An Analysis of Extracted Cross Sections from Heritage Site Point Clouds Using Mesh-free Point-based Techniques

by

Jerome Craig Aaron

ARNJER002

Supervised by Prof Patrick Marais

A Thesis submitted for the degree of MIT by coursework and dissertation (CSC5004W)

in the

Department of Computer Science

University of Cape Town

November 2024

Declaration of Authorship

I, Jerome Craig Aaron, declare that this work presented is my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Signed by candidate

Date: 2024-11-01

Abstract

In the digital age cultural heritage has evolved to incorporate 3D virtual models of heritage buildings and sites. It is common to use laser scanning to acquire the 3D point cloud data for these models, with this data usually being processed and “meshed” to form a surface mesh model, however, this process is computationally costly and generally creates new data (beyond the original, correct, surface samples). For heritage conservation purposes, accurate and original data is preferred. An important issue when using the points directly, rather than the 3D surface model, is the suitability of a direct point representation for rendering and image processing. One important task is the production of floorplan, elevation, and cross-section views from a 3D digital heritage model of a site.

In this work, the viability of a Point Based Technique (PBT) approach, for cross-section recovery from heritage point cloud models, is evaluated. The solution developed involves slicing the 3D point cloud, applying binary morphology operations on this 2D point cloud slice to close gaps in the cross-section profile, and filling holes in the cross-section profile. Post processing operations add minor but necessary improvements (such as filling gaps on the ground level due to slicing through vegetation and removing noise). Finally, to produce the desired image, the cross-section is overlaid onto the rendering of the scene (removing points in front of the cross-section slicing plane).

The results are assessed by registering the point-based cross-section against the mesh-based equivalent. The Intersection over Union (IoU) metric, a measure of similarity between two images, is calculated on the registered images. The tabulated IoU and IoU loss (dissimilarity between the registered cross-sections) is also depicted graphically.

The conclusion drawn on analysing the results, is that the fidelity of the point-based cross-section is a close approximation to the mesh-based cross-section (even when considering possible errors from manual registration of the cross-sections), thus validating the point based approach. The average IoU “score” across the 24 cross-sections recovered from 4 point cloud, cultural heritage models is 94.6%. An argument may be made for the point-based cross-section being an improvement over the mesh-based cross-section, where the latter connects points incorrectly in the mesh reconstruction of the cross-section.

A positive relationship is found between the point density and the fidelity of the point-based cross-section. An examination of the data and results reveals that a higher point density relates to a higher fidelity of the cross-section. The cross-section profile in a higher density point cloud is more likely to have no gaps in the profile, after binary morphology Closing (distance between points are smaller at higher point densities). Note that high point density is typical of current scanning technology (LIDAR, Photogrammetry) but the output is typically downsampled for practical use (e.g. uploading to repositories with file size limitations, to make the data publicly available), relying instead on mesh reconstruction to model a surface.

For the task of recovering cross-sections from 3D point clouds of cultural heritage sites we conclude that point-based techniques offer a similar accuracy to mesh-based techniques - provided that the model possesses a sufficiently high point density. The advantage of using points directly is the avoidance of (i) additional effort and computational cost of reconstructing a mesh, and (ii) the possible creation of undesirable “new” points (in addition to the original points) in the mesh reconstruction process.

Acknowledgements

I would like to thank my wife for her understanding and support over the period of completing this work. I especially thank Professor Patrick Marais for his support, guidance, understanding and supervision throughout. I cannot express enough my appreciation for being given this learning opportunity.

Contents

Table of Figures

Table of Tables

Chapter 1

Introduction

Cultural heritage and conservation enrich our understanding of our natural history. These can, for example, improve our perspective of the world, increase economic and socio-cultural benefit (e.g. through tourism) and may even lead to present day innovation (Otero, 2022).

The documenting of cultural heritage and conservation has evolved in the digital age, where documenting of building sites utilises electronic data capture methods such as photogrammetry and laser scanning which yields building representations in a point cloud format - a three-dimensional (3D) model with cartesian coordinate (x, y, z) points (also called vertices) that sample the surface of the modelled object.

The sample data used in this work is of ancient building heritage sites, publicly available at The Zamani Project repository and the Global Digital Heritage repository. The point cloud data is captured using LIDAR, a type of laser scanning. The output of LIDAR is a 3D set of points referred to as the raw point cloud (Börcs, Nagy & Benedek, 2017). Raw point clouds lack information of how any one point is connected to other points in the cloud (also referred to as an unstructured point cloud). The way in which points are connected is used to fit a surface to the points. Such topology can be useful for image processing but also comes at a cost of computation and speed. If the image processing can be done without the need for calculating a topology this would be an advantage.

Working with only the raw, unstructured point cloud may be of value for practitioners. The advantages would be:

- No costly meshing required - a lower requirement for computer processing power, lower storage requirements, and the possibility of achieving geometric processing goals in fewer steps i.e. not necessarily having to perform a mesh computation step.
- No potentially “spurious connectivity” introduced - meshing invariably makes assumptions about the underlying surface and these may lead to points being connected erroneously.

- Each point sample is exactly what the scanner returns - not a created sample at an estimated position, which may happen when mesh simplification techniques are used to fit or smooth the surface (the mesh). The advantage is that the original sampled surface point positions are preserved. This is an important benefit for conservation practitioners who prefer actual rather than interpolated data.

It is worth noting that the raw point cloud output of modern LIDAR systems is a high density point cloud (which we do not have access to). This high-density output is usually downsampled (the data we do have access to), preferring mesh models to approximate a closed surface on the discrete points. It is thought that there is no value in using points directly and that for mesh models fewer points are needed.

1.1. Architectural views

Historic cultural heritage buildings are documented for their study and as part of conservation work. In the digital age this documentation has moved from physical paper formats to electronic digital formats (e.g. point cloud models). A fundamental part of the documentation is to determine the architectural drawings which allow experts to better understand the building design, geometry, rooms, space utilisation and so forth.

In construction, architectural drawings form a plan for a building to be physically developed. The plan is a 2D representation of the desired building. An expert team would typically use these 2D architectural drawings to develop and construct the physical 3D building. These 2D architectural plans are represented in conventional or standard views which together provide the full information needed (such as structure, dimensions and connectivity) to appreciate the qualities of and construct the building. In historic cultural heritage such plans do not exist. Instead, the study for conservation and possible restoration of sites, requires that one reverse engineer the 2D architectural views from the 3D physical building. To construct a 3D object a minimum of three orthographic views are needed - Top, Front and Side (Lu et al., 2005). To acquire architectural views from a 3D object then, one would likewise capture Top, Front and Side views.

For the architectural drawings of buildings, one needs to consider both the exterior as well as the interior.

Standard architectural views of buildings therefore comprise cross-sections, elevations and floor plans (Yin et al., 2009).

Cross-section:

A view, orthogonal to either a horizontal or vertical plane as if cut through the building, of the orthographic projection of the building at the position of the plane. Cross-sections may also be simply referred to as sections.

Elevation:

A view of the building seen from one side and referred to as the compass direction the facade faces (e.g. North elevation)

Floor plan:

A view from above, taken from a horizontal cross-sectioning of a building.

Horizontal or vertical cross-sections of the point cloud may be derived at any user selected position, for any of Top, Front and Side views. In this work cross-sections of the building structure are recovered and examined since these have broad interest to practitioners and for applications like virtual heritage display (Banfi et al., 2019), (Pletinckx et al., 2000).

1.2. Similar work

Recovering a cross-section from a 3D point cloud has been done for the external facade of buildings by (Sui et al., 2015), however the work only recovers the external facade and excludes the internal structural layout. The work calculates vertex normal to identify points to connect by line segments. The method works well for recovering the cross-section of the external facade alone, but one needs to consider the level of complexity needed, compared to other potential methodology options, to compute, label and refine segments of normal to extract line segments to recover the floorplan.

(Kresslein et al., 2018) use a method combining skeletonization, segmentation and spline (curve) fitting to recover the shape of an object then overlaying cross-sections from the 3D point cloud onto the shape,

providing only a top view. This method does not provide a cross-section view and only depicts the top-view, external shape and points (from the 3D point cloud) depicting the cross-section at regular intervals in the shape.

The approach of slicing the 3D point cloud and recovering a cross-section is commonly used in point cloud volume estimation (Adhikary & Gurumoorthy, 2016;Li, Zhao, et al., 2019). These methods, surveyed in literature, follow an approach of slicing the point cloud, then working with the 2D slice image, fitting a curve by geometry extraction (convex or concave hull) or spline fitting techniques, to the recover the shape of the cross-section (Ramamurthy et al., 2015; Yuwen et al., 2006;G. He & Yang, 2019). The contour fitting techniques in these methods may incorrectly connect points and adds a computational cost to the method versus point-based techniques. Instead, one might adopt a point-based technique method which do not suffer these issues (the approach surveyed in Chapter 2 literature review).

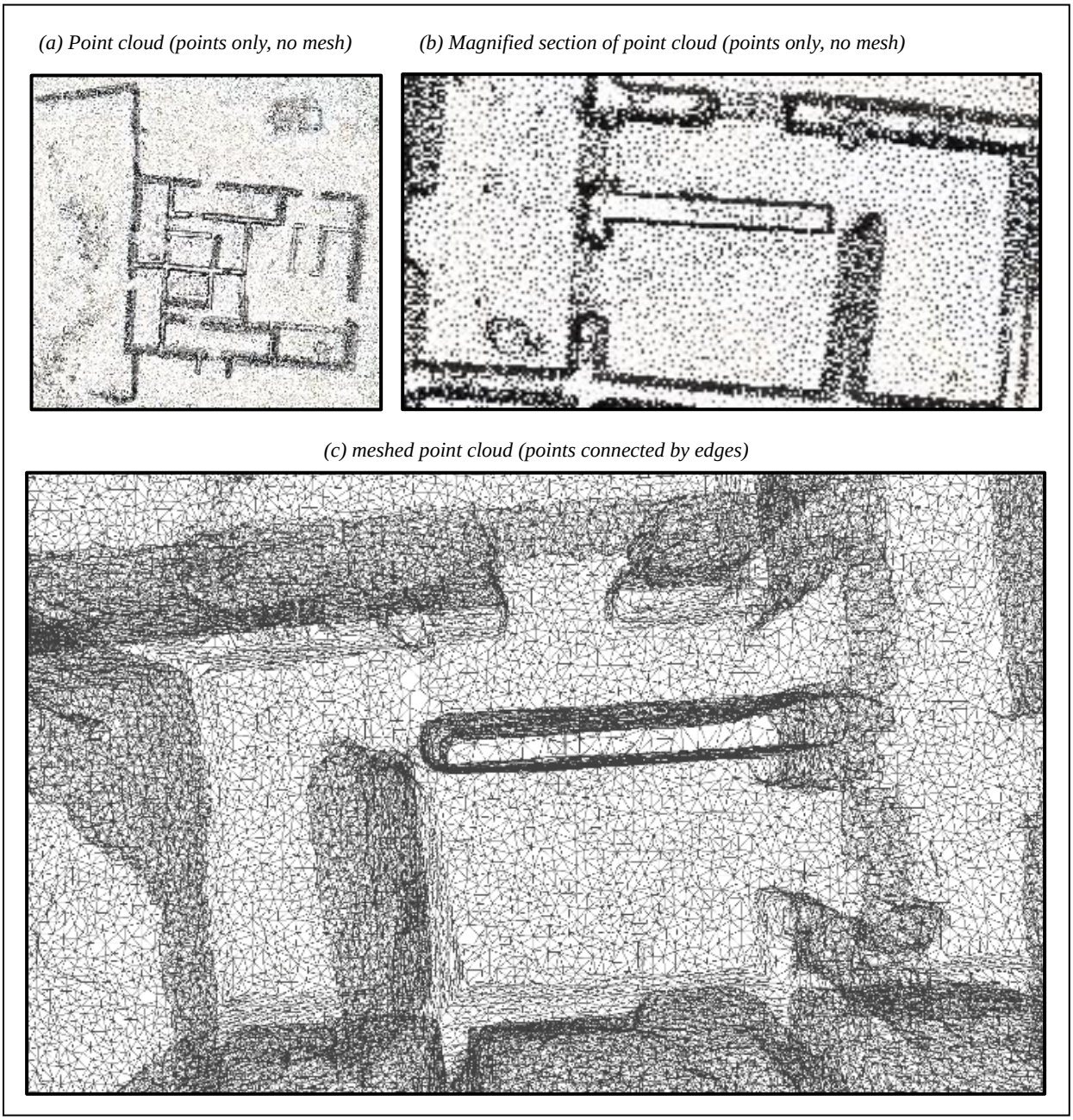
The literature review reveals a lack of tools in cultural heritage for point-based, cross-section techniques. Point-based techniques (PBTs) may achieve comparable or better cross-section recovery in cultural heritage building applications (compared to mesh-based cross-section recovery) when considering the high point density output of LIDAR scanning but also the noise introduced as part of the scanning process.

1.3. Research aims

This work investigates the extent to which the points (vertices) of point clouds, without a mesh, can be used to recover complete cross-section images of the point cloud. In doing so it is necessary to consider the impact of point density when using meshfree methods. Point density refers to the number of points within a defined local neighbourhood of the point cloud. A measure of point density is the distance between neighbouring points. The lower the point density the larger the distance between neighbouring points and conversely, the higher the point density the smaller the distance between neighbouring points.

An example of topology is the calculating of a point cloud mesh. The mesh connects each point of a point cloud to its immediately adjacent points in a polygon mesh, typically a triangle mesh. An example of a triangle mesh is shown in Figure 1.

Figure 1: Example of a grayscale point cloud showing vertices vs triangle mesh



Point clouds of buildings can comprise millions, tens of millions or even billions of points. Computing the mesh topology (edges and triangles), especially for large point clouds, is computationally expensive in terms of: (1) time to compute the mesh and (2) the computer processing resources required (CPU, RAM and storage space) (Suchde et al., 2022;Lv et al., 2022).

In general, cultural heritage projects would benefit from having a capability to recover architectural views (elevations, cross-section elevations and floor plans) from Top, Front and Side views of cultural heritage models. Working with the raw data point cloud may be of value to practitioners.

The above considerations lead to the following core research questions.

Research question 1 (RQ1): To what extent does a cross-section extracted from the vertices of a point cloud (without mesh information) match an equivalent cross-section from a mesh-based point cloud?

The early years of scanning and LIDAR produced a lower fidelity of data than is currently available. Practitioners relied on meshing to construct a surface from points of the point cloud. Given the improvements in scanning technology to produce a higher fidelity raw point cloud, for example in modern LIDAR scanning, the aim is to determine the extent to which the cross-section from a meshfree point cloud compares with the equivalent cross-section from the meshed point cloud.

Research Question 2 (RQ2): How does point density affect the fidelity of the cross-section?

In conjunction with examining research question 1, a complementary aim is to examine the impact of point density or distance between points in a local neighbourhood, on the fidelity or quality of the recovered cross-sections, given that point density is an important determinant of cross-section quality. An examination is undertaken, of the impact of point density or distance between points on the fidelity of the recovered cross-section.

1.4. Limitations

This work does not specifically address high quality point-based rendering to improve the aesthetics of the final image. Future work could address these methods for point-based rendering such as Surface Splatting or Gaussian Splatting. This work does not attempt advanced graphics techniques to resolve issues around rendering the graphics. The focus is on geometric processing rather than graphics processing. Nor does this work develop and compare multiple techniques for extracting cross-sections. The study is aimed at determining whether this is possible, rather than determining the optimal methods for constructing a cross section in this context. It is demonstrated that this can be successfully achieved.

1.5. Contributions

This work investigates the extent to which cross-sections of 3D point clouds, of architectural views of cultural heritage sites, may be recovered using mesh-free methods in computer vision.

1.6. Thesis layout

Chapter 2 lays out a survey of the literature on the most relevant methods and techniques relating to the research questions. Chapter 3 describes the methodology employed in this work followed by a presentation of the Data and Experiments conducted, in Chapter 4. Chapter 5 shares the results and discussion where the outcomes of the experiments are evaluated, followed lastly by our conclusions in Chapter 6.

Chapter 2

Background and related work

This chapter briefly introduces important foundational concepts in the domains of 3D point cloud image applications and processing, 2D image processing and geometry (shape) processing within the broader domains of computer vision and image processing. This is followed by a survey of related work.

2.1. Background

Virtual cultural heritage technologies afford a more accessible, immersive, 'life-like' experience and may attract a greater audience compared to use of simpler, traditional models alone. In this section a background of relevant digital technology for cultural heritage ensues before delving into pertinent techniques for achieving the research aims.

2.1.1. 3D representations of heritage data sets

Virtual heritage display is of interest to conservation practitioners, especially when considering their application in immersive technologies like Virtual reality or Augmented reality (Banfi et al., 2019; Pletinckx et al., 2000). Innovation in the digital representation of heritage data sets may stimulate increasing interest from a target audience, and the digital format makes for life-like, more accessible models.

Point clouds have been widely adopted for 3D data representation in fields such as construction, robotics, autonomous vehicles, medical fields, land surveying, cultural heritage and so forth. (Yang et al., 2022) survey the use of 3D point clouds for cultural heritage over the last two decades, noting the evolution and growing use across research disciplines, across several countries that lead research in cultural heritage. This scientometric study is offered as evidence of advancing a need for higher requirements and challenges in future such research, implying that improvement in point cloud techniques is of value and to be expected.

2.1.2. Methods to generate 3D representations of objects

The two primary methods for creating 3D point cloud models are Photogrammetry and Light Imaging Detection and Ranging (LIDAR), where the typical light source for buildings and cultural heritage sites is laser.

2.1.2.1. Photogrammetry and LIDAR

Photogrammetry is the use of optical sensors to process and combine a series of digital photos (which could be the series of frames from video media) to form a 3D model (Baltsavias, 1999). A high-resolution digital model can be achieved in photogrammetry, even from photosets taken with a consumer-grade digital camera (Westoby et al., 2012), for a cost-effective approach to 3D modelling. A photoset capturing a scene from multiple angles is processed to form a point cloud of the 3D scene. Being a visible light spectrum tool however means that results can be affected by low light conditions and shadows.

LIDAR directs light pulses at an object and receives the return pulse, in a way designed to calculate distance and position per point, in a scan which could comprises millions of points over the entire surface of the object being scanned. LIDAR has evolved to use a laser source which affords the measurement of small distances on the object (1 to 2 centimetres). In this way the entire surface of an object can be captured as a 3D point cloud image (Vierling et al., 2008). Laser has become the popular light source in LIDAR, able to capture fine-grain information of a scene or object in a 3D point cloud model. Benefits of LIDAR data capture are listed as: being high speed and high accuracy, non-contact and yielding completeness of information (Qiu, 2022).

The modelling of a 3D object can be accomplished by Photogrammetry alone, LIDAR alone or a combination of both LIDAR and Photogrammetry (Hermosilla et al., 2011). Laser-based LIDAR is used for capturing the point cloud data sets examined in this work, from a Terrestrial LIDAR Scan (TLS) platform. “Terrestrial” describes a ground-based scanning equipment (in comparison to satellite or aircraft-based LIDAR platforms).

2.1.2.2 3D Data structures - raw point clouds, voxels, meshes

The output of either photogrammetry or LIDAR is a 'raw' 3D model of the scanned object - where objects can be cultural heritage sites, ancient structures (e.g. castles), urban buildings, roads or pavements, vegetation or any small object. The raw 3D point cloud consists of points with x, y, z coordinates in a cartesian coordinate system, and can comprise millions or even billions of points representing the outer and or inner surfaces of an object.

Raw Point clouds

A raw point cloud (also called an unstructured point cloud) has no topological information (Y. Xu et al., 2021) but may include vertex colour and size attributes. In this research the point cloud is assumed to be unstructured, meaning that points are not stored in any particular order, with no information about connectedness to neighbouring points. Typical point cloud data formats are binary PLY or OBJ files (these are therefore the two file formats supported in the software developed for this research).

Point clouds, and improvements in the capturing thereof, offer benefits to conservation such as digital modelling for advanced analysis, documenting, visualization, capturing shape and appearance (Balzani et al., 2017).

Voxels

If a 3D grid is overlaid onto the point cloud, then each cell which has at least one point of the cloud in it is a voxel. A voxel is a 3D analogue of what a pixel is for 2D images. Structuring the point cloud as voxels can be useful, making some processing algorithms easier, but comes at a great cost in terms of storage and processing time. For this reason, it is often preferred, where geometric accuracy and speed are important, to avoid "voxelising" the point cloud.

Meshes

A 3D mesh model approximates a surface from the vertices of a point cloud. Two vertices are connected by a straight line, called an edge. Edges connect two points which are immediate neighbours. In some surface reconstruction algorithms it may be that new points are added, interpolated between the original points, for a smoother surface. The surface is made up of polygon faces (typically triangle polygons),

which is a closed set of edges (3 edges form a triangle face). A high polygon count mesh correlates to an image that is more photo-realistic than a low polygon mesh but also requires greater computational effort in image processing.

Meshing methods were derived for the purpose of approximating a surface, connecting the vertices of the point cloud, referred to as mesh modelling or surface reconstruction (Lin et al., 2004; Wang et al., 2011). Fitting a mesh to the points involves a further stage of computation after generating the raw point cloud. The closer the fit through a maximum number of vertices, the better the surface reconstruction. An early mesh modelling technique, Delaunay Triangulation (Boissonnat & Ghosh, 2010), has been improved on, both in terms of time and the quality of fit, by methods such as Poisson surface reconstruction (Kazhdan & Hoppe, 2013).

An unstructured point cloud does not have information about how neighbouring points connect to one another to form a surface. The mesh is an estimation of the connectivity of points (called edges), forming a surface.

2.1.3. Point-based Techniques (PBTs)

The 3D point cloud as a representation for objects has increasingly gained attention and become established in computational methods for computer vision over the last two decades (Kobbelt & Botsch, 2004; Sainz & Pajarola, 2004;Han et al., 2017). Point-based techniques (“meshfree” techniques) is a class of computational methods which use the point primitive of the point cloud to process the cloud.

A survey of the literature reveals techniques for slicing and extracting cross-sections, for both point clouds as well as mesh models. Point cloud methods and Point-based Techniques (PBTs) are the areas of interest.

There are several applications that adopt slicing and extracting cross-sections from meshed point clouds, such as: Feature Extraction (Zhi et al., 2016;Adhikary & Gurumoorthy, 2016), Cross-section Shape Extraction (Kresslein et al., 2018) and Volume Estimation (Zhang et al., 2024).

Meshing adds to computational cost. Improving this computational cost is the subject of research. For example, (Minetto et al., 2017) contrasts algorithms, for optimisation, to improve the computational cost, time or quality of methods for slicing meshed models. With regards to computational cost, note however that improving the cost of meshing would always be second-best to methods which eliminate the need for meshing altogether.

The literature holds fewer methods using PBTs than for mesh-based methods, for slicing and extracting cross-sections. A common approach is to first slice the point cloud orthogonally then draw a closed curve fitted over the boundary points, forming a cross-section profile from the 2D points (Kyriazis et al., 2009;Kyriazis & Fudos, 2013;Sui et al., 2015;Kokab & Urbanic, 2019; Li, Wei, et al., 2019; G. He & Yang, 2019). The variations within this approach are on the type of boundary curve estimated - convex hull, concave hull or spline fitting.

Over time point-based methods have improved and evolved. Currently there is interest in both point-based as well as mesh-based techniques, depending on the field of use. For applications where the extra computation effort for calculating a mesh is undesirable, one would likely prefer to work with the points directly.

In this work the aim is to examine the extent to which PBTs can be used for deriving cross-sections from the point cloud. Methodologies considered are those applied in computer graphics for shape modelling, particularly techniques related to deriving cross sections from a 3D point cloud, pertinent to point clouds of buildings and cultural heritage sites.

