock faces at the outcrop scale. The geological information we are interested in extracting is fracture orientation, surface roughness and specific rock feature detection. Fracture orientation is important in determining the strength of intact rock masses as well as the stresses it has received. The surface roughness is also important in determining the strength and stability of rock masses. The strength and stability of rock masses is especially important for safe mining environments. The specific rock feature we aim to detect is known as the shatter cone. Detecting shatter cones was explored to determine if 3D images could be used to extract a specific rock type. Two different methods for fracture orientation mapping is presented. Three different roughness measures are proposed. One method for specific rock detection is explored.

Acknowledgements

Words cannot describe my appreciation and gratitude to both of my supervisors Prosenjit Bose and Claire Samson. Their passion for research allowed me the opportunity to explore the area of Earth sciences in conjunction to Computer science. They have opened my mind to the interconnectivity of seemingly unrelated fields and the joy of exploring the unknown. Understanding the problems in one field and relating it to another has been a worthwhile challenge. I thank them for their guidance, ideas, encouragement and unyielding support. I am also deeply grateful to them for having faith in me even during times when I had lost faith in myself. This thesis would not have been possible without them.

I would like to thank the technical staff at Neptec Design Group for training and allowing me to use their instrument as well as Mr. R.J. Selle, owner of R.J. Selle and Son, Sand and Gravel of Douglas, ON, for access to his quarry.

I thank Christopher Fry for aiding my field work at the Selle quarry. I also thank Dave Melanson and Alexandre Boivin for aiding my field work in Sudbury, ON.

I thank my mother, father and sister for their love and support.

Finally, I am grateful for the financial support from Carleton University.

Contents

1	Introduction	1
1.1	Motivation and Objectives	1
1.1.1	Rock Mass Characterization	1
1.1.2	Feature Selection	2
1.1.3	Research Objectives	3
1.2	Previous Work	3
1.3	Summary of Contributions	4
1.4	Thesis Outline	4
2	Preliminaries	7
2.1	Terrestrial laser scanners (TLS)	7
2.1.1	Scanner types	8
2.1.2	Data format	8
2.2	The rock cycle and the three types of rock	9
2.2.1	Igneous rocks	9
2.2.2	Sedimentary rocks	10
2.2.3	Metamorphic rocks	10
2.3	Rock mass characterization	10
2.3.1	Fracture Orientation	11
2.3.2	Surface roughness	13

3	Data Acquisition and Point Cloud Pre-Processing	15
3.1	Instrument	15
3.2	Measuring instrument error	16
3.3	Merging multiple scans	18
3.4	Survey Sites and Datasets	18
3.4.1	Quartzite road cut at Ivy Lea	18
3.4.2	Limestone and dolostone at the Selle quarry	19
3.4.3	Sudbury shatter cones	21
3.5	Decimation	22
3.5.1	Simple	26
3.5.2	ε -nets	27
4	Surface Reconstruction and Mesh Operations	31
4.1	Triangulated Meshes and Planar Graphs	31
4.2	Delaunay triangulation	33
4.2.1	Properties	33
4.2.2	Construction	35
4.3	Poisson reconstruction	38
4.4	Comparison of Delaunay and Poisson using acquired datasets	40
4.5	Smoothing	47
4.5.1	Geometric laplacian	47
5	Planar Orientation Mapping	49
5.1	Automated strike and dip	50
5.2	Outer normal orientation	51
5.3	Hue Saturation Lightness color model	52
5.4	Color-coded geological scenes	54

6	Roughness Mapping in Point Clouds	58
6.1	Local window	59
6.2	Intuitive definition of curvature	60
6.3	Visualization of curvature on meshes	62
6.4	Roughness through curvature difference	64
6.5	Quad-tree roughness ratio	67
6.5.1	Clustering curvature	69
6.5.2	Quad-trees	71
6.5.3	Results	72
6.6	Roughness sampling	76
6.7	Summary and Discussion	80
7	Feature Selection	82
7.1	Normal orientation to produce a polar plot	82
7.2	Shatter cone markers	86
7.2.1	Quad-trees revisited	89
7.2.2	Results	91
8	Conclusion and Open Problems	95
A	Roughness Measures in Other Fields	97
A.1	Statistical	97
A.2	Electromagnetic	101
	Scattering Models Overview	102
	Physical Optics Approximation	103
	Applications	104
A.2.1	Refraction Methods	105

B Formal Curvature Definition	108
B.1 First and Second Fundamental Forms	108
B.2 Extension to triangulated meshes	114
C Gravitational Clustering	118
C.1 Approximation of Movement due to Gravity	118
C.2 Algorithm	120
C.3 Algorithm Parameters	123
D Quad Tree Operations	125
E Implementation Details	129
E.1 Wavelength Color Scale	130

List of Figures

1.1	Work-flow from point cloud to analysis.	6
2.1	A visualization of the rock cycle. Source: Monroe and Wicander 1992 [55].	9
2.2	The pink rectangle is the plane being measured and the blue circle represents the horizontal cutting plane. The directions North, East, South and West are labelled on the axes.	11
2.3	The purple and green planes are dipping towards the East and West directions respectively. The dip angle is shown as blue.	12
2.4	The green and red planes belong to one wall while the blue and orange planes belong to another wall. The red planes are thus the overhang of the green planes while the orange planes are the overhang of the blue planes.	13
2.5	JRC surface roughness	14
3.1	Photograph of the author and the LCS imaging shatter cones along Ramsay Lake Road, Sudbury, Ontario. The LCH is mounted on the tripod. The LCC is in the foreground, powered by two 12 volt car batteries.	16

- 3.2 Best-fit planes through 16,411 points representing a metal plate. Points are colored red if they are above the plane and green if they are below. Vertical axis is the camera-target distance. The units are in millimetres. 17
- 3.3 Photograph of the Ivy Lea survey site [51] with the three colors representing different joint sets (strike and dip in degrees in brackets): yellow (056/80), blue (139/88), and green (154/01). The dotted lines represent the boundaries of individual scans. 19
- 3.4 Each scan is labelled with their scan number. Number 1 is the first scan taken, number 2 is the second, and so on. The purple regions indicate the overlay between two scans and the shaded red region shows the laser scanning the first area and sixth area. The boulders can be seen in front of the LCS. 20
- 3.5 Photograph of the Selle quarry survey site. The pink line separates the limestone on the left from the dolostone on the right [58]. The blue shaded region represents the area scanned by the LCS (approximately 9.6 m wide by 3.2 m high). Also seen in the photo is the author wearing an orange vest, the LCS mounted on a tripod and the large boulders preventing safe camera movement. 21
- 3.6 Map of the Sudbury, Ontario, retrieved on June 29, 2013 from www.maps.google.ca. Red dots represents the actual imaging site locations. 23
- 3.7 Each scan is labelled with their scan number. Number one is the first scan taken, number is the second, and so on. The purple regions indicate the overlay between two scans and the shaded red region shows the laser scanning a portion of the scan. The green arrow indicates the direction that the LCS was moved to capture the multiple scans. . . . 24

3.8	Photographs of shatter cones found in Sudbury. (a) Detailed shatter cones from Laurentian University campus, photographed from a distance of 1 m. (b) Group of several shatter cones imprinted on an outcrop along Ramsey Lake Road, photographed from a distance of 3 m. (c) A photograph of the scene on Gibbison road. These more massive shatter cones are very distinct.	25
3.9	Comparisons between the same 1024×1024 point cloud decimated by (3.9a) factor of $n = 4$, resulting in a resolution of 512×512 and (3.9b) factor of $n = 16$, resulting in a resolution of 256×256 . Note how the data is organized in vertical lines.	26
3.10	Example of how decreasing ε increase the accuracy of the ε -net. . . .	27
3.11	Example of shattering point sets with half-planes.	28
3.12	Single image of a rock wall from the limestone/dolostone dataset is decimated using different parameters to produce ε -nets. The resulting point clouds have been covered with meshes to improve visualization. The parameters used are as follows: (a) original image (b) $\varepsilon = 0.0001$, $d = 3$, $c = 1$ (c) $\varepsilon = 0.0005$, $d = 3$, $c = 2$ (d) $\varepsilon = 0.001$, $d = 3$, $c = 1$. . .	30
4.1	(a) Embedding of the a graph where no edges overlap. (b) Graph where it is impossible to move the vertices such that none of the edges overlap making the graph non-planar. (c) Example of a non-planar embedding of (a).	32
4.2	(a) An example of an illegal edge. (4.2b) Part of a triangulation is shown as well as the triangles circumcircles. Note that each circle does not contain any points despite overlapping.	34
4.3	An example of how “flipping” an edge can maximize the minimum angle and restore the empty circle property.	35

4.4	An example of the two main cases one encounters when inserting a point into a Delaunay triangulation. The dashed red lines indicate the new edges to be added.	36
4.5	Visualization of a bounding triangle over some point set.	37
4.6	Intuitive illustration of Poisson reconstruction in 2D. Image source: Kazhdan et al [42].	39
4.7	A comparison between the 2.5D Delaunay triangulation mesh and the poisson mesh for the limestone/dolostone dataset. The region contained within the red square can be seen in greater detail in Figure 4.8. Outer and degenerate triangles are removed manually in Figure 4.7b.	42
4.8	The above shows a close up comparison of the sections highlighted in Figure 4.7. Figures 4.8a and 4.8c are from the Poisson reconstruction and Figures 4.8b and 4.8d are from the 2.5D Delaunay triangulation. Note that the larger triangles in Figures 4.8b and 4.8d represent occlusion zones. In the Poisson mesh the over the same area is represented with smaller triangles which fill the occlusion.	43
4.9	The Poisson reconstruction produces a watertight surface as see in Figure 4.9a. Note the additional degenerate “blob meshes” highlighted in red. These are outliers and since the normal estimation only looked at the nearest $k = 15$ neighbouring points, the points are assumed to be separate objects in space.	44
4.10	A problematic dataset for the Poisson reconstruction.	45
4.11	A comparison between the 2.5D Delaunay triangulation mesh and the Poisson mesh for the shatter cone dataset.	46
4.12	A comparison between the original mesh (top) and the smoothed geometric laplacian version (bottom).	48

5.1	Visualization of the plane being measured (blue) and the horizontal cutting plane (purple).	51
5.2	An illustration of α and β	52
5.3	Hue colour wheel.	54
5.4	Delaunay mesh of the Ivy Lea scene (a) before and (b) after smoothing using the geometric laplacian. Strike is color-coded according to the color wheel shown in Figure 5.3. Dip ranges from light (sub-vertical planes) to dark (sub-horizontal planes). (c) shows the removal of the overhangs.	56
5.5	Only triangles with a strike between $45 \pm 10^\circ$ and a dip between $80 \pm 10^\circ$ have been colored to highlight the most prominent joint set. This particular joint set corresponds to the yellow patches in Figure 3.3 . .	57
5.6	A close-up photo of the shatter cones is shown in 5.6a and 5.6b to 5.6d show the Delaunay mesh color coded according to their outer normal orientation. Figure 5.6b shows α which ranges from light (sub-vertical planes) to dark (sub-horizontal planes). 5.6c shows β which is color-coded according to the color wheel shown in Figure 5.3 and 5.6d show α and β superimposed.	57
6.1	Visual example of a local window and a part of a mesh projected onto a 2D plane. Figure 6.1a shows the local window of a vertex v . Edge points are green and vertices within the local window are blue. An edge is highlighted and expanded in Figure 6.1b. The terms d_1 and d_2 are the distances from vertices v_1 and v_2 to an edge point v_e	59
6.2	Examples of circles fitted to a curve illustrating how the circle radius decreases as the curve is “bent” more sharply.	60

6.3	An illustration of different planes cutting through p creating different curves (red). The curves with the maximum and minimum curvature defines the mean curvature. The arrow shows the rotation of the plane to cover all possible 2D curves.	61
6.4	A visual of how orientation of a curve affects the sign of the curvature.	61
6.5	Visualization of curvature for the limestone/dolostone dataset. A color key is shown at the top.	63
6.6	Note that Figure 6.6b has a larger range than Figure 6.6a meaning that larger roughness values were computed. The roughness values were scaled using the square-root function thus the color scale represents non-linear values. The effect of this scaling is that a few extreme values will not skew the coloring.	66
6.7	The red dots represent vertices of extreme curvature while the blue dots represents non-extreme curvature. The top two squares correspond to a flat surface as they have a few number of red dots. The bottom two squares correspond to a rough surface because they have a large number of red dots. It does not matter how the red dots are distributed in space, both the bottom squares are “rough” areas.	67
6.8	Application of gravitational clustering visualized.	70
6.9	Visualization of a quad-tree structure.	71

- 6.10 An example of quad-tree roughness ratio applied to one scan of the limestone/dolostone dataset. Figure 6.10a is a real world photograph of the surface which the scan represents. Figures 6.10b, 6.10c and 6.10d are visualization of the roughness ratio for each of the quad-nodes at depths 4, 5 and 6. As indicated in Figure 6.10e purple regions have a very low ratio of extreme curvature vertices (i.e: it is not rough) while red regions have a very high ratio of extreme curvature vertices (i.e: it is rough). Note that we can see bedding lines as well as the limestone bedding layers as smooth on the left while the crumbled rock as rough on the right. 73
- 6.11 The quad-tree roughness ratio applied to the limestone/dolostone dataset. Figure 6.11a shows the original coloring of the ratios of each quad-node. Figure 6.11b show the ratios scaled by the cube root function so that the ratios are colored on a non-linear scale. The color key is the same as in Figure 6.10. 74
- 6.12 Figure 6.12a shows the mesh model of the plate. Figure 6.12c shows the curvature mapped onto the plate. Note that while the deviation of the curvature values within the mesh is not great, the way that the coloring is computed (see Appendix E) allows us to see the variations between the different curvature values. Figures 6.12c to 6.12f show the results of applying the quad-tree roughness ratio. Since there is very little extreme curvature values the plate is almost entirely purple indicating that it is flat. 75
- 6.13 The vertex v that is being sampled is the blue dot and all the other dots are the neighbours of v . The magenta dots have an Euclidean distance that is less than or equal to $\varepsilon|\kappa(v)|$ while the grey dots do not. 77

6.14	(a) is the base image before varying ε and δ . (b) and (c) show the effects of varying ε while (d) and (e) show the effects of varying δ . . .	79
6.15	Flowchart describing the work-flow of roughness metrics.	81
7.1	A photograph of a very well defined shatter cone broken off from a larger rock. Note the clear cone shape as well as the striations on the cone where the shock force travelled through the rock. Image taken from [66].	83
7.2	Illustration of how the normals draw an arc shape the surface of a cone. The arc shape of the cone on the right is transposed in the circle plot to the left as the magenta dots. Each dot represents one of the normals.	84
7.3	Highlighting different parts of the shatter cone image (left) by selecting the corresponding cluster (right). The black box represents the selection area. The purple regions on the left images represent the triangles selected by the black box. The shatter cone corresponds to the large arc-shaped region near the center of the circle as see in (c) and (d). The α for that region ranges from 200° to 360° while the β ranges from 20° to 45°	85
7.4	Note that red colored streaks represents a side of a single shatter cone.	86
7.5	A visualization of both types of markers. The end markers have triangles whose orientation spans over the interval $[120^\circ, 154^\circ]$ while the start markers span over the interval $[0^\circ, 60^\circ]$	87
7.6	Red and green streaks represent start and end markers, respectively. When the proper markers are chosen correctly we can select the shape of the shatter cone.	88

- 7.7 A visual of a quad-tree grid overlaid on top of the **end** (left) and **start** (right) markers. The nodes and edges forming a MST which represent the markers are seen in blue (**end**) and orange (**start**). A single quad-strip is highlighted in pink. 89
- 7.8 An example quad-tree with depth 2. At depth 1 we have top quadrants labelled 0 and the bottom labelled 1 which can be seen as the left most 0 and 1. Since the definition is recursive the same applies at level 2. Note that the strip id (top-right number) of each of the quad-nodes is labelled. 90
- 7.9 Example of how the orientation of a shatter cone as defined on the left can move due to natural forces to the shatter cone oriented on the right. 92
- 7.10 The red circle highlights a poorly selected marker. The cone associated with this marker is the brown marker to the right of the red circle. Note that the brown marker is much longer in size and captures the side of the shatter cone while the poorly selected marker is short and does not capture the entire side of the shatter cone. 93
- 7.11 The red circle highlights the marker (same location as 7.10) which stops the brown markers from advancing at that location. Due to it's smaller size it is not able stop the brown markers properly resulting in a bad detection. Note that the two overlapping cones which have distinct markers are detected properly. 93
- 7.12 (a) Photograph of shatter cones along Gibbson road (b) The color overlays represent the areas which are was detected as a shatter cone. Each color corresponds to a different detected cone. 94

A.1	Two different views of a surface profiling instrument. Figure A.1a shows a 2D view where the rectangular gray shape represents an object which the brown stylus moves over as indicated by the arrows. Figure A.1b shows a 3D view where the line traced on the surface is indicated in red and the blue plane which intersects the object is shown. This plane represents a subsection which one can apply following roughness measures.	98
A.2	Example of an EM wave hitting a surface and scattering. The coherent part is represented by the red arrow moving point away from the surface and the purple arrows represent the scattering of the wave in all the other directions.	102
A.3	An aerial photo view of the ocean surface along with the waves which define the surface.	104
A.4	Visual of how higher wind speeds generate greater backscatter.	105
A.5	An example of an instrument which measures the slopes of the waves using a laser. Note that the refraction is exaggerated for illustration purposes.	106
B.1	On the left we have the parameter space (u, v) and on the right we have a surface s which takes in the two parameters u and v . The red and blue lines shows a single curve drawn in parameter space (B.1a) and how they behave in the surface space (B.1b). The partial derivatives s_u and s_v define vectors which point along the direction of their respective curves from the point $s(u, v)$ and together they form a tangent space. The cross product of s_u and s_v produces the normal vector of the tangent space.	110

- B.2 A computation of the normal cycle, $N(P)$, of a union of two line segments denoted as P . $N(P)$ is highlighted in blue. Note that since $N(P)$ is additive the normal cycles of the end points of the two line segments has the shape of a circle. The line itself is just the two lines above and below the line and thus together a single line segment has the shape of a rectangle with two half circles at the ends. Note the values $+1, 0, -1$. The $+1$ represents a curved region and the 0 represents the flat regions. The -1 represents a region where the arrow must change orientation by travelling along the edge until it reaches a point on the normal cycle of the center vertex where it can then travel along the circle in the opposite direction until it reaches the point where it can reach the edge for the other line segment. Since it must travel along part of the center vertex in the opposite direction the value for this region is negative as opposed to positive. 115
- B.3 The normal cycles of an edge and triangle in 3D are visualized in blue. 116
- B.4 Illustration of the “mass” of the normal cycles for each of the simplices is shaded in green. In Figure B.4b edge is highlighted in magenta and angle $\beta(e)$ is shown as the shaded red region. Figure B.4a shows the mass for a vertex with adjacent to three edges. Note that the mass (green) does not include any of the edge masses (blue). 117
- E.1 The application created and used for analysis. Pictured is the shatter cone scene found in Section 7.2, Figure 7.4. 130

Chapter 1

Introduction

This thesis investigates how 3 dimensional (3D) laser images of rock faces can be processed to highlight several characteristics of the rock mass in three different geological environments and to detect specific features at the outcrop scale.

1.1 Motivation and Objectives

Cheap terrestrial laser scanners (TLS) are starting to be a standard tool for mining engineers in assisting rock mass characterization and for digitally storing their working environment. The main motivation for this thesis is to determine if one can use 3D laser imaging to give insight to the imaged subjects. We examine this through a geological context so the insight we would like is related to properties and features of rocks. In the following subsections we detail the properties and features which this research is focused on.

1.1.1 Rock Mass Characterization

During the preliminary design phase of an engineering project - especially if it involves underground excavations - it is vital to estimate the strength of the rock mass and

this is commonly called *rock mass characterization*. Rock masses are composed of intact competent rock and discontinuities, usually in a few preferential orientations, called *joint sets*. The orientation, spacing and the surface roughness of the joint sets all play a role in the overall strength of the rock mass. These properties are used as input parameters for rock mass characterization schemes such as the Rock Tunnelling Quality Index (Q) [3] or the Rock Mass Rating (RMR) system [7]. The schemes assign a grade to the rock which ranges from “very poor” (unstable and weak) to “very good” (stable and strong). An underestimation of the strength of the rock masses leads to unsafe working environments and ultimately a poorly designed project. Overestimation leads to increased costs and over-engineering. Typically the parameters in rock mass characterization schemes are empirically determined by geologists/mining engineers who visually examine the rock mass. Due to time constraints, they usually only achieve a few point measurements at key locations. A TLS can complete a scan in a short amount of time and images produced are typically very high resolution. Imaging analysis developed using these images would then be performing a number of measures relative to the resolution of the sensor. The measurements are much more detailed as the laser covers the entire rock face in a uniform fashion. Determining rock mass characterization parameters from 3D laser images is described in Chapters 5 and 6.

1.1.2 Feature Selection

Depending on the type of rock, and the forces it is subjected over time, different geological structures are formed. Rivers and mountain ridges are examples of geological structures formed due to the tectonic forces. In this thesis, we focus on shatter cones, centimetre-scale structures which are formed due to the hypervelocity impact of meteors. The force of impact ripples through the host rock in the form of a shock wave. The intense pressure from the meteor morphs the rock into cone-like structures. The

objective of the research detailed in Chapter 7 is to demonstrate how shatter cones can be automatically detected with a scene based on their typical morphology.

1.1.3 Research Objectives

To demonstrate the applicability of 3D laser imaging in a wide range of geological environments, this thesis features examples from different types of rock: quartzite (composed of a sandstone protolith having experienced regional metamorphism), limestone and dolostone (chemical sedimentary rocks), and shatter cones (host rock altered by shock forces). More details of the different types of rocks can be found in Chapter 2 and details about the acquired datasets can be found in Chapter 3. Each of the different types of rocks gives rise to a different challenge. Using a TLS we scan the scenes and reconstruct the underlying surface using surface reconstruction algorithms. Determining if we can use these reconstructed surfaces to extract more geological information aside is the focus of this thesis. The quartzite rocks are large and cubic which made it the perfect candidate to explore visualization of the joint set orientation. The limestone and dolostone rocks are two different types of chemical sedimentary rocks which differ by one chemical element. Despite this, the two rocks are visually very similar except the dolostone is more friable than the limestone. We are interested if TLS can pick up this difference. The shatter cones are a special kind of metamorphic rock with a specific shape. Our aim with this scene is to determine if these rocks can be identified.

1.2 Previous Work

The majority of research on TLS for rock mass characterization has been focused on large scale structures using time-of-flight variety of TLS [70] [47] [46] [50] [24]. There has been some work using trigonometric triangulation-based TLS in relation

to fracture detection [9] and rock mass characterization [51] [53]. Detection of such large scale features using Laser (LiDAR) data [44] and satellite imagery has been explored. Recently K. Di et al. [21] use Mars rover imagery and 3D point cloud data for automatic rock detection. Their work involves the detection of potential rock candidates through the use of shadows in imagery and pruning of candidates is driven by the 3D point cloud data. To the best knowledge of the author, detection of specific rock features at the outcrop scale has not been explored.

1.3 Summary of Contributions

This thesis presents several image processing approaches to extract additional geological information from the data produced by 3D laser scanners. Firstly we extend the work of Mah et al [51] and color-code the triangles of triangulated meshes with their geological orientation. These color-coded meshes are of particular interest to mining engineers and geologists and the orientation of fracture within a rock mass can give insight to the strength of the rock mass as well as the stresses the rock masses has experienced [40]. Secondly we present three different measures to help determine the “roughness” of rock walls at the outcrop scale. Using these measures of roughness we look at how it can be used to differentiate two different types of rocks within the same rock wall. These measures have not been used to characterize rock walls. Lastly we examine feature detection in point clouds for the geological context. We present an original method for detecting a feature imprinted in rocks called the shatter cone.

1.4 Thesis Outline

To understand exactly how different types of rocks can have different effects on the stability of the environment we begin by giving a brief background on the field of geology. What rocks are made of, how they are formed and how their constituents can

contribute to the rock mass characterization models is explored in Chapter 2. The instrument used to capture the different geological structures studied in this thesis, the point clouds it produces and pre-processing methods are described in Chapter 3. Once we have the point clouds, and it has undergone some pre-processing, we reconstruct the surface that it represents. Chapter 4 covers planar graphs, triangulations and surface reconstruction algorithms used. Planar graphs and triangulations are covered in more detail in order to give us a bound on the size of the mesh models generated relative to the size of the point clouds. Once we have a reconstructed surface we look at automatically computing some of the parameters used in rock mass characterization models. The details of how these parameters are computed is covered in Chapters 5 and 6. Detection of a special type of geological feature is examined in Chapter 7. The results of rock mass characterization model parameter computation and feature selection is detailed in their respective chapters. Finally Chapter 8 provides a summary of what was covered in this thesis and future research directions. The software created for the algorithms used in Chapters 5, 6 and 7 is described in Appendix E. Motivations and key design decisions are also detailed. Figure 1.1 is a flowchart which describes the work-flow for this thesis.

Figure 1.1: Work-flow from point cloud to analysis.

Chapter 2

Preliminaries

In this chapter we give a summary of terrestrial laser scanners and the data that they produce. We also provide a brief background on selected topics in geology to give context to the methods developed in Chapters 5, 6 and 7.

2.1 Terrestrial laser scanners (TLS)

In order to model geological structures an instrument is needed to capture their characteristics with high fidelity. Traditionally, manual measurements made with instruments such as compasses, inclinometers, profilometers, and straightedges, and photographic and stereographic images were used to produce a record of the structure. Terrestrial laser scanning technologies are becoming the standard as they allow one to easily scan a structure and record its topology in detail. The scanner can also reach inaccessible areas or potentially hazardous areas safely and the digital records can be analysed after the field data acquisition has been completed.

2.1.1 Scanner types

The basic concept behind TLS is the use of light to capture the topology of the imaging target. Exactly how the TLS uses the laser to measure distance gives rise to two main types of scanners: time-of-flight (TOF), and trigonometric triangulation-based (TTB). TOF scanners emit pulses of laser light and the distance is determined by the time it takes for the light to be reflected back to the scanner. TOF scanners are typically long-range scanners (distance $\approx 10 - 1000$ m). Their accuracy is related to their ability to detect reflected pulses and is not a function of distance. TTB scanners project either a laser line or a single laser point to scan across an object. The light that is reflected off the target is focused onto a charge-coupled device (CCD), and using trigonometry, the scanner calculates the distance from the target to the scanner. TTB TLS are short-range scanners (distance $\approx 1 - 25$ m) and their accuracy degrades exponentially with distance. The TLS we use for this study was provided by Neptec Design Group [67] and it is a TTB scanner. More information will be provided in Section 3.1.

2.1.2 Data format

Regardless of the type of TLS, the data produced are in the form of *point clouds*. Point clouds are unstructured collections of *voxels* which are volumetric image elements with the 3D Cartesian coordinates (x, y, z) of where the laser has illuminated the target surface. Some scanners also record the intensity of the reflected laser light, I , with each voxel. This thesis does not consider intensity so for our purposes point clouds only contain 3D spatial information. Once a point cloud is obtained, the surface that it represents can be reconstructed using a variety of different algorithms two of which are described in Chapter 4.

2.2 The rock cycle and the three types of rock

Rocks are classified in three main types: igneous, sedimentary, and metamorphic [55]. Their type is determined by how they are formed within the rock cycle (see Figure 2.1). In this section we examine these different classifications of rocks.

Figure 2.1: A visualization of the rock cycle. Source: Monroe and Wicander 1992 [55].

2.2.1 Igneous rocks

Igneous rocks are crystallized from a magma. When the magma is trapped below the Earth's surface it cools slowly allowing growth of large crystals that are visible to the naked eye. These crystals range from 1 – 10mm in diameter and the corresponding rocks are termed *intrusive*. Magma reaching the surface cools and solidifies too rapidly for large crystals to be formed. The corresponding rocks are called *extrusive* and have very fine grain sized crystals which are less than 1 mm. They can also have crystals which are invisible to the naked eye or they can be glassy.

2.2.2 Sedimentary rocks

Sedimentary rocks are formed by deposition of material on the Earth's surface and within bodies of water (clastic sedimentary rocks) or by precipitation (chemical sedimentary rocks). Clastic sedimentary rocks are naturally layered and the rocks themselves are a product of their environment of deposition. Thus, the study of the different layers of sedimentary rock is the main source of scientific information about the Earth's paleoenvironments. A bed is the smallest division of layering in a sedimentary sequence and bedding planes mark the separation between different beds.

2.2.3 Metamorphic rocks

Metamorphic rocks are the result of pre-existing rocks (protoliths) that have undergone high pressure and temperature. It can be thought of as an intermediate stage between igneous and sedimentary rocks. When protoliths are subjected to large changes in temperature and pressure, the minerals within are recrystallized and new minerals are formed. This alteration of a protolith into another in response to fluids, and temperature and pressure (P-T) conditions, without changing from a solid into a liquid (ie: magma), is called *metamorphism*. There are different types of metamorphism which encapsulate the conditions which triggered the alterations, for example, interactions with fluids, regional changes in P-T conditions and shock forces.

2.3 Rock mass characterization

Rock masses are comprised of intact rock and discontinuities of various sizes. The intact rock might be very strong and resistant to stress but, when combined with discontinuities, the resulting rock mass can become weak. Discontinuities with little or no movement between rock blocks are called *joints*. A group of joints with a common orientation within a rock mass is commonly called a *joint set*. During the

feasibility and preliminary design phases of mining engineering projects, there is very little detailed information available on the rock mass. Measuring properties of the joint sets can give insight into the overall strength of the rock mass which would then affect ground control design in the case of weak rock masses.

2.3.1 Fracture Orientation

The geological orientation of a joint is given by two angles: strike and dip. These two angles are measured by imagining a horizontal plane cutting into the joint set. Both the strike and dip angles are measured by taking a horizontal plane and cutting it with the joint. The angle the joint makes with respect to the North direction is the strike and the inclination of the joint set with respect to the cutting plane is the dip. The strike of a plane ranges from 0° to 360° while the dip ranges from 0° to 90° . Figure 2.2 illustrates these definitions. The orientation of joints are traditionally measured manually using compasses (for strike) and inclinometers (for dip). Work has already been done in automatically computing the strike and dip for triangulated meshes [51] and we extend it by color coding the triangles with this information (see Chapter 5).

Figure 2.2: The pink rectangle is the plane being measured and the blue circle represents the horizontal cutting plane. The directions North, East, South and West are labelled on the axes.

Since strike and dip uses a compass system as the axis, planes can dip in either the East or the West direction and Figure 2.3 illustrates this scenario. The strike angle for these two planes would be exactly 180° apart. For the example in Figure 2.3, the strike angles would be 0° and 180° . For consistency geologists follow a “right hand” rule to determine which plane is associated with which angle. If one were to view the plane along the strike direction, then the plane would always dip towards the right. In Figure 2.3 the purple plane, which points along the north direction, would be assigned a strike angle of 0° and the green plane, which points along the south direction, would be assigned an angle of 180° . The dip angle remains unchanged and the direction that the plane is dipping does not need to be specified. Thus any plane with a strike angle, s , that is $\geq 180^\circ$ is considered to be the overhang version of the plane with a strike angle of $s - 180^\circ$. Figure 2.4 illustrates this concept.

Figure 2.3: The purple and green planes are dipping towards the East and West directions respectively. The dip angle is shown as blue.

Figure 2.4: The green and red planes belong to one wall while the blue and orange planes belong to another wall. The red planes are thus the overhang of the green planes while the orange planes are the overhang of the blue planes.

2.3.2 Surface roughness

The surface roughness of a joint set helps in defining the strength of a rock mass. Traditionally the surface roughness in mining engineering is determined by taking a horizontal straightedge and placing it on the rock face. The largest perpendicular distance between this straightedge and the rock face is recorded as the asperity amplitude (see Figure 2.5a). Using the length of the straightedge and the asperity amplitude, the joint roughness coefficient (JRC) is found using the chart in Figure 2.5b. A joint surface that is “smooth” has a lower JRC compared to a joint surface that is “rough”. Smoother joint sets have a lower coefficient of friction compared to rougher joint sets and a rock mass comprised of smooth joint sets is weaker than a rock mass comprised of a rough joint sets. Surface roughness of point clouds measured using the JRC approach has been studied in the mining engineering context [52]. Our contributions in Chapter 6 aim to look at roughness in point clouds in a more general context with the goal of differentiating between two different types of sedimentary rocks.

(a) Measuring the asperity amplitude

(b) JRC Chart

(c) JRC Roughness Examples

Figure 2.5: JRC surface roughness

Chapter 3

Data Acquisition and Point Cloud Pre-Processing

In this chapter we will describe the TLS used to collect the data for this thesis and present the rationale for choosing the survey sites for scanning. We will also cover the pre-processing used to prepare the point clouds for the surface reconstruction algorithms presented in Chapter 4.

3.1 Instrument

The trigonometric triangulation-based Laser Camera System (LCS) developed by Neptec Design Group of Ottawa, Ontario, was selected for the 3D image acquisition. Detailed LCS specifications can be found in Table 1 of [67]. The LCS consists of the Laser Camera Head (LCH) which is the actual camera which fires the laser and the Laser Camera Controller (LCC) which is a laptop computer containing the software that controls the imaging parameters of the LCH.

Images were captured using the LCS mounted on a tripod. The LCS uses an infrared laser tuned to a specific frequency (source wavelength = 1500 nm, detector sensitive to 1500 ± 3 nm) making it immune to ambient light allowing one to effectively

collect data outdoors in bright sunshine or in the shade. The data collected are point clouds acquired with a $30^\circ \times 30^\circ$ field of view. Each image typically contains a million voxels per image (1024×1024 resolution) and takes approximately 15 minutes to acquire. It is important to note that the image processing techniques developed in Chapters 5, 6 and 7 are not hardware-dependent and can be applied to high-resolution 3D images recorded by other TLS variants.

Figure 3.1: Photograph of the author and the LCS imaging shatter cones along Ramsay Lake Road, Sudbury, Ontario. The LCH is mounted on the tripod. The LCC is in the foreground, powered by two 12 volt car batteries.

3.2 Measuring instrument error

To estimate instrument error in field conditions, a flat smooth metal plate was scanned from a distance of 3 m when acquiring point clouds at the Selle quarry (see Section 3.4.2). A subset of voxels ($N = 16,411$) corresponding to the metal plate was extracted manually and a best-fit plane was computed using principal component analysis (PCA). A best-fit plane minimizes the sum of the squared errors (SSE). The

SSE of the best fitting plane in Figure 3.2 was found to be 1.8701×10^5 . Since the SSE is the summation of all the squared errors of each point, we can find the standard error of the mean (SE) as follows:

$$SE = \text{sqrt}\left(\frac{SSE}{N}\right) \quad (3.1)$$

$$= \text{sqrt}\left(\frac{1.8701 \times 10^5}{16,411}\right) \quad (3.2)$$

$$= 3.3757 \quad (3.3)$$

From Equation 3.1 the standard error was found to be equal to 3.4 mm for a camera-target distance of 3 m.

Figure 3.2: Best-fit planes through 16,411 points representing a metal plate. Points are colored red if they are above the plane and green if they are below. Vertical axis is the camera-target distance. The units are in millimetres.

3.3 Merging multiple scans

Scenes which are too large for one scan will require multiple scans to be fully captured. The set of scans which make up the scene will need to be merged into one large point cloud. This process is called point cloud *alignment* and *registration*. Each scan in the scene requires some overlap with neighbouring scans. The overlap allows the user to provide rough markers to steer the alignment algorithm so that adjacent scans can be merged properly. Once two scans have been “roughly” aligned manually, an iterative closest point (ICP) algorithm is used to minimize the difference between the two scans. Registration of point clouds is a well studied topic and not the focus of this thesis. More details can be found in the paper by Rusinkiewicz and Levoy [64].

3.4 Survey Sites and Datasets

The first set of images was taken at Ivy Lea along Provincial Highway 137, near Kingston, Ontario [51]. The scene features large cubic quartzite rock blocks from a road cut which made it a prime candidate for the automated color-coding of strike and dip. The second scene shows a layered sedimentary sequence composed of a massive limestone and more friable dolostone from the Selle sand and gravel quarry in Douglas, Ontario, making it interesting to study how well point clouds capture the differences between the two types of calcite minerals. The third scene exhibits shatter cone structures imprinted in an outcrop in the Sudbury impact crater, in northern Ontario. This data set was chosen to test if image processing strategies could enhance the subtle fan-like patterns associated with the shatter cones.

3.4.1 Quartzite road cut at Ivy Lea

This image dataset was studied in detail by Mah et al. [51]. The rock face exhibits large cubic blocks (side $\approx 0.5 - 1.0$ m) fractured during large-scale regional tectonic

events, including the Grenville orogeny (approx 1.1 Ga ago) [40]. Mah et al. [51] developed a method for computing the strike and dip for each of the triangular elements in meshes generated from the dataset. The coloured patches (yellow, green and cyan) overlaid in Figure 3.3 represent areas where manual measurements were made for validation purposes. We re-examine this scene as it gives us a point of reference so that we can validate the automation of strike and dip color-coding presented in Chapter 5.

Figure 3.3: Photograph of the Ivy Lea survey site [51] with the three colors representing different joint sets (strike and dip in degrees in brackets): yellow (056/80), blue (139/88), and green (154/01). The dotted lines represent the boundaries of individual scans.

3.4.2 Limestone and dolostone at the Selle quarry

The second scene exhibits a layered sedimentary sequence composed of a massive limestone and a more friable dolostone from the Selle sand and gravel quarry in Douglas, Ontario. Due to the fact that large boulders were present in front of the scene, the LCS on its tripod was placed in a central location and the LCH rotated to take multiple scans. A total of 12 scans were required to completely capture the scene.

Each of the scans contains roughly 20% overlap to allow for point cloud alignment and registration. The extra points in the overlapping regions were removed so the entire merged scene contains 10,334,288 points. Figure 3.4 illustrates the LCS setup and the overlapping scans.

Figure 3.4: Each scan is labelled with their scan number. Number 1 is the first scan taken, number 2 is the second, and so on. The purple regions indicate the overlay between two scans and the shaded red region shows the laser scanning the first area and sixth area. The boulders can be seen in front of the LCS.

The scene which was captured by the LCS has been studied by Nurkhanuly [58]. The entire scene is part of the Paleozoic era (≈ 541 to 252.2 million years ago) of sedimentary rocks. The main mineral which makes up limestone is calcite which is chemically known as calcium carbonate (CaCO_3). The main mineral which makes up dolostone is dolomite which is chemically known as calcium magnesium carbonate ($\text{CaMg}(\text{CO}_3)_2$). Despite the fact that the dolostone only differs from the limestone by one element, magnesium (Mg), the dolostone is weaker. In field the two rocks are difficult to tell apart at a glance although close examination of their crystalline

structure will reveal differences. Another way to differentiate limestone and dolostone is to apply hydrochloric acid (HCL) to the areas of potential difference. The limestone rock will react chemically, which can be seen as effervesce, while the dolostone rock will not. Nurkhanuly [58] determined the locations of the limestone and dolostone through the HCL test and the pink line in Figure 3.5 marks the contact between the two different types of rock. Since the limestone and dolostone are difficult to differentiate macroscopically it would be interesting to study how well point clouds capture subtle differences in surface texture between the two types of rock. It is also a prime candidate for developing an approach for measuring roughness in point clouds as the more friable dolostone is rougher than the limestone.

Figure 3.5: Photograph of the Selle quarry survey site. The pink line separates the limestone on the left from the dolostone on the right [58]. The blue shaded region represents the area scanned by the LCS (approximately 9.6 m wide by 3.2 m high). Also seen in the photo is the author wearing an orange vest, the LCS mounted on a tripod and the large boulders preventing safe camera movement.

3.4.3 Sudbury shatter cones

The third scene exhibits shatter cone structures imprinted in an outcrop in the Sudbury impact crater, in northern Ontario. This structure was chosen to determine if

image processing strategies could enhance the subtle fan-like patterns associated with the shatter cones. Shatter cones are metamorphic rocks which are formed through shock metamorphism. Shatter cones have a distinct conical shape and they can range in size from microscopic to several meters. They are fairly rare as they are only found beneath meteorite impact craters or underground nuclear explosions. Thus the presence of shatter cones is direct evidence that the host rock has been subjected to a shock-wave with pressures in the range of 2 – 30 GPa [28] [65]. The direction of the shock wave can be determined from the orientation of the cone which, assuming no subsequent large-scale deformation, points toward the centre of the crater.

Three different locations were chosen to image shatter cones. Figure 3.6 is a map with each of the survey sites labelled with a number. The first survey site was on the campus of Laurentian University, the second survey site was on a bike path along Ramsey Lake, the third survey site was determined from a map provided by Deitz et al. [22] and is located on Gibson road. Figure 3.8 shows three photographs of shatter cones each from a different survey site. Instead of keeping the LCS in one location and rotating the camera head to take multiple scans, the LCS was moved laterally as seen in Figure 3.7. A total of 3 scans were taken at Laurentian University, 5 scans were taken on the bike path along Ramsey Lake and 2 scans were taken along Gibson road. This gives us 2,453,659 points for the scene at Laurentian University, 4,680,305 points for the scene along Ramsey Lake and 1,691,041 points for the scene at Gibson road.

3.5 Decimation

The point clouds acquired by the LCS are very large and thus require a large amount of processing time. Decimation was introduced to reduce the time required for advanced image processing. Decimation is a well studied topic and the methods fall

Figure 3.6: Map of the Sudbury, Ontario, retrieved on June 29, 2013 from www.maps.google.ca. Red dots represents the actual imaging site locations.

into two main categories: mesh-based and direct point cloud. We first review some algorithms found in the literature for both categories of decimation before introducing our decimation methods. For the definition of a mesh, see Chapter 4, Section 4.1.

Mesh-based Mesh based decimation first computes a mesh over the entire point set and then uses properties of the mesh or some shape descriptor of the mesh to perform decimation through re-meshing. The results can then be exported. Schrooder et al. [68] decimated meshes through geometric and topological characterization of the meshes locally. The approach removed vertices based on their deviation from the local average plane. Berg and Dobrindt [18] examined a hierarchy of levels of detail for a given mesh. This approach was motivated for a need to visualize detail when examining a mesh up close and less detail when it is examined from a distance. The greater the distance the less elements make up the mesh. Hoppe et al. [34] used complex energy functions to find the appropriate edges to collapse. A series of edge collapses would then produce a simplified mesh and the resulting points can be

Figure 3.7: Each scan is labelled with their scan number. Number one is the first scan taken, number is the second, and so on. The purple regions indicate the overlay between two scans and the shaded red region shows the laser scanning a portion of the scan. The green arrow indicates the direction that the LCS was moved to capture the multiple scans.

exported. This approach works well for uniform and noise-free data. Alliez et al. [2] have compiled a detailed survey on re-meshing algorithms all of which can be used to reduce the size of meshes. In general, mesh based decimation processes require a generation of an initial mesh on the very large dataset beforehand. On point clouds which contain noise this can be error-prone.