2.1.4. Image processing

Image processing techniques are used to improve the suitability of an image or form part of a set of techniques to improve the quality of the image processing outcome (Coady et al., 2019).

2.1.4.1. Filtering

One of the most common tasks in image processing is filtering. A Filter is applied by convolving a source image with a filter mask (kernel), to achieve processing objectives such as Edge detection, Noise

reduction, Smoothing, Sharpening, Up/Down-sampling (increasing/decreasing image resolution) and so forth. These filtering techniques, both linear and non-linear, are used in such methods as noise reduction, segmentation and feature extraction. Filtering may be the core image processing technique or a pre-processing step, used to prepare an image for other processing techniques to be applied e.g. anti-aliasing, a smoothing filter to reduce jagged edges in an image.

2.1.4.1. Fitting a curve

A trivial curve fitting would be drawing a line connecting one point to its nearest neighbouring point until a closed loop is formed, however this could lead to many false positive connections. A way to minimise such errors would be to employ a geometry bounding technique such as a convex or concave hull algorithm (Barber et al., 1996;Moreira & Santos, 2007; Asaeedi et al., 2017). The computation cost and complexity of reducing false positive connections remains, even with improved methods for drawing a high fidelity boundary curve around the 2D cross-section of points (Mineo et al., 2019).

Binary Morphology offers a method which does not require calculation nor fitting of a curve. Binary morphology methods can avoid the complexity of drawing curves to form a closed boundary, forming the cross-section profile, while preserving the inherent geometry of the 2D point cloud cross-section.

2.1.4.2 Binary morphology

Mathematical morphology is a type of non-linear filtering techniques used in Computer Vision for image processing. These techniques have been around since at least the late 1980s and are still of value in image processing today. See Figures 11 (ii)(c) and 11(iii).

Mathematical morphology is based on set-theory, geometry and concepts in topology (Haralick et al., 1987; Heijmans, 1995). Initial research incorporating mathematical morphology was in characterising physical properties of certain materials. The general concept is applied in processing the geometric structure or shape of an image. This is considered a useful technique for this work. The morphology methods can be performed on colour, grayscale or binary images. Depending on the image processing needs, colour or grayscale morphology may be used or alternatively a colour image may be converted to grayscale or binary. In this work a cross section of a cloud can be converted to a binary image to enable the adoption of binary morphology in our methodology, a common method used to sharpen or

improve boundaries in images, by either removing connected pixels or connecting disconnected pixels within a window of pixels.

Typical operations in binary morphology are Dilation, Erosion, Closing and Opening (Haralick et al., 1987; Heijmans, 1995;Amer, 2002).

Binary morphology techniques result in the filtering or modification of one or more pixels in a neighbourhood. Where this modification is to be performed over the entire image, the operation uses a second image, called the structuring element, which size is the set to the size of the neighbourhood to be modified. The modification produces a third image which is the binary transformation of the original (source) image and the structuring element, to produce the transformed (output) image.

The characteristics of the structuring element, size and shape, play a role in obtaining the desired effect from the transformation. The smallest structuring element can be two pixels in size. This may be useful where the transformation need not be symmetrical. The smallest structuring element for a symmetrical transformation is of size 3x3 pixels (rows=3, columns=3). A pixel in the structuring element is defined as the anchor point, typically the centre pixel in a symmetrical structuring element. The anchor is placed over each pixel in the source image and the transformation then applied. Structuring elements may also be shaped to achieve the desired transformation, and can be square, rectangular, circular, elliptical or custom shaped.

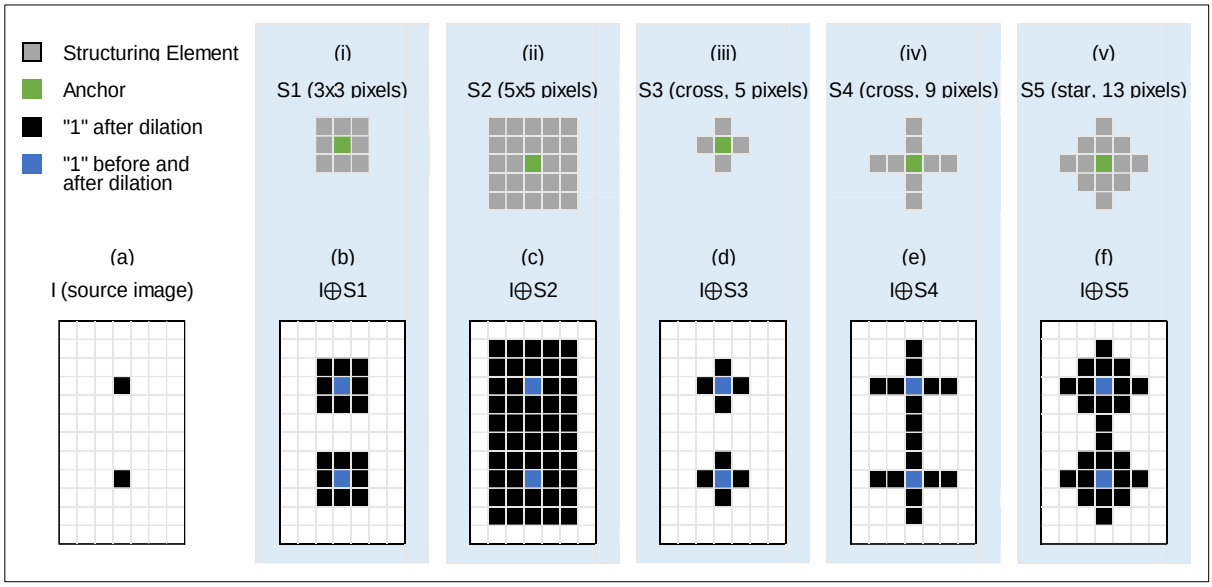
Dilation (usually depicted by the symbol $\oplus$) expands the connected set of ‘1’s from the source image, see Figure 2 depicting the effect of dilation. The source image is transformed by overlaying the anchor of the structuring element over each pixel in the source image. When the anchor encounters a ‘1’ the dilation then makes every pixel which the structuring element overlays a ‘1’.

Erosion (usually depicted by the symbol $\ominus$) reduces the connected set of ‘1’s in the binary source image. Where the anchor of the structuring element encounters a ‘1’ in the source image:

(i) if all other pixels of the structuring element overlay onto a ‘1’ in the source image, then the transformation keeps the ‘1’.

(ii) if any of the other pixels of the structuring element overlay onto a '0' in the source image, then the transformation changes the '1' of the source image to a '0' i.e. eroding the pixel.

Figure 2: Dilation: Source image (a), structuring element (i to v) and respective transformed image (b to f)



The binary morphology description above may be described by the following equations (Nadadur & Haralick, 2000).

Let Z = {z | 0 <= 0 <= ∞} be the set of integers. Let A, B, C and K be sets in Z^2 and let O be the origin of Z^2 ($O \in Z^2$)

Binary Erosion of a set A by a structuring element K is denoted by $A \ominus K$ and is defined as:

$$A \ominus K = \{x \in Z^2 \mid x + b \in A \mid \text{for every } b \in A\}$$

Binary Dilation of a set A by a structuring element K is denoted by $A \oplus K$ and is defined as:

$$A \oplus K = \{x \in Z^2 \mid x = a + b, \text{ for some } a \in A \text{ and } b \in K\}$$

Closing is performing Dilation (of order N) followed by Erosion (of order N). Opening is performing Erosion (of order N) followed by Dilation (of order N). Where N refers to the number of recursive Erosions/Dilations, N is an integer and N > 0.

As an example of Dilation, Figure 2 (b) to (f) is the output of dilating the source image Figure 2 (a) with the respective structuring elements Figure 2 (i) to (v). For an example of Erosion, consider each image in Figure 2 (b) to (f) as the input image which, when eroded by the respective structuring element Figure 2 (i) to (v), results in the source image Figure 2 (a).

Dilation can be used to remove gaps in an image, for example by filling in missing pixels. Erosion can be used to reduce salt and pepper noise. A combination of Dilation and Erosion, called Closing, can be used to fill in missing pixels and return the image to its initial state with the missing pixels now filled in (Closed). The size of the image is conserved (although small distortions are possible). The closing transformation fills gaps smaller than the structuring element size.

2.1.5. Views of 3D models

There are two main parts to the process of generating a 2D or 3D image using a computer programme - Modelling and Rendering (McReynolds & Blythe, 2005).

A model is created, which sets up the object of the scene for the graphics processing system. The model depicts the shape and appearance of the object. Rendering takes the model as input and generates pixel values for the display (e.g. computer screen).

OpenGL is the most widely used open-source graphics standard (https://developer.nvidia.com/opengl) and is primarily used to render objects. A rendering of a point cloud image can be seen in Figures 11 (i). The actual model data and data structures are set up in the software application. The model is rendered and drawn specifying point, line or polygon (usually triangle) primitives. Each vertex is described by an x, y, z position and may also contain colour, intensity and normal information. OpenGL is designed to accept data structures from the application, representing position, colour, intensity, point size and triangle faces. In this way a 3D point cloud object is rendered, fully depicting the scene. Note: A rendering pipeline may also contain textures and light position.

2.1.6. Camera view: Object Space and Model View Projection transforms

The vertex position values, as they exist in the application, are said to be in Object Space (or Local Space), without the context of positioning in World Space coordinates.

The Model transform (M) translates, rotates and/or scales the object to place it in World space. Applying a Viewing transform (V) after applying M, places the object in Eye space, also called Camera Space.

The Projection transform (P) specifies the range of vertex coordinates in each of x, y and z dimensions. This transform is then applied to convert from View Space to Clip Space. The Projection transform defines a ‘box’, called a Frustum, wherein each vertex coordinate is transformed into the range -1.0 to +1.0. Any transformed coordinate outside of the -1.0 to +1.0 range is clipped (not displayed)

The camera position can therefore be set according to which view of the point cloud (Top, Front, Side) the user desires. To view the entire object the camera position must be set at a position outside of the object coordinates. The user may traverse through the inside of the object by positioning the camera within the object coordinates.

2.1.7. Evaluation

The evaluation of ‘similarity between two arbitrary images’ is a subset of measuring the similarity between two fuzzy sets. Such mathematical tools for the evaluation may be found in Set Theory. Besides, classical quality measures such as Mean Square Error (MSE) or Peak Signal to Noise Ratio (PSNR) do not perform well as measures of geometric image similarity (Van Der Weken et al., 2004), sometimes producing the same value for very different images.

Veltkamp & Hagedoorn (2001), describe the state of the art in geometric shape similarity measurement, between two arbitrary 2D images, as a measure of the area of overlap. One such metric, using Set Theory, is the Intersection over Union (IoU) (Choi et al., 2019; J. Xu et al., 2019).

The IoU of two images A and B is calculated as the ratio of the intersection (overlap) of A and B and the union of A and B (the total of the area of overlap and non-overlap), given as the equation:

$$\text{IoU} = \frac{\text{Area of } (A \cup B)}{\text{Area of } (A \cup B)}$$

A perfect match of both images would produce an IoU score of 100%. A score of 0% indicates no overlap. The IoU, calculated after registering the two cross-section images, provides a way to evaluate the methodology. The metric suits 2D, binary, registered images although variations of the IoU may be applied to 3D, greyscale and colour image comparisons.

2.2. Developing a method for PBT cross-section recovery

The method implemented for recovering cross-sections combines relevant PBTs from the literature in 3 main steps:

- Slicing the point cloud - with consideration to slice thickness
- Forming a closed cross-section profile or boundary from the discrete points of the 2D slice
- Addressing holes after forming the cross-section boundary

In the literature, cross-section methods have found use in applications such as 3D printing (Oropallo et al., 2018), medical imaging (Kokab & Urbanic, 2019), feature extraction (Kyriazis et al., 2009) and so forth. However, compared to our data sets these methods use small models such as objects for 3D printing, human organs of the body or the head of a sculpture. In contrast the 3D point cloud data from cultural heritage sites are noisy and complicated with complex boundaries.

2.2.1. Slicing the point cloud

The actual slicing of the point cloud is an operational task however choosing the slice thickness needs consideration. The literature describes choosing the slice thickness as either a fixed or variable value. A slice of a point cloud can be seen in Figures 11 (ii) (b).

Wu et al. (2004) propose adaptive slicing for processing point cloud data for reverse engineering 3D printing applications, where it is desirable to minimise the number of slices. The adaptive thickness is determined as a function of: (i) maximum distance between the points of the point cloud, (ii) user defined feature points and (iii) the desired thickness for the application. Advancement in point-based

methodologies and the supporting technology (e.g. improvements in processing power and storage costs) is available in mainstream computing and is continuously improving. This supports more complex algorithms should adaptive slicing be necessary for the application.

Alternatively, a constant slice thickness is proposed by (Oropallo et al., 2018), also for the application of 3D printing. Their research aim being to eliminate “error prone algorithms” as part of their work of revisiting point cloud slicing. A constant slice thickness seems suitable for the purposes of examining the impact of point density on the quality of cross-section profiles.

2.2.2. Forming a closed boundary from discrete points of the 2D slice

A slice of the 3D point cloud yields a 2D set of discrete points (see Figure 11 (ii) (c)), for which 2D image processing techniques may be employed. A closed 2D surface is required. The next step then is to form a profile or closed boundary of the cross-section from the discrete points of the point cloud slice.

One approach from the literature is to draw a curve through the 2D point cloud which best fits the points. Curve fitting of boundary points add a layer of effort and complexity versus using the points directly, and may connect points incorrectly (Kyriazis et al., 2009; Yuwen et al., 2006). Spline fitting interpolates a curve when connecting a cluster of points and may fit part of the curve between rather than through the original points. Concave hull curve fitting may miss convex boundary connections and convex hull fitting may miss concave connections.

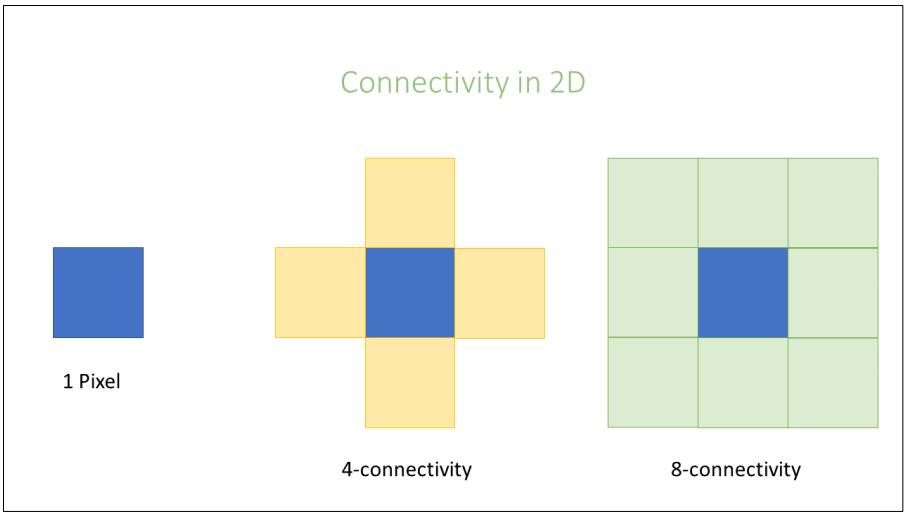
Binary morphology can be used to close the gap between any two adjacent points in a point cloud slice, provided that the gap is smaller in size than the structuring element used. The technique offers a way to form a closed boundary from the discrete points of the 2D slice (Amer, 2002; Chen & Haralick, 1995; Snyder, 1992) without interpolating ‘new’ points. The challenge with binary morphology is in minimise the merging and undesirable connecting of fine detail in the image. One way to minimise such an error is to set the structuring element to a minimum size. Loss of small detail in recovering cross-sections from cultural heritage sites may or may not be acceptable but in general this should be avoided if possible.

2.2.3. Addressing holes after forming the cross-section boundary

A suitable technique for filling holes is flood-filling (Burtsev & Kuzmin, 1993; Fishkin & Barsky, 1984; Khudeev, 2005). The technique requires either a binary mask to specify the region to fill or a seed point, which is the starting point for filling, ending at the first closed boundary encountered, enclosing the seed point.

To perform flood-filling, first identify holes in the cross-section being formed. This can be done using a Connected Components Analysis to identify and label the holes in the image (Bolelli et al., 2020; L. He et al., 2017; Di Stefano & Bulgarelli, 1999). The analysis uses a rule (or pattern) to determine what a connected component is - typically 4 connected or 8 connected (see Figure 3). The output of the connected component analysis is sufficient to complete the flood-filling, where the centroid of the hole is used as the seed (blue pixel in Figure 3) for the flood-fill. Note that the seed point can be any point within the hole.

Figure 3: Connected Component Analysis – "4-connected" and "8-connected"



In this chapter, a motivation for the use of point clouds as a 3D data representation for cultural heritage is identified. The literature was surveyed to better understand the background to point cloud models and related image processing tasks, pertinent to the research aims. This informed the method for this work which combines image filtering techniques: binary morphology, connected component analysis and flood-filling.

Chapter 3

Methodology

The research aims are examined by applying both quantitative methods and qualitative methods. A target application for this methodology is in the conservation of ancient buildings. The methodology recovers cross-sections from several cultural heritage point cloud models, employing Point-based Techniques (PBTs). These are compared to the equivalent cross-sections from a mesh-based model for comparison.

The sample data from which cross-sections are recovered is of ancient building heritage sites captured as point cloud by the Zamani project and the Global Digital Heritage project.

Problem statement

This work examines the extent to which using the points directly may produce a cross-section of high fidelity compared to an equivalent cross-section from the mesh-based model. Further, the impact of point density on the fidelity of the PBT cross-section is examined.

3.1. Design principles and constraints

The PBT methodology developed for this work, based on the literature, combines techniques into one method, incorporating: slicing the point cloud, developing the cross-section boundary, filling the cross-section and post-processing.

Slicing the point cloud

In this work a constant thickness is preferred. This will allow the effect of point density to be examined, as a factor in the extent to which a point-based approach is effective.

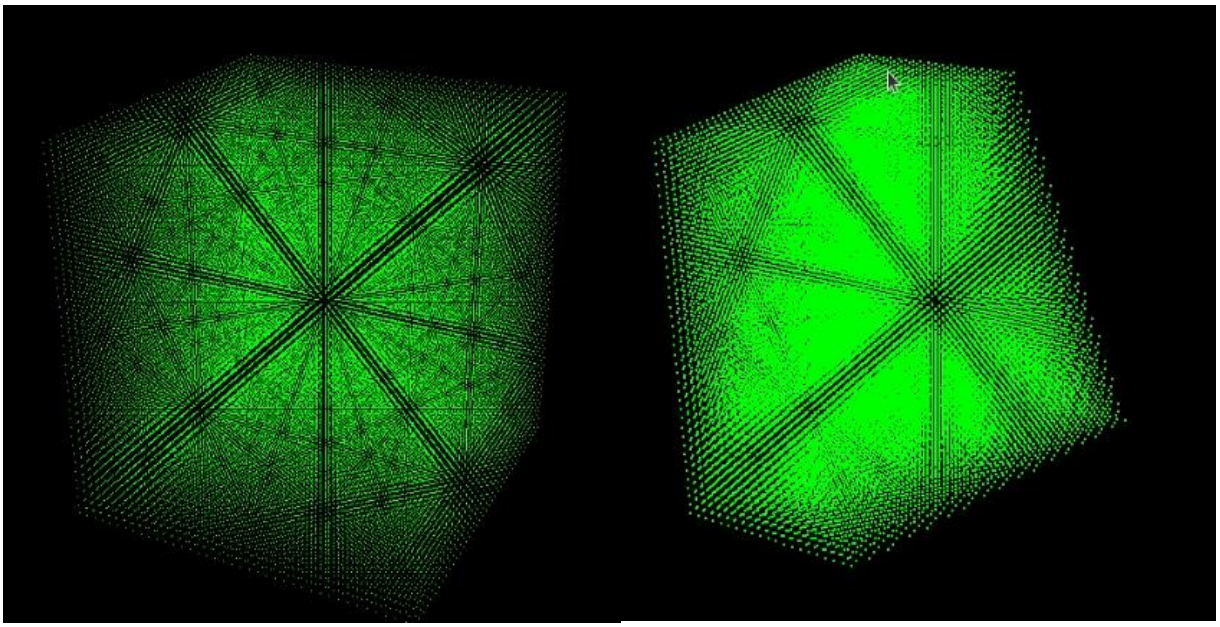
The thickness of the slice should be large enough to capture enough points (projected onto the slicing plane, in orthographic projection) to develop the boundary of the 2D slice, yet small enough to capture only the desired detail, avoiding detail outside the practitioners' interest.

Perspective division, dividing the x, y and z coordinates by the w coordinate, transforms 4D Clip Space coordinates to 3D normalized device coordinates. By setting the w component to 0 that point is then 'clipped' or hidden. This is one way in which a slice of the point cloud can be taken. Points lying outside or behind a slice plane (traversed by the user through the cloud), can be clipped by setting the w component to 0 (see Figure 4 below), or alternatively, setting the alpha (intensity) component in the RGBA value to 0 (McReynolds & Blythe, 2005).

Figure 4: Example of clipping the point cloud

(a) Point cloud - whole cube

(b) Point cloud - right side of cube clipped ("w" coordinate value set to 0)



Developing the cross-section boundary

Binary Morphology is adopted to close the gaps between the discrete points of the slice (to form a closed cross-section boundary), as opposed to connecting boundary points by curve fitting (convex/concave hull or spline fitting techniques), offers efficiency over complexity and a better fit to the points. Curve fitting requires a higher complexity of mathematical computation, may not connect points correctly and may interpolate a curve between points which does not reflect the structure as accurately as using the points directly. Binary Morphology may be applied recursively (a series of dilations followed by an equal number of erosions. A single (non-recursive) Closing may also be used to reduce or eliminate noise or spurious points (from background) in the cross-section image.