Direct point cloud Direct point cloud decimation attempts to reduce the size of the point clouds without generating any intermediate meshes. While these methods do not generate meshes they do use data structures to help partition the points. Lee et al. [49] first compute the normals for each point before using an octree, a form of 3D grid, and to extract a representative point for each of the 3D cells. The drawback

Figure 3.8: Photographs of shatter cones found in Sudbury. (a) Detailed shatter cones from Laurentian University campus, photographed from a distance of 1 m. (b) Group of several shatter cones imprinted on an outcrop along Ramsey Lake Road, photographed from a distance of 3 m. (c) A photograph of the scene on Gibson road. These more massive shatter cones are very distinct.

of such approaches is the reliance on the cell sizes. It is possible to join unconnected regions if the cell size is too large. Pauly et al. [59] use a moving least squares (MLS) method and an iterative decimation process similar to the the progressive meshes

of Hoppe et al [34]. Similar to how a least-squares method can fine a line to a set of points, the moving least squares method fits a surface to a set of points. Points with the largest error are removed and the MLS surface is recomputed. The iterative recomptuation of the MLS surface makes this method computationally expensive. In the literature, to the best knowledge of the author, direct point cloud decimation methods require some underlying structure to drive the decimation process.

3.5.1 Simple

A simple and unbiased way to reduce the resolution of a point cloud, without additional underlying data structures, is to simply select every n^{th} point. Because the LCS executes a raster pattern when acquiring voxels, point clouds decimated in this fashion exhibit organized patterns introducing an undesired directional bias to the data (Figure 3.9). Thus this sort of method is dependant on how the scanner acquires data. It also requires that the point clouds have no missing data as a small patch of missing data will affect the ordering of the points that are dropped.

Figure 3.9: Comparisons between the same 1024×1024 point cloud decimated by (3.9a) factor of $n = 4$, resulting in a resolution of 512×512 and (3.9b) factor of $n = 16$, resulting in a resolution of 256×256 . Note how the data is organized in vertical lines.

3.5.2 ε -nets

To address the issues described in Section 3.5.1 and to provide a better way to control the decimation process, we introduce a random sampling method known as ε -nets [79]. This method is designed to give good approximations of the input set relative to a geometric object. Given a set of points, P , in space it is easy to imagine that large convex shapes laid over top of P are so “fat” they cannot avoid “hitting” a certain number of points of P . An ε -net is a subset of P such that, when given a set of large convex shapes, C , some fraction of the ε -net points can be found within each of the shapes of C . This concept is best illustrated by an example. Let us define S as the square with a side length of 1 as shown in Figure 3.10. The red dots (which forms a square of side length $\frac{1}{2}$) contained within S can then be thought of as an ε -net of S where $\varepsilon = \frac{1}{2}$. A convex shape which intersects at least $\frac{1}{2}$ of S must contain one or more of the dots. Note that since S is a square, it can be represented as an infinite number of points similar to how a line can be represented by an infinite number of points. The $\frac{1}{2}$ -net thus captures the shape of S using only 4 points. By decreasing ε , we decrease the size of the convex shape which intersects with S and thus the size of the ε -net which represents S increases and the accuracy of the ε -net approximation increases.

Figure 3.10: Example of how decreasing ε increase the accuracy of the ε -net.

The choice of convex shape also has an impact on the size of the ε -net. Different convex shapes have different VC-dimension [79] which reflects how complex of a set the shape can shatter. The shattering function of a set K can be thought as a special kind of covering where one attempts to use the selected shape and fit all possible subsets of K with it. For example, if we choose the half-plane, we can show that it has a VC-dimension of 3 since that is the maximum number of points the half-plane can shatter. In Figures 3.11a to 3.11d we have a set of 3 points which we will denote as X .

Figure 3.11: Example of shattering point sets with half-planes.

We can see that all the subsets of X which contain just one point can be separated by using one half-plane (Figure 3.11b). All the subsets containing two points can be also separated by using one half-plane (Figure 3.11c). All points can be separated by placing the half-plane away from all the points (Figure 3.11d). Figure 3.11e shows 4 points on the plane and a possible subset containing two points is highlighted in red. This subset cannot be within the range of one half-plane without including other points of X . Thus, the subset shown in Figure 3.11e (and in general any 4 points on a plane) cannot be shattered by half-planes. The more complex the shape, the higher the VC-dimension and a shape with high VC-dimension requires a larger ε -net to capture shapes. It can be shown that for any set S an ε -net exists with the following number of points:

$$c \times \left(\frac{d}{\varepsilon}\right) \ln\left(\frac{d}{\varepsilon}\right) \quad (3.4)$$

where d is the VC-dimension of the covering convex shape and $c \geq 1$ is a real number

which allows for a finer control over the number points in the ε -net. Thus when given a large set of points P we can approximate it by selecting the number of points given by Equation 3.4 at random. The selected sample, S , would then be guaranteed to be large enough such that there is at least one point in S which will be covered by all shapes which cover an ε portion of P . The strength of using ε -nets is in their ability to capture the geometric shape of an object while remaining simple to implement. They also give an easy way to control how accurate the resulting decimated point cloud will be through the ε parameter. No underlying structures are needed to drive the decimation process and VC-dimension theory guarantees an ε covering of the input point set. The accuracy of the ε -net can also be found through algorithms described by Berg et al [19].

Figure 3.12: Single image of a rock wall from the limestone/dolostone dataset is decimated using different parameters to produce ε -nets. The resulting point clouds have been covered with meshes to improve visualization. The parameters used are as follows: (a) original image (b) $\varepsilon = 0.0001$, $d = 3$, $c = 1$ (c) $\varepsilon = 0.0005$, $d = 3$, $c = 2$ (d) $\varepsilon = 0.001$, $d = 3$, $c = 1$

Chapter 4

Surface Reconstruction and Mesh Operations

In this chapter we describe two surface reconstruction algorithms which were used to create a triangulated mesh for the image processing algorithms discussed in Chapters 5, 6 and 7. Triangulated meshes generated from both algorithms were used as a way to validate that the methods developed in Chapters 5, 6 and 7 do not depend on a specific algorithm. A mesh operation known as smoothing is detailed. The content presented in this section is not an original contribution but is required for the processing methods described in Chapters 5, 6 and 7. The author implemented the mesh smoothing algorithm and the two surface reconstruction algorithms were implemented in other software which the author uses to generate the meshes.

4.1 Triangulated Meshes and Planar Graphs

A graph, $G = \{V, E\}$, is a pair of sets where one of the sets, V , contains only vertices and the other set, E , contains edges where an edge is a 2-element subset of V . Given a set of points P in the plane, an *embedding* of a graph $G = \{V, E\}$ is a mapping $\phi = V \rightarrow P$ where each edge $(x, y) \in E$ is represented by the line

segment with endpoints $\phi(x)$ and $\phi(y)$. An embedding of a graph is planar provided that adjacent edges intersect only at their common endpoints and non-adjacent edges do not intersect. That is not to say that a certain representation of a planar graph cannot have edges that intersect, there just needs to be a way of embedding the graph such that no edges intersect. Fáry [23] proved that every planar graph has a planar embedding with line segment edges. See Figure 4.1 for visual examples of 4.1a a planar, 4.1b a non-planar graph and 4.1c a planar graph that has edges that cross. For more details on planar graphs see [17] [72].

Figure 4.1: (a) Embedding of the a graph where no edges overlap. (b) Graph where it is impossible to move the vertices such that none of the edges overlap making the graph non-planar. (c) Example of a non-planar embedding of (a).

A plane graph of a point set P is a planar embedding of a graph. Throughout this thesis, we assume that edges are embedded as line segments. A triangulated mesh is a planar graph where every face except the outer face is a triangle. Any triangulated mesh over a point set P is considered a *triangulation* of P . Since triangulations are planar embeddings of point sets we can apply Euler's formula for planar graphs to get an upper bound on the number of triangles and edges that a triangulation can contain [72]. To determine this upper bound using Euler's formula we need to make use of a structure known as the *convex hull* of a point set P . The convex hull of P , denoted as $CH(P)$, is the smallest convex set containing P [72]. See [17] [72] for

formal definitions and proofs of the properties of convex hulls. With the convex hull defined we can use the following theorem [72]:

Theorem 1. *Let P be a set of n points in the plane such that no points are collinear and let k be the number of points in which lie on the boundary of the convex polygon produced by $CH(P)$. Any triangulation of P contains $2n-2-k$ triangles and $3n-3-k$ edges.*

Theorem 1 allows us to conclude that *any* triangulated mesh will have a linear number of triangles with respect to the number of points which make up the mesh. Since a single scan of the LCS produces a million points, and a reconstructed surface typically contains $\leq 2n$ triangles (see Chapter 4 for surface reconstruction algorithms) where n is the number of points, we are performing the equivalent of a million measurements. The amount of time it takes for a single scan at a resolution of 1024×1024 is 20 minutes which means it is able to perform a million measurements in 20 minutes. This is a large decrease in time when compared to taking manual measurements [51].

4.2 Delaunay triangulation

This section will briefly detail the properties of a Delaunay triangulation [20] and how it can be constructed. Since the Delaunay triangulation is a form of triangulation, it has a linear number of faces and edges with respect to the number of points used to create the triangulation. For this thesis we use a 2.5D Delaunay triangulation where the z coordinate of the points represents the camera-target distance.

4.2.1 Properties

In this section we detail the properties which separates the Delaunay triangulation from other triangulations. We will see later in Section 4.2.2 how some of these properties can be used in the construction of a Delaunay triangulation.

Empty Circle

Let $e = (p_a, p_b)$ be an edge that is incident to triangles $\Delta p_a p_b p_c$ and $\Delta p_a p_b p_d$ as shown in Figure 4.2. The edge e is considered to be *illegal* if p_d lies in the interior of the circumcircle of $\Delta p_a p_b p_c$ or p_c in that of $\Delta p_a p_b p_d$. In a Delaunay triangulation, no illegal edges are allowed and thus the circumcircle of any triangle within the triangulation is empty and does not contain any points in the original input point set. A direct result of having no illegal edges means that each triangle within the triangulation has an empty circumcircle (see Figure 4.2b).

Figure 4.2: (a) An example of an illegal edge. (4.2b) Part of a triangulation is shown as well as the triangles circumcircles. Note that each circle does not contain any points despite overlapping.

Local Maximum Minimum Angle

A Delaunay triangulation maximizes the minimum angle. That is, compared to any other triangulation of a point set, the smallest angle in the Delaunay triangulation is at least as large as the smallest angle in any of the others [17]. This property is important as it means that the triangulation will minimize the number of thin triangles. For example consider the quadrilateral formed by the two triangles $\Delta p_a p_b p_c$ and $\Delta p_a p_b p_d$

as shown in Figure 4.3a. The smallest angles contained within the triangles are smaller than the angles found in the triangles which make up the quadrilateral in Figure 4.3b. By “flipping” the edge p_ap_b we obtain a different triangulation over the same point set which maximizes the minimum angles. See [17] for a formal proof.

Figure 4.3: An example of how “flipping” an edge can maximize the minimum angle and restore the empty circle property.

4.2.2 Construction

In Section 4.2.1 we observed the interplay between the empty circle and local maximum minimum angle properties. What ties them together is the existence of illegal edges. The existence of an illegal edge shared by two triangles implies that the circumcircles of the triangles are not empty and thus the triangles have small angles. These angles can then be maximized and the empty circle criterion can be satisfied by flipping the illegal edge. In this section we take a look at how the Delaunay triangulation can be constructed by randomized incremental insertions of points. We cover this particular construction algorithm as it is the one used in the software library which generates the meshes [8].

Randomized Incremental

Suppose we already have a Delaunay triangulation T over a point set P and we would like to insert one point, v , into the triangulation. There are three cases of how this scenario can play out (ignoring the possibility of duplicate points). The point v could fall:

1. inside a triangle of T
2. onto an edge of T
3. outside of T

In the first case we would need to find the triangle of T that contains v and then add the appropriate edges. In the second case we would simply need to add the edges to connect v to the remaining point which makes up the two triangles that the edge is incident to. For a visual example of these two cases see Figure 4.4. It should be noted that an edge can become illegal only if one of its incident triangles has changed. Thus while the newly added edges are legal, as the previous triangulation contained only legal edges, the outer edges can become illegal causing a chain reaction of edge flips. See [17] for a more detailed explanation as to why newly added edges are legal.

Figure 4.4: An example of the two main cases one encounters when inserting a point into a Delaunay triangulation. The dashed red lines indicate the new edges to be added.

To handle the third case we start the out with a large triangle enclosing our entire point set. This large triangle allows us to reduce the third case into the first case as the entire point set is contained within the large triangle. As the name of this section implies, the algorithm is incremental thus we start with no points in T and we add one point at a time until all of P has been added into T . Since T is a delauny triangulation at each step, when the algorithm terminates T is a delauny triangulation of the entire point set P . When checking for illegal edges we ignore those of the large bounding triangle as it is there to help with construction and will be removed at the end. To ensure that the large bounding triangle encloses the point set we can choose the vertices to be $v_1 = (3M, 0)$, $v_2 = (0, 3M)$ and $v_3 = (-3M, -3M)$ where M is the maximum absolute value of any coordinate in the input point set. See Figure 4.5 for a visualization. To handle the possibility of having a large amount of edge flips per point inserted into T we randomize which points we add. It can be shown that by randomizing the order in which the points are inserted, we only flip on average a constant number of edges [17].

Figure 4.5: Visualization of a bounding triangle over some point set.

There is one more issue that we need to address before we can complete the algorithm. We need a way to find out which triangle or edge that a point falls onto. The approach is essentially the same as the point location algorithms used for finding out which face a point falls in for planar graphs. Detailed analysis of such algorithms can be found in [17] and [73]. Now that we have covered the basics, we can give an overview of the randomized incremental algorithm:

Algorithm 1 Randomized Incremental

- 1: Initialize triangulation DT with the large bounding triangle.
 - 2: Initialize point location structure D .
 - 3: Choose a random point v from P .
 - 4: Find the Δt that contains v using D .
 - 5: Subdivide the Δt into smaller triangles each of which has v as a vertex.
 - 6: Update D to include these new triangles.
 - 7: Legalize all the edges that might have become illegal and update D again.
 - 8: Repeat steps 3 to 7 until all points in P have been added to DT .
-

4.3 Poisson reconstruction

In this section we provide a brief summary of the Poisson surface reconstruction algorithm as an alternative to the Delaunay triangulation. We will not go into detailed analysis of the algorithm as it was not implemented by the author. The author does use existing software [13] which utilizes this algorithm to generate meshes. It should be noted that unlike the Delaunay triangulation, the Poisson reconstruction requires the each of the input points to have an associated normal vector. If the points do not have an associated normal vector then a pre-processing step is required to orient the points.

This surface reconstruction method was introduced by Kazhdan et al [42] and is part of the family of algorithms that rely on solving implicit functions in order

to reconstruct a surface. The implicit function approach to surface reconstruction defines a function $f(S)$ for a surface S such that $f(S) = 0$ outside the surface S and $f(S) > 0$ inside the surface S . Such a function is called the *indicator* function. Thus if one were to extract the zero-set from the indicator function, we would have the boundary of the surface. The approach taken by Kazhdan et al [42] is to first assume that the input set of points P are oriented. Thus each point $p \in P$ has an associated normal and there exists a normal field of P . Given this normal field the gradient of the indicator function is approximated and the indicator function is derived from the gradient. Once we have the indicator function the surface of the object represented by the input can then be extracted. Figure 4.6 illustrates this concept visually for a 2D point set. Details on how the gradient and indicator functions are computed can be found in their original paper [42]. The surface is extracted by computing an octree, the 3D analog of quad-trees, over the input point set and then using the marching cubes algorithm. The details of the surface extraction method through octrees and marching cubes can be found in another of Kazhdan et al's papers [43].

Figure 4.6: Intuitive illustration of Poisson reconstruction in 2D. Image source: Kazhdan et al [42].

4.4 Comparison of Delaunay and Poisson using acquired datasets

In this section we compare the meshes generated from a 2.5D Delaunay triangulation approach and the Poisson surface reconstruction approach using the datasets described in Chapter 3. Figure 4.7 shows a comparison between the two meshing algorithms using the Selle quarry dataset. Since the Poisson reconstruction assumes the point cloud represents a fully enclosed surface the algorithm usually outputs a large number of degenerate triangles in an attempt to form a water-tight closed mesh [42]. This effect is visible in Figure 4.9.

The Ivy Lea dataset also posed problems for the Poisson reconstruction as seen in Figure 4.10. Note that the problems for the Ivy Lea dataset are a result of a poor fitting of the indicator function which is why part of the data was considered to be “inside”. Fine tuning the parameters for the algorithm is required to get the indicator function just right. While the Poisson reconstruction did have some problems with the datasets, it also has advantages as seen in Figure 4.8d. The same problems presented in Figures 4.9 and 4.10 is also what allows the Poisson reconstruction to handle the occlusion zones better than the 2.5D Delaunay triangulation approach. By assuming that the point cloud is made up of fully 3D objects the occlusion zones are then filled in with similar size triangles as to represent a solid surface rather than the large triangles which represents empty space. The Poisson reconstruction also handles 3D noise better than the 2.5D Delaunay triangulation approach as seen in Figure 4.11. The red rectangle highlights data which represents grass that was in front of the shatter cone outcrop. This data was kept intentionally as a test to see how the meshing algorithms as well the methods presented in Chapters 5, 6 and 7 would handle this sort of noise. The Poisson reconstructs “blobs” while the 2.5D Delaunay triangulation method produces very sharp triangles.

Since our datasets are primarily outcrops of rock walls, and the LCS was not fully rotated to capture a full 3D object, a 2.5D Delaunay triangulation is good choice as smaller holes are filled and manual removal of degenerate triangles is not needed. The 2.5D Delaunay triangulation also does not require oriented points like the Poisson reconstruction removing the additional step of determining the normals of the point cloud. The problems the Poisson reconstruction had with the Ivy Lea dataset found in Figure 4.10 and the fact that the Delaunay triangulation is well studied, well understood and required no additional parameter adjustment further motivated the author to use the 2.5D Delaunay meshes as the main surfaces for the analysis found in Chapters 5, 6 and 7. The Poisson surfaces were used to validate that the methods were not dependant on the reconstruction algorithm.

(a) 2.5D delaunay triangulation

(b) Poisson

Figure 4.7: A comparison between the 2.5D Delaunay triangulation mesh and the poisson mesh for the limestone/dolostone dataset. The region contained within the red square can be seen in greater detail in Figure 4.8. Outer and degenerate triangles are removed manually in Figure 4.7b.

(a) Poisson mesh view

(b) 2.5D TIN mesh view

(c) Poisson wireframe

(d) 2.5D TIN wireframe

Figure 4.8: The above shows a close up comparison of the sections highlighted in Figure 4.7. Figures 4.8a and 4.8c are from the Poisson reconstruction and Figures 4.8b and 4.8d are from the 2.5D Delaunay triangulation. Note that the larger triangles in Figures 4.8b and 4.8d represent occlusion zones. In the Poisson mesh the over the same area is represented with smaller triangles which fill the occlusion.

(a) Wireframe and triangles partially transparent.

(b) Outer triangles are deleted revealing the actual surface inside.

Figure 4.9: The Poisson reconstruction produces a watertight surface as see in Figure 4.9a. Note the additional degenerate “blob meshes” highlighted in red. These are outliers and since the normal estimation only looked at the nearest $k = 15$ neighbouring points, the points are assumed to be separate objects in space.

(a) Point cloud view.

(b) Reconstructed surface.

(c) Overlay of Figures 4.10a and 4.10b.

Figure 4.10: A problematic dataset for the Poisson reconstruction.

(a) 2.5D Delaunay triangulation

(b) Poisson

Figure 4.11: A comparison between the 2.5D Delaunay triangulation mesh and the Poisson mesh for the shatter cone dataset.

4.5 Smoothing

To reduce the noise of the instrument and improve visualization smoothing algorithms were used on the reconstructed surface. This section details two smoothing algorithms that were used to smooth meshes. More detailed analysis of various smoothing algorithms can be found in the paper by A. Belyaev and Y. Ohtake [5]. The main smoothing algorithm used to reduce the noise of the point clouds is detailed in the next section.

4.5.1 Geometric laplacian

The *geometric laplacian* was originally introduced by Karni and Gotsman [41] as a way to measure the smoothness of a vertex v in a mesh M based on the Euclidean distances of the neighbours of v . Formally it is defined as follows:

$$GL(v) = v - \frac{\sum_{i \in n(v)} l_i^{-1} v_i}{\sum_{i \in n(v)} l_i^{-1}} \quad (4.1)$$

where $n(v)$ is the set of indices which correspond to the neighbours of v and l_i is the Euclidean distance from v to v_i . The term $GL(v)$ represents the difference vector between v and its new position after a single laplacian smoothing step. The greater the difference vector the less smooth the vertex. To smooth a mesh M one would find take each vertex v in M and offset it by $GL(v)$ for n iterations. The greater n is, the smoother the resulting mesh. See Figure 4.12 for an example of the geometric laplacian applied to a single scan of the shatter cone dataset.