Filling holes after developing the cross-section boundary

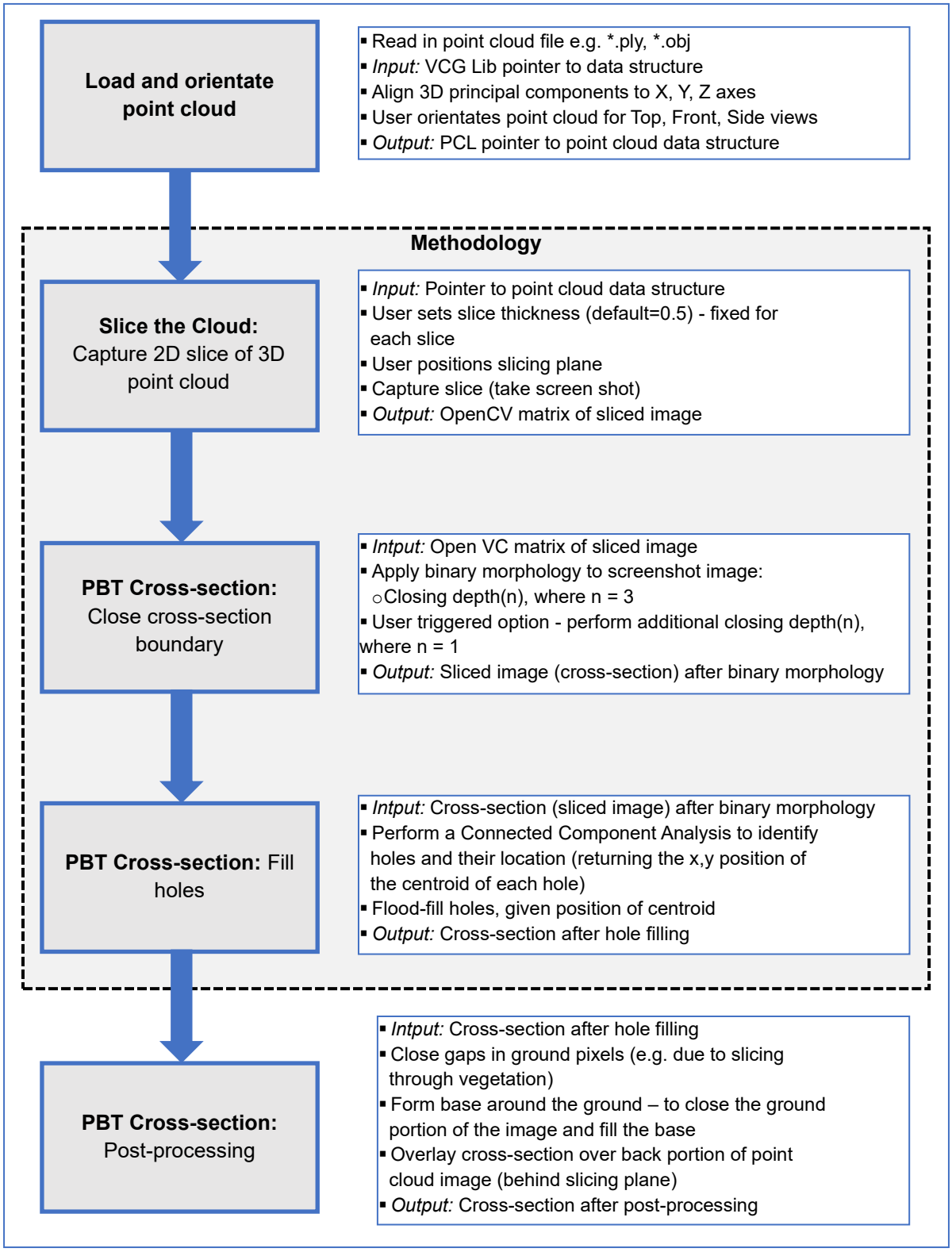
Once a 2D cross-section closed boundary is formed, further processing to fill possible holes in the 2D cross-section profile could be needed. Some holes may not be closed by the previous binary morphology step. In particular, the inner cavity of a wall, between the interior and exterior laser scan of the wall, will likely contain holes. The cross-section slices through the wall, revealing wall cavities which form part of the cross-section being depicted.

Filling holes requires a prior identification of the hole positions. This is achieved by applying a Connected Component Analysis.

3.2. Solution overview

The system architecture is shown in Figure 5, followed by the pseudo code of the methodology. The key steps in the methodology are described for a better understand of the PBT methodology. The solution pseudocode is included in Appendix 1: Pseudocode.

Figure 5: Methodology - system architecture



3.3. Implementation

The research aims are examined in two parts. First, by recovering cross-sections using our point-based methodology and evaluating these against their equivalent from the mesh-based model. Second, by examining the impact of point density on the fidelity of the PBT cross-sections. Cross-sections are recovered from the point cloud sample data sets by applying the point-based methodology software developed for this work. The cross-section recovered is then compared to the equivalent cross-sections recovered from the meshed point cloud in Meshlab (https://www.meshlab.net/).

In the first part of our experimentation, each cross-section comparison is evaluated. The comparison yields a score (a percentage), measuring the extent to which the cross-sections match. A score of 0% indicates no match at all and 100% indicates a perfect match. For the comparison the “Intersection over Union” (IoU) metric is calculated. Variable elements in the point-based methodology (slice thickness: t, and structuring element size: K) are set and varied to better examine their impact.

In the second part of our experimentation, point density is examined to determine the extent to which point density impacts the ability to recover a high fidelity cross-section. In the case of a cross-section evaluation initially scoring an IoU less than 90%, the point density is then increased (densified). For each densified point cloud, the cross-section is recovered and compared with the equivalent mesh-based cross-section. Variable elements in the point-based methodology are kept constant (e.g. slice thickness, structuring element size used in binary morphology) to examine the effect of point density. If our cross-section method breaks down (the cross-section profile is not formed), then a qualitative analysis of the cross-section is done.

Cross-sections from the point clouds of 4 different cultural heritage sites are recovered and compared with their meshed equivalent. The experiments are described in Chapter 4 *Data and experiments*.

3.4. Method for recovering cross-section

Based on the literature review, the method comprising our point-based methodology is:

(i) orient the 3D point cloud,

(ii) slice the cloud - the slice is treated as a 2D point cloud projected onto the slicing plane,

(iii) apply the most appropriate techniques from the literature to process the 2D slice into a solid cross-section - Binary Morphology, Flood-filling, Connected Component Analysis,

(iv) post-processing as needed, and

(v) render the completed cross-section overlaid onto the portion of the 3Dpoint cloud behind the slice.

An overview of each step in the method is shared below. The keyboard user input keys for recovering a cross-section are available in Appendix 2.

3.4.1. Orient the 3D point cloud

The software reads in and orients the point cloud to present the desired architectural views (cross-section elevations and floor plan - Top, Front and Side views). Choosing an appropriate or 'good' view and slice position is not always obvious, so these are left to the expertise of the user and is not automated.

The camera setup determines how the scene is rendered on screen to the user. The camera is set to provide an Orthographic projection, meaning there is no perspective of distance from the camera and the rendered scene appears the same size no matter how close or far away the camera is positioned. The alternative is Perspective projection which would not provide a true scale of the scene dimensions as this projection is reflects a perception of depth in 2D (like railway tracks "merging" in the distance).

A Principal Component Analysis (PCA) is available in the software, to determine the primary direction for each dimension, along which the data points (vertices of the point cloud) lie. The PCA provides the information to orient the cloud for each dimension so that the user sees the scene rendered orthogonally to the camera.

3.4.2. Slice the cloud

The user can traverse an imaginary slicing plane through the point cloud. The slice plane thickness is set to a fixed value for each slice. The slice thickness (t) can be adjusted by the user, between 0.1

(10cm) and 1.5 (150cm), before a slice is captured. The user positions the slicing plane according to the desired cross-section to be examined. The cross-section is captured and processed to form a closed boundary, after which this cross-section profile is then filled.

3.4.3. Developing the point-based cross-section recovery method in software

The method implemented in software for recovering a cross-section comprises algorithms to perform these point-based techniques.

(i) Binary morphology - to form the cross-section profile or boundary from the discrete points of the slice.

(ii) Flood-fill - to fill 'holes' after forming the cross-section boundary.

(iii) Connected Component Analysis - to identify holes in the cross-section profile and determine the coordinates of the holes to be filled.

The Binary Morphology, Connected Components and Flood-filling algorithms are available as API calls in OpenCV (https://opencv.org). The OpenCV flood-filling and Connected Component Analysis algorithm is developed by Bolelli et al., 2020.

Figure 6: Overview of software functionality

(i) Orientate the 3D point cloud
Calculate bounding box
Translate point cloud to origin
Orient point cloud for Top View (default view)
Render 2 views:
(i) Contextual Perspective View and
(ii) Slice Plane Orthographic View
(ii) Slice the cloud (at a user selected point)
User traverses slicing plane through cloud
Position slice plane for cross-section
Capture 2D point cloud cross-section
(iii) Apply PBTs to recover cross-section
Recover boundaries and contours
Binary morphology - recursive dilation and erosion
Fill holes
Identify holes and point within the hole
Connected Component Analysis
Flood-fill
Post processing
(iv) Overlay completed cross-section onto back-section of sliced point cloud
Merge cross-section onto point cloud
Render cross-section and remaining portion of point cloud
(v) Evaluation
Recover cross-section from PBTs software (this work)
Recover equivalent cross-section from meshhed point cloud
Perform cross-section comaprison (Intersection over Union)

The key software modules (Orientation, Methodology and Evaluation) are shown in Figure 6.

3.4.4. Post-processing

A post-processing stage addresses any further image processing needed to complete the cross section. Available features are:

- De-noising - Erosion of depth n=1, where n refers to the number of recursive Erosions, n is an integer and n>0.
- Closing horizontal gaps in the ground plane e.g. a slicing through vegetation on the ground may create gaps in the cross-section which must be addressed for 'filling' (in the next step) to work.
- Bounding and filling in the ground plane.

These features allow the cross-section profile to be closed and holes to be filled for the construction of the completed cross-section.

3.4.5. Overlay and render the cross-section

The recovered cross-section needs to be displayed in the context of the point cloud. The portion of the point cloud behind the slice plane is displayed with the cross-section overlaid.

The point cloud vertices are rendered, meshfree, as points. A perspective view highlighting the position of the slice plane is displayed alongside the orthographic cross-section view.

3.5. System and software

The techniques to recover a cross-section from the 3D point cloud was implemented on computer with Ubuntu Linux 23.04 OS, an Intel i7-11700 CPU, 16GB RAM, 2.5(min)/4.9(max)GHz 16 core CPU and NVIDIA GeForce GTX 670 graphics card. No multi-threading was employed in the software.

Implementation was done using the following libraries:

(i) Visual Computing Graphics (VCG) Library (https://vcglib.net) - to read in the point cloud vertices, triangle faces, colour values (RGBA) and normal per vertex.

(ii) Point cloud Library (PCL) - for principal component analysis, calculating eigenvectors for use in orientating the point cloud.

(iii) OpenGL - to render the point cloud to screen

The code was written in C++ and compiled with GCC using a makefile.

3.6. Evaluation

The evaluation of our methodology requires an analysis of the experiment results - comparing the output of the point-based methodology against the equivalent mesh-based cross-section. The approach taken is to first register the two images (overlay and align them to achieve a best effort fit) then calculate the Intersection over Union (IoU) metric (see section 2.1.7 Evaluation*).*

To evaluate the extent of “fit” between the point-based and mesh-based cross-sections, the registration (alignment) of the cross-sections is done manually, using scaling as necessary (no rotation or skewing of the images is permitted) in a graphics processing app, GIMP (www.gimp.org). After registration, the Intersection over Union (IoU) is calculated, based on the following definitions:

Intersection:

Number of overlapping pixels for both point-based cross-section and mesh-based cross-section (indicated in green in the IoU evaluation images, see Figure 7).

False positive (FP):

Number of pixels in the point-based cross-section, which are not in the mesh-based cross-section (indicated in red in IoU evaluation images, after each table of results below)

True negative (TN)

Number of pixels in the Mesh-based cross-section, which are not in the point-based cross-section (indicated in blue in IoU evaluation images after each table of results below).

Intersection over Union

IoU = Intersection / (FP + TN + Intersection)

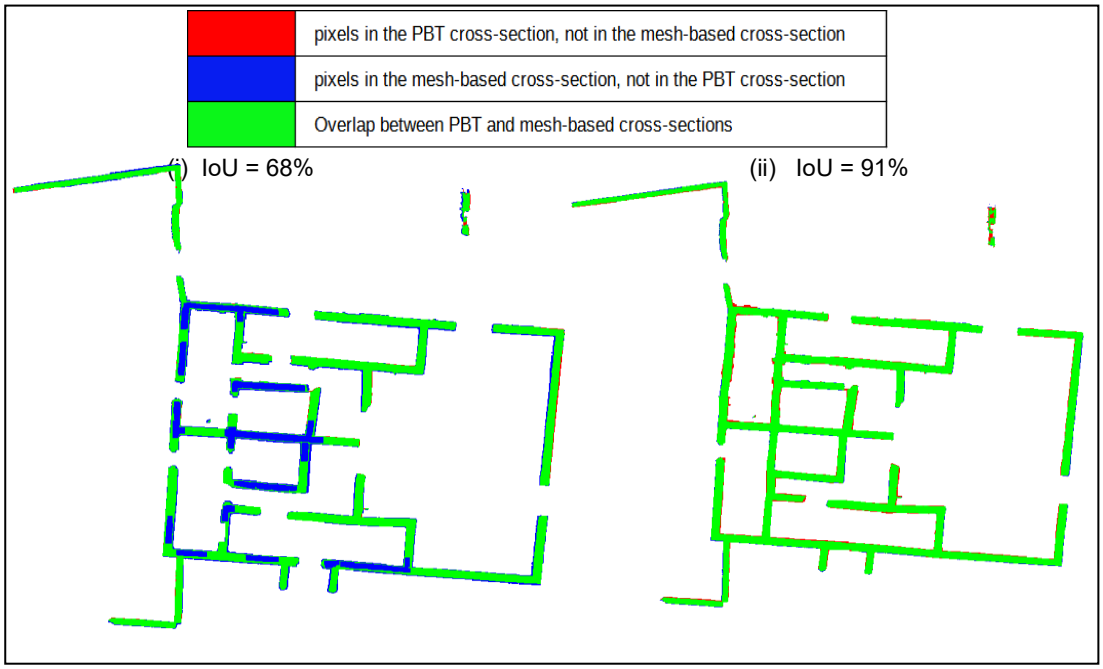
No prior work is found in the literature to compare our PBT methodology against, therefore evaluating the point-based cross-sections is undertaken by utilising the IoU metric and impart a meaning to the range of these IoU scores. Applying a qualitative visual inspection of the results (“fit” between point-

based and mesh-based cross-sections) and the corresponding IoU score, the following interpretation for evaluation is used:

- an IoU score of 90% and above is taken as a validation of our PBT methodology.
- An IoU score of between 80 and 90% is considered is “promising” but needs improvement - either improving on developing the cross-section profile or removing sources of error in post-processing, for example, small clusters of pixels which are not part of the cross-section targeted by the user, but which the slice plane captures in the slice.
- An IoU core of less than 80% is considered an indication that our PBT methodology is inadequate compared to mesh-based techniques.

Figure 7 illustrates our interpretation of the IoU metric. Figure 7(i) is a low fidelity PBT cross-section (versus the mesh-based cross-section), scoring an IoU of 68% or an IoU Loss (1 - IoU) of 32%, displaying the dissimilarity between the 2 cross-sections in blue (TN) and red (FP). A high fidelity of point-based cross-section with an IoU of 91%, is considered a validation (a sufficient match) to the mesh-based cross-section (as is any IoU of 90% and over). An IoU of < 90% and > 60% is considered an acceptable match to the mesh-based cross-section but in such cases, improvement of the point-based cross-sections is undertaken by any of these factors: increasing slice thickness, increasing structuring element size, additional Closing, increasing point density.

Figure 7: Visual examples of Intersection over Union (IoU) metric “scores”



Chapter 4

Data and Experiments

This chapter presents the data sets and describes the experiments performed on the data, for the purpose of examining the research aims.

4.1. Data

Of the 4 point cloud models forming the data sets, 3 were downloaded from the Zamani repository at and 1 from the Global Digital Heritage Data (GDH) repository. The point clouds are models of cultural heritage sites of historic interest.

Our methodology based on Point-based Techniques (PBTs) was applied to the point cloud data sets (see Table 1), to recover cross-sections. When referring to the "original point cloud" in this work, the data set with the "#Vertices", as listed in Table 1, is being referred to. Any resampling of the data takes these original data sets as input.

Table 1:Original 3D point cloud Data sets

#	3D pointcloud model	# Vertices	#Triangle faces	Xmin	Xmax	Ymin	Ymax	Zmin	Zmax
1	Mosque 3 (Kua Ruins, Tanzania)	1,765,606	3,568,287	-7.72882	7.72882	-6.22656	6.22656	-2.21221	2.21221
2	Mosque 2 (Kua Ruins, Tanzania)	4,027,661	8,175,111	-10.1104	10.1104	-12.5834	12.5834	-2.95808	2.95808
3	House 4 (Kua Ruins, Tanzania)	2,411,523	4,821,969	-19.2587	19.2587	-17.5552	17.5552	-2.42451	2.42451
4	Santa Maria de Melque (Toledo, Spain)	4,749,848	9,496,681	-18.4185	18.4185	-21.3815	21.3815	-7.85925	7.85925

Although triangle face data is available in the data sets, the experiments process only the points (see # vertices) and does not use mesh data (see # triangle faces). Triangle faces are required when densifying (increasing the point density of a model) for the purpose of experimentation. The mesh (triangle faces) is needed in up-sampling so that the original image is not deformed. Figure 8 illustrates the 3D point clouds in perspective view for each of these 4 data sets, used for the experiments.

The original point clouds, used to generate the sample data, were captured using LIDAR laser scanning. An important characteristic of these point clouds is the capture of both the interior and exterior wall surfaces of the structure (such models are not easily available). The default point size, set at 1.0, remains unchanged throughout the experiments. Vertex colour, from the LIDAR scan output, was not recorded for the walls in the models but a simple greyscale rendering is available to give a sense of the shape of the surfaces (the greyscale values are often produced from the vertex normal or projected onto vertices from texture files). The vertex normals data is available in the data set but is used only for rendering the cross-section (normals facing away from the camera are not rendered to screen nor are these points captured in the cross-section for processing).

Figure 8: Point cloud models

(i) Mosque 3 Kua Ruins

(ii) Mosque 2 Kua Ruins

(iii) House 4 Kua Ruins

(iv) House 4 Kua Ruins

4.2. Experiments

Two categories of experiments were undertaken to:

1. Assess the fidelity of the cross-sections, extracted using PBTs - a quantitative assessment. Four (4) cultural heritage point cloud models were used for sample data sets.
2. Examine the extent to which point density affects the fidelity of cross-sections recovered from vertices - a qualitative assessment.

A total of 37 cross-sections were recovered from the 4 models of which 24, which comprise the sample data for this work, are well spaced across the model and illustrate a range of challenging cases. The cross-sections were recovered and compared with the equivalent cross-section derived from the mesh-based model. The Meshlab.net application was used to recover the cross-section from the mesh-based model for comparison with our point-based cross-section.

Note: Models with both interior and exterior surface points are difficult to access. Zamani data was initially the only appropriate data accessible. All 12 of the 3D point cloud models from the Zamani

repository were assessed for their suitability to examine the research aims. Of these, 6 lacked both colour and grayscale vertex values (display as white i.e. no shading, difficult to discern features), a further 3 were malformed and not useable, leaving 3 models for the sample data sets. These models are ruins of heritage sites and have no roof structure. A further model was sourced from the Global Digital Heritage foundation which does have a roof structure. Models which fit the criteria (full structure including roof, possessing interior / exterior surface points, vertex colour and normals), would be preferred but could not be sourced with all these attributes.

4.2.1. Experimental variables

The point-based methodology was applied to the 4 point cloud models, from which a sample data of cross-sections are recovered and evaluated. The cross-sections selected illustrates a range of cases of interest for our methodology.

Initial experimentation revealed that an appropriate starting point, for examining the research aims, are the following variable settings:

i) structuring element size, K = 1 yields a 3x3 symmetrical structural element (passed as a parameter to the Binary Morphology API in the OpenCV software library)

ii) slice thickness, t=0.5.

The first step in the image processing of the cross-section, is to close the boundary of the cross-section slice. When the structuring element size (K) and slice thickness (t) fail to produce a closed cross-section profile (yielding a low fidelity cross-section), their values are then increased to improve the fidelity of cross-section.

Note that the values chosen for K and t are dependent on image resolution of the 2D slice, which is determined by a combination of:

i) the point density of the 3D point cloud and

ii) the camera position - closer or further away from the rendered scene determines lower or higher image resolution respectively (while optimally positioning the camera to view the entire image on screen, as large as possible).

Note: If the image resolution were to significantly change, the values for K and t would need to be recalibrated e.g. if the unit distance between points correspond to a metre in physical terms, versus say centimetres. Automating this calibration however is out of scope for this work and is suggested as future work.

In this work a sufficiently high resolution is needed - a criterion which may be simulated from the data sets should it not exist in the original data. The values chosen for K and t are indicative of our use case.

4.2.2. Experiment 1: Fidelity of Cross-sections

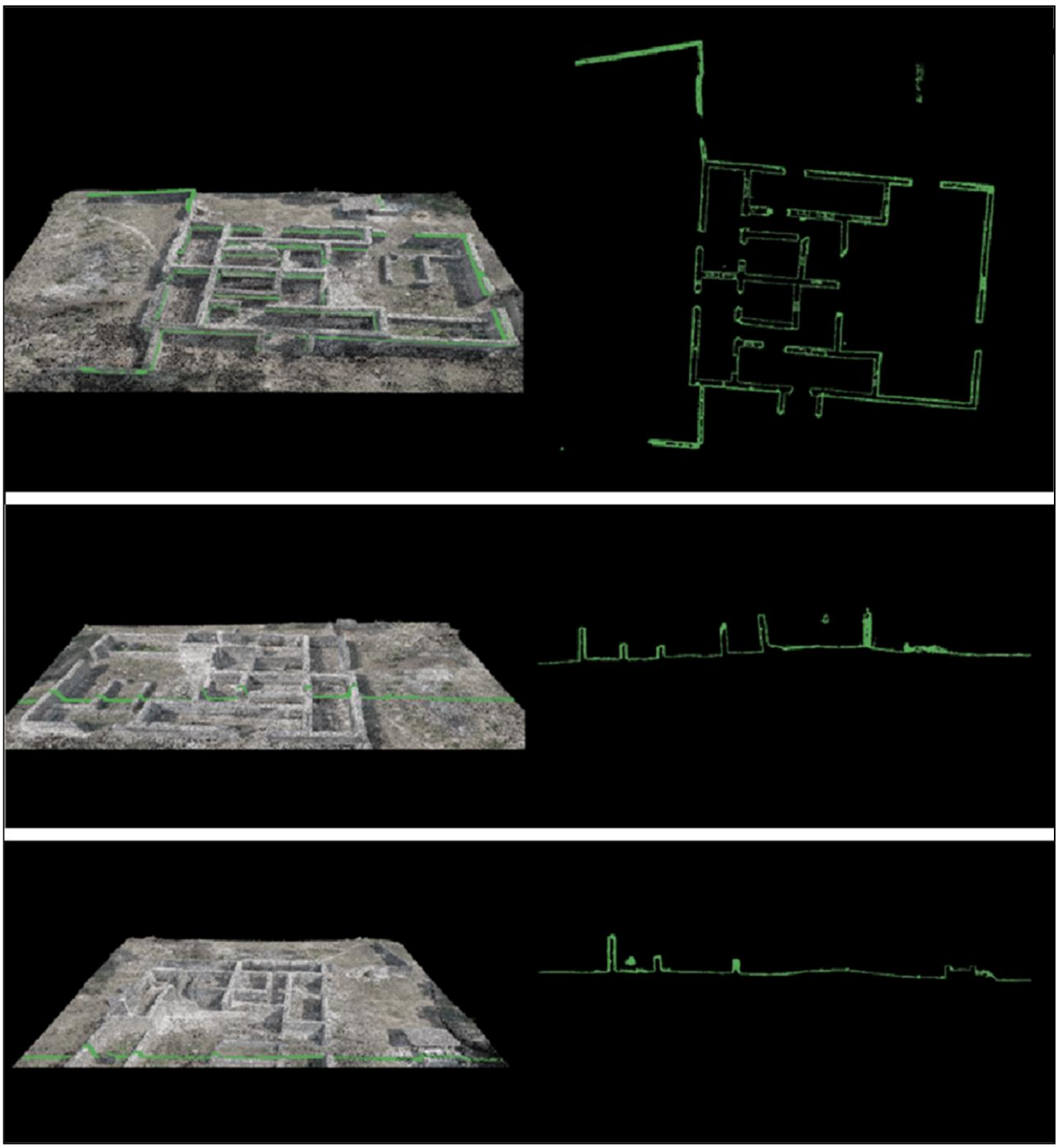
Cross-sections were recovered, using our point-based methodology and the fidelity of the cross-sections examined. For point cloud model, 3 to 8 cross-sections were recovered, chosen to illustrate the range of interesting features or challenging cases. A metric for assigning a quantitative measure of fidelity, the Intersection over Union (IoU), is described in section 3.6 Evaluation. The meaning ascribed to the IoU metric is that the fidelity of the cross-section is considered a convincing match to the comparable mesh-based cross-section if the IoU metric has a value of 90% or higher.

In a real-world application, an expert user would determine the orientation and positioning of the slice plane at which a cross section is recovered (positioning the slicing plane along the x, y or z axes). For these experiments, the researcher chose points of interest which reflect the expected use of the proposed methodology. Cross-sections which did not aid in the understanding of the method's efficacy were avoided.

Figure 9 displays an example of positioning the slicing plane for a cross-section recovery in each of Top, Front and Side Views for House 4 at Kua Ruins. Once the slicing plane is positioned, a screenshot is captured, and the resulting image is then processed by applying the methodology described in Chapter 3. For each data set, cross-sections are captured and processed. The results are tabulated and illustrated in Chapter 5. Section 5.1 presents the visual output (and description) of the key steps in our

point-based methodology. See Figure 11 Visual output of key steps in the methodology (House 4 - Side View, #vertices = 2,411,523)

Figure 9: Kua Ruins: House 4 Cross-sections – Top, Front and Side Views



4.2.3. Experiment 2: Examine the effect of point density

The second category of experiments examines the effect of point density on the cross-section fidelity. Point density refers to the number of points, within a local neighbourhood of the point cloud. The distance between neighbouring points, as a measure of point density, is of interest. High quality laser scans (which this work does not have access to) can have a very high density of points (small distance between points, typically 1 to 3 cm), which are often downsampled for image processing. Essential in this experiment is the impact of point density on fidelity or quality of cross-sections.

To calculate point density, a 3D “grid” is imposed onto the 3D point cloud (each cell of the grid is called a “bin”). Each dimension of the point cloud is divided into 100 intervals. The number of points are counted in each bin and the highest number of points defines the “peak point density”.

After performing the experiments described in 4.2.2 Experiment 1: Fidelity of Cross-sections, any cross-section evaluation with an IoU score below 90% is a subject for examination of the point density, by up-sampling the model to increase higher point density (densification) then repeating the cross-section. The densification is done in Meshlab (www.meshlab.net). A “midpoint sub-division” filter is applied, halving the distance between points - each edge of a triangle face is sub-divided at the midpoint in the original point cloud forming 4 new faces for each original face (no duplication of points), to achieve densification of the original point cloud. The point density histogram can be calculated and visualised in Meshlab.net where the peak of the histogram is the “Peak Point Density” value, calculated per data set. This Peak Point Density value is used as a proxy to describe the overall increase in point density for the data set, see Table 2: Point density Densification data set - House 4 Kua Ruins and Figure 10. Note: In this second category of experiments, examining the impact of point density, the interest is on the “distance between points” in the local neighbourhood of any “breakdown” in developing a closed, point-based, cross-section profile. The “distance between neighbouring points” and “point density” are inversely related.

4.2.3.1. The challenges of examining point density

In considering how best to examine the point density of the models, and particularly the point density of the slice of a model, the challenge encountered is of relating the gap in a cross-section profile to the

point density in the corresponding local neighbourhood. Ideally, each gap in a cross-section profile (after binary morphology closing) should be related to the corresponding point density or distance between points (in the corresponding local neighbourhood of the point cloud), for closer examination.

Consider the 3D point cloud is projected onto a 2D screen using OpenGL (https://www.opengl.org/). Rendering the 3D point cloud onto a 2D screen requires one to apply a Model View Projection transformation described in 2.1.6 Camera view: Object Space and Model View Projection transforms. For the reverse then, to relate a pixel from the 2D rendered image back to the specific point in the 3D point cloud, the inverse of the Model View Projection transformation is applied. This could be achieved with the OpenGL API call:

gluUnProject(float **winx,** float **winy,** float **winz,** FloatBuffer **modelMatrix,** FloatBuffer **projMatrix,**
IntBuffer **viewport,** FloatBuffer **obj_pos)**

Where winx, winy and winz are the screen coordinates to be transformed back to world coordinates. However, in orthographic view there is no depth dimension hence the corresponding coordinate value would need to be estimated. For example, in the 2D rendering of a Top View, the camera and slicing plane are positioned along the Z axis. Therefore, the Z position of the coordinate relating to the pixel in question, would need to be estimated. At best, the Z value from the slice plane position along the Z axis can be estimated, lying somewhere within the range $\mathbf{Z_{slice\text{-}plane}}$ **- slice thickness (t).** Our estimate would only be as accurate as the slice thickness which would not however be accurate enough for an examination. Note: Technically the desired conversion is from "window coordinates" (x,y, z_depth) per fragment (from the rendering perspective the context is the fragment shader), back to world coordinates - and this could possibly be done by an API call to gluUnproject() which returns the necessary matrix. However, once you have only the final framebuffer, the depth information is not preserved.

The slice of the point cloud is examined, relating any gap in the 2D cross-section profile image back to the local neighbourhood in the 3D slice, by visual inspection. In conjunction, the distance between points in that local region of the point cloud slice is examined, in world coordinates using Meshlab,net - comparing this to a region in the slice that relates to no gaps in the cross-section profile. The aim is to confirm/reject a relationship between cross-section fidelity and point density, rather than define an exact point density which result in gaps. An initial analysis of the data sets reveals 2 models for which the

original point clouds produce cross-section profiles with gaps. These 2 point cloud models and their densified versions are described below, in terms of overall point density and distance between points. The local region/s of interest are then described in 5.4.1 Discussion - examination of point density impact.

Table 2: Point density Densification data set - House 4 Kua Ruins & Santa Maria de Melque (Spain)

			Point Density	Distance between neighbouring points (cm)				
#	3D pointcloud model	#Vertices	Peak	Mode	Median	Average	75%	Max.
House 4, Kua Ruins								
1	Original	2,411,523	451,795	2.5 - 3.75	5.7	29.9	32.0	3115.4
2	Densification (1)	3,744,313	544,486	2.5 – 3.75	13.8	27.9	41.4	1557.7
3	Densification (2)	5,686,065	674,751	2.5 – 3.75	19.7	21.7	36.2	778.9
4	Densification (3)	6,626,825	744,753	2.5 – 3.75	20.6	21.0	32.6	389.4
Santa Maria de Melque (Toledo, Spain)								
5	Original	473,890	104,272	8.75 - 10.0	9.6	9.9	12.6	29.3
6	Densification (1)	4,749,848	1,014,181	2.5 - 3.75	3.1	3.7	3.4	9.9

The values in Table 2 are indicators of point density and distance between points, calculated from an analysis of the point cloud models. The #vertices and “Peak Point Density” values provide an indication of the densification of the original models. The distance between neighbouring points (connected in the triangle mesh) are analysed for each model. The measures of central tendency (Mode, Median, Average) as well as the 75th percentile and Maximum values, are calculated, to better understand the point density of the models on which the point-based methodology is applied and experiments analysed.

The distance between neighbouring points can have large variation from 2.5cm to 3115.4cm (mode vs Max.) for the original House 4 point cloud model. Recursive densification reduces this to a range of 2.5cm to 389.4cm. The smaller the distance between points the more likely the cross-section profile will be closed under binary morphology, although this depends on where exactly in the slice the larger point “separation” lies. This will be a key consideration when analysing gaps in the cross-section profile in section 5.4 Examination of point density impact. Note that the densification algorithm settings in Meshlab.net are chosen to only add mid-points if the distance is larger than the smallest 1% across the model. It is unnecessary to add new points between existing points which are already close together (the smallest 1% in the model). As such the Mode remains the same for most of the densifications. Counter-intuitively, the Median and 75th percentile measurements increase with densification; however,

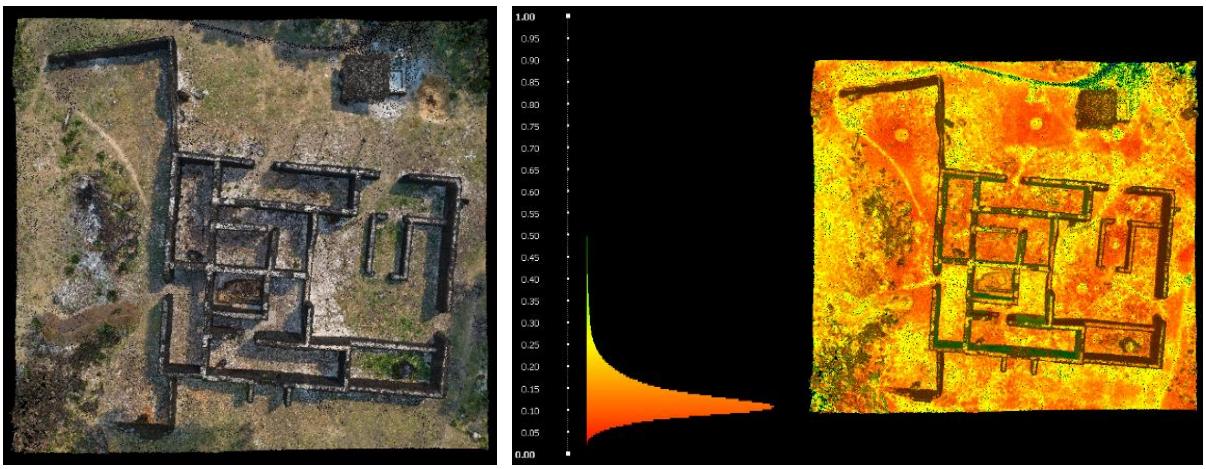
this is an indication of the high positive skewness of the original data distribution and the densification reducing this skewness. As a reference, the maximum distance between neighbouring points in a typical raw, high density point clouds (the models we do not have access to) is typically 2 to 3 cm.

Figure 10 is a visualisation of the point density of House 4, Kua Ruins (done in Meshlab.net). The point density varies across the image. This variation can affect the fidelity of a cross-section of the building structure should there be high density in some parts of the slice and low density in other parts. As the original image is increasingly densified there is less variation in point density (fewer yellow and green patches vs red).

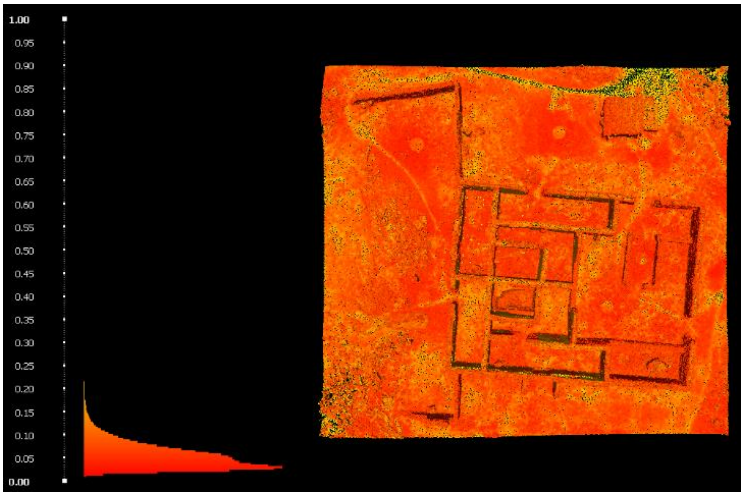
Note that densification is a tool to help us examine the impact of point density or distance between points on our point-based methodology (it is not part of our point-based methodology). In addition, note that densification will not necessarily close holes in models which are not watertight (contains non-manifold edges or vertices) since our chosen densification method (Subdivision Surfaces: Midpoint in Meshlab.net) uses the mesh to add points. Areas of non-manifold vertices and edges ("holes") will not be densified and not be closed by densification of the point cloud model.

Figure 10: Densification data set - House 4 Kua Ruins

(i) Original point density (**# Vertices:** 2,411,523 **Peak point density:** 451,795)

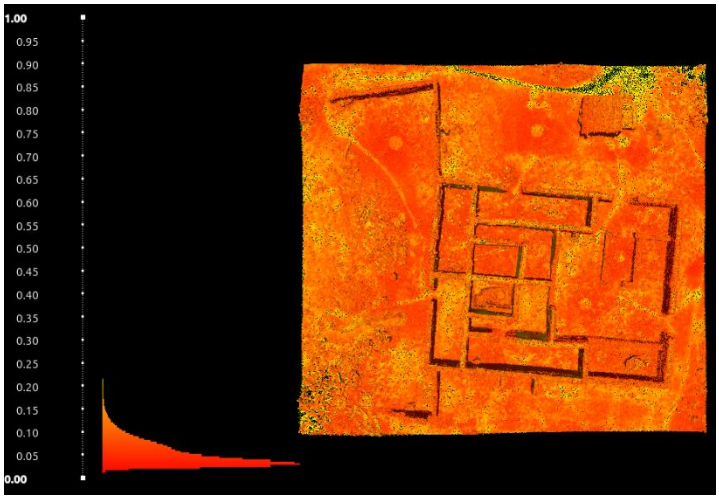


(ii) Densification (level 1) (**# Vertices:** 3,744,313 **Peak point density:** 544,486)



(iii) Densification (level 2) (**# Vertices:** 5,686,065 **Peak point density:** 674,751)

(iv) Densification (level 3) (**# Vertices:** 6,626,825 **Peak point density:** 744,753)



4.3. Conclusion

Experiments were successfully conducted to (i) examine the fidelity of cross-sections extracted by point-based methods and (ii) examine the effect of point density on the fidelity were successfully implemented. All 24 cross sections recovered by point-based techniques, from the 4 models chosen, were registered against cross sections recovered by mesh based techniques and the fidelity examined. Further, the point density of the 4 models were varied and the experiments repeated. The experiment results are evaluated next, in Chapter 5.

Chapter 5

Results and Discussion

The results are categorised into two parts: (i) The fidelity of “Point-based Technique (PBT) cross-section” versus “mesh-based cross-section” and (ii) an examination of the impact of point density on the fidelity of the recovered cross-section. Before presenting the results, a visual output of the key steps in the methodology is first described.

5.1. Visual output of key steps in the methodology

The output of the key steps in our methodology is presented here to clarify the methodology and support the description of the results (presented in the next section). Figure 11 (i) to (vii) are aligned to the key steps in the methodology, previously described in section 3.4 Method for recovering cross-section and Figure 5: Methodology - system architecture.

Description of key steps in the methodology:

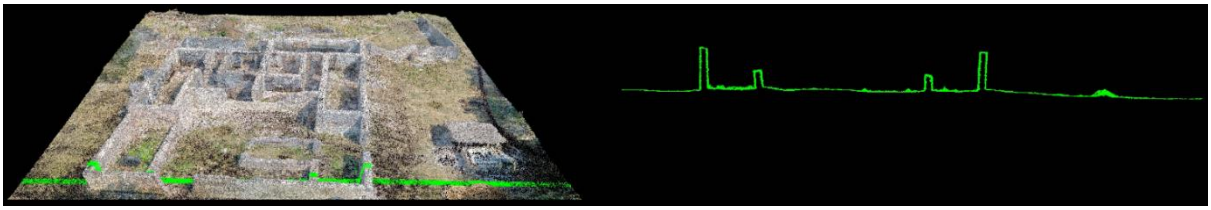
- In Figure 11 (i), a preparatory step to applying the methodology, the loading and orientation of the point cloud is depicted. The user orientates the point cloud for a Top, Front or Side view. On the left is a perspective view of the point cloud which provides context to the user for positioning the slice plane. An orthographic view is displayed on the right side of the image.
- In Figure 11 (ii) (a to c) the user positions the slicing plane at a point of interest and takes the screenshot of the slice as shown in Figure 11 (ii) (b). The 2D slice of the point cloud is captured, comprising discrete points, see Figure 11 (ii) (c).
- In Figure 11 (iii) a cross-section profile is formed from the discrete points of the slice, by applying a binary morphology Closing operation of Depth n = 3, (3 successive Dilations, followed by 3 successive Erosions).
- Figure 11 (iv) (a) displays the holes which may remain after forming the cross-section profile (highlighted in red) and in addition, highlighted in white, gaps between wall structures and the ground (in Front and Side Views).

- In Figure 11 (iv) (b) the holes, previously highlighted in red, are now filled, leaving the gaps previously highlighted in white still to be addressed.
- Figure 11 (iv) (b) further depicts another type of gap (highlighted in blue), typical the result of slicing through groundcover/vegetation.
- In Figure 11 (v), highlighted in green, the gaps in the ground are closed as part of post-processing.
- To address the gaps highlighted in white in Figure 11 (iv) (a), a base is formed to enclose the ground level, see Figure 11 (v). The enclosed based is then filled.
- The completed cross-section is shown in Figure 11 (vi).
- In Figure 11 (vii) the recovered cross-section is overlaid onto the portion of point cloud behind the slice plane.

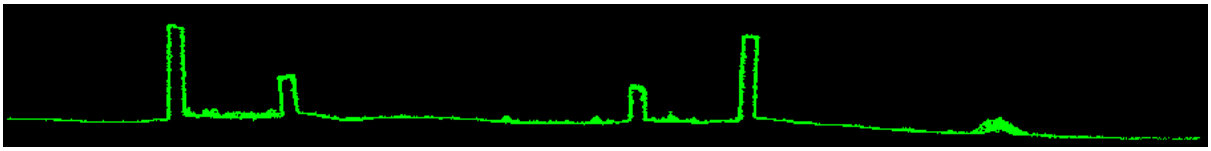
Figure 11: Visual output of key steps in the methodology (House 4 - Side View, #vertices = 2,411,523)

(i) Load and orientate the point cloud

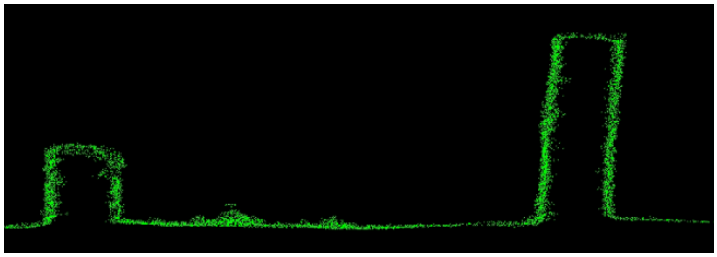
(ii) (a) Positioning the slice plane



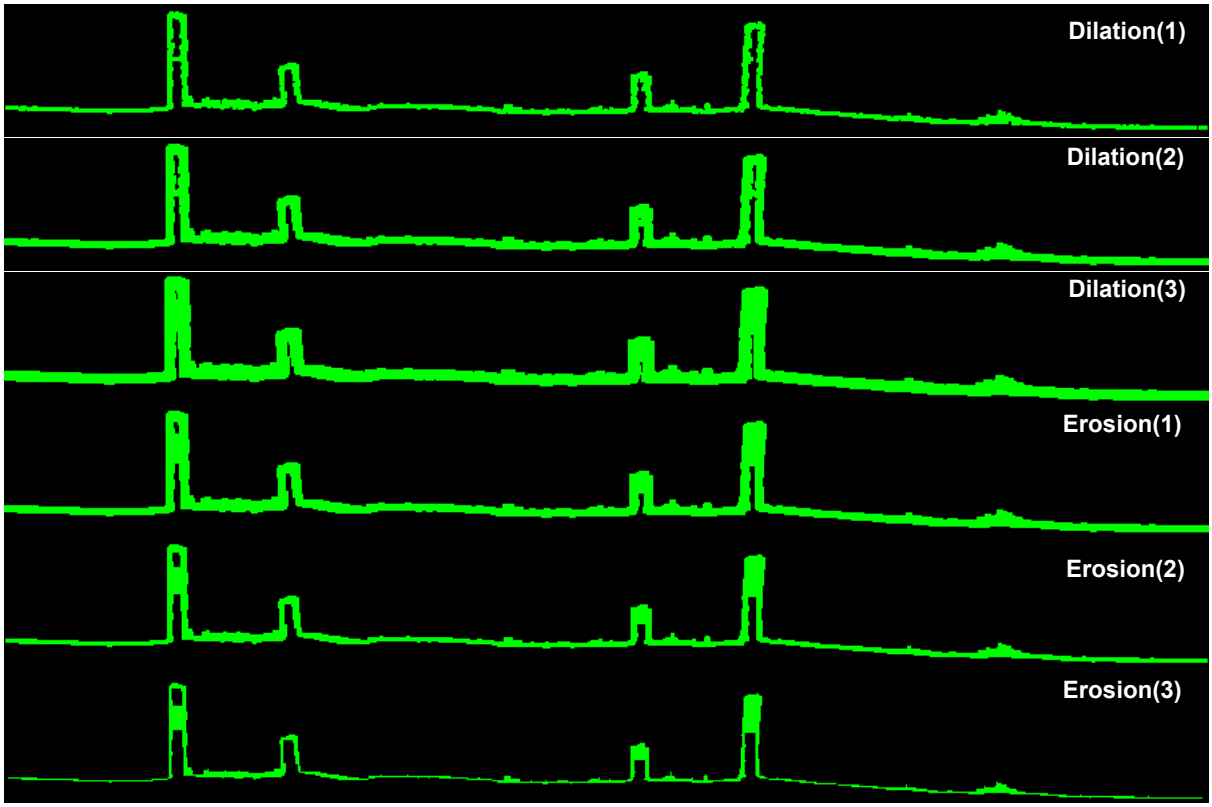
(ii) (b) 2D Slice of the point cloud



(ii) (c) Close-up of a section of the slice, showing discrete points (before binary morphology)



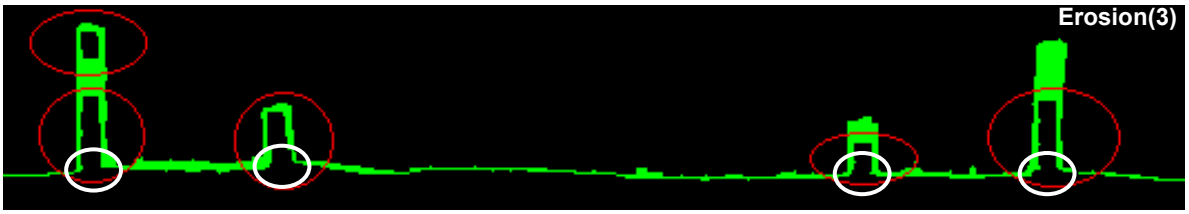
(iii) Close the discrete points of the cross-section boundary (applying binary morphology: Closing depth n = 3



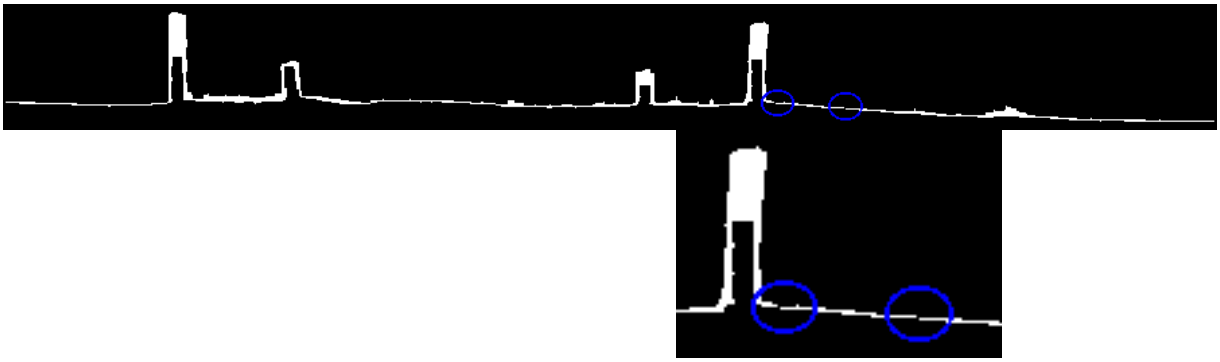
(iv) Fill holes (after binary morphology)

- Perform “Connected Component Analysis” to identify holes and their location (return x, y position of centroid for each hole)
- Flood-fill holes, given position of centroids of holes.