(a) Original Mesh.

(b) Geometric laplacian.

Figure 4.12: A comparison between the original mesh (top) and the smoothed geometric laplacian version (bottom).

Chapter 5

Planar Orientation Mapping

In this chapter we examine methods for labelling the triangles within a mesh according to their orientation, more specifically, according to strike and dip (Section 5.1) as well as the normal orientation (Section 5.2). The mapping of planar orientation is important for geomechanical structural analysis since it is an indicator of rock mass strength [33] [32] as well as the stresses it has received [40]. The stronger the rock mass, the more stable the environment. Stable environments are important for the safety of mining crews as well as the structural integrity of the mine itself.

Mapping fracture orientation for geomechanical structural analysis has been explored. Publicly available tools are COLTOP 3D and Split-FX. Both have their limitations with fully 3D point clouds [39] [71]. COLTOP is not suited for discrete fracture analysis while Split-FX is limited to a 2.5D lattice gridded mesh resulting in large regions of occlusion and the inability to detect all the joint surface orientations (see Section 2.3 for definition of joints and joint sets). More recently Lato et al [47] has explored detecting planes in rock masses by looking at the angular difference of vertex normals. For each vertex, v , the mean angular difference is computed with the surrounding neighbours. Any vertex with a large mean angular difference is considered a “sharp” vertex and is excluded from the planar set. The set of vertices which

are not sharp is then processed again to find disjoint sets which represent individual planes. Planes which have a low surface area are excluded and the orientation of each of these planes is computed and grouped together. Each group is then considered a joint set for the rock mass and is color-coded with a separate color. The requirement for vertex normals means that in raw point clouds they need to be computed. In our approaches we use the normals of each triangle to compute the geological strike and dip as well as another variant of planar orientation. The main motivation and contribution for this is not to detect discrete planes but to visually enhance the orientation of the overall geological structure in support of geological interpretation.

5.1 Automated strike and dip

In this approach for mapping planar orientation, we consider the plane containing each triangle and thus the strike and dip can be computed for each triangle. To measure the strike and dip of a plane P we first take a horizontal plane H and intersect it with P (see Figure 5.1). We then compute the orthogonal vector with respect to $\vec{n}$ that is also contained by H which we denote as $\vec{v}$. We also compute the angle $\vec{n}$ makes with respect to H which we denote as θ . The angle that $\vec{v}$ makes with respect to the North direction is the strike angle. The North direction is known from measuring the orientation of the LCH in the field during data acquisition. The dip is found by subtracting θ from 90° . The actual computation of strike and dip was integrated from the method developed by Mah et al [51], and our contribution here is to take the orientations and map them onto the mesh itself. The advantage of this approach is that one can visualize the joint sets directly on the mesh.

Figure 5.1: Visualization of the plane being measured (blue) and the horizontal cutting plane (purple).

5.2 Outer normal orientation

In this variant for mapping planar orientation, we take the normal vector of each triangular mesh element and measure the angle of deviation, α , with respect to the z -axis. Figure 5.2 illustrates this using a cone with normal vectors and the z -axis going through the center of the cone. When the normal vector is parallel to the z -axis, α is equal to 0° . When the normal vector is perpendicular to the z -axis, α is equal to 90° . Thus the angle α measures a normal vectors deviation with respect to the z -axis. The angle β is measured by projecting the normal vector directly onto the xy -plane as in Figure 5.2. Thus β ranges from 0° to 360° . Together the angles α and β represent the orientation of any normal vector $N = (nx, ny, nz)$. We formally define α and β below:

$$\alpha = \arctan2(\sqrt{nx^2 + ny^2}, nz) \times \frac{180}{\pi} \quad (5.1)$$

$$\beta = \arctan2(nx, ny) \times \frac{180}{\pi} \quad (5.2)$$

The $\arctan2(y, x)$ function for Equations 5.1 and 5.2 has two arguments. It returns the angle in radians between the positive x -axis of a plane and the point given by the coordinates (x, y) , thereby indicating the appropriate quadrant of the computed

angle, an information not available when the single-argument arctangent function is used.

Figure 5.2: An illustration of α and β .

5.3 Hue Saturation Lightness color model

The Hue Saturation and Lightness (HSL) color model was chosen for coloring the orientation of the triangles according to their orientation. The hue corresponds to a color wheel (see Figure 5.3). Saturation is described as the perceived intensity of a specific color. The greater the saturation, the more vivid the color. Intuitively, lightness can be thought of how much light is shining on an object [1].

Since the strike and the β -angle of the outer normal range from 0° to 360° , they can be mapped directly to the hue (H) component of the color model. The dip and the α -angle of the outer normal, which range from 0° to 90° are mapped to the saturation (S) component which ranges from 0 (light grey) to 1 (the color represented by hue). The lightness (L) component also ranges from 0 to 1 and is kept at a constant value of $\frac{1}{2}$.

Since the main color model used for rendering computer graphics is the Red Green Blue (RGB) color model, the HSL color values need to be converted into RGB. Given a HSL color with hue $H \in [0^\circ, 360^\circ)$, saturation $S \in [0, 1]$ and lightness $L \in [0, 1]$ we first compute what is known as the chroma C as well as the values H' and X :

$$\begin{aligned} C &= (1 - |2L - 1|) \times S \\ H' &= \frac{H}{60^\circ} \\ X &= C(1 - |H' \bmod 2 - 1|) \end{aligned}$$

The RGB values unadjusted by light are then computed as follows:

$$(R', G', B') = \begin{cases} (0, 0, 0) & \text{if } H \text{ is undefined} \\ (C, X, 0) & 0 \leq H' < 1 \\ (X, C, 0) & 1 \leq H' < 2 \\ (0, C, X) & 2 \leq H' < 3 \\ (0, X, C) & 3 \leq H' < 4 \\ (X, 0, C) & 4 \leq H' < 5 \\ (C, 0, X) & 5 \leq H' < 6 \end{cases}$$

Finally, to adjust the (R', G', B') we compute:

$$\begin{aligned} m' &= L - \frac{1}{2}C \\ (R, G, B) &= (R' + m, G' + m, B' + m) \end{aligned}$$

Figure 5.3: Hue colour wheel.

5.4 Color-coded geological scenes

Using the different planar orientation measures (Section 5.1 and 5.2) we now color-code geological scenes to enhance features of interest. The Ivy Lea scene, with its large cubic blocks, affords an opportunity to enhance planar orientation. The 6 images taken at Ivy Lea had already been merged and reduced in size [51], thus ε -net decimation was not applied.

Figure 5.4 shows the combined strike and dip overlaid on the Delaunay mesh before and after smoothing using the geometric laplacian. This site has been studied by Mah et al. [51] who identified three joint sets (strike/dip in degrees) using the pole density contouring method: 043/84 (most prominent), 132/86, and 154/10 (less prominent). These joint sets can be easily recognized in Figure 5.4a and 5.4b as patches of yellow/blue, green/magenta and dark grey, respectively. Figure 5.4c displays the results of subtracting 180° from any triangle that has a strike of greater than 180° . The result of this is that overhangs are removed (see Section 2.3.1) and the major joint sets are further enhanced. Figure 5.5 shows the joint set 043/84 highlighted.

Figure 5.6 shows the outer normal orientation mapping on the shatter cone dataset from Laurentian University. This dataset shows a close-up view of a shatter cone and its typical fanning shape is visible. The main differences between the strike and dip and the outer normal orientation is the axis which is used to measure the angles. The strike and dip requires a North direction while the outer normal orientation always uses the Cartesian axis. Since the strike and dip are used in geology, they are well suited for measuring fracture orientations for rocks and applications which require georeferencing. The normal orientation is useful for when georeferencing information is not available or not required.

Figure 5.4: Delaunay mesh of the Ivy Lea scene (a) before and (b) after smoothing using the geometric laplacian. Strike is color-coded according to the color wheel shown in Figure 5.3. Dip ranges from light (sub-vertical planes) to dark (sub-horizontal planes). (c) shows the removal of the overhangs.

Figure 5.5: Only triangles with a strike between $45 \pm 10^\circ$ and a dip between $80 \pm 10^\circ$ have been colored to highlight the most prominent joint set. This particular joint set corresponds to the yellow patches in Figure 3.3

Figure 5.6: A close-up photo of the shatter cones is shown in 5.6a and 5.6b to 5.6d show the Delaunay mesh color coded according to their outer normal orientation. Figure 5.6b shows α which ranges from light (sub-vertical planes) to dark (sub-horizontal planes). 5.6c shows β which is color-coded according to the color wheel shown in Figure 5.3 and 5.6d show α and β superimposed.

Chapter 6

Roughness Mapping in Point Clouds

To the best knowledge of the author, there is no roughness measure defined for point clouds of geological context. In this chapter, we will look at image processing algorithms for reconstructed surfaces from point clouds in order to define a roughness measure. The algorithms presented are based on a key geometric property of surfaces, *curvature*, for which we will give an intuitive definition. Formal definitions of curvature can be found in Appendix B. We will introduce an operation on surfaces known as the *local window* (see Section 6.1) which is required for determining the curvature of vertices as curvature is not defined for a vertex. Once we know how to compute the curvature for surfaces, we provide a visualization of curvature mapped onto the limestone/dolostone dataset. We look at three different methods for determining roughness using the curvature of the surface: curvature difference (Section 6.4), quad-tree roughness ratios (Section 6.5), and roughness sampling (Section 6.6). The main contributions of this chapter is the application of the curvature difference roughness measure to geological structures, the quad-tree roughness ratio framework and the roughness sampling method for measuring roughness in point clouds.

6.1 Local window

Local window is a concept taken from Lavoué [48] which selects the neighbouring elements of a mesh from a source vertex. Thus for each vertex v in a mesh M we can define a local window which we denote as $W_r(v)$ where r represents the radius of a sphere centered on v . $W_r(v)$ would then contain a set of objects that are within this sphere (see Figure 6.1a). The boundary of the sphere would also intersect edges and thus for all the edges that touch the sphere we add additional points which are called *edge points*. Interpolation is required for the edge points (see Figure 6.1b) to determine their position and any other attributes such as their curvature. A graph searching algorithm such as Breadth-First Search (BFS) [45] can be used to find all the objects contained within $W_r(v)$. In this thesis the local window radius is taken to be a percentage (most commonly 1% to 5%) of the bounding box diagonal.

Figure 6.1: Visual example of a local window and a part of a mesh projected onto a 2D plane. Figure 6.1a shows the local window of a vertex v . Edge points are green and vertices within the local window are blue. An edge is highlighted and expanded in Figure 6.1b. The terms d_1 and d_2 are the distances from vertices v_1 and v_2 to an edge point v_e .

6.2 Intuitive definition of curvature

Intuitively, the curvature of a surface, S , can be thought of as the amount from which S deviates from being flat. This deviation can be measured by looking at the rate of change on S which gives us an idea of how much S bends at any point on the surface. Before we look at how curvature can be measured for a 3D surface let us first look at the curvature for a 2D curve. A straight line would be considered flat and thus has zero curvature at all points on the line. A curved line should have a curvature relative to how much it bends. We can capture this concept by approximating the curve with a circle. This circle would have to have an infinite radius to fit a straight line. The circle's radius would decrease as we attempt to fit it to a curved line. These concepts are illustrated in Figure 6.2. Thus the curvature of a point on 2D curves can be defined as $\kappa = \frac{1}{r}$ where r is the radius of the circle which fits the curve at that point.

Figure 6.2: Examples of circles fitted to a curve illustrating how the circle radius decreases as the curve is “bent” more sharply.

For the curvature of a point p on a 3D surface we look at all the 2D curves made by cutting a plane with p . Figure 6.3 illustrates this concept visually. Given all the possible curves created through this process we can find two curves which have a

maximum (κ_{max}) and minimum (κ_{min}) curvature. These two values are the principal curvatures and $\kappa_{mean} = (\kappa_{max} + \kappa_{min})/2$ defines the mean curvature. Since a surface, S , embedded in Euclidean space $\mathbb{R}^3$ is orientable [74] the curvature is also oriented (see Figure 6.4 for a 2D example). Formal definitions of curvature can be found in Appendix B.

Figure 6.3: An illustration of different planes cutting through p creating different curves (red). The curves with the maximum and minimum curvature defines the mean curvature. The arrow shows the rotation of the plane to cover all possible 2D curves.

Figure 6.4: A visual of how orientation of a curve affects the sign of the curvature.

6.3 Visualization of curvature on meshes

The point clouds of the 12 adjacent images acquired at Selle quarry were aligned and merged in Polyworks and the resulting 10,334,288 points exported. The size of the merged point clouds were then reduced with ε -nets with $\varepsilon = 0.0005$, $d = 3$ and $c = 2$ which results in a dataset of 104,394 points. A mesh was then generated using software written by Boaz [8] who uses the incremental Delaunay triangulation method. Figure 4.7a shows a 2.5D TIN mesh of the limestone/dolostone rock face which was smoothed after 4 iterations of the geometric laplacian. Figure 6.5 displays the curvature maps for two different local window radii. The curvature maps show the limestone to be relatively flat and the dolostone to be more irregular, reflecting its rougher surface. For the shorter radius map (Figure 6.5b) lines of higher curvature coincide with the limestone bedding planes. By increasing the size of the local window used to compute the curvature, we increase the processing time and the more subtle details of the surface are averaged. The main purpose of visualizing curvature on meshes is to give the reader intuition as to how the curvature of a surface can define regions which are considered to be geologically rough. The main contributions of this chapter is to use this intuition to develop and apply methods which use curvature to define a roughness measure.

Figure 6.5: Visualization of curvature for the limestone/dolostone dataset. A color key is shown at the top.

6.4 Roughness through curvature difference

One approach for measuring roughness of a mesh M , which was explored by Lavoué [48], takes a look at the difference between the curvature of M and the curvature of a smoothed version of M which we will denote as M^s . We denote $M(V)$ to be the vertices of M and similarly for M^s . Lavoué defined an *adaptive smoothing* operation which acts on all vertices contained within a local window. For a radius r and a vertex v , the smoothed vertex v' is defined as follows:

$$v^m = (v + \lambda s(v)) \text{ with } s(v) = \frac{1}{n} \sum_{e_i \in W_r(v)} (e_i - v)$$

$$v' = (v^m + \mu s(v^m))$$

where e_i is the i^{th} edge point of the local window $W_r(v)$, n is the number of edge points in $W_r(v)$, λ and μ are defined to be 0.6307 and -0.6732 . These values were suggested by Taubin as part of the two-setp Taubin filter [77].

Listed below are the steps to computing the roughness map as defined by Lavoué:

1. Mesh is smoothed using adaptive smoothing with local window radius r .
 2. The curvature values κ_{max} for each vertex v is computed for M and the M^s .
 3. The average curvature values κ_{avg} is computed for each vertex v with a local window radius of $2r$. This radius is chosen as it can correct some of the noise that has been suppressed by the adaptive smoothing [48].
 4. The roughness values are computed for each vertex v using the asymmetric difference between the κ_{avg} values of the original mesh and the smoothed mesh.
- This gives us the following relation for the roughness at a given vertex v and

its corresponding vertex on the smoothed mesh v^s :

$$R(v) = \begin{cases} \kappa_{avg}(v) - \kappa_{avg}(v^s) & : \text{if } \kappa_{avg}(v) > \kappa_{avg}(v^s) \\ 0 & : \text{otherwise} \end{cases}$$

The collection of all $R(v)$ values for all v in the mesh is the roughness map $R(M)$.

Figure 6.6 shows the results of applying the curvature difference method to the limestone/dolostone dataset with $r = 84$ cm and $r = 120$ cm. As seen in Figure 6.5, increasing the local window radius, r , produces larger patches of higher curvature values. This effect can be seen in the curvature map as the differences are now cover a larger area.

Figure 6.6: Note that Figure 6.6b has a larger range than Figure 6.6a meaning that larger roughness values were computed. The roughness values were scaled using the square-root function thus the color scale represents non-linear values. The effect of this scaling is that a few extreme values will not skew the coloring.

6.5 Quad-tree roughness ratio

In this section we look at using a combination of clustering curvature and quad-trees to define roughness. Using a clustering algorithm and a space partitioning data structure together to provide a measure of roughness is our main contribution. Section 6.5.1 provides an overview of the clustering algorithm used in this approach and Section 6.5.2 defines the quad-tree. Since the curvature captures how much a surface bends at a given point, a surface which is rough should contain vertices which exhibit a wide range of curvature. Given any $S \subset M$ we can say that S is rough with respect to M if it contains many vertices which are of “extreme” curvature. Figure 6.7 illustrates this concept visually.

Figure 6.7: The red dots represent vertices of extreme curvature while the blue dots represents non-extreme curvature. The top two squares correspond to a flat surface as they have a few number of red dots. The bottom two squares correspond to a rough surface because they have a large number of red dots. It does not matter how the red dots are distributed in space, both the bottom squares are “rough” areas.

Using this intuitive concept, we introduce the pre-processing step of the algorithm which:

1. Determines which of the vertices of V are of extreme curvature
2. Sub-divide the Euclidean space that V is embedded in so that we can assign a roughness value to portions of space

We pre-process a mesh $M = (V, E)$ for vertices V and edges E as follows:

Algorithm 2 Quad-tree roughness preprocessing

Require: $M = (V, E)$, local window radius r , maximum width w of a quad-node

Ensure: Clustering of V into two sets, quad-tree Q over V

for $v \in V$ **do**

$E = W_r(v)$

$v.curvature = curvature(v, E)$

end for

 Cluster V into two sets H which contains extreme curvature vertices and $L = V - H$

$Q \leftarrow \text{build-quad-tree}(V)$

Once the mesh M has been pre-processed, we can label any subspace (i.e: quad-node) with a roughness value. The size of the subspaces is determined by the depth of the quad-tree. Following our initial intuition the roughness value is then defined as the ratio of the total number of vertices within the subspace verses the number of extreme curvature vertices in the same subspace (see Algorithm 3). How well this method performs depends entirely on the clustering algorithm used to determine exactly what an “extreme curvature” vertex is. Rationale for our clustering algorithm choice is found in Section 6.5.1 and Appendix C covers the specific details.

Algorithm 3 Quad-tree roughness ratio

Require: Quad-tree Q , extreme curvature vertex set H , $i \leq height(Q)$

Ensure: A roughness mapping for each quad-node q at level i in Q

1: $P \leftarrow$ all quad-nodes of Q at height i

2: **for** $p \in P$ **do**

3: $S \leftarrow p.elements$

4: $X \leftarrow S \cap H$

5: **for** $v \in S$ **do**

6: $v.roughness = \frac{|X|}{|S|}$

7: **end for**

8: **end for**

6.5.1 Clustering curvature

Gravitational clustering [29] was used to separate the vertices of a mesh into extreme curvature and non-extreme curvature. This algorithm (details found in Appendix C) was chosen because it is unsupervised and clusters are formed naturally based on the force of gravity. The input parameters of the algorithm determine the final clustering. The description of these parameters and their sensitivity to change is summarized in Table C.1. The parameter values used in this thesis is summarized and discussed in Table C.2.

Since gravity is applied to bodies in space with mass, we let each vertex v of a mesh M represent a particle in space. We will denote the particle of v as $p(v)$. Since each particle, $p(v)$, will move according to gravity, using just the spatial coordinates of $p(v)$ will produce a single large cluster as point clouds are fairly dense. To overcome this, and have the clustering reflect the curvature of the surface, we add a 4th spatial component whose value is the curvature of v . This allows vertices with non-extreme curvature to group together easily while the vertices with extreme curvature will be separated. Thus the largest cluster produced by the gravitational clustering algorithm will contain all the vertices with non-extreme curvature. The remaining much smaller clusters will contain the vertices which have extreme curvature. Figure 6.8 shows the results of applying gravitational clustering to a single scan of the limestone/dolostone dataset. Note that in Figure 6.8c we can see empty spaces which correspond to areas of extreme curvature. These points are missing from the largest cluster and are thus considered to be rough vertices.

(a) Mesh with curvature visualized.

(b) Projection of the mesh onto the plane.

(c) Largest cluster found.

Figure 6.8: Application of gravitational clustering visualized.

Figure 6.9: Visualization of a quad-tree structure.

6.5.2 Quad-trees

Quad-trees are a well studied space-subdivision data structure where each internal node of the tree has exactly 4 children [25]. Each of the 4 children divide their parent into 4 quadrants (hence the name quad-tree). See Figure 6.9 for a visualization. The quadrants are canonically named North-East (NE), North-West (NW), South-East (SE) and South-West (SW). The north/south direction indicates that the quadrant is on top or on the bottom and the east/west direction indicates if the quadrant is to the left or to the right. The type of data which the quad-tree stores can be just the 2D plane itself or geometric objects which lie on the 2D plane. The quad-tree can be built in $O(n \log n)$ time and queries can be answered in $O(\log n)$ time [25] [17]. There are many variants of the quad-tree and they mainly differ in the subdivision stopping condition and/or the type of data they store. An example of two different stopping conditions are: continuously subdivide the space until each node reaches a certain area and stopping when there are no points left. Appendix D details the exact quad-tree algorithms used in this chapter.