(iv) (a) Holes (highlighted in red), post-processing needed to address gaps (highlighted in white)



(iv) (b) After filling holes – note further gaps in ground (highlighted in blue), to be addressed in post-processing

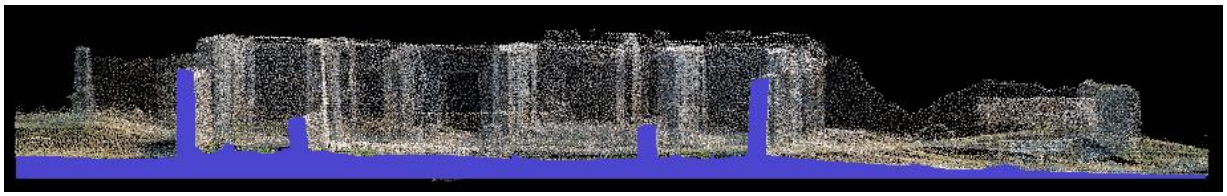


(v) Post-processing

- Gaps in ground closed (highlighted in green – such gaps are typically the outcome of slicing through ground cover/vegetation)
- Form base around the ground – to close gaps highlighted in white in (iv) (a)

(vi) Recovered cross-section

(vii) Overlay cross-section onto back portion of point cloud image (behind slicing plane)



5.2. Settings and evaluation

The results of all recovered cross-sections are presented in tabular form below. The tables include the variable settings (slice thickness and structuring element size), and results of the IoU evaluations, measuring the similarity and dissimilarity between point-based and mesh-based cross-sections, for each point-based cross-section recovered. The IoU metric is described in section 3.6 Evaluation.

Each table of results below, is followed by an accompanying visual representation of the IoU evaluation, displaying the extent to which the point-based cross-section matches an equivalent mesh-based cross-section.

A discussion of the results then follows, analysing the case for our point-based methodology, any advantages or disadvantages over the equivalent mesh-based cross-section (e.g. any advantages or disadvantages in point/mesh-based cross-section recovery), and particular issues that need to be considered such as special cases or exceptions or errors in the evaluation process.

5.3. Fidelity of cross-section

The PBT methodology was applied to the 4 data sets, from which sample data of 24 cross-sections were recovered and evaluated. This sample data illustrates a range of cases of interest (see section *4.1* Data). The results of the first category of experiments, to examine the fidelity of our point-based cross-sections, can be found in Table 3,Table 4, 5 and 6, supported respectively by Figure 12, Figure 13, Figure 14 and Figure 15.

Initial experimentation revealed that an appropriate starting point, for examining the research aims, are the following variable settings:

iii) structuring element size K = 1 yields a 3x3 symmetrical structural element (passed as a parameter to the Binary Morphology API in the OpenCV software library)

iv) slice size of t=0.5.

The first step in the image processing of the cross-section, is to close the boundary contour of the cross-section image. If the structuring element size (K) and slice thickness (t) fail to produce a closed cross-section, their values are then increased to examine the impact on the fidelity of cross-section.

The original Mosque 3 Kua Ruins model has 1,765,606 vertices. The cross-section profiles form without gaps and are all in the high fidelity range (>90%) as shown in Table 3 and Figure 12.

Table 3: Results of PBT cross-section evaluation - Mosque 3 Kua Ruins data set

	t	K	IoU
Top View			
Cross-section(TV01)	0,5	1	**90,7%**
Front View			
Cross-section(FV01)	0,5	1	**95,1%**
Cross-section(FV02)	0,5	1	**95,5%**
Side View			
Cross-section(SV01)	0,5	1	**96,8%**
Cross-section(SV02)	0,5	1	**95,8%**

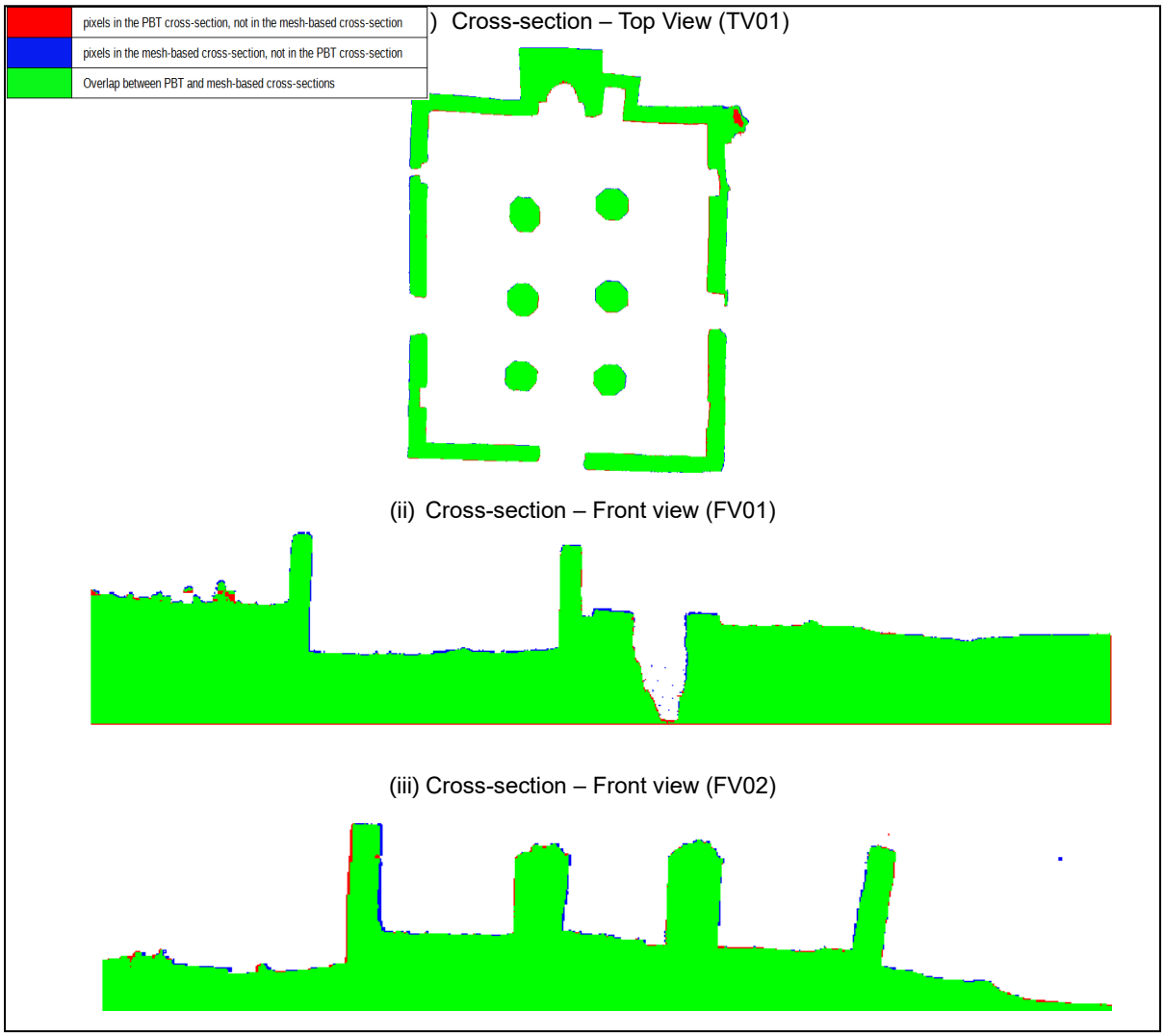
Figure 12: IoU Evaluation graphical representation - Mosque 3 Kua Ruins

The original Mosque 2 Kua Ruins model has 4,027,661 vertices. The cross-section profiles form without gaps and are all in the high fidelity range (>90%) as shown in Table 4 and Figure 13.

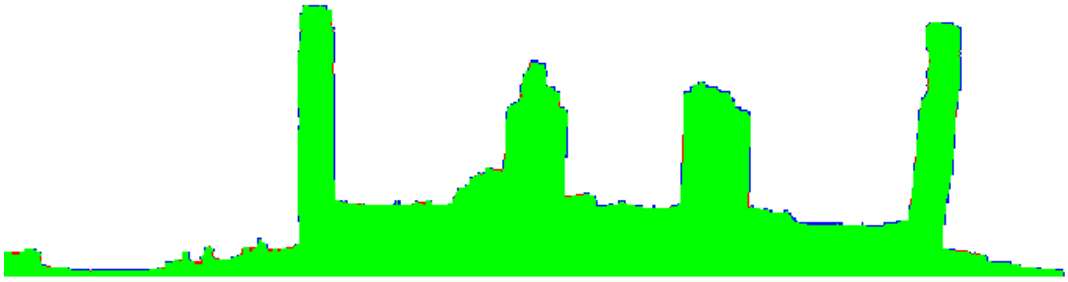
Table 4: Results and settings of PBT cross-section evaluation - Mosque 2 Kua Ruins data set

	t	K	IoU
Top View			
Cross-section(TV01)	0,5	1	**95,1%**
Front View			
Cross-section(FV01)	0,5	1	**95,9%**
Cross-section(FV02)	0,5	1	**97,2%**
Cross-section(FV03)	0,5	1	**97,2%**
Cross-section(FV04)	0,5	1	**94,5%**
Side View			
Cross-section(SV01)	0,5	1	**97,7%**
Cross-section(SV02)	0,5	1	**98,5%**
Cross-section(SV03)	0,5	1	**98,9%**

Figure 13: IoU Evaluation graphical representation - Mosque 2 Kua Ruins

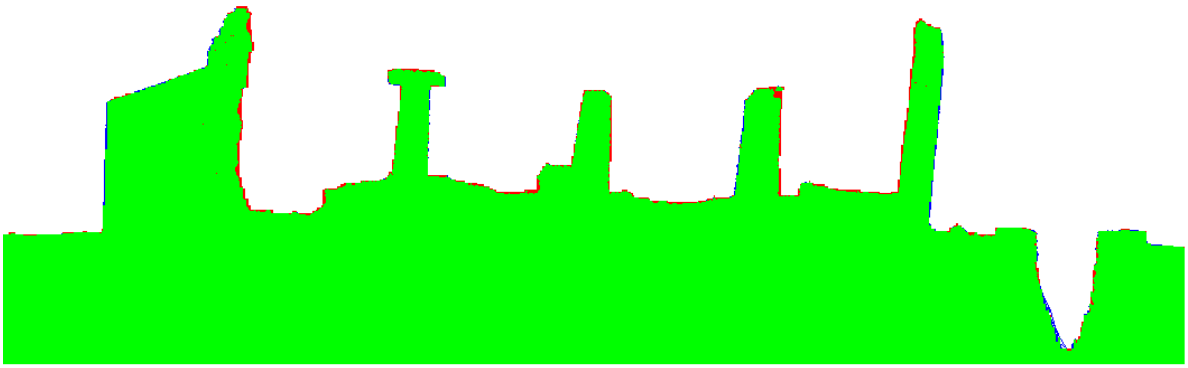


(iv) Cross-section – Front view (FV02)



(v) Cross-section – Side view (SV01)

(vi) Cross-section – Side view (SV02)



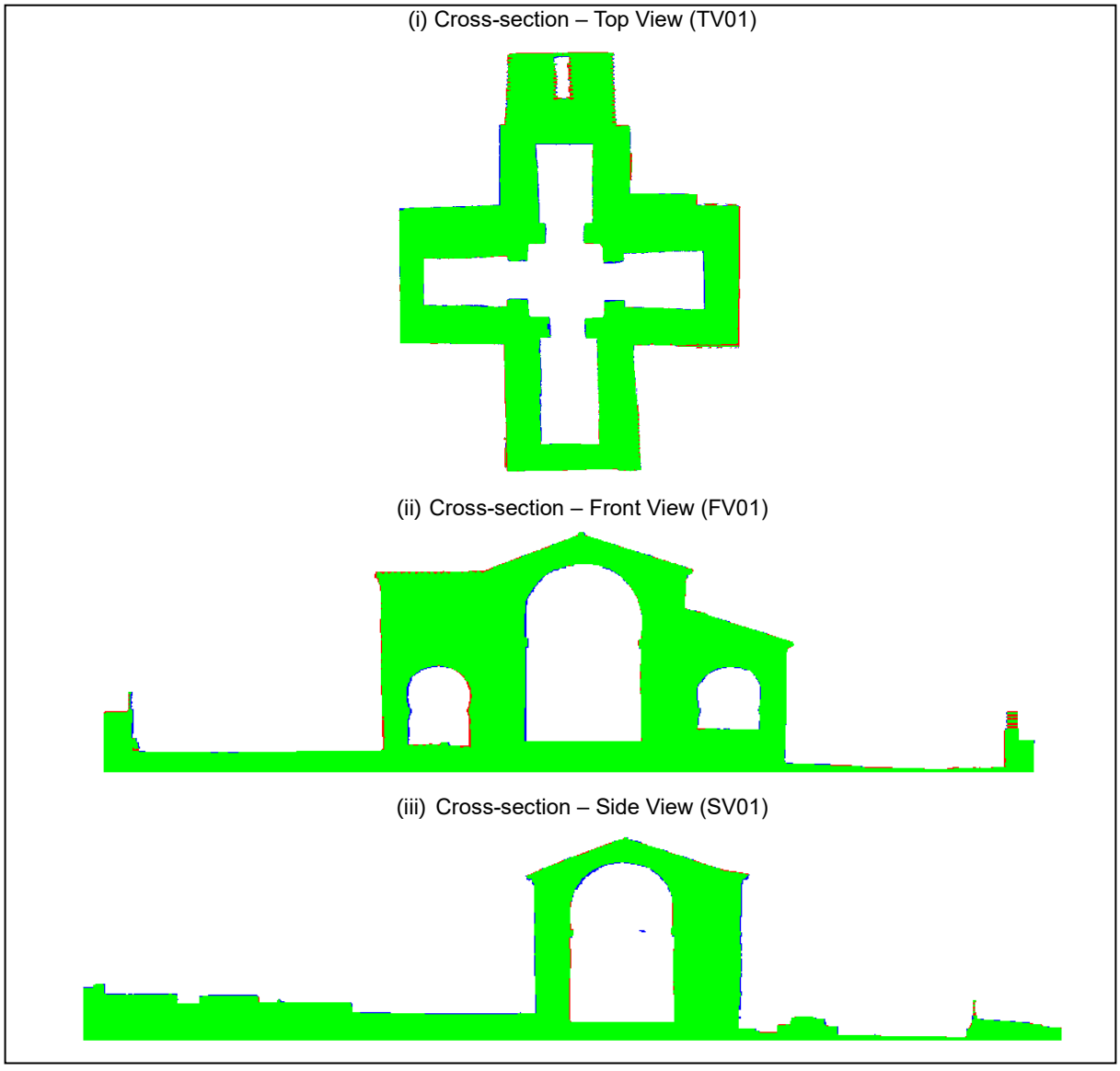
(vii) Cross-section – Side view (SV03)

The original Santa Maria de Melque model is a low density point cloud (473,890 points). The model has holes (manifold edges and vertices) which could not be repaired in Meshlab.net nor comprehensively improved by densification of the model. To recover cross-sections with closed profiles the slice thickness (t) is set to a value of t=1.0, producing the results in Table 5 and Figure 14.

Table 5: Results and settings of PBT cross-section evaluation - Santa Maria de Melque

	t	K	IoU
Top View			
Cross-section(TV01)	1.0	1	**96.7%**
Front View			
Cross-section(FV01)	1.0	1	**97.5%**
Side View			
Cross-section(SV01)	1.0	1	**97.5%**

Figure 14: IoU Evaluation graphical representation - Santa Maria de Melque (Spain)

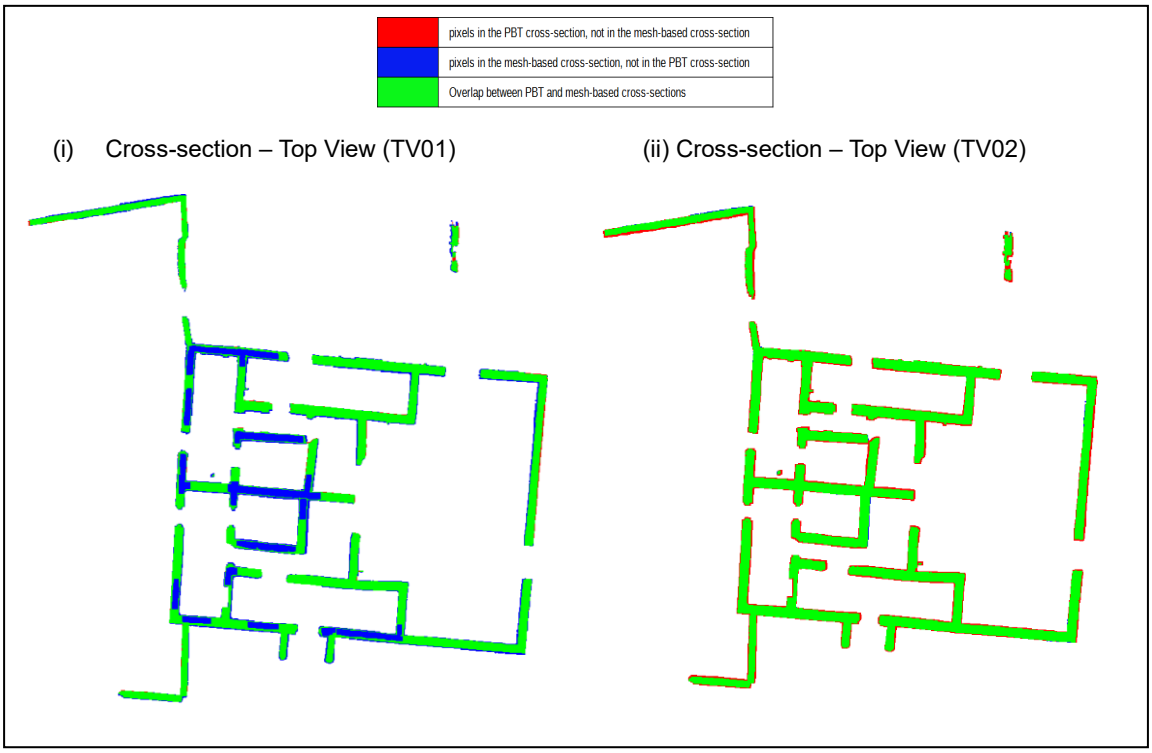


The original point cloud model, House 4 Kua Ruins has 2,411,523 vertices. Although this may not be considered a low density point cloud, a low fidelity cross-section is recovered (with a slice thickness of t=0.5 and a structuring element size of 3x3 with K=1), see Table 6 and Figure 15. This is due to the point density varying considerably across the model, affecting areas of the relevant slice - slices TV01, TV02, TV03 and TV04 are taken at the same position (after accounting for slice thickness), see table with slice plane position in .

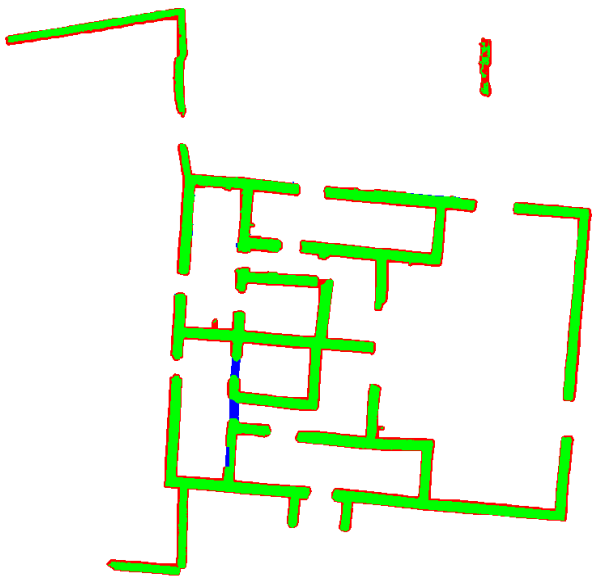
Table 6: Results and settings of PBT cross-section evaluation - House 4 Kua Ruins data set

	t	K	IoU
Top View			
Cross-section(TV01)	0,5	1	**67,9%**
Cross-section(TV02)	0,5	1	**82,6%**
Cross-section(TV03)	0,5	2	**73,6%**
Cross-section(TV04)	1,0	1	**88,3%**
Front View			
Cross-section(FV01)	0,5	1	**90,1%**
Cross-section(FV02)	0,5	1	**94,6%**
Cross-section(FV03)	0,5	1	**94,6%**
Side View			
Cross-section(SV01)	0,5	1	**93,3%**
Cross-section(SV02)	0,5	1	**90,1%**
Cross-section(SV03)	0,5	1	**94,7%**
Cross-section(SV04)	0,5	1	**91,1%**

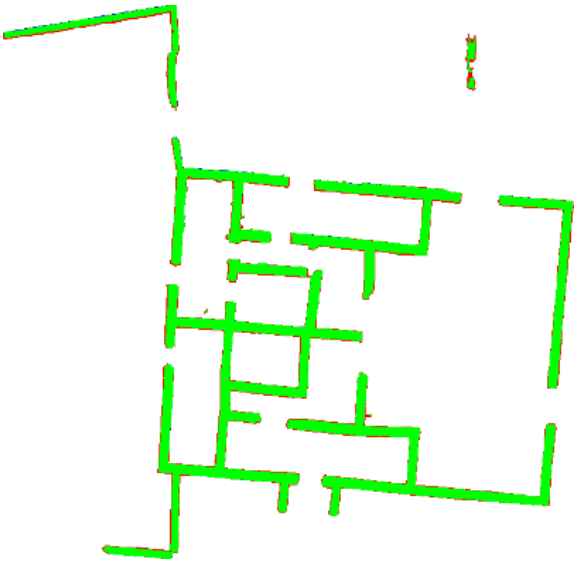
Figure 15: IoU Evaluation graphical representation - House 4 Kua Ruins



(iii) Cross-section – Top View (TV03)

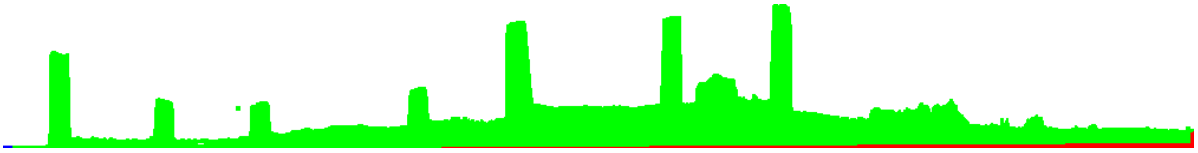


(iv) Cross-section Top View (TV04)

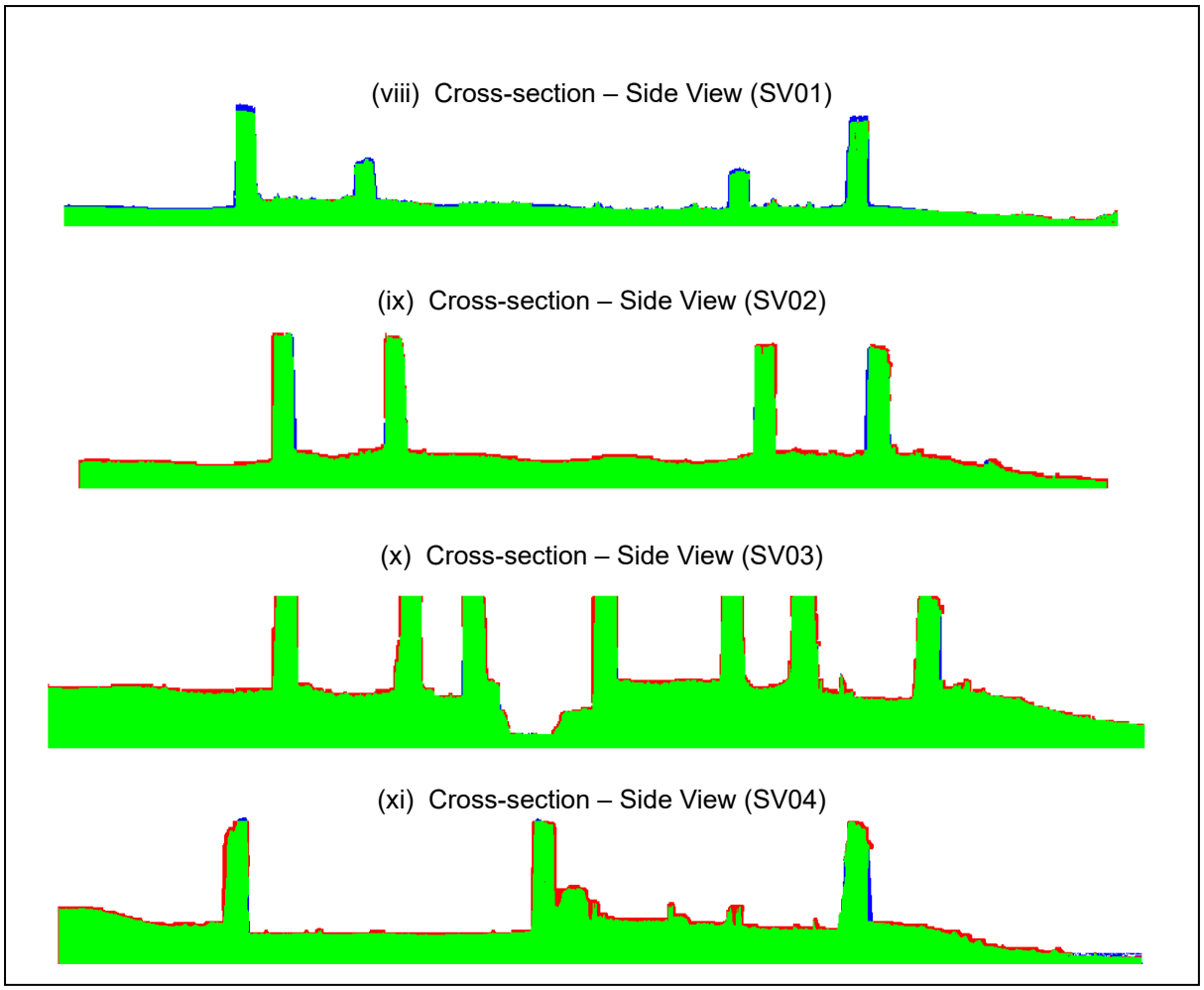


(v) Front View (FV01)

(vi) Front View (FV02)



(vii) Front View (FV03)



5.3.1. Discussion - fidelity of cross-section (original data set point density)

The Intersection over Union (IoU) measure is a quantitative means of assessing the fidelity of the recovered cross-sections. The average IoU, calculated from the total number of cross-sections recovered, is 95.3%. The recovered cross-sections captured points of interest in the 4 data sets, amounting to 24 cross-sections recovered and analysed. The graphical illustration of the IoU metric further supports the analysis of the point-based methodology.

Recall, from section *3.63.6* Evaluation, an IoU of 90% is considered a convincing validation of the point-based methodology, making the case for our point-based methodology over mesh-based methodologies for cross-sectioning applications. An IoU score between 80% and 90% is interpreted as a promising case for our point-based methodology but needing improvement, additional image processing, or post-processing. A score between 60% and 80% does not make an adequate case for

our point-based methodology and requires further analysis to inform cause of the shortcoming and possible solutions.

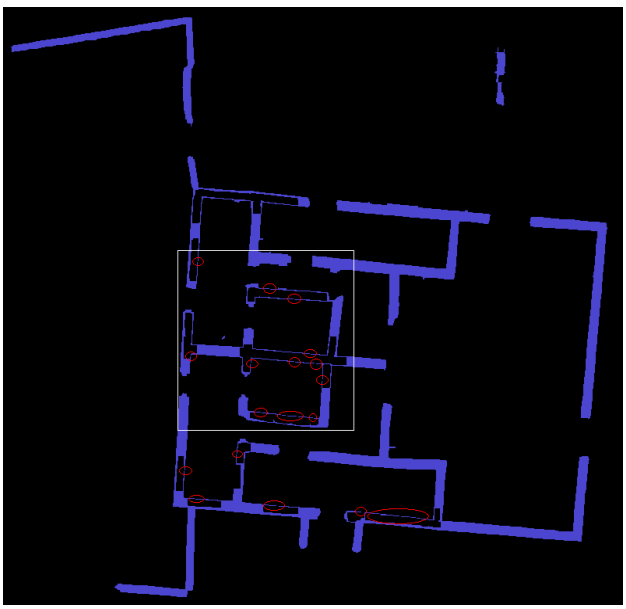
The results for the Top View cross-sections for House 4 Kua Ruins lie in the range 60% to 90% (see Table 6 and Figure 15 (i) to (iv)). For the data set, the values of t=0.5 (a real-world thickness of 50cm) and K=1 do not recover a closed cross-section, see Figure 15 (i). The holes in the cross-section are not filled because the boundary contour, the output of the binary morphology step, has gaps which are then not filled in the flood-fill step see Figure 16 where the gaps in the cross-section profile are highlighted.

The gap in the boundary means that the hole in the cross-section “connects” to and is interpreted as background, which is excluded from flood-filling in the filling algorithm, leaving the cross-section incomplete. In our point-based methodology, there are three options to improve the fidelity of the cross section, before considering densification of the model.

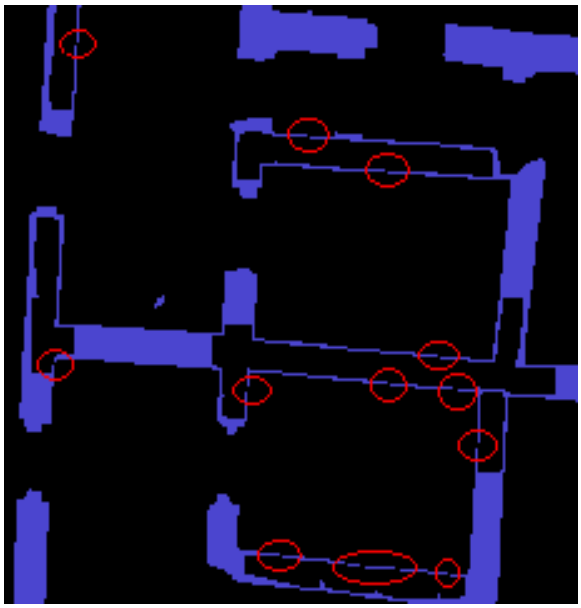
The first approach to achieving a higher fidelity cross-section is to perform a further single round of binary morphology Closing of depth n =1 (1x Dilation followed by 1x Erosion). This improves the IoU score from 67.9% to 82.6%. See Figure 15 (ii). This additional image processing step closes most but not all boundary gaps and therefore not all holes are filled. Note: The methodology aims to automate

Figure 16: Example of cross-section profile with gaps

(i) After binary morphology Closing



(ii) closeup of gaps



hole filling, avoiding the need for manual hole filling. It is not part of the methodology to manually close gaps in the cross-section profile.

A second approach to achieve a higher fidelity is to increase the structuring element from K=1 (which applies a 3x3 structuring element), to K=2, which applies a 5x5 structuring element. For K=2, the IoU improves from 67.9% to 73.5%, see Figure 16 (iii). The output image is “oversized” compared to mesh-based cross-section (red pixels along the boundary). The binary morphology output from using a larger structuring element is “fatter”. While this will close more holes that were missed using the smaller sized structuring element (K=1), there is also a loss of finer detail.

Increasing slice thickness from t=0.5 to t=1.0 is a third approach to improve fidelity and improves the cross-section fidelity from an IoU score of 67.9% to 88.3%. The fidelity for the larger slice thickness, although close to yielding a high fidelity IoU score (>90%), may also include more of the building structure’s geometry in the slice than is desired by the user. For the data set in question there is little additional geometry resulting from an increased slice thickness (see Figure 15 (iv)) however this may vary from model to model.

It should be noted that some error may be introduced when registering (aligning and scaling) the point-based cross-section against the mesh-based cross-section. This error is estimated to be minimal (around 2-5%). Even if this error is introduced, the results and analysis are still valid. The registration error may arise from the manual scaling of the point-based cross-section during the manual registration against the mesh-based cross-section. The aim is to demonstrate the feasibility of a point-based methodology which the manual registration of the cross-sections does not detract from.

Note: The reason for using an external software app and not recovering the mesh-based cross-section in our software is as follows. The OpenGL software library is used to render the point cloud images to screen. When drawing a mesh, the technique used to “clip” points, for slicing the point cloud, sets the alpha value of the vertex colour to 0. Setting the alpha value to 0 results in edges connected to the clipped point being blended in black (indistinguishable from the background which is also black) in the 2D sliced image. The black portion of the edge shows as a gap, similar to the gaps indicated in Figure 16. The options to resolve this issue are to either (i) employ more advanced graphics processing or (ii)

develop mesh-reconstruction algorithms, are both out of scope for this work which aims are geometry processing and not advanced graphics processing. The mesh-based cross-section was therefore recovered in a separate software package (meshlab.net) but needs scaling to register the cross-sections for comparison and evaluation, given that Meshlab has different camera settings to our software.

Across all the 24 cross-sections recovered from the 4 data sets, for all but one cross-section our methodology does produce a high fidelity cross-section making the case for the point-based methodology. Where the methodology falls short, additional processing may be applied to improve the fidelity.

Gaps in the boundary of the cross-section and the resultant lower fidelity, may be due to an “insufficient” point density. The second part of the experiments examines this possibility.

5.4. Examination of point density impact

In this section the impact of point density is investigated, by densifying (increasing the point density) of the data set: House 4 Kua Ruins. The point-based cross-section is recovered at different densification levels and compared with the corresponding mesh-based cross section.

To recap, in the first part of the experiment (see Table 6) an IoU metric of 67.9% (low fidelity) is measured, setting slice thickness t=0.5 and a 3x3 structuring element for K=1. Increasing the size of the structuring element, from 3x3 with K=1 to 5x5 with K=2, improves the IoU metric from 67.9% to 73.6%. When an additional binary morphology Closing, is introduced, of depth(1) this increases IoU from 67.9% to 82.6% (t=0.5, K=1). If instead the slice thickness is increased to t=1.0 the IoU metric improves from 67.9% to 88.3%, however increasing slice thickness may not always be desirable as it may unintentionally include parts of the structure or ground not intended to be part of the cross-section. A high fidelity cross-section is desired while keeping the slice thickness as small as possible. To examine the impact of point density, a slice thickness of t=0.5 and K=1 is maintained while increasing the point density.

5.4.1. Discussion - examination of point density impact

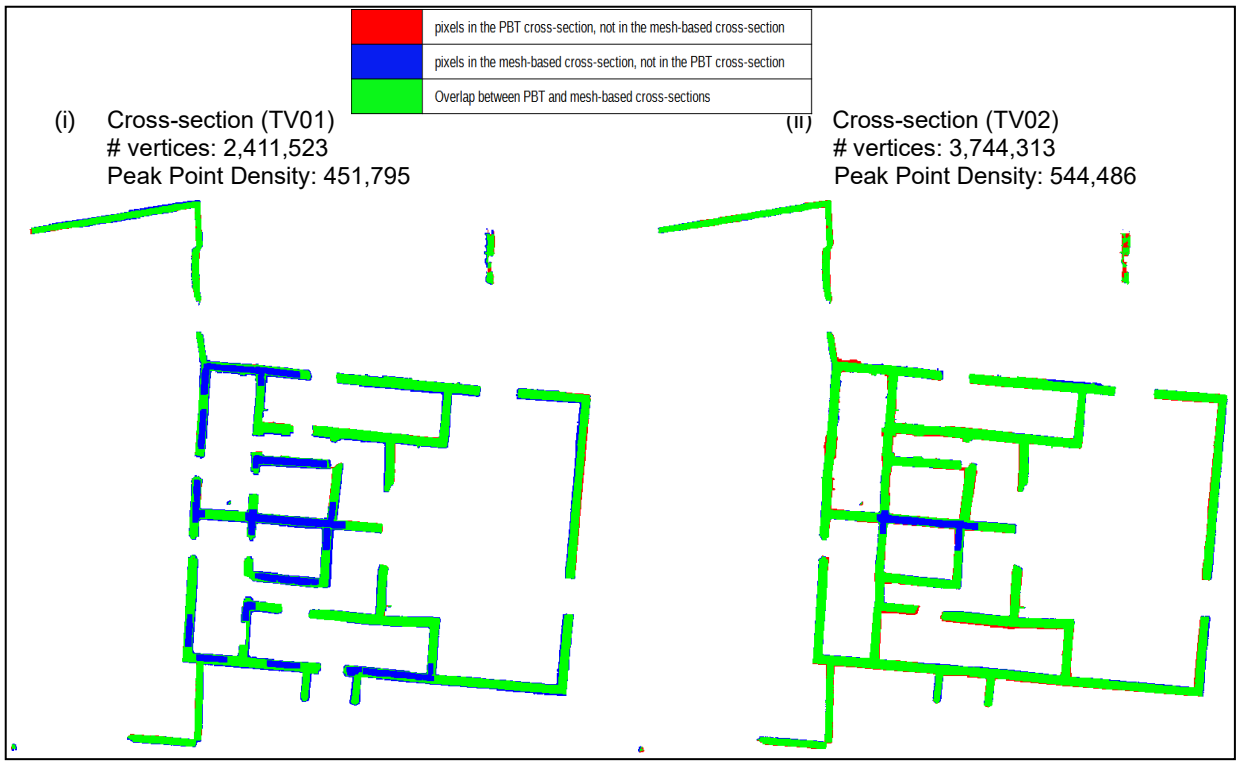
The House 4 Kua Ruins model was recursively densified and cross-section results captured in Table 7 and Figure 17. The tabulated results and Figure 17 reveal a clear correlation between the point cloud density and the fidelity of the cross section. The cross-section from the point cloud with the original point density (as downloaded from the Zamani repository) scores an Intersection over Union (IoU) of 67.9%. Increasing the point density (using Meshlab.net) improves the IoU score for each increase in point density, to 87%, 90% and 91% respectively.

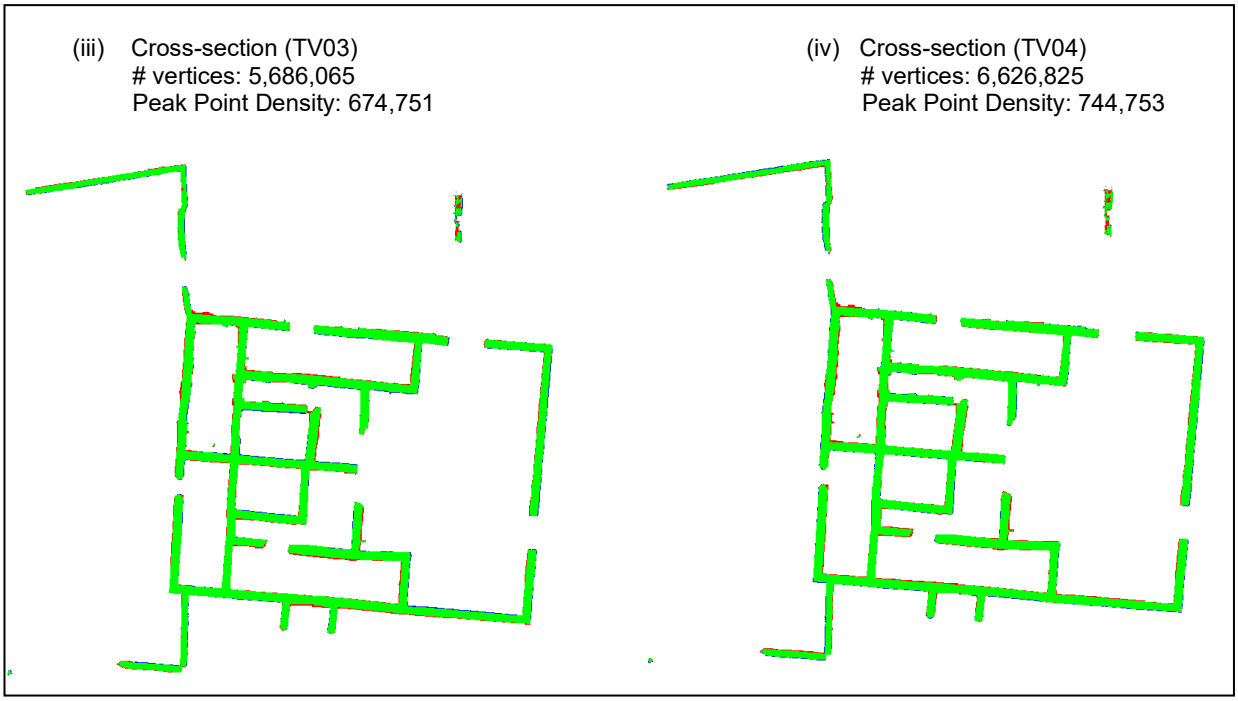
The higher the point density (meaning that points are closer), the lower the likelihood of gaps remaining after the binary morphology step, the better for holes in the 2D cross-section (after binary morphology) to be filled and the higher the fidelity of the recovered cross-section.

Table 7: Impact of point density

Top View	#Vertices	Peak Point Density	t	K	IoU
Cross-section(TV01)	2,411,523	451,795	0.5	1	**67.9%**
Cross-section(TV02)	3,744,313	544,486	0.5	1	**87.0%**
Cross-section(TV03)	5,686,065	674,751	0.5	1	**90.0%**
Cross-section(TV04)	6,626,825	744,753	0.5	1	**91.0%**

Figure 17: Impact of increasing Point density on cross-section fidelity – Top View, House 4 Kua Ruins





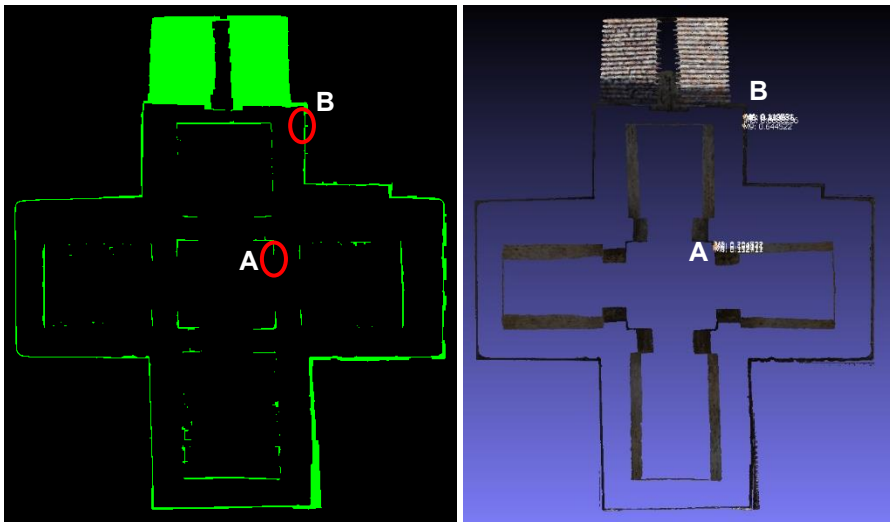
The “distance between points” (point separation) can be used as a measure of point density. The higher the point density the smaller the distance between points. Conversely, the lower the point density the larger the distance between points. An examination of point separation reveals a direct relationship with our point-based methodology.

Figure 18 illustrates an example of the impact of distance between points, and therefore point density, on our point-based methodology. In Figure 18 (i) two areas of a cross-section are shown, labelled A and B, where A is an area of large point separation or low point density (compared to other areas of the point cloud) and B is an area of small point separation or high point density. The area of large point separation (A - 15cm to 47cm) corresponds to an area of the point-based cross-section where the profile has gaps after performing Binary Morphology Closing (see Figure 18 (ii) for a close-up of the point separation, measured in Meshlab.net). An example of an area of small point separation (B - 2.9cm to 3.6cm) corresponds to an area of the point-based cross-section where the profile does close (no gaps) after performing Binary Morphology Closing (see Figure 18 (iii) for a close-up of the point separation).

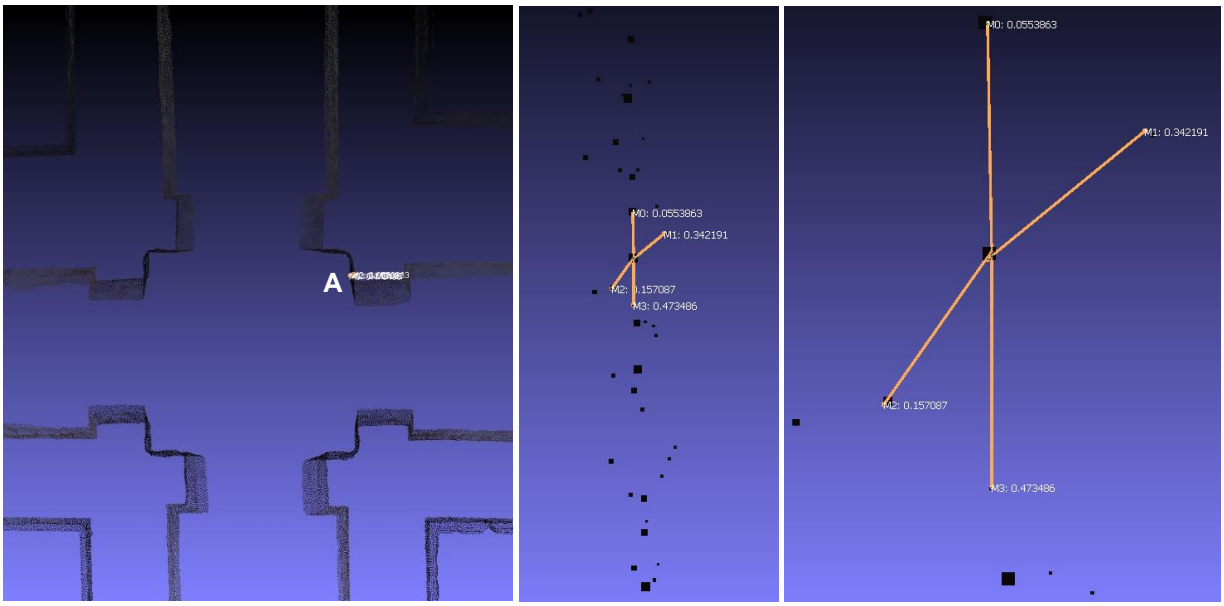
This closer examination of point separation or point density is observed in all areas where gaps in the cross-section profile exist after Binary Morphology Closing. This indicates that high point density is a necessary condition for the effectiveness of our point-based methodology.