6.5.3 Results

Figure 6.10 shows the quad-tree roughness ratio for a single scan of part of the limestone/dolostone rock wall. This scan was chosen as it exhibits the massive limestone strata with well-defined bedding planes on the left and more friable dolostone on the right. The limestone beds are smooth while the friable dolostone would be considered geologically rougher. Note that we can see the bedding lines as well as the limestone bedding layers as smooth on the left while the crumbled rock as rough on the right. Since a single scan covers an area of 1.6 m^2 , the area covered by each quad node is roughly 10 cm^2 , 5 cm^2 , and 2.5 cm^2 for quad-tree depths 4, 5 and 6. Even with a low quad-tree depth (4) the contact between the two types of rock is clearly delineated and the dolostone is mapped as rougher than the limestone. Since we are using the ratio of extreme curvature points to total number of points within the quad-nodes, increasing the depth decreases the size of quad-nodes and increases the amount of detail that can be seen.

Figure 6.11 shows an application of the quad-tree roughness ratio to the entire limestone/dolostone rock wall. Note that the measure detects that it is rougher on the right side (dolostone) compared to the left side (limestone). In Figure 6.11a a single extreme roughness ratio (seen in red) skews the coloring for the rest of the quad-nodes blending them into the background. To enhance the perception of low roughness ratios we scale the roughness ratios of all quad-nodes using a cube root function. The color key remains unchanged. The result of this scaling can be seen in Figure 6.11b. The dolostone on the right is enhanced while the limestone on the left remains mostly unchanged as there were very few quad-nodes that had any extreme curvature values.

Lastly we revisit the smooth metal plate that was used in Section 3.2 to determine the instrument error as a validation of this method for measuring roughness. From Figure 6.12 we can see that the plate is considered to be mostly smooth overall.

Figure 6.10: An example of quad-tree roughness ratio applied to one scan of the limestone/dolostone dataset. Figure 6.10a is a real world photograph of the surface which the scan represents. Figures 6.10b, 6.10c and 6.10d are visualization of the roughness ratio for each of the quad-nodes at depths 4, 5 and 6. As indicated in Figure 6.10e purple regions have a very low ratio of extreme curvature vertices (i.e: it is not rough) while red regions have a very high ratio of extreme curvature vertices (i.e: it is rough). Note that we can see bedding lines as well as the limestone bedding layers as smooth on the left while the crumbled rock as rough on the right.

(a) No Scale

(b) Cubic Root Scale

Figure 6.11: The quad-tree roughness ratio applied to the limestone/dolostone dataset. Figure 6.11a shows the original coloring of the ratios of each quad-node. Figure 6.11b show the ratios scaled by the cube root function so that the ratios are colored on a non-linear scale. The color key is the same as in Figure 6.10.

Figure 6.12: Figure 6.12a shows the mesh model of the plate. Figure 6.12c shows the curvature mapped onto the plate. Note that while the deviation of the curvature values within the mesh is not great, the way that the coloring is computed (see Appendix E) allows us to see the variations between the different curvature values. Figures 6.12c to 6.12f show the results of applying the quad-tree roughness ratio. Since there is very little extreme curvature values the plate is almost entirely purple indicating that it is flat.

6.6 Roughness sampling

Using curvature as a basis again we look at a method for sampling the “rougher” points of a given point cloud. The main concept we use to determine whether or not a vertex v in a mesh M is rough enough to be sampled is to compare the distances of v to its neighbours and the curvature of v . We will denote the curvature of v as $\kappa(v)$. Intuitively, when $\kappa(v)$ is flat (i.e: close to zero) then the distance from v to its farther neighbours would larger in comparison. When $\kappa(v)$ is very curved then the distance from v to it’s neighbours should not exceed the value of $\kappa(v)$. The Euclidean distances between points on a flat line and a bent line do not change much but the curvature of the line will. Thus while the neighbours of v remain at a similar distance to v , $\kappa(v)$ will not. We use this intuition to capture roughness by defining two real numbers ε and δ such that $0 < \varepsilon < 1$ and $0 < \delta < 1$. Our main contribution is to use these two numbers to define a voting system which looks at the neighbourhood of a vertex v determines whether or not v is rough.

The number ε determines whether or not a given neighbouring vertex of v will “agree” that v is rough. More precisely a vertex u “agrees” that v is rough if the Euclidean distance from u to v is less than or equal to $\varepsilon|\kappa(v)|$. The number δ represents the portion of neighbours surrounding v who are required to “agree” that v is rough before v is labelled a rough vertex. For example if $\delta = 0.5$ then a vertex v requires at least 50% of its neighbours to be agree that v is rough before it can be labelled as rough. Using the illustration in Figure 6.13 with $\delta = 0.5$ we can see that Figure 6.13a only contains 2 out of 6 neighbours that is close enough to it. Thus the vertex being sampled in Figure 6.13a is not labelled a rough vertex. Figure 6.13b however has 8 out of 10 neighbours that are close enough to it thus the vertex being sampled in Figure 6.13b is labelled a rough vertex. A rough sample is then the collection of all vertices which are labelled as rough. Figure 6.13 illustrates this intuition for a 2D curve and Algorithm 4 is the formal algorithm.

Figure 6.13: The vertex v that is being sampled is the blue dot and all the other dots are the neighbours of v . The magenta dots have an Euclidean distance that is less than or equal to $\varepsilon|\kappa(v)|$ while the grey dots do not.

Algorithm 4 Roughness sampling

Require: Mesh $M = (V, E)$, $0 < \varepsilon < 1$, $0 < \delta < 1$, local window radius r

Ensure: $P \subset V$ representing the rougher vertices

- 1: Compute the curvature of M using local window size W_r
 - 2: $P \leftarrow \emptyset$
 - 3: **for** $v \in V$ **do**
 - 4: $S \leftarrow W_r(v)$
 - 5: $Q \leftarrow \emptyset$
 - 6: **for** $s \in S$ **do**
 - 7: **if** $\text{dist}(p, s) \leq \varepsilon|\kappa(v)|$ **then**
 - 8: $Q \leftarrow Q \cup s$
 - 9: **end if**
 - 10: **end for**
 - 11: **if** $|Q| > \delta|S|$ **then**
 - 12: $P \leftarrow P \cup v$
 - 13: **end if**
 - 14: **end for**
-

The choice of parameters ε and δ in Algorithm 4 affects the size of the resulting sample size. Reducing ε means that vertices in the neighbourhood of v which are too far away (i.e: greater than $\varepsilon|\kappa(v)|$) will not agree that v is rough. This results in a smaller roughness sample. Increasing ε has the opposite effect. Reducing δ means that we require a smaller portion of the neighbours of v to agree that v is rough before

labelling v as rough which increases the size of the rough sample. Increasing δ has the opposite effect.

Figure 6.14 shows results of selecting a rough sample using Algorithm 4 from a single scan of the limestone/dolostone rock wall. This is the same scan which was chosen for visualizing the quad-tree roughness ratio (see Figure 6.10). We chose the same scan so that we can see the behaviour how vertices are sampled as well as providing a comparison to the quad-tree roughness ratio method outlined in Section 6.5. The red regions indicate the vertices which have been sampled as rough by Algorithm 4 while the blue regions indicates smooth areas which were not sampled. Note that the flat planar regions of the mesh are avoided. The effects of varying ε and δ can be seen in Figure 6.14 can be seen as well. Note that both follow a similar progression in terms of the resulting sample size as well as the actual areas that are mapped. This means that one can fix one of the parameters and varying the other to compute the roughness sample. The only difference is that with ε we increase to have the sample cover a larger portion of the mesh while with δ we decrease. It should also be noted that compared to the quad-tree roughness ratio method, this method did not label as many areas on the right as rough. The major fractures are first outlined. As the sample covers a greater area it appears to expand from the fractures of the rocks. This is in contrast to the quad-tree roughness ratio where a lower depth corresponds to a larger area covered by each quad-node which increases the averaging effect.

(a) $\varepsilon = 0.30, \delta = 0.50$ (b) $\varepsilon = 0.25, \delta = 0.50$ (c) $\varepsilon = 0.35, \delta = 0.50$ (d) $\varepsilon = 0.30, \delta = 0.40$ (e) $\varepsilon = 0.30, \delta = 0.60$

Figure 6.14: (a) is the base image before varying ε and δ . (b) and (c) show the effects of varying ε while (d) and (e) show the effects of varying δ .

6.7 Summary and Discussion

In this chapter we have presented three different roughness measures for point clouds. Figure 6.15 describes the work flow for measuring roughness through a flow chart. Each of the roughness measures utilizes the curvature of the surface that the point cloud represents. The first measure, curvature difference (Section 6.4), is due to Lavoué [48]. The second measure, quad-tree roughness ratio (Section 6.5), and the third measure, roughness sampling (Section 6.6) was created by the author.

The quad-tree roughness ratio is the most general measure of the three roughness measures. This is because the only requirement for the quad-tree roughness ratio is the separation of the point set into one set which represents rough points and another set which represents non-rough points. The roughness value for any region of space (as determined by the depth of the quad-tree) is then the ratio of rough points to total points contained in the space. The method for separation used in the quad-tree roughness ratio as described in Section 6.5 is through the clustering of curvature. This is not the only way to separate the point set into a rough set and a non-rough set. If we first use the roughness sampling method we can skip the separation and clustering step. Thus the output of the roughness sampling method can be used as the input for the roughness ratio (visualized as a dashed grey arrow in Figure 6.15). The same concept applies to using the curvature difference. As for the quality of the roughness measures, the author believes the quad-tree roughness ratio using curvature clustering performs the best. The rationale is that the quad-tree roughness ratio method is able to clearly determine the bedding planes of the limestone layers as well as map the more crumbled areas which are considered to be geologically rougher (see Figures 6.10 and 6.11). The use of a space partitioning data structure also allows for a range of roughness values for a given curvature mapping. This range in turn allows for various levels of detail. The curvature difference produces a roughness mapping which is dependant on a smoothed mesh. Thus when given a mesh which is already

smoothed, the algorithm will compute the curvature difference between the smooth mesh and an even smoother mesh which will produce a roughness mapping that may not be representative of the original structure. The roughness mapping is defined for each point rather than a portion of space and thus visually the mapping appears irregular increasing the difficulty of interpretation. The roughness sampling method requires additional experimentation of parameters to produce a good sampling. The method also does not sample all the points which make up the bedding planes (see Figure 6.14). Potential future work in the area of roughness measures is discussed in Chapter 8.

Figure 6.15: Flowchart describing the work-flow of roughness metrics.

Chapter 7

Feature Selection

This chapter looks at selecting a specific geological structure known as shatter cones through a mesh of the point cloud. As discussed in Section 3.4.3, shatter cones are metamorphic rocks which are formed through shock metamorphism and they are rare structures that only occur in the vicinity of impact craters where rocks experienced extreme and rapid pressure [28] [65]. The direction of the shock wave can be determined from the orientation of the cone which, assuming no subsequent large-scale deformation, points toward the centre of the crater. Visually, well defined shatter cones are easy to identify with the human eye as they are cone-like in shape, with striations fanning out from a common apex. Figure 7.1 is an up close photograph of a very well defined shatter cone.

7.1 Normal orientation to produce a polar plot

An initial approach was made by observing that any cone-like structure within a mesh has normals which form the pattern as illustrated in Figure 7.2. Thus in this approach we compute the outer normal orientation mapping as described in Section 5.2 and we plot the orientations about a circle. Recall that the outer normal orientation mapping labels each triangle within a mesh M , with two angles α and β . These two angles

Figure 7.1: A photograph of a very well defined shatter cone broken off from a larger rock. Note the clear cone shape as well as the striations on the cone where the shock force travelled through the rock. Image taken from [66].

are used to form a polar coordinate $p = (\theta, r)$ where α is mapped to the angular coordinate θ and β is mapped to the radial coordinate of r . Now every triangle t in the mesh M is mapped to a polar coordinate p which represents the orientation of t . Using these coordinates we can plot the orientations of all the triangles onto a circle as seen in Figure 7.3.

Chapter 5, Figure 5.6 displays a visualization of the outer normal orientation mapping and the results of converting the outer normal orientation of the triangles into the polar coordinate plot about a circle is shown in Figure 7.3. While the polar plot does display a large arc-shared cluster which represents the surface of the shatter cones, it is not able to separate it from the background plane nor is it able to detect separate individual cones within the same scene. This is due to the fact that all the normals are converted into a polar coordinate and so the normals which represent just

the cone is lost within all the other normals which also point in the same direction. There is potential for this method to be a local measure as it will be able to describe a sub-portion of a mesh and determine whether or not it is a curved surface based on the shape of the clusters formed. This potential is left as future growth.

Figure 7.2: Illustration of how the normals draw an arc shape the surface of a cone. The arc shape of the cone on the right is transposed in the circle plot to the left as the magenta dots. Each dot represents one of the normals.

Figure 7.3: Highlighting different parts of the shatter cone image (left) by selecting the corresponding cluster (right). The black box represents the selection area. The purple regions on the left images represent the triangles selected by the black box. The shatter cone corresponds to the large arc-shaped region near the center of the circle as see in (c) and (d). The α for that region ranges from 200° to 360° while the β ranges from 20° to 45° .

7.2 Shatter cone markers

In this section we detail an approach, which is the main original contribution of this chapter, to selecting shatter cones which address some of the issues found in Section 7.1. In Figure 7.4 we can see red streaks which are exactly along the edges of a shatter cone.

(a) Real photograph

(b) Color coded with β component of the normal orientation (Section 5.2).

Figure 7.4: Note that red colored streaks represents a side of a single shatter cone.

Using this observation we note that this is due to the shape of the cone and reason that if there exists a “streak” on one side of the cone, there must exist a streak on the other side of the cone. The orientation of the triangles which form these two types of streaks were determined by visual inspection. We will denote the streaks as a *marker* and markers come in two types: **end** and **start** markers (see Figure 7.5).

(a) **end** markers (b) **start** markers (c) **end** (gray) and **start** (purple) markers superimposed

Figure 7.5: A visualization of both types of markers. The **end** markers have triangles whose orientation spans over the interval $[120^\circ, 154^\circ]$ while the **start** markers span over the interval $[0^\circ, 60^\circ]$.

Once the orientation range was determined for the **end** and **start** markers, we then select all the triangles within both ranges and place them into two sets: E for

end markers and S for **start** markers. In Figure 7.5a and 7.5b we can see that within the two sets of markers are separate disjoint components. To find these disjoint sets for the two marker types we use a minimum spanning tree (MST). Given a set of vertices V , the MST over V connects each of the vertices using an edge such that the total number of edges is $|V| - 1$ and the total length of the edges is minimized. Thus by building a MST over the marker sets we connect each component in their respective set. Once we have a MST we can use it to find the separate components by dropping all longest edges and running a graph searching algorithm. More details of the MST and finding connected components can be found in Cormen's book [16]. How the long edges were chosen is described in Section 7.2.1. Thus for the marker sets E and S we can find two new sets P and Q where P would contain disjoint sets of triangles of the **end** type and Q would contain disjoint sets of triangles of the **start** type. What remains is to connect the each of the disjoint sets from P to the appropriate disjoint set in Q . Figure 7.6 illustrates this concept visually. The question remains as to how exactly does one select the appropriate disjoint sets from $p \in P$ and $q \in Q$ such that together they enclose the shatter cone. The approach used to connect two marker types is described in Section 7.2.1.

Figure 7.6: Red and green streaks represent **start** and **end** markers, respectively. When the proper markers are chosen correctly we can select the shape of the shatter cone.

7.2.1 Quad-trees revisited

The motivation for building a quad-tree over the triangles of a mesh M came from the need to define the length of the edges we wish to drop in order to cluster the markers. Since the marker sets are initially one large set of triangles within a range of orientation, each of the triangles could be considered a node and one can build a graph, G , where the triangles are the vertices. The number of nodes in G is proportional to the number of triangles within the mesh. This number can potentially become very large. By building quad-tree over the triangles first, and treating each quad-node as a node in G , the size of G is now proportional to number of quad-nodes at a given level of the quad-tree. This also allows us to place a bound on the weight of the edges as the distance from the centre of one quad-node to another at the same level is constant. Since a streak as in Figure 7.5 is continuous then the quad-nodes that contain those triangles should be right beside each other as in Figure 7.7. Thus any edge that is greater than $\varepsilon \times (\max(w, h)/2^d)$ where d is the current level of quad-tree, w and h are the width and height of a quad-node at level d and $0 < \varepsilon \leq 2$ is an edge that should be dropped. Figure 7.7 provides an illustration.

Figure 7.7: A visual of a quad-tree grid overlaid on top of the **end** (left) and **start** (right) markers. The nodes and edges forming a MST which represent the markers are seen in blue (**end**) and orange (**start**). A single quad-strip is highlighted in pink.

Now that we know how to find individual markers through the MST built from the quad-nodes representing subsets of the markers, we introduce the quad-strip as a way to select the quad-nodes that are part of a single shatter cone.

Quad-strips We define a quad-strip, s , as a single horizontal row of quad-nodes at some depth d of the quad-tree. A quad-strip is ordered in the sense that the strips at the top come before the strips at the bottom. Within each strip, the quad-nodes are ordered from left to right. Given any non-leaf quad-node we associate the top two quadrants of the quad-node with 0 and the bottom two quadrants with a 1. This allows one to recursively label quad-nodes with their proper strip id such that we keep the ordering of $s_{top} > s_{bottom}$. See Figure 7.8 for an example. Algorithm 9 (found in appendix D) is the procedure for finding quad-strips at depth i of a quad-tree.

Figure 7.8: An example quad-tree with depth 2. At depth 1 we have top quadrants labelled 0 and the bottom labelled 1 which can be seen as the left most 0 and 1. Since the definition is recursive the same applies at level 2. Note that the strip id (top-right number) of each of the quad-nodes is labelled.

Since each quad-node is associated with their own strip id, given a quad-node q we can find the list of quad-nodes that are also on the same strip as q . Since markers are composed of quad-nodes, given any two markers we can determine which of their quad-nodes are in the same strip. Now to determine which of the quad-nodes in

a quad-tree belong to which shatter-cone, we can use the **end** marker set and loop through the quad-nodes that make up each of the markers. The quad-node uses its strip id to find the quad-strip that it belongs to and moves left until it reaches another quad-node that is part of the **start** marker set. If we do not reach such a marker then no quad-nodes are labelled. If we do, we label all the quad-nodes in-between associating them with either the **end** or the **start**. It does not matter which one we choose as long as we are consistent. This method is visualized in Figure 7.6.

7.2.2 Results

Here we detail the results of the finding shatter cones through matching their **end** and **start** markers. Figure 7.10 shows the results of finding the connected components of both markers from Figure 7.5c. A different color corresponds to a different marker. The coloring is for visualization purposes to show that they are indeed disjoint. Note that the markers do not all have the same size. In fact the light blue marker (circled in Figure 7.10) which is closest to the brown marker is fairly small. It would appear that one side of the shatter cone is more defined than the other for certain cones. This might be due to the local erosion effects.

Using the markers in shown in Figure 7.10, we use the quad-strip structure to determine the shatter cones using the method presented in Figure 7.6. Figure 7.11 shows the result of applying the quad-strip method. As in Figure 7.10 a different color corresponds to a different shatter cone. Figure 7.12 shows the results of applying the shatter cone marker method to the Gibson road dataset. Note that the main central shatter cone is detected while the more broken shatter cones around the the main cone are also detected but are considered to be different cones due to fracturing as well as markers which are not well-defined.

While the method described in this section can detect shatter cones, not all are detected. The method requires the use of the distinct cone markers and some of the

cones do not exhibit these markers due to weathering. The orientation of the markers are also experimentally determined thus it would be useful to find a way to determine the markers automatically as shatter cones are not always vertical in direction. The way they are oriented may be due to the direction of force of impact but they can be moved over time through natural geological processes (see Figure 7.9). An alternate approach to matching the markers would be to fit a line to the triangles for each of the markers and using the intersection of lines as a basis for matching. The main problem with this approach is still that the markers may not be very distinct as seen in Figure 7.10 making it difficult to fit a line that matches the actual side of the cone.

Figure 7.9: Example of how the orientation of a shatter cone as defined on the left can move due to natural forces to the shatter cone oriented on the right.

Figure 7.10: The red circle highlights a poorly selected marker. The cone associated with this marker is the brown marker to the right of the red circle. Note that the brown marker is much longer in size and captures the side of the shatter cone while the poorly selected marker is short and does not capture the entire side of the shatter cone.

Figure 7.11: The red circle highlights the marker (same location as 7.10) which stops the brown markers from advancing at that location. Due to its smaller size it is not able to stop the brown markers properly resulting in a bad detection. Note that the two overlapping cones which have distinct markers are detected properly.

(a) Photograph

(b) Shatter cones detected.

Figure 7.12: (a) Photograph of shatter cones along Gibson road (b) The color overlays represent the areas which are was detected as a shatter cone. Each color corresponds to a different detected cone.

Chapter 8

Conclusion and Open Problems

We have described a number of techniques that can be used to determine some of the parameters of rock mass characterization models. We have described two different methods for computing and visually enhancing the planar orientations of rock masses as well as using curvature as a basis for measuring roughness. Since roughness for point clouds in a geological setting has not been explored, we have given a number of potential measures that can be used. We have also described and examined a method for detecting shatter cones structures embedded in a plane. The effectiveness of these methods will need to be further validated through a series of field trials and empirical studies using a larger variety of datasets.