Figure 18: Impact of Point "separation" (a measure of Point Density)

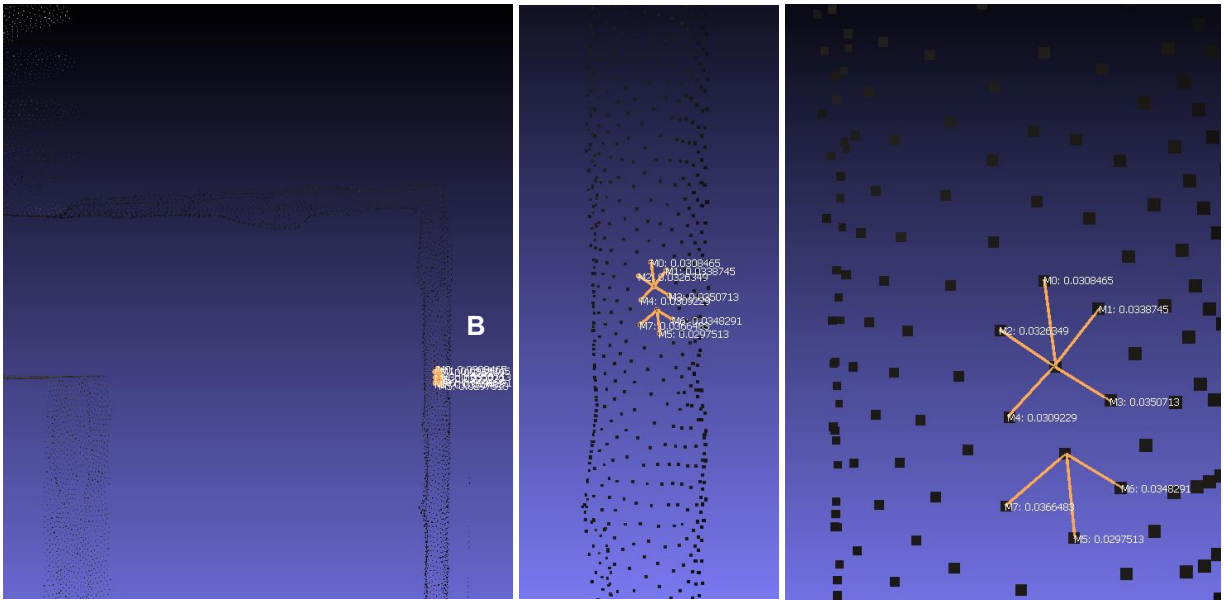
(i) Point-based and mesh-based cross-sections



(ii) Cross-section profile: Example of point “separation” where binary morphology does not Close (leaves a gap)



(iii) Cross-section profile: Example of point “separation” where binary morphology does close (no gap)



5.5. Discussion

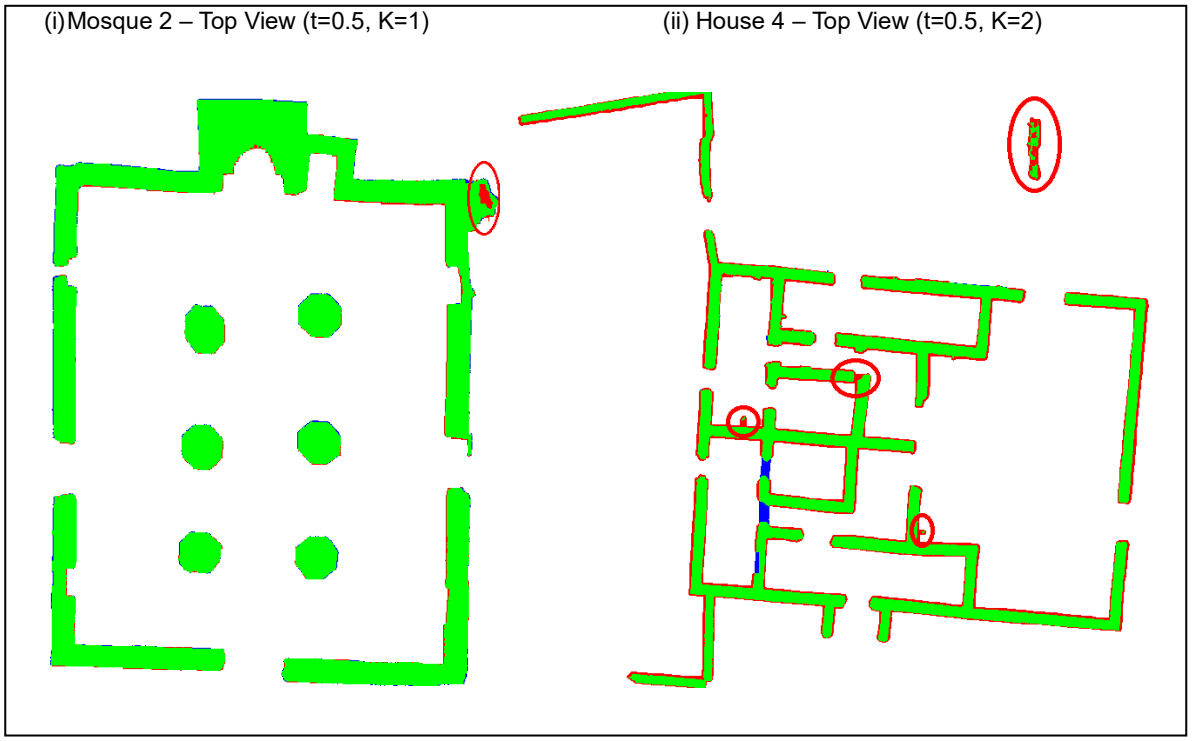
Overall it can be seen, across Figure 12, Figure 13, Figure 14 and Figure 15, that IoU loss (dissimilarities, indicated by red or blue pixels in the IoU image results) is not clustered in specific areas and somewhat evenly distributed. See Figure 19 as an example of the type of dissimilarity across all cross-sections recovered (IoU loss highlighted in red). The distribution of IoU loss, “over” in some places and “under” in others, across the entire model, is an indication that the point-based methodology is not biased or favouring certain areas of the image (which might appear as a cluster of red or blue). Note: The red pixels indicate where the PBT cross-section has pixels which are not in the mesh-based cross-section and blue indicates the converse, where the mesh-based cross-section pixels are not in the point-based cross-section.

Figure 19: Example of “evenly distributed” IoU loss (red and blue pixels)

5.5.1. Binary Morphology Closing merges detail in the cross-section

Binary morphology Closing (Dilation followed by Erosion) is performed with a depth of 3 (3 successive Dilations followed by 3 successive erosions) in our methodology the Dilation steps are intended to form the discrete points of the cross-section into a closed profile (the erosion steps remove the pixels added by Dilation) however a possible side effect is that image detail, the finer features close to each other in the image, is merged thus losing that detail. Erosion will not recover the gaps between the detail after they are merged in the Dilation steps. See Figure 20, loss of detail (red pixels) highlighted by red ellipsoids. Note that this loss of detail may or may not be acceptable for the application depending on the practitioners needs.

Figure 20: Binary Morphology Closing - loss of detail

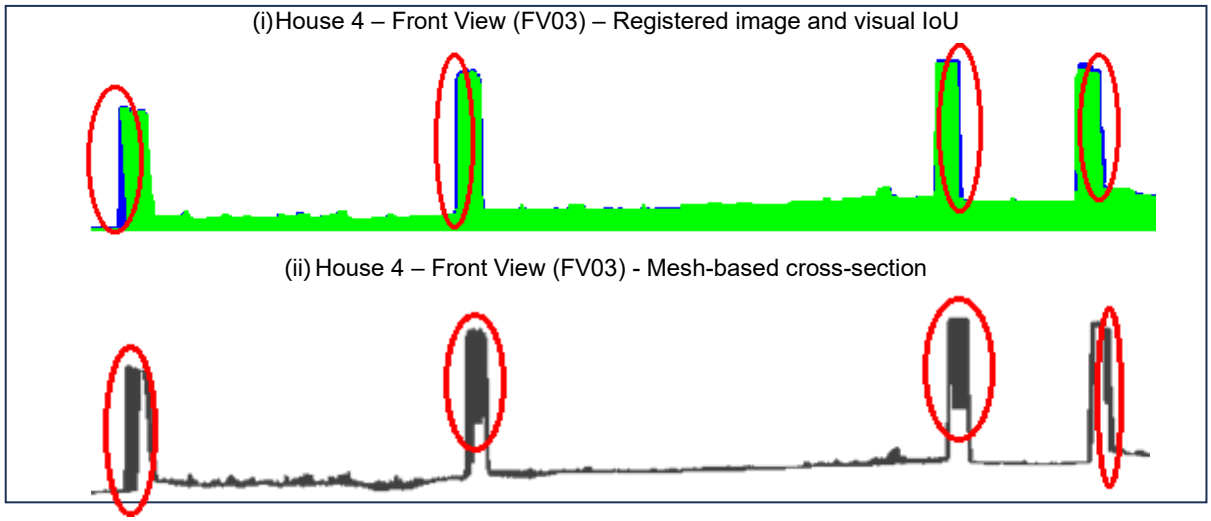


5.5.2. Mesh reconstruction incorrectly connects points

The mesh-based cross-section is recovered in an external software application (mehlab.net). The key reason for this is that, after slicing the point cloud the mesh needs to be reconstructed on the slice (for registration and evaluation against the point-based cross-section), a task which is out of scope for the software developed in this work. Instead, since meshlab.net provides mesh reconstruction algorithms the mesh-based cross-section is recovered in meshlab.net.

Mesh reconstruction may however connect points incorrectly. This becomes especially evident when slicing the point cloud, which inevitably “breaks” triangle faces. The remeshing after slicing may incorrectly connect points which otherwise would not have been connected had the sliced points still been present. See Figure 21, incorrect mesh points connected. The incorrect mesh points are apparent in Figure 21(i) the IoU of the registered images (blue pixels) and Figure 21(ii) the meshed-based cross-section. One downside of the mesh-based cross-section is the incorrect connecting of points. In this regard, the point-based methodology provides an improvement over the mesh-based cross-section.

Figure 21: House 4 – Front View (FV03) - Incorrect points connected in mesh reconstruction slice

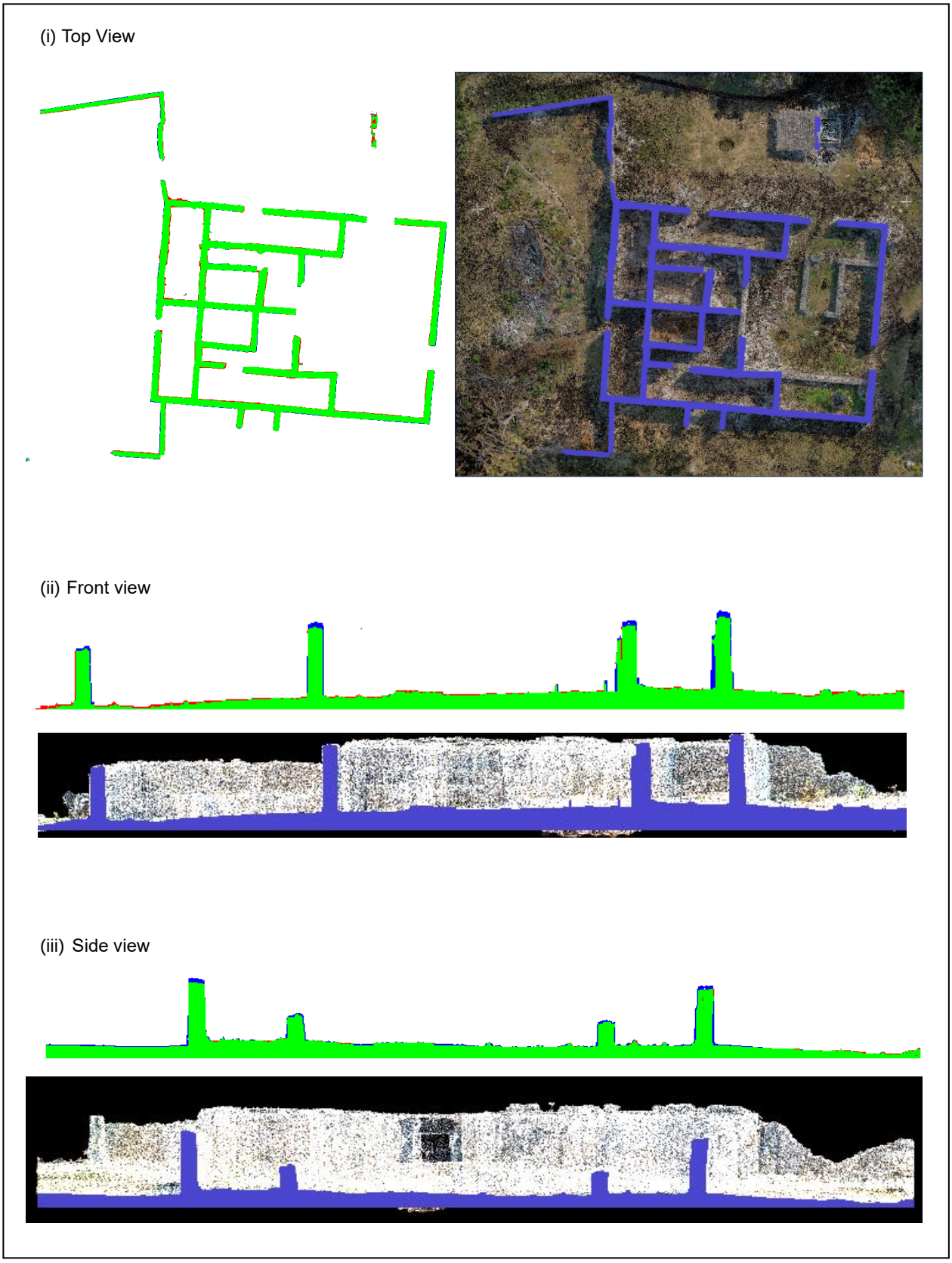


5.5.3. Cross-sections overlaid onto point cloud

The final step after completing the formation of the cross-section is to overlay it on onto the point cloud. This provides an opportunity to view the cross-section within the context of the overall point cloud, for a better understanding of the structure revealed by the cross-section. This can add value to viewing the structure as well as knowledge sharing and discussion about the structure - layout, geometry, dimensions and so forth.

The walls of the point cloud models unfortunately do not have colour, merely a grayscale shading which makes it difficult to visualise the context of the actual site for Front and Side Views, see Figure 22, however the cross-section accuracy remains a validation of the research aims.

Figure 22: Examples of cross-section overlaid onto point cloud



Chapter 6

Conclusion

This work sets out to evaluate the viability of a Point Based Technique (PBT) approach for cross-section construction. The extent to which the points alone (without a meshed point cloud) can be used to extract cross-sections is investigated, for architectural views from cultural heritage 3D point cloud models. In addition, the impact of point density on the fidelity of the cross-section was examined. The fidelity of the cross-section recovered using a point-based methodology was compared to the cross-section recovered from a mesh-based model. Although our method is applied (and evaluated) on heritage point cloud models, the method should be applicable in recovering cross-sections from any building point cloud.

A total of 24 cross-sections were recovered from 4 point cloud models. The IoU metric (measure of similarity) between the point-based cross-section and the comparable mesh-based cross-section for all 24 data sets are categorised as high fidelity. The overall average IoU of the sample data sets (24 cross-sections) is 94.6%, with a maximum average of 96.8% and a minimum average of 91.8%. The IoU metric scores are all above 90% with a minimum of 90.7%. The average IoU metric score for the data sets are, Mosque 3: 94.8%, Mosque 2: 96.9%, House 4: 92.1% and Santa Maria de Melque: 97.3%. An IoU score of 90%+ is a validation of the point-based approach and the relative accuracy of each IoU score is considered conclusive evidence that a point-based approach for cross-section recovery is indeed viable.

For the point-based cross-section, it is necessary for the cross-section contour to be fully closed (leaving no gaps) for the cross-section IoU to be 90% or higher relative to the mesh-based cross-section. On closer examination, where the cross-section IoU falls below 90%, the point density is not high enough. Point density is therefore a direct factor in the fidelity of the cross-section recovered. By increasing the point density, the IoU metric for a low fidelity cross-section is improved, from 67.9% to 91% for House 4 Kua Ruins, as an example. The gaps in the cross-section profile are not closed in the binary

morphology stage of the point-based methodology, however these gaps are closed at a higher point density where the distance between points is smaller.

The results also show that the mesh-based cross-section may connect points incorrectly and this directly increases dissimilarity of the cross-sections recovered by these two methodologies. The trade-off in our point-based methodology is that fine detail is lost (in the binary morphology Closing step) versus points incorrectly connected using a mesh-based methodology.

The output of modern LIDAR systems is a high point density (which we do not have access to in this work) and is typically downsampled to save on processing resources needed (time, computational effort). It is demonstrated that by increasing the point-density, a high fidelity cross-section is recovered where initially a low fidelity cross-section was recovered at a lower point density. The point density has a direct positive impact on the cross-section fidelity.

Possible applications of this methodology could be in Virtual or Augmented Reality (VR/AR) display of cultural heritage sites. A user could, for example, navigate a VR model advancing the camera forward, slicing through the facade to reveal the cross-section and interior of rooms beyond the facade. This would be more appealing and realistic when combined with the suggestions made in the next section, future work, particularly the suggestion on point-based rendering (models would still need to capture colour to render more realistically).

The evaluation demonstrates that a meshfree, point-based approach for extracting cross-sections is a valid approach, providing a close approximation to the cross-section derived from the meshed point cloud. This work suggests that if the full resolution point cloud is available, it should be possible to produce high fidelity cross-section images.

6.1. Future work

While developing this work some aspects were deemed out of scope but may be beneficial and worth further study.

The following suggestions are made for future work:

- Develop a mesh reconstruction algorithm to apply on the recovered slice. This would be used in the evaluation of the point-based cross-section against the mesh-based cross-section, thus avoiding the need to use an external application (Meshlab.net) for mesh-reconstruction of the slice. The need for manual registration would then also be avoided, removing any possible error in manual registration of the cross-sections, since both the point-based cross-section and mesh-based cross-section would be recovered within the same system (coordinate system and camera settings).

- For rendering of the data sets, employ point-based rendering of the scene e.g. Gaussian Splatting.

- Automate calibration of variable K (structuring element size used in binary morphology Closing) and slice thickness t, according to the point density of the model.

References

Adhikary, N. & Gurumoorthy, B. 2016. A slice based approach to recognize and extract free-form volumetric features in a CAD mesh model. Computer-Aided Design and Applications. 13(5):587-599. DOI: 10.1080/16864360.2016.1150703.

Amer, A. 2002. New binary morphological operations for effective low-cost boundary detection. *International Journal of Pattern Recognition and Artificial Intelligence*. 17(2):1-13. Available: http://www.ece.concordia.ca/~amer [2024, January 23].

Asaeedi, S., Didehvar, F. & Mohades, A. 2017. α-Concave hull, a generalization of convex hull. *Theoretical Computer Science*. 702:48-59. DOI: 10.1016/J.TCS.2017.08.014.

Baltsavias, E.P. 1999. A comparison between photogrammetry and laser scanning. *ISPRS Journal of Photogrammetry and Remote Sensing*. 54(2-3):83-94. DOI: 10.1016/S0924-2716(99)00014-3.

Banfi, F., Brumana, R. & Stanga, C. 2019. Extended reality and informative models for the architectural heritage: from scan-to-BIM process to virtual and augmented reality. *Virtual Archaeology Review*. 10(21):14-30. DOI: 10.4995/VAR.2019.11923.

Barber, C.B., Dobkin, D.P. & Huhdanpaa, H. 1996. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*. 22(4):469-483. DOI: 10.1145/235815.235821.

Boissonnat, J.D. & Ghosh, A. 2010. Manifold reconstruction using tangential Delaunay complexes. *Proceedings of the Annual Symposium on Computational Geometry*. 324-333. DOI: 10.1145/1810959.1811013.

Bolelli, F., Allegretti, S., Baraldi, L. & Grana, C. 2020. Spaghetti Labeling: Directed Acyclic Graphs for Block-Based Connected Components Labeling. *IEEE Transactions on Image Processing*. 29(1):1999-2012. DOI: 10.1109/TIP.2019.2946979.

Burtsev, S. V. & Kuzmin, Y.P. 1993. An efficient flood-filling algorithm. *Computers & Graphics*. 17(5):549-561. DOI: 10.1016/0097-8493(93)90006-U.

Chen, S. & Haralick, R.M. 1995. Recursive Erosion, Dilation, Opening, and Closing Transforms. *IEEE Transactions on Image Processing*. 4(3):335-345. DOI: 10.1109/83.366481.

Choi, H., Lee, H.J., You, H.J., Rhee, S.Y. & Jeon, W.S. 2019. Comparative analysis of generalized intersection over union and error matrix for vegetation cover classification assessment. *Sensors and Materials*. 31(11):3849-3858. DOI: 10.18494/SAM.2019.2584.

Coady, J., O'Riordan, A., Dooly, G., Newe, T. & Toal, D. 2019. An overview of popular digital image processing filtering operations. *Proceedings of the International Conference on Sensing Technology, ICST*. 2019-December. DOI: 10.1109/ICST46873.2019.9047683.

Fishkin, K.P. & Barsky, B.A. 1984. A family of new algorithms for soft filling. *ACM SIGGRAPH Computer Graphics*. 18(3):235-244. DOI: 10.1145/964965.808604.

Han, X.F., Jin, J.S., Wang, M.J., Jiang, W., Gao, L. & Xiao, L. 2017. A review of algorithms for filtering the 3D point cloud. *Signal Processing: Image Communication*. 57:103-112. DOI: 10.1016/J.IMAGE.2017.05.009.

Haralick, R.M., Sternberg, S.R. & Zhuang, X. 1987. Image Analysis Using Mathematical Morphology. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. PAMI-9(4):532-550. DOI: 10.1109/TPAMI.1987.4767941.

He, G. & Yang, J. 2019. Application of the slicing technique to extract the contour feature line. *Cluster Computing*. 22(6):13937-13943. DOI: 10.1007/s10586-018-2144-9.

He, L., Ren, X., Gao, Q., Zhao, X., Yao, B. & Chao, Y. 2017. The connected-component labeling problem: A review of state-of-the-art algorithms. *Pattern Recognition*. 70:25-43. DOI: 10.1016/J.PATCOG.2017.04.018.

Heijmans, H.J.A.M. 1995. Mathematical Morphology: A Modern Approach in Image Processing Based on Algebra and Geometry. *SIAM Review*. 37(1):1-36. Available: http://www.jstor.org.ezproxy.uct.ac.za/stable/2132751.

Hermosilla, T., Ruiz, L.A., Recio, J.A. & Estornell, J. 2011. Evaluation of Automatic Building Detection Approaches Combining High Resolution Images and LiDAR Data. *Remote Sensing 2011, Vol. 3, Pages 1188-1210*. 3(6):1188-1210. DOI: 10.3390/RS3061188.

Kazhdan, M. & Hoppe, H. 2013. Screened poisson surface reconstruction. *ACM Transactions on Graphics (TOG)*. 32(3). DOI: 10.1145/2487228.2487237.

Khudeev, R. 2005. A new flood-fill algorithm for closed contour. *IEEE International Siberian Conference on Control and Communications 2005, SIBCON 05 - Proceedings*. 2005:170-174. DOI: 10.1109/SIBCON.2005.1611214.

Kobbelt, L. & Botsch, M. 2004. A survey of point-based techniques in computer graphics. *Computers & Graphics*. 28(6):801-814. DOI: 10.1016/J.CAG.2004.08.009.

Kokab, H.S. & Urbanic, R.J. 2019. Extracting of Cross Section Profiles from Complex Point Cloud Data Sets. *IFAC-PapersOnLine*. 52(10):346-351. DOI: https://doi.org/10.1016/j.ifacol.2019.10.055.

Kresslein, J., Haghighi, P., Park, J., Ramnath, S., Sutradhar, A. & Shah, J.J. 2018. Automated cross-sectional shape recovery of 3D branching structures from point cloud. *Journal of Computational Design and Engineering*. 5(3):368-378. DOI: 10.1016/j.jcde.2017.11.010.

Kyriazis, I. & Fudos, I. 2013. Building Editable Free-form Models from Unstructured Point Clouds. *Computer-aided Design and Applications*. 10(6):877-888. Available: 10.3722/cadaps.2013.877-888%0A1.

Kyriazis, I., Fudos, I. & Palios, L. 2009. Extracting CAD features from point cloud cross-sections. 137-144. Available: https://otik.uk.zcu.cz/handle/11025/10899.

Li, B., Wei, J., Wang, L., Ma, B. & Xu, M. 2019. A comparative analysis of two point cloud volume calculation methods. *International Journal of Remote Sensing*. 40(8):3227-3246. DOI: 10.1080/01431161.2018.1541111.

Li, B., Zhao, X., Chen, Y., Wei, J., Wang, L. & Ma, B. 2019. Journal of Geodesy and Geoinformation Science, (2019), 31-43, 2(4). Available: https://doi.org/10.11947/j.JGGS.2019.0404 [2024, May 20].