It should be noted that the quad-tree roughness measure developed in Section 6.5 is versatile can use any other measure as a basis. For example, instead of computing the curvature of the mesh and then clustering the vertices of the mesh into extreme and non-extreme curvature, one can apply the curvature difference (see Section 6.4) and apply the clustering algorithm to separate the vertices into a non-extreme curvature difference set and an extreme curvature difference set. Now the quad-tree roughness ratio will be determined using the curvature difference rather than just curvature. The roughness sampling algorithm (see Section 6.6) can also be used with

the quad-tree roughness ratio measure. No clustering would be required as the sample of rough vertices is already provided. One can also use any of the roughness measures from remote sensing and material sciences as described in Appendix A. Statistical measures found in Section A.1 can be used to first obtain a global measure of the surface and then find the ratios of the local measure within a quad-node with respect to the global. The electromagnetic measures for computing scattering coefficients in Section A.2 could be used to assign each quad-node a scattering coefficient. Ray-tracing can be used to simulate light rays to determine the scattering coefficients. The quad-nodes that cover a rougher area would scatter more light than smoother areas. Lastly, instead of using the quad-tree one can also use an octree as the basis for the subdivision of space. Experimentation with these variants could be a possible direction for future work.

Since laser imaging for geological structures is an emerging research area, there are several directions one can take that are beyond the geological topics covered in this thesis. One can image rocks at short range to determine the size and potentially the shape of crystal grains on the order of 1 mm in diameter. While we have explored the roughness at a larger scale, the size of crystal grains is related to the roughness of the surface at a local scale. It would be interesting to see if laser imaging could automatically categorize different types of minerals or if embedded clasts could be distinguished from their matrix. It would also be interesting to examine a series of laser images of a scene over a period of time as well as laser imaging for rover-based autonomous planetary exploration.

Appendix A

Roughness Measures in Other Fields

A.1 Statistical

In this section we detail some common measures of roughness in material sciences [37]. These measures usually deal with a *patch* of a 2D plane where each vertex within the plane is associated with a elevation value e . Thus a patch can be thought of as a collection of 2D points within the 2D plane where each point p_i in the patch has an elevation value e_i . A point within a patch, p_a , is considered to be higher than another point p_b in the patch if $e_a > e_b$. Since many of these measures are used in material sciences it is worth briefly discussing how this field handles roughness and why there is interest in it. In material sciences, and especially in manufacturing, the roughness of a surface is measured using a surface profiling instrument which can be seen in Figure A.1. This device has a pen or styles which measures the height of the surface at the point of the stylus. The instrument is moved across the surface and measurements are recorded. Multiple across surface measurements results in the ability to select a patch of the surface. Generally the material sciences are interested in roughness for

the manufacturing process where one may want to have the surface being very smooth or at a certain roughness so that the material has a certain coefficient of friction. It should be noted that the stylus profiler is not the only type of instrument but it is one of the more common instruments used. Regardless of the instrument used the measures found in this section are the most commonly used throughout all fields.

Figure A.1: Two different views of a surface profiling instrument. Figure A.1a shows a 2D view where the rectangular gray shape represents an object which the brown stylus moves over as indicated by the arrows. Figure A.1b shows a 3D view where the line traced on the surface is indicated in red and the blue plane which intersects the object is shown. This plane represents a subsection which one can apply following roughness measures.

Next we list several different ways of measuring roughness once we have the appropriate data using any instrument or measuring techniques. These measures of roughness can be found throughout many fields in science and more details can be found in the Surface Metrology Guide [37] and in the Measurement Good Practice Guide [62].

Average Surface Roughness :

Approximates surface roughness by computing the mean absolute elevation values from a given patch of vertices using the following formula:

$$R_a = \frac{1}{N} \sum_{i=1}^N |h_i|$$

R_a is thus the roughness value of the patch and N is the number of points within the patch. The value h_i is the difference between e_i and e_{avg} where e_{avg} is the

average elevation value of the patch and e_i is the elevation of the i th vertex. This measure of roughness is by far the most commonly used throughout literature and there are often variations of this formula to suite a certain kind of data set [37] [62] [30] [83].

Root-Mean-Square Deviation of the Surface :

A variant of the Average Surface Roughness method. This is often used as the equivalent of the sample standard deviation used in statistics. The roughness is given by the following formula:

$$R_q = \sqrt{\frac{1}{N} \sum_{i=1}^N h_i^2}$$

Tenpoint-Height :

This method defines the roughness of a patch to be the average value of the absolute heights of the five highest and lowest points within the patch (hence the name “Tenpoint”). The formula for this measure of roughness is given below:

$$TH = \frac{\sum_{H=1}^5 e_H + \sum_{L=1}^5 e_L}{5}$$

Here TH is the roughness value for the patch and e_H represents the elevation values from the five highest points while e_L represents the elevation values from the five lowest points.

Density of Summits :

Uses the number of summits based on the eight neighbour summit definition. Here the patch must be defined to have eight neighbours. Such a patch is usually extracted from a DEM model of terrain and it makes this measure less practical

as not all models will have the eight neighbour connectivity.

$$S_{ds} = \frac{S}{a}$$

Here S_{ds} is the roughness value for the patch and S is the number summits while a is the area of the patch.

Skewness of Topography Height Distribution :

Skewness is a measurement of the symmetry of the surface deviations with respect to the mean reference plane. The equation for this measurement is given below. The S_{sk} value describes the shape of the topography height distribution. For example, a surface with a random (or Gaussian) height distribution and a symmetric topography has a skewness of zero. Thus S_{sk} represents the degree of bias in either the upward or downward direction of an amplitude distribution curve. A symmetrical profile gives an amplitude distribution curve that is symmetrical about the center of the data and an unsymmetrical profile results in a skewed curve. The direction of this skewed curve depends on whether or not a large enough portion of the data is above the mean plane (negative skew) or below the mean plane (positive skew). The S_{sk} value is useful for when two surfaces have the same R_a value [62].

$$S_{sk} = \frac{1}{NR_q^3} \sum_{i=1}^N h_i^3$$

Kurtosis of Topography Height Distribution :

Measures the “sharpness” or “peaky-ness” of the surface height distribution. The S_{ku} characterizes the spread of the height distribution. A centrally distributed surface has a kurtosis value that is larger than three while a value that is less than three would have a well spread distribution. A surface with a gaus-

sian height distribution has a kurtosis value of three. Thus a spikey surface will have a high kurtosis value and a bumpy surface will have a low kurtosis value. The S_{ku} is useful for predicting physical component performance with respect to wear and lubrication retention [62]. It should be noted that S_{ku} cannot tell the difference between valleys or peaks.

$$S_{ku} = \frac{1}{NR_q^4} \sum_{i=1}^N h_i^4$$

A.2 Electromagnetic

This section details how the interactions between electromagnetic (EM) waves and a surface can and has been used to determine the roughness of a surface. The first property we will look at is the reflection of EM waves and the second property is the refraction of EM waves. The former can be used on solid and liquid surfaces and the latter is mainly used with liquid surfaces as a refraction index is needed. Solids which are transparent (i.e: glass or plastic) can also make use of the refraction method.

Reflection Methods

The reflection methods for measuring the roughness of a surface through EM waves primarily deal with the scattering of the wave. The reflection of an EM wave and a given surface produces a coherent part in the specular direction and a scattering part in the other directions (see Figure A.2 for an example). Exact solutions for the EM field can be difficult to find (as well as being computationally expensive to compute) so approximations are often used to model the scattering produced by rough surfaces. Extensive studies have been done to develop models which are valid over a range of incident scattering angles and applications [4] [14] [38] [61] [80] [83]. Validity for reflection models refers to the ability of the model in question to reproduce the

scattering realistically. The validity of the models are then ranked based on factors such as the angle in which the EM wave hits the surface and so on. Generally the surface roughness for these methods is characterized by the R_q value (see Section A.1 for definition) of the height distribution of the input data. The R_q value for reflection methods is related to wavelength of the measuring EM wave and the size of the surface that is being measured. Thus these methods do not produce a unique roughness value for each individual method but rather are used to extract the backscattering data from reflected EM waves and from the backscattering data we determine the R_q values. Despite not producing a unique roughness value these methods are important to study as they are used across a wide range of studies and is related closely to the measuring instruments.

Figure A.2: Example of an EM wave hitting a surface and scattering. The coherent part is represented by the red arrow moving point away from the surface and the purple arrows represent the scattering of the wave in all the other directions.

Scattering Models Overview

Early rough surface scattering models include the physical optics (PO) approximation and the small perturbation method (SPM). PO is valid within the range of large electromagnetic frequencies and SPM is valid within the range of low frequencies [54]. These models are still in use because of their robustness and simplicity of use [54] [80]. Recent scattering models are adapted to their application for various kinds of rough surfaces such as the ocean surface [14] [83] to soil surfaces [80]. These new scattering models are based on perturbation expansions and they are typically named

after the parameter which the field (electromagnetic) quantities are expanded. Some examples of these models include the small slope approximation (SSA) [10] [6], phase perturbation [11] [81], Kirchhoff approximations [4] [76] [78] and the unified perturbation expansion [63]. The detailed analysis of scattering models is out of the scope of this thesis but we briefly will cover the basis of these models, PO approximation, in order to gain an understanding of how scattering models are used to determine the roughness of a surface.

Physical Optics Approximation

The PO model approximation requires a thorough understanding of the wave theory found in physics thus the precise definition will not be provided as it is out of the scope of this thesis. The focus for this thesis is the measure of roughness based off the PO model. A brief description on what it is will be provided instead and precise definitions can be found in the papers by Floyd W Millet and Karl F Warnick [54] and by the lecture notes of Jun-ichi Takada [36].

The approximation is an intermediate method between geometric optics, which ignores wave effects such as diffraction and interference, and the fully developed and precise wave theory of electromagnetism [36]. This approximation method estimates the EM field on a surface and then integrates the field over the surface to compute the scattered field. Thus the PO approximation is a stochastic model as it uses the statistical properties of the surface itself [54]. Another way to view the PO approximation is by viewing the surface as a series of tangent planes which approximate the actual surface. Indeed the PO approximation is closely related to the Kirchhoff approximation which is also commonly known as the tangent plane approximation. The difference between the Kirchhoff approximation and PO approximation is that the PO approximation also accounts for shadowing [4].

Applications

The surface of the ocean consists of waves and which is produced when there is wind. These waves are the components in determining how “rough” the ocean surface is and most techniques focus on the slope of these waves. Figure A.3 shows a real world photo showing how waves on the ocean surface can be related to roughness. The faster the winds, the rougher the ocean surface. The majority of methods used to determine ocean roughness relies on the optical properties of the measuring instruments modifying the methods found in Section A.2 to accommodate for the specific instrument.

Figure A.3: An aerial photo view of the ocean surface along with the waves which define the surface.

There has been use of space LiDAR (via CALIPSO satellite) to measure ocean surface roughness and wind speed [83]. The principles are the same in the sense that one sends a signal to the ocean surface to measure the roughness through the various changes in the slope. The method found in [83] looks for the back scatter of the reflected LiDAR measurements. There has also been use of reflected GPS signals as shown by Clarizia et al [14]. Figure A.4 sums up the physics used in this method in an illustration.

Reflection based methods have also been used in a wide variety of land applications involving the soil with the roughness of the soil as one of the characteristics of soil that EM wave reflections can determine [80]. Indeed the roughness of soil

Figure A.4: Visual of how higher wind speeds generate greater backscatter.

or oceanic surfaces is but one of many applications for reflection based methods as GlobeSAR, a project which uses airborne synthetic aperture radar (SAR) imagery [27] [26]. Reflection methods are not limited to natural sciences and can also be used in manufacturing processes. Instead of the profiler instrument found in Section A.1 a laser system can be used and the principles of the reflection based methods can still apply.

A.2.1 Refraction Methods

This section will deal with the refraction based methods for determining surface roughness. Much like the reflection methods found in Section A.2, the actual roughness values is characterized by the R_q value of the surface. Thus, similar to the reflection methods found in Section A.2, the refraction methods are measurement methods which measure the surface using EM waves to determine the R_q value. The only difference is that we are using a the refraction property of the EM wave. For the rest of this section we will use the ocean surface as the primary example in detailing the refraction method though any surface that has a refractive index can be used as well.

The ocean waves are measured by specially designed instruments which measure the refraction of laser beams from the water to air barrier as there is a difference in the refraction of the laser beam. This difference of refraction from water to air follows

Snell's law of wave propagation. Thus we can monitor the changing position of the refracted beam and determine the slope of the surface at the position that the laser beam penetrates the water to air barrier [35] [60]. See Figure A.5 for an example.

Figure A.5: An example of an instrument which measures the slopes of the waves using a laser. Note that the refraction is exaggerated for illustration purposes.

Since the laser beams refraction obeys Snell's Law we have the follow relation between the angles of incidence and refraction for Figure A.5:

$$\begin{aligned} n_A \sin \theta_A &= n_W \sin \theta_W \\ \theta_A &= \sin^{-1} \left(\frac{n_W}{n_A} \sin \theta_W \right) \end{aligned}$$

Where n is the refractive index of the substance and θ is the angle between the beam path and the surface normal. The subscripts W and A refer to the beam travelling through underwater and the beam travelling through air. Since we are dealing with water and air we can substitute $n_W = \frac{4}{3}$ and $n_A = 1$ as these are their refraction indices [31]. This gives us $\theta_A = \sin^{-1}(1.333 \sin \theta_W)$. Now in Figure A.5 we also have an additional angle labelled just θ and it is equal to $\theta_A - \theta_W$. We substitute $\theta_A = \sin^{-1}(1.333 \sin \theta_W)$ and we obtain $\theta = \sin^{-1}(1.333 \sin \theta_W) - \theta_W$ allowing us to express the angle θ using just θ_W . The slope of the ocean wave, S , is determined by $\tan \theta_W$. With repeated measurements over an area we can determine the roughness of

the ocean by observing the changing slopes per area. Like in the material sciences the series of measurements is run through an Average or Root-Mean-Square as defined in Section A.1.

The instrument displayed in Figure A.5 is but one way to measure the surface slope of the ocean waves. One could also position the laser over the ocean surface and measure the refraction but then one must also consider the ambient light as well [35] [60]. Regardless of the instrument the basic principles are the same thus making this method attractive as it is simple, easy to implement and to collect measurements. When covering a large area one must also consider the amount of time that the instrument takes to scan an area. If the instrument moves much quicker than the motion of the waves then the surface is effectively frozen in place. Using simple scanning patterns such as a line or a rectangular grid one can also apply data analysis procedures to the data such as the Fourier transformation.

Appendix B

Formal Curvature Definition

In this appendix we give a formal definition of curvature using concepts from differential geometry. Since the field of differential geometry is very large and out of the scope of this thesis, we will provide brief derivations of the necessary concepts and much more detail can be found within the notes on differential geometry by Shirin [69] and the book Generalized curvatures by Jean-Marie Morvan [57]. Before we even begin to talk about curvature we must formally introduce the concept of a smooth surface.

B.1 First and Second Fundamental Forms

Definition 1. *A smooth surface in $\mathbb{R}^3$ is a subset $X \subset \mathbb{R}^3$ such that each point has a neighborhood $U \subset X$ and a map $S : V \rightarrow \mathbb{R}^3$ from an open set $V \subset \mathbb{R}^2$ such that:*

- *$S(u, v) = (x(u, v), y(u, v), z(u, v))$ has derivatives of all orders*
- *At all points the first partial derivatives $S_u = \frac{\partial S}{\partial u}$ and $S_v = \frac{\partial S}{\partial v}$ are linearly independent.*

Intuitively we can think of smooth surfaces as graphs of two variable functions. Now that we have a definition for smooth surfaces we can begin to define the two

concepts we will need from smooth surfaces to define curvatures. They are listed below along with a brief description describing the concepts as well as some useful notation:

First Fundamental Form Completely describes the metric properties of a surface.

It enables one to compute the lengths of curves on the surface and the areas of sub-regions on the surface itself.

Second Fundamental Form Relates the way a surface “fits” into $\mathbb{R}^3$. Together with the first fundamental form it defines the principle curvatures κ_{min} and κ_{max} .

Notation For some f we denote the following partial derivatives:

$$\begin{aligned} f_u &\leftrightarrow \frac{\partial f}{\partial u} \\ f_v &\leftrightarrow \frac{\partial f}{\partial v} \\ f_{uu} &\leftrightarrow \frac{\partial^2 f}{\partial u^2} \\ f_{uv} &\leftrightarrow \frac{\partial^2 f}{\partial v \partial u} \end{aligned}$$

We begin by defining the first fundamental form, denoted as $\mathbf{I}(U, V)$, which is the inner product on the tangent space of a surface in $\mathbb{R}^3$ (see Figure B.1 for an example). Given the parametric representation of a surface, $S(u, v)$, the inner product of the two tangent vectors $U = aS_u + bS_v$ and $V = cS_u + dS_v$ is:

$$\mathbf{I}(aS_u + bS_v, cS_u + dS_v) = \langle aS_u + bS_v, cS_u + dS_v \rangle \quad (\text{B.1})$$

$$= ac\langle S_u, S_u \rangle + (ad + bc)\langle S_u, S_v \rangle + bd\langle S_v, S_v \rangle \quad (\text{B.2})$$

$$= Eac + F(ad + bc) + Gbd \quad (\text{B.3})$$

where E , F and G are called the coefficients of $\mathbf{I}$ and note that $|U|^2$ can be expressed as $\mathbf{I}(U, U) = Ea^2 + Fab + Gb^2$. Using the coefficients the first fundamental form can also be expressed as a symmetric matrix:

$$\mathbf{I}(x, y) = x^T \begin{pmatrix} E & F \\ F & G \end{pmatrix} y \quad (\text{B.4})$$

Figure B.1: On the left we have the parameter space (u, v) and on the right we have a surface s which takes in the two parameters u and v . The red and blue lines show a single curve drawn in parameter space (B.1a) and how they behave in the surface space (B.1b). The partial derivatives s_u and s_v define vectors which point along the direction of their respective curves from the point $s(u, v)$ and together they form a tangent space. The cross product of s_u and s_v produces the normal vector of the tangent space.

Now we define the unit normal vector n with respect to the $S(u, v)$ as follows:

$$n = \frac{s_u \times s_v}{|s_u \times s_v|} \quad (\text{B.5})$$

where $\times$ denotes the cross product of two vectors. Suppose now we take our surface $S(u, v)$ and push or pull along its normal, we would have the following one-parameter

family of surfaces:

$$R(u, v, t) = S(u, v) - tn(u, v) \quad (\text{B.6})$$

where

$$R_u = S_u - tn_u, \quad R_v = S_v - tn_v \quad (\text{B.7})$$

Now we know that the first fundamental form gives us a metric for determining the length of a curve on a surface, the second fundamental form tells us how the first fundamental form of $R(u, v, t)$ changes as t changes. That is the second fundamental form tells us *how* the length of the curve changes as the surface moves along it's normal vector. The means we need to differentiate the first fundamental form with respect t . Now let us define the following:

$$E(t) = \langle R_u, R_u \rangle \quad F(t) = \langle R_u, R_v \rangle \quad G(t) = \langle R_v, R_v \rangle \quad (\text{B.8})$$

Note that $E(t)$, $F(t)$ and $G(t)$ are analogs to the first fundamental form but for the family of surfaces $R(u, v, t)$. The second fundamental form of $S(u, v) = R(u, v, 0)$ is then defined as:

$$\mathbf{II} = \frac{1}{2} \frac{\partial}{\partial t} (E(t)du^2 + 2F(t)dudv + G(t)dv^2)|_{t=0} \quad (\text{B.9})$$

Expanding the computing the expressions $E(t)$, $F(t)$ and $G(t)$ we get the following:

$$E(t) = \langle R_u, R_u \rangle \quad (\text{B.10})$$

$$= \langle S_u - tn_u, S_u - tn_u \rangle \quad (\text{B.11})$$

$$= \langle S_u, S_u \rangle - 2t\langle S_u, n_u \rangle + t^2\langle n_u, n_u \rangle \quad (\text{B.12})$$

$$\frac{\partial E}{\partial t}|_{t=0} = -2\langle S_u, n_u \rangle \quad (\text{B.13})$$

$$F(t) = \langle R_u, R_v \rangle \quad (\text{B.14})$$

$$= \langle S_u - tn_u, S_v - tn_v \rangle \quad (\text{B.15})$$

$$= \langle S_u, S_v \rangle - t(\langle S_u, n_v \rangle + \langle S_v, n_u \rangle) + t^2\langle n_u, n_v \rangle \quad (\text{B.16})$$

$$\frac{\partial F}{\partial t}|_{t=0} = -(\langle S_u, n_v \rangle + \langle S_v, n_u \rangle) \quad (\text{B.17})$$

$$G(t) = \langle R_v, R_v \rangle \quad (\text{B.18})$$

$$= \langle S_v - tn_v, S_v - tn_v \rangle \quad (\text{B.19})$$

$$= \langle S_v, S_v \rangle - 2t\langle S_v, n_v \rangle + t^2\langle n_v, n_v \rangle \quad (\text{B.20})$$

$$\frac{\partial G}{\partial t}|_{t=0} = -2\langle S_v, n_v \rangle \quad (\text{B.21})$$

The expressions in can be simplified by noting that S_u and S_v are always orthogonal to the normal vector n (Equation B.5) as seen in Figure B.1b. Thus this gives us the following simplifications for Equation B.13:

$$0 = (\langle S_u, n \rangle)_u \quad \text{recall } f_u \text{ is a partial derivative} \quad (\text{B.22})$$

$$0 = \langle S_{uu}, n \rangle + \langle S_u, n_u \rangle \quad \text{recall that } \frac{d}{dx}xy = x'y + xy' \quad (\text{B.23})$$

$$-\langle S_u, n_u \rangle = \langle S_{uu}, n \rangle \quad (\text{B.24})$$

$$-2\langle S_u, n_u \rangle = 2\langle S_{uu}, n \rangle \quad (\text{B.25})$$

$$\frac{\partial E}{\partial t}|_{t=0} = 2\langle S_{uu}, n \rangle \quad (\text{B.26})$$

Similar arguments can be made for B.17 and B.21 which leads us to the following

simplifications:

$$\frac{\partial E}{\partial t}|_{t=0} = -2\langle S_u, n_u \rangle = 2\langle S_{uu}, n \rangle \quad (\text{B.27})$$

$$\frac{\partial F}{\partial t}|_{t=0} = -(\langle S_u, n_v \rangle + \langle S_v, n_u \rangle) = 2\langle S_{uv}, n \rangle \quad (\text{B.28})$$

$$\frac{\partial G}{\partial t}|_{t=0} = -2\langle S_v, n_v \rangle = 2\langle S_{vv}, n \rangle \quad (\text{B.29})$$

Substituting Equations B.27, B.28 and B.29 back into Equation B.9 we get the following:

$$\mathbf{II} = \frac{1}{2} \frac{\partial}{\partial t} (E(t)du^2 + 2F(t)dudv + G(t)dv^2)|_{t=0} \quad (\text{B.30})$$

$$= \frac{1}{2} \left(\frac{\partial E}{\partial t} du^2 + 2 \frac{\partial F}{\partial t} dudv + \frac{\partial G}{\partial t} dv^2 \right) |_{t=0} \quad (\text{B.31})$$

$$= \frac{1}{2} (2\langle S_{uu}, n \rangle du^2 + 4\langle S_{uv}, n \rangle dudv + 2\langle S_{vv}, n \rangle dv^2) |_{t=0} \quad (\text{B.32})$$

$$= (\langle S_{uu}, n \rangle du^2 + 2\langle S_{uv}, n \rangle dudv + \langle S_{vv}, n \rangle dv^2) |_{t=0} \quad (\text{B.33})$$

We let $L = \langle S_{uu}, n \rangle$, $M = \langle S_{uv}, n \rangle$ and $N = \langle S_{vv}, n \rangle$ and we have the second fundamental form defined as follows:

$$\mathbf{II} = Ldu^2 + 2Mdudv + Ndv^2 \quad (\text{B.34})$$

Using the coefficients of $\mathbf{II}$ we can express the second fundamental form in a matrix much like the first fundamental form:

$$\begin{pmatrix} L & M \\ M & N \end{pmatrix} \quad (\text{B.35})$$

Note that the above second fundamental form was calculated by taking the derivative at $t = 0$ and thus it is the second fundamental form only for the specific surface $R(u, v, 0) = S(u, v)$ and not the family of surfaces.

Now that we have defined the first and second fundamental forms we can define the *principle curvatures* of a surface $S(u, v)$ by defining the matrix $\tau = \mathbf{I}^{-1}\mathbf{II}$. The principle curvatures κ_1 and κ_2 are then the eigenvalues of τ . The matrix τ is also known as the curvature tensor.

B.2 Extension to triangulated meshes

While we have defined curvature formally for smooth surfaces in Section B.1, it is not immediately clear how curvature can be defined for triangulated meshes which are approximations to smooth surfaces. Steiner et al [15] show that the second fundamental form arises naturally from the normal cycle of the mesh. Thus once we have the normal cycle of the mesh we can estimate the curvature tensor at each vertex of the mesh. The complete and formal definition of normal cycles is beyond the scope of this thesis and more details can be found in Steiners paper [15] [82] [84]. We will give an intuition of what and how normal cycles are used to define and compute curvatures a triangulated mesh through examples.

Normal cycles are based off of the theory of *currents* which is a subset of the larger field of geometric measure theory. Detailed analysis and formal definitions with proofs can be found in [56]. Intuitively currents are similar to electrical currents in the sense that they have a direction and a magnitude. Normal cycles are then a special type of current that is integral. We denote the normal cycle of $A \in \mathbb{E}^n$, where $\mathbb{E}^n$ is n -dimensional Euclidean space, to be $\mathbf{N}(A)$. A key property of the normal cycle is that it is additive. Thus $\mathbf{N}(A \cup B) = \mathbf{N}(A) + \mathbf{N}(B) - \mathbf{N}(A \cap B)$. Figure B.2 provides an example of a computed normal cycle in 2D.

Figure B.2: A computation of the normal cycle, $N(P)$, of a union of two line segments denoted as P . $N(P)$ is highlighted in blue. Note that since $N(P)$ is additive the normal cycles of the end points of the two line segments has the shape of a circle. The line itself is just the two lines above and below the line and thus together a single line segment has the shape of a rectangle with two half circles at the ends. Note the values $+1, 0, -1$. The $+1$ represents a curved region and the 0 represents the flat regions. The -1 represents a region where the arrow must change orientation by travelling along the edge until it reaches a point on the normal cycle of the center vertex where it can then travel along the circle in the opposite direction until it reaches the point where it can reach the edge for the other line segment. Since it must travel along part of the center vertex in the opposite direction the value for this region is negative as opposed to positive.

If we decompose a triangulated mesh, M , into it's parts we would have a set of vertices V , a set of edges E and a set of faces F . We use the term *simplex* to refer to the different types of objects in the decomposition of M . Since the normal cycle is additive, and the objects from a decomposed $M = (V, E, F)$ are all subsets of $\mathbb{E}^n$, the normal cycle of M is the summation of the normal cycle of the V , E and F . Figure B.3 illustrates the normal cycle in 3D for an edge and a triangle.

Figure B.3: The normal cycles of an edge and triangle in 3D are visualized in blue.

Given this intuition Steiner et al [15] show that the “mass” (see Figure B.4) for each of the simplices can be computed as follows:

Vertex The mass of a single vertex v within a triangulation is less than the area of a sphere around v . When v is connected to just one edge, it is a half-sphere. When connected to more than one edge it is the area of the sphere that is not covered by the union of mass of the edges (see Figure B.4a).

Edge Since it is composed of lines and vertex end points, we just need to determine the mass for the line. The mass of a single edge e is $|\beta(e)|length(e)$ where $\beta(e)$ is the signed angle (positive when e is convex, negative if concave) between the normals of the incident triangles to the edge e (see Figure B.4b).

Triangle Since it is composed of edges and vertices, we only need to worry about the triangular plane itself. The mass is equal to area of the planes directly on top of the triangle (see Figure B.4c).

Figure B.4: Illustration of the “mass” of the normal cycles for each of the simplices is shaded in green. In Figure B.4b edge is highlighted in magenta and angle $\beta(e)$ is shown as the shaded red region. Figure B.4a shows the mass for a vertex with adjacent to three edges. Note that the mass (green) does not include any of the edge masses (blue).

Putting all of this together Steiner et al [15] show the curvature measure for a subset of a triangular mesh $M = (V, E)$ is defined as the following:

$$C(M) = \sum_{e \in E} \beta(e) \text{length}(e) \mathbf{vec}(e) \otimes \mathbf{vec}(e) \quad (\text{B.36})$$

where $u \otimes v$ is the matrix computed from $u.v^t$, $\mathbf{vec}(e)$ is a vector in the direction of the edge e and $\beta(e)$ is the signed angle between the incident triangles of e . The matrix $C(M)$ is then the curvature tensor for M .

Appendix C

Gravitational Clustering

The gravitational clustering algorithm was first proposed by Wright as a way to merge particles (data points) until only one particle (cluster) remained. It was then extended by Gomez et al. [29] who allowed for multiple clusters as well as some modifications on how the gravitational forces interacted with the particles. This appendix details the clustering algorithm used for separating the points of extreme curvature which was used as a sub-routine in the quad-tree roughness ratio measure found in Section 6.5.

C.1 Approximation of Movement due to Gravity

Before we discuss the algorithm we must first understand the theory of gravity. The most widely accepted theory of gravity was developed by Isaac Newton and it explains the motion of celestial bodies as well as the large scale structure of the Universe. Everything in the universe with mass exerts a gravitational pull on everything else in the universe. The strength of this pull is called the gravitational force and is modelled by the following equation:

$$F(t) = \frac{Gm_xm_y}{d(x(t), y(t))^2} \quad (\text{C.1})$$

where m_x and m_y are the masses of object x and y , $d(x(t), y(t))$ is the Euclidean distance between x and y at time t , and G is the Universal gravitational constant (6.67×10^{-11}). Thus an object y under the gravitational influence of an object x is pulled towards x with the directional vector $d(t) = x(t) - y(t)$. Using the directional vector we can rewrite Equation C.1 as follows:

$$F(t) = \frac{Gm_x m_y}{|d(t)|^2} \quad (\text{C.2})$$

To find the new location of object y with respect to the gravity of an object x we need to use Newtons second law, $F = ma$, in conjunction with straight line motion laws. Straight line motion tells us that that the position of an object x with non-constant acceleration at time t is modelled as follows:

$$x(t + \Delta t) = x(t) + v(t)\Delta t + \frac{a(t)\Delta t^2}{2} \quad (\text{C.3})$$

where $v(t)$ and $a(t)$ is the velocity and acceleration of x at time t . Using Newtons second law ($F = ma$) and the law of gravitation (Equation C.2) we can find the acceleration due to gravity which will allow us to use Equation C.3 to determine the position of a particle under the influence of gravity at any point in time.

$$F_g(t) = m_y a(t) \quad (\text{C.4})$$

$$\frac{Gm_x m_y}{|d(t)|^2} = m_y a(t) \quad (\text{C.5})$$

$$a(t) = \frac{Gm_x}{|d(t)|^2} \quad (\text{C.6})$$

Thus the acceleration of an object y under the influence of gravity of an object y is modelled by Equation C.6. Since the acceleration is a complex differential equation finding the position function of y under one or more gravitational fields is non-trivial.

We approximate the acceleration vector with the following equation:

$$a(t) = d(t) \frac{Gm_x}{|d(t)|^3} \quad (\text{C.7})$$

Substitute Equation C.7 into Equation C.3 and we get the following:

$$y(t + \Delta t) = y(t) + v(t)\Delta t + d(t) \frac{Gm_x \Delta t^2}{2|d(t)|^3} \quad (\text{C.8})$$

which represents the position of object y under the influence of the gravitational field of object x from time t to time $t + \Delta t$.

C.2 Algorithm

The algorithm takes as input a set of N data points in n -dimensional space and moves them according to the force of gravity for k iterations. The basic idea is as follows:

1. A data point within some cluster will exert a greater amount of gravitational force on a data point within the same cluster than a data point not within the same cluster.
2. Data points which do not belong to any cluster will experience very little gravitational force from other points making it effectively immobile.
3. Once two data points are within some ε distance they are considered to be in the same cluster.
4. After a set amount of time (i.e: iterations) the data will form clusters through the force of gravity.

To save processing time and memory we use a simplified version of Equation C.8 to find the new position of the data points for each iteration. We let the mass of each

point within the data be 1, discretize the change in time (i.e: let $\Delta t = 1$) and we let the velocity of the data points at any time be zero. This allows the algorithm to not require any additional memory for storing the velocity vector of each data points and the runtime is decreased as the number of operations decreases. The simplified equation is as follows:

$$x(t+1) = x(t) + d(t) \frac{G}{|d(t)|^3} \quad (\text{C.9})$$

Now that we have discussed the basic outline of the algorithm and exactly how to move data points on each iteration we need to briefly discuss how to extract the clusters after they have been moved. The algorithm uses a UNION-FIND structure on top of the input to allow for each data point to still move according to gravity but still be contained with the same cluster. This is the structure which is used to extract the clusters after the algorithm is finished.

The main parameters which influence the clusters found is the gravitational constant G , the minimum distance ε for merging clusters, and the number of iterations the algorithm will run for. The formal definition of the gravitational clustering algorithm is stated in Algorithm 5. In each iteration the algorithm moves every data point with respect to a random data point. The random selection of data points prevents selection bias patterns from occurring which allows the data to cluster according only to gravity. Once two data points are with an ϵ distance of each other they are merged together in the underlying UNION-FIND structure. The process is repeated for M iterations and on each iteration the gravitational constant G is reduced by a decaying term ΔG . This term prevents the algorithm from clustering all the data points as one cluster after a very large (infinite) number of iterations.

Since the UNION-FIND structure assigns every data point (noise included) to their own cluster it is necessary to define an extract parameter α when extracting valid clusters. We can assign α to be some percent of N and thus only clusters which have a size $> \alpha N$ would be considered to be a valid cluster.

Algorithm 5 Gravitational Clustering

Require: data points p , G , ΔG , M , ε **Ensure:** clusters C

```

1: Initialize UNION-FIND structure  $U$  with 1 to  $n$ .
2: Initialize  $C$  to be a mapping between integers and lists.
3: for  $i = 1$  to  $M$  do
4:   for  $j = 1$  to  $N$  do
5:      $p_k =$  random point such that  $k \neq j$ 
6:     MOVE( $p_j, p_k$ ) (via Equation C.9)
7:     if  $\text{dist}^2(p_j, p_k) \leq \varepsilon$  then
8:        $U.\text{union}(j, k)$ 
9:     end if
10:  end for
11:   $G = (1 - \Delta G) \times G$ 
12: end for
13: for  $i = 1$  to  $N$  do
14:   $c_{id} = U.\text{find}(i)$ 
15:  if  $C.\text{contains}(c_{id})$  then
16:     $C[c_{id}].\text{add}(p_i)$ 
17:  else
18:     $C[c_{id}] = \{p_i\}$ 
19:  end if
20: end for
21: return  $C$ 

```

Parameter	Description	Sensitivity to change
G	Gravitational constant. It is the empirical physical constant involved in the calculation of gravitational force between two bodies.	If this value is too large the algorithm will form one cluster. If it is too small the algorithm will not form clusters at all. No universal value is known [29] and is dependant on the data set and must be chosen carefully.
ΔG	Gravitational decay. Since gravity is a constant force, after a large (infinite) number of iterations, all points will be moved into the same position thus producing a single cluster. This term is introduced to prevent this scenario [29].	If this term is too large the algorithm will decrease the force of gravity before the data points have a chance to form clusters. If it is too low then the algorithm will decrease the force of gravity too slowly and the data will form one cluster.
ε	Minimum distance between two clusters in-order for them to be merged into one cluster.	The algorithm is not very sensitive to this parameter since the minimum distance can be overcome by increasing the number of iterations as well as changing the ΔG or G parameter. Only when the minimum distance is over some 1% of the maximum distance between any two data points will the ε parameter begin to force the algorithm to produce one large cluster [29].
α	Maximum size to be considered a cluster	Used to eliminate noise in datasets. Clusters which are too small (i.e: a single point) would be considered outliers.

Table C.1: Description of parameters used in gravitational clustering.

C.3 Algorithm Parameters

Table C.1 summarizes the main parameters which can affect the clusters produced by gravitational clustering and table C.2 summarizes the parameter values used in Section 6.5.

Parameter	Value	Rationale
G	10^{-4}	Empirically determined.
ΔG	0.001	Empirically determined.
ε	Diagonal of the 3D bounding box of the input data.	This value was chosen to be variable specially because we wanted the clustering to reflect the curvature rather than the spatial distance. By choosing to merge any points which have a distance that is less than the diagonal we immediately place all points which have zero curvature into one cluster in the first iteration. The subsequent iterations will then merge all the non-extreme curvature points. The points which have extreme curvature will gravitate towards others which are similar in curvature value and thus do not approach the merging threshold of the large non-extreme cluster.
α	0.03	As suggested by [29].

Table C.2: Values of the parameters used in Section 6.5.

Appendix D

Quad Tree Operations

Algorithm 6 Build-Quadtree

Require: quad-node Q , stopping condition $s \in \{\text{width}, \text{height}, \text{surface area}\}$,
stopping value v

Ensure:

```

1: if ( $s$  is width and  $Q.bounds.width < v$ )
   or ( $s$  is height and  $Q.bounds.height < v$ )
   or ( $s$  is surface area and  $Q.bounds.area < v$ ) then
2:    $Q.isLeaf = \text{True}$ 
3:   return
4: end if
5:  $NE, NW, SE, SW \leftarrow Q.bounds.split$ 
6:  $Q.NE = \text{new QuadNode}(NE)$ 
7:  $Q.NW = \text{new QuadNode}(NW)$ 
8:  $Q.SE = \text{new QuadNode}(SE)$ 
9:  $Q.SW = \text{new QuadNode}(SW)$ 
10: Build-Quadtree( $Q.NE$ ,  $s$ ,  $v$ )
11: Build-Quadtree( $Q.NW$ ,  $s$ ,  $v$ )
12: Build-Quadtree( $Q.SE$ ,  $s$ ,  $v$ )
13: Build-Quadtree( $Q.SW$ ,  $s$ ,  $v$ )
14: return

```

Algorithm 7 Find-Elements

Require: quad-node n , list L **Ensure:** list L will contain all the elements underneath quad-node n

```

1: if  $n.isLeaf = \text{True}$  then
2:    $L \leftarrow L \cup n.elements$ 
3: else
4:   Find-Elements( $n.NE$ ,  $L$ )
5:   Find-Elements( $n.NW$ ,  $L$ )
6:   Find-Elements( $n.SE$ ,  $L$ )
7:   Find-Elements( $n.NW$ ,  $L$ )
8: end if

```

Algorithm 8 Find-QuadNodes

Require: quad-node n , list L , $i < \text{height of the quad-tree}$ **Ensure:** list L will contain all the quad-nodes at level i

```

1: if  $i > 0$  then
2:   Find-QuadNodes( $n.NE$ ,  $L$ ,  $i - 1$ )
3:   Find-QuadNodes( $n.NW$ ,  $L$ ,  $i - 1$ )
4:   Find-QuadNodes( $n.SE$ ,  $L$ ,  $i - 1$ )
5:   Find-QuadNodes( $n.NW$ ,  $L$ ,  $i - 1$ )
6: else
7:    $L \leftarrow L \cup n$ 
8: end if

```

Algorithm 9 Find-QuadStrip

Require: Quad-node q , Mapping S , levels of quad-tree remaining h , current row r

Ensure: S contains a mapping of $r \rightarrow$ list of quad-nodes

 Compute the curvature of M using local window size W_r

if $h > 0$ **then**

 Find-QuadStrip($q.NE$, S , $h - 1$, $r \times 10 + 0$)

 Find-QuadStrip($q.NW$, S , $h - 1$, $r \times 10 + 0$)

 Find-QuadStrip($q.SE$, S , $h - 1$, $r \times 10 + 1$)

 Find-QuadStrip($q.SW$, S , $h - 1$, $r \times 10 + 1$)

else

if $S.hasKey(r) == \text{False}$ **then**

$S.put(r, \text{new List}())$

end if

$S.get(r).add(q)$

end if

Appendix E

Implementation Details

In this appendix we discuss the software that was created and used for analysing the captured datasets (see Figure E.1). The software was used for reconstructing and smoothing surfaces, mapping the orientation of the mesh elements, developing and testing the various algorithms as described in Chapters 4, 5, 6 and 7. The software also renders the reconstructed surface along with any additional color mapping applied to the surface. The software was written to allow geologists, mining engineers or anyone other interested end users to install and use with ease. Thus the programming language JAVA was chosen for the ease of cross-platform distribution. Any major operating system (MacOS, Windows, UNIX-based) will support the Java Runtime Environment (JRE). OpenGL was chosen as the API for rendering triangulated meshes and the Lightweight Java Game Library (LWJGL) was chosen as the library for interfacing with OpenGL commands in JAVA. The 2.5D Delaunay triangulation described in Chapter 4 was computed using a library created by Boaz [8]. The Poisson surface reconstruction was computed using MeshLab [13] and the results exported. All algorithms described in Chapters 5, 6 and 7 were designed and implemented by the author with the exception of the automated computation of strike and dip (Section 5.1) which was written by Mah et al [51] in MATLAB. The strike and dip code

was translated by the author and imported into the author’s software.

Figure E.1: The application created and used for analysis. Pictured is the shatter cone scene found in Section 7.2, Figure 7.4.

E.1 Wavelength Color Scale

Since the values of curvature range from $-\infty$ to $+\infty$, a common approach to visualize curvature is to assign negative curvature values a color (i.e: blue) and positive curvature values another color (i.e: red). All of the curvature values inbetween are assigned a color which is blended from the positive and negative color values. We do not take this approach as fine grain detail is lost (i.e: the difference between two positive curvature values) and instead map the curvature range of a mesh M onto the range of visible light. The wavelength range of visible light for humans is approximately

400nm to 700nm [75]. Thus given a range $[a, b]$ where $a < b$ and $a \in [-\infty, +\infty]$ and $b \in [-\infty, +\infty]$ we remap a to 400 and b to 700. Then given any value $\lambda \in [a, b]$ we compute its wavelength and convert it to a RGB value using Algorithm 10 which is due to Bruton [12]. This color scale is again used for the roughness algorithms presented in Chapter 6.