Lin, H.W., Tai, C.L. & Wang, G.J. 2004. A mesh reconstruction algorithm driven by an intrinsic property of a point cloud. *Computer-Aided Design*. 36(1):1-9. DOI: 10.1016/S0010-4485(03)00064-2.

Lu, T., Tai, C.-L., Li, B., Su, F. & Cai, S. 2005. 3D Reconstruction of Detailed Buildings from Architectural Drawings. *Computer-Aided Design & Applications*. 2:527-536. Available: moz-extension://6c849b50-98c0-4b95-b2e6-a4f5e4e3d3f4/enhanced-reader.html?openApp&pdf=https%3A%2F%2Fwww.cad-journal.net%2Ffiles%2Fvol_2%2FCAD_2(1-4)_2005_527-536.pdf [2024, January 13].

Lv, C., Lin, W. & Zhao, B. 2022. Voxel Structure-Based Mesh Reconstruction from a 3D Point Cloud. *IEEE Transactions on Multimedia*. 24:1815-1829. DOI: 10.1109/TMM.2021.3073265.

McReynolds, T. & Blythe, D. 2005. *Advanced graphics programming using OpenGL*. Elsevier. Available: moz-extension://6c849b50-98c0-4b95-b2e6-a4f5e4e3d3f4/enhanced-reader.html?openApp&pdf=https%3A%2F%2Fwww.r-5.org%2Ffiles%2Fbooks%2Fcomputers%2Falgo-list%2Frealtime-3d%2FTom_McReynolds-Advanced_Graphics_Programming-EN.pdf [2024, January 31].

Mineo, C., Pierce, S.G. & Summan, R. 2019. Novel algorithms for 3D surface point cloud boundary detection and edge reconstruction. *Journal of Computational Design and Engineering*. 6(1):81-91. DOI: 10.1016/j.jcde.2018.02.001.

Minetto, R., Volpato, N., Stolfi, J., Gregori, R. & da Silva, M. 2017. An Optimal Algorithm for 3D Triangle Mesh Slicing. *Computer-Aided Design*. 92. DOI: 10.1016/j.cad.2017.07.001.

Moreira, A. & Santos, M.Y. 2007. Concave hull: a k-nearest neighbours approach for the computation of the region occupied by a set of points. *Proceedings of the 2nd International Conference on Computer Graphics Theory and Applications (GRAPP 2007), Barcelona, Spain*. (March, 8). Available: https://repositorium.sdum.uminho.pt/handle/1822/6429 [2024, January 22].

Oropallo, W., Piegl, L.A., Rosen, P. & Rajab, K. 2018. Point cloud slicing for 3-D printing. *Computer-Aided Design and Applications*. 15(1):90-97. DOI: 10.1080/16864360.2017.1353732.

Otero, J. 2022. Heritage Conservation Future: Where We Stand, Challenges Ahead, and a Paradigm Shift. *Global Challenges*. 6(1):2100084. DOI: 10.1002/GCH2.202100084.

Pletinckx, D., Callebaut, D., Killebrew, A.E. & Silberman, N.A. 2000. Virtual-reality heritage presentation at Ename. *IEEE Multimedia*. 7(2):45-48. DOI: 10.1109/93.848427.

Qiu, Y. 2022. 3D Reconstruction and Intelligent Digital Conservation of Ancient Buildings Based on Laser Point Cloud Data. *Journal of Electrical and Computer Engineering*. 2022. DOI: 10.1155/2022/7182018.

Ramamurthy, R., Harding, K., Lucas, V., Du, X., Jia, T., Liao, Y. & Paul, R. 2015. Geometric and topological feature extraction of linear segments from 2D cross-section data of 3D point clouds. In *Proceedings of SPIE - The International Society for Optical Engineering*. V. 9489. GE Global Research, China. DOI: 10.1117/12.2179987.

Sainz, M. & Pajarola, R. 2004. Point-based rendering techniques. *Computers & Graphics*. 28(6):869-879. DOI: 10.1016/J.CAG.2004.08.014.

Snyder, W. 1992. Closing gaps in edges and surfaces. *Image and Vision Computing*. (January, 1). Available: https://www.academia.edu/10978474/Closing_gaps_in_edges_and_surfaces [2024, February 28].

Di Stefano, L. & Bulgarelli, A. 1999. A simple and efficient connected components labeling algorithm. *Proceedings - International Conference on Image Analysis and Processing, ICIAP 1999*. 322-327. DOI: 10.1109/ICIAP.1999.797615.

Suchde, P., Jacquemin, T. & Davydov, O. 2022. Point Cloud Generation for Meshfree Methods: An Overview. *Archives of Computational Methods in Engineering 2022 30:2*. 30(2):889-915. DOI: 10.1007/S11831-022-09820-W.

Sui, W., Wang, L., Fan, B., Xiao, H., Wu, H. & Pan, C. 2015. Layer-Wise Floorplan Extraction for Automatic Urban Building Reconstruction. *IEEE Transactions on Visualization and Computer Graphics*. 22:1. DOI: 10.1109/TVCG.2015.2505296.

Veltkamp, R.C. & Hagedoorn, M. 2001. State of the art in shape matching. *Principles of visual information retrieval*. 87-119. Available: https://webspace.science.uu.nl/~veltk101/publications/art/vir2001.pdf [2024, March 06].

Vierling, K.T., Vierling, L.A., Gould, W.A., Martinuzzi, S. & Clawges, R.M. 2008. Lidar: Shedding new light on habitat characterization and modeling. *Frontiers in Ecology and the Environment*. 6(2):90-98. DOI: 10.1890/070001.

Wang, Y., Li, H., Ning, X. & Shi, Z. 2011. A new interpolation method in mesh reconstruction from 3D point cloud. *Proceedings of VRCAI 2011: ACM SIGGRAPH Conference on Virtual-Reality Continuum and its Applications to Industry*. 235-242. DOI: 10.1145/2087756.2087790.

Van Der Weken, D., Nachtegael, M. & Kerre, E.E. 2004. Using similarity measures and homogeneity for the comparison of images. *Image and Vision Computing*. 22(9):695-702. DOI: 10.1016/J.IMAVIS.2004.03.002.

Westoby, M.J., Brasington, J., Glasser, N.F., Hambrey, M.J. & Reynolds, J.M. 2012. 'Structure-from-Motion' photogrammetry: A low-cost, effective tool for geoscience applications. *Geomorphology*. 179:300-314. https://uct.primo.exlibrisgroup.com/permalink/27UCT_INST/o2vtft/cdi_openaire_primary_doi_dedup_fbd852140cd9c49c61b4f1a9e881f9ae [2024, May 20].

Wu, Y.F., Wong, Y.S., Loh, H.T. & Zhang, Y.F. 2004. Modelling cloud data using an adaptive slicing approach. *Computer-Aided Design*. 36(3):231-240. https://www-sciencedirect-com.ezproxy.uct.ac.za/science/article/pii/S0010448503000976 [2024, May 20].

Xu, J., Ma, Y., He, S. & Zhu, J. 2019. 3D-GIoU: 3D Generalized Intersection over Union for Object Detection in Point Cloud. *Sensors 2019, Vol. 19, Page 4093*. 19(19):4093. https://www.mdpi.com/1424-8220/19/19/4093/htm [2024, May 20].

Xu, Y., Tong, X. & Stilla, U. 2021. Voxel-based representation of 3D point clouds: Methods, applications, and its potential use in the construction industry. *Automation in Construction*. 126:103675. Available: https://www-sciencedirect-com.ezproxy.uct.ac.za/science/article/pii/S0926580521001266?via%3Dihub [2024, May 20].

Yang, S., Xu, S. & Huang, W. 2022. 3D Point Cloud for Cultural Heritage: A Scientometric Survey. *Remote Sensing 2022, Vol. 14, Page 5542*. 14(21):5542. Available: https://www.mdpi.com/2072-4292/14/21/5542/htm [2024, May 20].

Yin, X., Wonka, P. & Razdan, A. 2009. Generating 3D Building Models from Architectural Drawings: A Survey. *IEEE Computer Graphics and Applications*. 29(1):20-30. Available: https://ieeexplore-ieee-org.ezproxy.uct.ac.za/document/4736453 [2024, May 20]

Yuwen, S., Dongming, G., Zhenyuan, J. & Weijun, L. 2006. B-spline surface reconstruction and direct slicing from point clouds. *International Journal of Advanced Manufacturing Technology*. 27(9/10):918-924. Available: http://ezproxy.uct.ac.za/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=aph&AN=19537685&site=ehost-live [2024, May 20].

Zhang, Y., Xia, B. & Taylor, S. 2024. High-resolution 3-D geometry updating of digital functional models using point cloud processing and surface cut. *Computer-Aided Civil and Infrastructure Engineering*. 39(1):3-19. Available: https://onlinelibrary-wiley-com.ezproxy.uct.ac.za/doi/full/10.1111/mice.13076 [2024, May 20].

Zhi, Y., Zhang, Y., Chen, H., Yang, K. & Xia, H. 2016. A Method of 3D Point Cloud Volume Calculation Based on Slice Method. *BT - 2016 International Conference on Intelligent Control and Computer Application (ICCA 2016)*. Atlantis Press. 155-158. Available: https://www.semanticscholar.org/paper/A-Method-of-3D-Point-Cloud-Volume-Calculation-Based-Zhi-Zhang/d33e81c3cacc7518ff37328984608426a280e596 [2024, May 20].

Appendix 1: Pseudocode

Input variables are highlighted in blue.

Output variables are highlighted in orange.

Pseudocode: ***Slice the Cloud:*** *Capture 2D slice of 3D point cloud*

```
Void screenshot(void)
{
// Input variable: cv::Mat screencaptureData {cv::Size(img_width, img_height), CV_8UC4, cv::Scalar(0,0,0,255)};
// This function captures the screen after setting the position of the slicing plane. The user positioning of the
// slicing plane clips the points (removes them from the screen). The screen then displays only those points
// within the bounds of the slicing plane.

    //initialise a matrix in OpenCV to store the screenshot of the 2D slice, 4 channels for RGBA

    screencaptureData = {cv::Size(img_width, img_height), CV_8UC4, cv::Scalar(0,0,0,255)};


    //capture the screen - glreadpixels(0, 0, img.cols, img.rows, gl_bgr, gl_unsigned_byte, img.data);

    glReadPixels(window_width*wScaleL, //capture the portion of the screen where the slice is drawn

                 0,

                 screencaptureData.cols,

                 screencaptureData.rows,

                 GL_RGBA,                    //number of channels = 4 for RGBA colour

                 GL_UNSIGNED_BYTE,          //RGBA data structure

                 screencaptureData.data);


    add_alpha_channel(); //if screenshot does not have an alpha channel add one

    convert_RGBA_to_BGRA(); //OpenCV uses BGRA format for colour

    set_folder_to_store_cross-section(); //create a new folder e.g. "./TopView/1" for storing the cross-section

    while(key != ESC)

        show_cross-section_in_new_window();

// Output variable: cv::Mat screencaptureData {cv::Size(img_width, img_height), CV_8UC4, cv::Scalar(0,0,0,255)};
// with alpha channel added
}
```

```
void crossSection(glm::vec3 clip_plane, float sliceSize)
{
// Input variable: cv::Mat screencaptureData {cv::Size(img_width, img_height), CV_8UC4, cv::Scalar(0,0,0,255)};
// glm::vec3 clip_plane – the slice plane as defined by the point (xyz coordinates) stored in clip_plane.
// float sliceSize – the thickness of the slice plane.
// The user positions the slicing plane, the point cloud orientation is based on the user selected camera angle
// (Front View, Side View or Top View), and the section of the point cloud outside the bounds of the slicing plane
// are removed (i.e. “clipped”) i.e. removed from view / set to not be displayed. The screen then displays only
// those points within the bounds of the slicing plane.

    if(cameraOrientation == CAMERA_FACING_NEG_Z_AXIS) //Top View, clip plane moves along Z axis

        user_input_moves_clip_plane_along_Z_axis();

    else if(cameraOrientation == CAMERA_FACING_NEG_Y_AXIS) //Front View, clip plane moves along Y axis

        user_input_moves_clip_plane_along_Y_axis();

    else if(cameraOrientation == CAMERA_FACING_NEG_X_AXIS) //Front View, clip plane moves along X axis

        user_input_moves_clip_plane_along_X_axis();


    for (j = 0; j < vertex_buffer_data1.size(); j++) //iterate through all points in the point cloud
    {

        if(clip_plane_lies_within_the_pointcloud)

            set_points_lying_within_the_clip_plane_to_green(); //highlight the slice


        if (showslice)

            clip_points_lying_outside_the_clip_plane(); //set point so that it does not render to screen

        else if(show_cloud_behind_place)

            clip_points_in_front_of_clip_plane(); //show the portion of the point cloud lying behind the

                                                  //clip plane

    }

// Output variables:
// std::vector<glm::vec4> vertex_buffer_data1 //x,y,z,w, std::vector<glm::vec4> color_buffer_data; //x,y,z,w
// std::vector<glm::vec4> vertex_buffer_data2; //x,y,z,w
// The two output variables are for displaying (rendering) the orientation and slice result of the slice plane - on
the left and right sides of the visual display (screen) respectively. On the left is the point cloud showing the
position of the slicing plane – entire point cloud is intact and slice plane position highlighted in green. On the
right is the cross-section slice – only the points between the slice are visible on screen.
}
```

Pseudocode: **PBT Cross-section:** Close cross-section boundary

```
// Input variable:

void Close(int n)
{
    Dilation(n);

    Erosion(n);

}


//Binary morphology: Dilation - "fatten" the image

void dilation(int n)
{
// Input variable: cv::Mat screencaptureData {cv::Size(img_width, img_height), CV_8UC4, cv::Scalar(0,0,0,255)};
// screencaptureData holds the slice

   int dilation_size = KERNEL_SIZE; //KERNEL_SIZE is a global variable
   cv::Mat dilation_dst;
    //set up the Structuring Element (SE) for the dilation
    cv::Mat element = getStructuringElement(MORPH_RECT, //rectangular shaped SE for symmetrical dilation
   Size(2 * dilation_size + 1, 2 * dilation_size + 1), //3x3 for K=1
   Point(dilation_size, dilation_size)); //set the anchor as the centre


    if(n==1) //depth n=1
    {
        cv::dilate(screencaptureData, dilation_dst, element, 1); //dilate with depth n=1
        cv::imwrite(crossSectionsFolder+"D1-Dilationx1.png", dilation_dst); //write image to file
    }
    else if(n==3)
    {
        //Apply the dilation operation, depth n=3 (3 successive dilations)
        cv::dilate(screencaptureData, dilation_dst, element, 3);
        cv::imwrite(crossSectionsFolder+"D3-Dilationx3.png", dilation_dst); //writ ethe image to file
    }
// Output variables:
// cv::Mat dilation_dst holds the result of the dilation operation performed on the cross-section
// screencaptureData;
}
```

```
//Binary morphology: Erosion - "thin" the image

void erosion(int n)
{
// Input variable: cv::Mat screencaptureData {cv::Size(img_width, img_height), CV_8UC4, cv::Scalar(0,0,0,255)};
// screencaptureData holds the slice and the result of recursive Dilations.

    int erosion_size = KERNEL_SIZE; // KERNEL_SIZE is a global variable


    cv::Mat erosion_dst;


    //set up the Structuring Element (SE) for the erosion

    cv::Mat element = getStructuringElement(MORPH_RECT, //rectangular shaped SE for

                                                        //symmetrical erosion

    Size(2 * erosion_size + 1, 2 * erosion_size + 1), //3x3 for K=1

    Point(erosion_size, erosion_size)); //set the anchor as the centre


    if(n==1) //depth n=1

    {

        cv::erode(screencaptureData, erosion_dst, element, 1); //erode with depth n=1

        cv::imwrite(crossSectionsFolder+"E1-Erosionx1.png", erosion_dst); //write image to file

    }

    else if(n==3)

    {

        //Apply the erosion operation, depth n=3 (3 successive erosions)

        cv::erode(screencaptureData, erosion_dst, element, 3); //erode with depth n=3

        cv::imwrite(crossSectionsFolder+"E3-Erosionx3.png", erosion_dst); //write image to file

    }

// Output variables:
// cv::Mat erosion_dst holds the result of the dilation operation performed on the cross-section
// screencaptureData;

}
```

Pseudocode: ***PBT Cross-section:*** *Fill holes*

```
void Fill_Holes_After_Morphology(void)
{
// Input variable: cv::Mat crossSection_img {cv::Size(img_width, img_height), CV_8UC4, cv::Scalar(0,0,0,255)};
// crossSection_img holds the slice after recursive (binary morphology) Opening and Closing operations.

    //read in image in grayscale
    crossSection_img = cv::imread(crossSectionsFolder+"E3-Closed.png",
                                        cv::IMREAD_GRAYSCALE);

    convert_img_to_binary(); //covert the grayscale image to binary, for 'Connected Components
                                    //Analysis' which in turn is needed for flood-filling of holes

    invert_binary_img(); //Invert image so that holes are '1' pixels

    getConnectedComponents(cv::Mat &crossSection_img);

    if(Oriented_for_Front_View or Oriented_for_Side_View)
        fillGround_maxMin2DCoOrds();

// Output variables:
// cv::Mat crossSection_img holds the cross section profile, now with the ground bounded and holes filled.
}


int getConnectedComponents(cv::Mat binary_negative_img)
{
// Input variable: cv::Mat binary_negative_img hold the cross section profile after binary morphology – here a
// connected component analysis is performed to identify holes in the cross -section profile.

    cv::Mat labels(crossSection_img.size(), CV_32S);

    cv::Mat stats, centroids;

    int nrLabels = 0, labelsFilled = 0;

    centroids = cv::Mat();

    stats = cv::Mat();

    labels = cv::Mat();
```

```
    unsigned int outlier = 0; //could be set by expert user to avoid filling in small rooms / enclosures
                              //not meant to be filled

    nrLabels = cv::connectedComponentsWithStats(binary_negative_img, labels, stats, centroids, 8);

    if(nrLabels > 1)  //nrLabels == 1 means no holes, just the background
    {
        outlier = calculate_and_remove_small_rooms(); //Do not fill 'small rooms' which may appear
                                                      //as "holes to fill" in the
                                                      //connectedComponentsWithStats(…)
                                                      //OpenCV API

        for(int i = 1; i < nrLabels; ++i) // note: i = 0 is the centroid of the background
        {
            if((stats.at<int>(i, cv::CC_STAT_AREA) <= outlier) &&
              (stats.at<int>(i, cv::CC_STAT_AREA) < userThreshhold))
            //Only fill "holes" in walls - not false positive enclosed areas e.g. a courtyard which is
             //enclosed by /walls and should be filled
            {
                cv::floodFill(binary_negative_img,
                              (cv::Point2d)centroids.row(i), cv::Scalar(0.0, 0.0, 0.0));
                labelsFilled++;
            }
        }
    }
    return(labelsFilled);
// Output variables:
// cv::Mat binary_negative_img hold the cross section profile, now with holes filled.
}
```

Appendix 2: Software functionality and associated keyboard user interface

Ctrl+Z - Orientate the viewing camera for Top View (camera facing negative Z direction)

Ctrl+Y - Orientate the viewing camera for Front View (camera facing negative Y direction)

Ctrl+X - Orientate the viewing camera for Side View (camera facing negative X direction)

+/= - increase camera position

-/_ - decrease camera position

C - capture cross-section

Shift+C - Capture cross-section and perform an additional round of closing (depth N=1) after the Closing of depth N=3.

1 - toggle between slice plane view and backdrop (behind slice plane) view

2 - toggle slice vs plane cross-section

3 - adjust slice thickness (scroll through 0.5 to 1.5 in increments of 0.1

4 - Highlight points (in red) where the distance between the points are >75% of the max.

5 - Output point density to terminal window. Each bin is 1/100th of the respective Bounding Box dimension.

O - Switch to Orthographic camera

P - Switch to Perspective camera

R - Reset view

0 - Dilate

9 - Erode

] - increase point size

[- decrease point size

In this work the terminology used when viewing a point cloud of a cultural heritage site is Top View, Front View and Side View - where Top View sets the view (camera position) facing down the negative Z axis, Front View facing down the negative Y axis and Side View facing down the negative X axis.

Appendix 3: Results (expanded tables)

Table 8: Results and settings of PBT cross-section evaluation - Mosque 3 Kua Ruins data set

	Clip Plane position	t	K	IoU	FP	TN	Intersection
Top View							
Cross-section(TV01)	Z = 0.3122	0,5	1	**90,7%**	3216	4227	72480
Front View							
Cross-section(FV01)	Y = 0.6265	0,5	1	**95,1%**	8643	1156	192027
Cross-section(FV02)	Y = -1.5734	0,5	1	**95,5%**	839	2195	64676
Side View							
Cross-section(SV01)	X = 2.6288	0,5	1	**96,8%**	1913	560	75216
Cross-section(SV02)	X = -0.5712	0,5	1	**95,8%**	1496	1315	63381

Table 9: Results and settings of PBT cross-section evaluation - Mosque 2 Kua Ruins data set

	Clip Plane position	t	K	IoU	FP	TN	Intersection
Top View							
Cross-section(TV01)	Z = 1.2581	0,5	1	**95,1%**	1865	1416	63941
Front View							
Cross-section(FV01)	Y = 8.3773	0,5	1	**95,9%**	2061	805	66658
Cross-section(FV02)	Y = 5.0773	0,5	1	**97,2%**	489	723	42462
Cross-section(FV03)	Y = -2.0227	0,5	1	**97,2%**	196	740	32857
Cross-section(FV04)	Y = -5.8227	0,5	1	**94,5%**	820	1161	34126
Side View							
Cross-section(SV01)	X = 4.0061	0,5	1	**97,7%**	1710	408	88863
Cross-section(SV02)	X = -0.6939	0,5	1	**98,5%**	1984	648	172400
Cross-section(SV03)	X = -1.3939	0,5	1	**98,9%**	1028	439	129227

Table 10: Results and settings of PBT cross-section evaluation - House 4 Kua Ruins data set

	Clip Plane position	t	K	IoU	FP	TN	Intersection
Top View							
Cross-section(TV01)	Z = 0.7245	0.5	1	**67.9%**	673	16470	36191
Cross-section(TV02)	Z = 0.7245	0.5	1	**82.6%**	10762	256	52405
Cross-section(TV03)	Z = 0.7245	0.5	2	**73.6%**	14715	630	42681
Cross-section(TV04)	Z = 1.2245	1.0	1	**88.3%**	5439	259	43061
Front View							
Cross-section(FV01)	Y = 3.5551	0.5	1	**90.1%**	563	546	10045
Cross-section(FV02)	Y = 1.5449	0.5	1	**94.6%**	1520	124	28764
Cross-section(FV03)	Y = -9.7449	0.5	1	**94.6%**	0	992	17436
Side View							
Cross-section(SV01)	X = 14.0587	0.5	1	**93.3%**	141	1062	16785
Cross-section(SV02)	X = 6.1587	0.5	1	**90.1%**	1828	236	18713
Cross-section(SV03)	X = -1.5413	0.5	1	**94.7%**	2329	118	44160
Cross-section(SV04)	X = -4.7414	0.5	1	**91.1%**	5221	640	60294

Table 11: Results and settings of PBT cross-section evaluation - Santa Maria De Melque (Spain) data set

	Clip Plane position	t	K	IoU	FP	TN	Intersection
Top View							
Cross-section(TV01)	Z = 0.2592	1.0	1	**96.7%**	1927	1789	109693
Front View							
Cross-section(FV01)	Y = 7.7814	1.0	1	**97.5%**	679	675	53841
Side View							
Cross-section(SV01)	X = 8.0185	1.0	1	**97.5%**	245	667	35791