Algorithm 10 Convert Wavelength to RGB

Require: wavelength $\lambda \in [400, 700]$

Ensure: (R, G, B) color

```

1:  $R, G, B, F \leftarrow 0, 0, 0, 0$ 
2: if  $\lambda \geq 380$  and  $\lambda \leq 440$  then
3:    $R, G, B \leftarrow \frac{-(\lambda-440)}{440-380}, 0, 1$ 
4: else if  $\lambda > 440$  and  $\lambda \leq 489$  then
5:    $R, G, B \leftarrow 0, \frac{\lambda-440}{489-440}, 1$ 
6: else if  $\lambda > 489$  and  $\lambda \leq 509$  then
7:    $R, G, B \leftarrow 0, 1, \frac{-(\lambda-509)}{509-489}$ 
8: else if  $\lambda > 489$  and  $\lambda \leq 509$  then
9:    $R, G, B \leftarrow 0, 1, \frac{-(\lambda-509)}{509-489}$ 
10: else if  $\lambda > 509$  and  $\lambda \leq 579$  then
11:    $R, G, B \leftarrow \frac{\lambda-509}{579-509}, 1, 0$ 
12: else if  $\lambda > 579$  and  $\lambda \leq 644$  then
13:    $R, G, B \leftarrow 1, \frac{-(\lambda-644)}{644-579}, 0$ 
14: else if  $\lambda > 644$  and  $\lambda \leq 780$  then
15:    $R, G, B \leftarrow 1, 0, 0$ 
16: end if
17: if  $\lambda \geq 380$  and  $\lambda \leq 419$  then
18:    $F \leftarrow 0.3 + 0.7(\frac{\lambda-380}{420-380})$ 
19: else if  $\lambda > 419$  and  $\lambda \leq 700$  then
20:    $F \leftarrow 1$ 
21: else if  $\lambda > 700$  and  $\lambda \leq 780$  then
22:    $F \leftarrow 0.3 + 0.7(\frac{780-\lambda}{780-700})$ 
23: end if
24:  $R \leftarrow 255 \times (R \times F)^{0.8}$ 
25:  $G \leftarrow 255 \times (G \times F)^{0.8}$ 
26:  $B \leftarrow 255 \times (B \times F)^{0.8}$ 
27: return  $(R, G, B)$ 

```

Bibliography

- [1] CIE Technical Committee 8-01. The ciecam02 color appearance model, 2002.
URL www.polybytes.com/misc/Meet_CIECAM02.pdf. [Online; accessed 9-January-2013].
- [2] Pierre Alliez, Giuliana Ucelli, Craig Gotsman, and Marco Attene. Recent advances in remeshing of surfaces. In *Shape Analysis and Structuring, Mathematics and Visualization*. Springer, 2008.
- [3] N. Barton, R. Lien, and J. Lunde. Engineering classification of rock masses for the design of tunnel support. *Rock mechanics*, 6(4):189–236, 1974. ISSN 0035-7448. doi: 10.1007/BF01239496. URL <http://dx.doi.org/10.1007/BF01239496>.
- [4] P. Beckmann and A. Spizzichino. *The scattering of electromagnetic waves from rough surfaces*. 1987.
- [5] Alexander Belyaev and Yutaka Ohtake. A comparison of mesh smoothing methods. In *Israel-Korea Bi-National Conference on Geometric Modeling and Computer Graphics*, pages 83–87, Tel-Aviv, Israel, 2003. Tel Aviv University.
- [6] G. Berginc. Small-slope approximation method: A further study of vector wave scattering from two-dimensional surfaces and comparison with experimental data. *Progress in Eletromagnetics Research*, 37:251–287, 2002. doi: doi:10.2528/PIER02070603.

- [7] Z.T. Bieniaswki. Engineering classification of jointed rock masses. *Transations of South African Instituion of Civil Engineers*, 15(12):335–344, 1973.
- [8] Ben-Moshe Boaz. Delaunay triangulation in java, September 2011. URL <http://www.cs.bgu.ac.il/~benmoshe/DT/Delaunay%20Triangulation%20in%20Java.htm>.
- [9] A. Boivin, P. Lai, C. Samson, E. Cloutis, S. Holladay, and F.A. Monteiro Santos. Electromagnetic induction sounding and 3d laser imaging in support of a mars methane analogue mission. *Planetary and Space Science*, 2013. ISSN 0032-0633. doi: 10.1016/j.pss.2013.03.005. URL <http://www.sciencedirect.com/science/article/pii/S0032063313000640>.
- [10] S L Broschat and E I Thorsos. An investigation of the small slope approximation for scattering from rough surfaces. part ii. numerical studies. *Journal of the Acoustical Society of America*, 101(5):2615, 1997. URL <http://link.aip.org/link/?JAS/101/2615/1&Agg=doi>.
- [11] S.L. Broschat. The phase perturbation approximation for rough surface scattering from a pierson-moskowitz sea surface. *IEEE Trans. Geosci. Rem. Sens.*, 31(1), Jan 1993.
- [12] Dan Bruton. Approximate rgb values for visible wavelengths, February 1996. URL <http://www.physics.sfasu.edu/astro/color/spectra.html>.
- [13] Paolo Cignoni. MeshLab. URL <http://meshlab.sourceforge.net/>.
- [14] Di Bisceglie M. Galdi C. Gommenginger C. Clarizia, M.P. and Srokosz M. What do reflected gps signals tell us about the ocean waves? a numerical study. In *GNSS-R Workshop 10*, 2010.

- [15] David Cohen-Steiner and Jean-Marie Morvan. Restricted delaunay triangulations and normal cycle. In *Symposium on Computational Geometry*, pages 312–321, 2003.
- [16] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction To Algorithms*. MIT Press, 2001. ISBN 9780262032933. URL http://books.google.ca/books?id=NLngYyWF1_YC.
- [17] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 2008. ISBN 9783540779735. URL <http://books.google.ca/books?id=tkyG8W2163YC>.
- [18] Mark de Berg and Katrin Dobrindt. On levels of detail in terrains. In *Symposium on Computational Geometry*, pages C26–C27, 1995.
- [19] Mark de Berg, Prosenjit Bose, Otfried Cheong, and Pat Morin. On simplifying dot maps. *Computational Geometry*, 27(1):43 – 62, 2004. ISSN 0925-7721. doi: <http://dx.doi.org/10.1016/j.comgeo.2003.07.005>. URL <http://www.sciencedirect.com/science/article/pii/S0925772103000634>.
[jce:title;Computational Geometry - EWCG'02; /ce:title;.](#)
- [20] Boris N. Delaunay. Sur la sphère vide. *Bulletin of Academy of Sciences of the USSR*, (6):793–800, 1934.
- [21] Kaichang Di, Zongyu Yue, Zhaoqin Liu, and Shuliang Wang. Automated rock detection and shape analysis from mars rover imagery and 3d point cloud data. *Journal of Earth Science*, 24(1):125–135, 2013. ISSN 1674-487X. doi: [10.1007/s12583-013-0316-3](http://dx.doi.org/10.1007/s12583-013-0316-3). URL <http://dx.doi.org/10.1007/s12583-013-0316-3>.
- [22] Robert S. Dietz and Louis W. Butler. Shatter-cone orientation at sudbury, canada. *Nature*, 204:280 – 281, October 1964. doi: [10.1038/204280a0](http://dx.doi.org/10.1038/204280a0). URL <http://www.nature.com/nature/journal/v204/n4955/abs/204280a0.html>.

- [23] István Fáry. On straight-line representation of planar graphs. *Acta Sci. Math. (Szeged)*, 11:229–233, 1948.
- [24] Diederichs M Fekete S and Lato M. Geotechnical and operational applications for 3-dimensional laser scanning in drill and blast tunnels. *Tunnelling and Underground Space Technology*, 2010.
- [25] R.A. Finkel and J.L. Bentley. Quad trees a data structure for retrieval on composite keys. *Acta Informatica*, 4(1):1–9, 1974. ISSN 0001-5903. doi: 10.1007/BF00288933. URL <http://dx.doi.org/10.1007/BF00288933>.
- [26] Canada Centre for Remote Sensing. Globe sar ocean imaging and applications, 2010.
- [27] Canada Centre for Remote Sensing. Globe sar land application, 2012.
- [28] B.M. French. *Traces of catastrophe: a handbook of shock-metamorphic effects in terrestrial meteorite impact structures*. LPI contribution. Lunar and Planetary Institute, 1998. URL <http://books.google.ca/books?id=LTJQAQAIAAJ>.
- [29] Jonatan Gómez, Dipankar Dasgupta, and Olfa Nasraoui. A new gravitational clustering algorithm. In *SDM*, 2003.
- [30] Ibrahim Ciftci Ulvi Seker Halil Demir, Abdulkadir Gullu. An investigation into the influences of grain size and grinding parameters on surface roughness and grinding forces when grinding. *Journal of Mechanical Engineering*, 56, 2010.
- [31] E. Hecht. *Optics*, 4/E. Pearson Education, 2008. ISBN 9788131718070. URL <http://books.google.ca/books?id=wcMWpBMMzIkC>.
- [32] E. Hoek. *Rock engineering*. 1999. URL <http://books.google.ca/books?id=zLPLSgAACAAJ>.

- [33] E. Hoek. Practical rock engineering. Course Notes, 2013.
- [34] Hugues Hoppe. Progressive meshes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, pages 99–108, New York, NY, USA, 1996. ACM. ISBN 0-89791-746-4. doi: 10.1145/237170.237216. URL <http://doi.acm.org/10.1145/237170.237216>.
- [35] Paul A. Hwang. Microstructure of ocean surface roughness: A study of spatial measurement and laboratory investigation of modulation analysis. *Journal of Atmospheric and Oceanic Technology*, 16:1619–1629, 1999.
- [36] Jun ichi Takada. Wave theory ii physical optics approximation. University Lecture, 2004.
- [37] Precision Devices Inc. Surface profile parameters, 1998. URL <http://www.predev.com/smgsd.pdf>. [Online; accessed 15-May-2012].
- [38] A. Ishimaru. *Wave Propagation and Scattering in Random Media: Multiple scattering, turbulence, rough surfaces, and remote sensing*. Number v. 2 in Wave Propagation and Scattering in Random Media. Academic Press, 1978. ISBN 9780123747020. URL <http://books.google.ca/books?id=NLMRAQAIAAJ>.
- [39] M Jaboyedoff, R Metzger, T Oppikofer, R Couture, MH Derron, J Locat, and D Turmel. New insight techniques to analyze rock-slope relief using dem and 3d-imaging cloud points: Coltop-3d software. *Rock Mechanics: Meeting Society's Challenges and Demands: Proceedings of the 1st Canada-US Rock Mechanics Symposium, Vancouver, Canada*, 27:31, 2007.
- [40] Katherine S. Kalenchuk, Mark S. Diederichs, and Steve McKinnon. Characterizing block geometry in jointed rockmasses. *International Journal of Rock Mechanics and Mining Sciences*, 43(8):1212 – 1225, 2006. ISSN 1365-1609.

- doi: 10.1016/j.ijrmms.2006.04.004. URL <http://www.sciencedirect.com/science/article/pii/S1365160906000566>.
- [41] Zachy Karni and Craig Gotsman. Spectral compression of mesh geometry. In *SIGGRAPH*, pages 279–286, 2000.
- [42] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, SGP '06, pages 61–70, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association. ISBN 3-905673-36-3. URL <http://dl.acm.org/citation.cfm?id=1281957.1281965>.
- [43] Michael Kazhdan, Allison Klein, Ketan Dalal, and Hugues Hoppe. Unconstrained isosurface extraction on arbitrary octrees. In *Proceedings of the fifth Eurographics symposium on Geometry processing*, SGP '07, pages 125–133, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association. ISBN 978-3-905673-46-3. URL <http://dl.acm.org/citation.cfm?id=1281991.1282009>.
- [44] Patric Keller, Oliver Kreylos, Marek Vanco, Martin Hering-Bertram, EricS. Cowgill, LouiseH. Kellogg, Bernd Hamann, and Hans Hagen. Extracting and visualizing structural features in environmental point cloud lidar data sets. In Valerio Pascucci, Xavier Tricoche, Hans Hagen, and Julien Tierny, editors, *Topological Methods in Data Analysis and Visualization*, Mathematics and Visualization, pages 179–192. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-15013-5. doi: 10.1007/978-3-642-15014-2_15. URL http://dx.doi.org/10.1007/978-3-642-15014-2_15.
- [45] Donald E. Knuth. *The art of computer programming, volume 1 (3rd ed.): fundamental algorithms*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1997. ISBN 0-201-89683-4.

- [46] Hutchinson D.J Harrap R Lato M, Diederichs M.S. Evaluating roadside rock-masses for rockfall hazards using lidar data: optimizing data collection and processing protocols. *Nat. Hazards*, 2012.
- [47] Voge M. Lato M. Automated mapping of rock discontinuities in 3d lidar models. *Int. J. of Rock Mech. And Min. Sci.*, 2012.
- [48] Guillaume Lavoué. A local roughness measure for 3d meshes and its application to visual masking. *ACM Trans. Appl. Percept.*, 5(4):21:1–21:23, February 2009. ISSN 1544-3558. doi: 10.1145/1462048.1462052. URL <http://doi.acm.org/10.1145/1462048.1462052>.
- [49] K. H. Lee, H. Woo, and T. Suk. Point data reduction using 3d grids. *The International Journal of Advanced Manufacturing Technology*, 18(3):201–210, 2001. ISSN 0268-3768. doi: 10.1007/s001700170075. URL <http://dx.doi.org/10.1007/s001700170075>.
- [50] Lato M. Remote monitoring of deformation using terrestrial laser scanning (tls or terrestrial lidar). *Geotechnical Instrumentation News*, 2012.
- [51] Jason Mah, Claire Samson, and Stephen D. McKinnon. 3D laser imaging for joint orientation analysis. *International Journal of Rock Mechanics and Mining Sciences*, 48:932–941, 2011. doi: 10.1016/j.ijrmms.2011.04.010.
- [52] Jason Mah, Claire Samson, Stephen D. McKinnon, and Denis Thibodeau. 3d laser imaging for surface roughness analysis. *International Journal of Rock Mechanics and Mining Sciences*, 58(0):111 – 117, 2013. ISSN 1365-1609. doi: 10.1016/j.ijrmms.2012.08.001. URL <http://www.sciencedirect.com/science/article/pii/S1365160912001724>.
- [53] McKinnon SD Aikman D. Mah J, Samson C. Triangulation-based 3d laser imaging for underground rock mass characterization: impact of operational param-

- eters. In *Proceedings of the 42nd US rock mechanics symposium*, pages 87–94. American Rock Mechanical Association, 2007.
- [54] Warnick K.F. Millet, F.W. Validity of rough surface backscattering models. *Waves Random Media*, 14(3):327–347, 2004. URL <http://www.scopus.com/inward/record.url?eid=2-s2.0-3342990115&partnerID=40&md5=b672c4cd5f92b6e9a820d9324dc3c302>. cited By (since 1996) 12.
- [55] J.S. Monroe and R. Wicander. *Physical Geology: Exploring the Earth*. Available Titles CengageNOW Series. Brooks/Cole, 2005. ISBN 9780534399870. URL <http://books.google.ca/books?id=QIognFPkjXQC>.
- [56] F. Morgan. *Geometric Measure Theory: A Beginner's Guide*. Elsevier Science, 2000. ISBN 9780080525600. URL <http://books.google.ca/books?id=f1VeGoxUBogC>.
- [57] J.M. Morvan. *Generalized curvatures*. Geometry and Computing, 2. Springer Berlin Heidelberg, 2008. ISBN 9783540737926. URL <http://books.google.ca/books?id=TyS1ohX0B0wC>.
- [58] Ulan Nurkhanuly. Structurally controlled hydrothermal dolomite, eganville-douglas paleozoic outlier, ottawa-bonnechere graben, eastern ontario. Master's thesis, Carleton University Dept. of Earth Sciences, May 2012.
- [59] Mark Pauly, Markus Gross, and Leif P. Kobbelt. Efficient simplification of point-sampled surfaces. In *Proceedings of the conference on Visualization '02*, VIS '02, pages 163–170, Washington, DC, USA, 2002. IEEE Computer Society. ISBN 0-7803-7498-3. URL <http://dl.acm.org/citation.cfm?id=602099.602123>.
- [60] C.S. Lin P.M. Smith. A laser slope gauge and a spar buoy wave gauge.

- [61] S. O. Rice. Reflection of electromagnetic waves from slightly rough surfaces. *Commun. Pure Appl. Math.*, 4:351, 1951.
- [62] Xiangqian Jiang Mike Conroy Darrian Mauger Richard Leach, Leigh Brown. *Measurement Good Practice Guide No. 108*. National Physical Laboratory, Hampton Road, Teddington, Middlesex, 2008.
- [63] Ernesto Rodríguez and Yunjin Kim. A unified perturbation expansion for surface scattering. *Radio Science*, 27(1):79–93, 1992. doi: doi:10.1029/91RS02638.
- [64] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *3DIM*, pages 145–152, 2001.
- [65] Amir Sagy, Jay Fineberg, and Ze’ev Reches. Shatter cones: Branched, rapid fractures formed by shock impact. *Journal of Geophysical Research: Solid Earth (1978–2012)*, 109(B10), 2004. doi: 10.1029/2004JB003016. URL <http://dx.doi.org/10.1029/2004JB003016>.
- [66] Amir Sagy, Jay Fineberg, and Ze’ev Reches. Shatter cones: Branched, rapid fractures formed by shock impact. *Journal of Geophysical Research: Solid Earth*, 109(B10):n/a–n/a, 2004. ISSN 2156-2202. doi: 10.1029/2004JB003016. URL <http://dx.doi.org/10.1029/2004JB003016>.
- [67] C. Samson, C. English, A. Deslauriers, I. Christie, F. Blais, and F. Ferrie. Neptec 3D Laser Camera System: From Space Mission STS-105 to Terrestrial Applications. *Canadian Aeronautics and Space Journal*, 50(2), June 2004.
- [68] William J. Schroeder, Jonathan A. Zarge, and William E. Lorensen. Decimation of triangle meshes. In *Computer Graphics (SIGGRAPH ’92 Proceedings)*, pages 65–70, 1992.

- [69] Theodore Shifrin. Differential geometry: A first course in curves and surfaces. University Course Notes, 2012.
- [70] S. Slob, H. R. G. K. Hack, and A. K. Turner. An approach to automate discontinuity measurements of rock faces using laser scanning techniques. In C. Denis da Gama and Sousa, editors, *Proceedings of the ISRM International Symposium on Rock Engineering for Mountainous Regions*, pages 87–94. Sociedade Portuguesa de Geotecnia, November 2002.
- [71] Siefko Slob, Bart van Knapen, Robert Hack, Keith Turner, and John Kemeny. Method for automated discontinuity analysis of rock slopes with three-dimensional laser scanning. *Transportation Research Record: Journal of the Transportation Research Board*, 1913(1):187–194, 2005.
- [72] Michiel Smid. Planar graphs, euler’s formula, crossing numbers, point-line incidences, and unit-distances. University Lecture, 2003.
- [73] Michiel Smid. Point location in planar subdivisions. University Lecture, 2003.
- [74] M. Spivak. *Calculus On Manifolds: A Modern Approach To Classical Theorems Of Advanced Calculus*. Westview Press, 1971. ISBN 9780813346120. URL <http://books.google.ca/books?id=P0IJJJcCyUkC>.
- [75] C. Starr, C.A. Evers, and L. Starr. *Biology: Concepts and Applications*. Available Titles CourseMate Series. BROOKS COLE Publishing Company, 2010. ISBN 9781439046739. URL http://books.google.ca/books?id=_16xbB2Py_UC.
- [76] Shira L. Broschat Taiqian Yang. A comparison of scattering model results for two-dimensional randomly rough surfaces. *IEEE Transactions on Antennas and Propagation*, 40, 1992.

- [77] Gabriel Taubin. A signal processing approach to fair surface design. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '95, pages 351–358, New York, NY, USA, 1995. ACM. ISBN 0-89791-701-4. doi: 10.1145/218380.218473. URL <http://doi.acm.org/10.1145/218380.218473>.
- [78] Eric I. Thorsos. The validity of the kirchhoff approximation for rough surface scattering using a gaussian roughness spectrum. *The Journal of the Acoustical Society of America*, 83, 1988.
- [79] V. N. Vapnik and A. Ya. Chervonenkis. On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities. *Theory of Probability and Its Applications*, 16, 1971. doi: 10.1137/1116025.
- [80] Xiaoyang Wen, Chao Wang, and Hong Zhang. The scattering study of rough surface by ray tracing technique. *Environmental Science and Information Application Technology, International Conference on*, 2:704–707, 2009. doi: <http://doi.ieeecomputersociety.org/10.1109/ESIAT.2009.552>.
- [81] Dale P. Winebrenner and Akira Ishimaru. Application of the phase-perturbation technique to randomly rough surfaces. *J. Opt. Soc. Am. A*, 2(12):2285–2293, Dec 1985. doi: 10.1364/JOSAA.2.002285. URL <http://josaa.osa.org/abstract.cfm?URI=josaa-2-12-2285>.
- [82] P. Wintgen. *Normal Cycle and integral curvature for polyhedra in Riemannian manifolds*. North-Holland, Amsterdam, 1982.
- [83] Carl Weimer. Youngxiang Hu. High resolution sea surface roughness and wind speed with space lidar (calipso). In *International Winds Working Group (IWWG) Workshop 10*, 2010.

- [84] M. Zähle. *Integral and current representation of Federer's curvature measures*. Forschungsergebnisse / Friedrich-Schiller-Universität Jena. Friedrich-Schiller-Univ., 1985. URL <http://books.google.ca/books?id=Iyn-ewEACAAJ>.