BUILT INFRASTRUCTURE POINT CLOUD DATA CLEANING: AN OVERVIEW OF GAP FILLING ALGORITHMS

Abbas Rashidi
Georgia Institute of Technology, USA

Ioannis Brilakis
University of Cambridge, UK

Patricio Vela
Georgia Institute of Technology, USA

ABSTRACT: *Video captured from infrastructure scenes can be used to generate point cloud data (PCD) as a potential solution for acquiring spatial information of built infrastructure; however, video based PCD is incomplete and includes gaps, outliers and poor/non-reconstructed areas. This phenomenon has a negative impact on both visualization and measurement practices and is mainly caused by a number of reasons including insufficient coverage of all views while videotaping the scene, lack of sufficient features on uniform surfaces and possible errors in calibration, matching and optimization algorithms.*

To tackle this issue, researchers suggested various post processing algorithms for reconstructing missing surfaces and filling gaps/holes. This paper provides an overview on these algorithms summarize their properties in terms of efficiency, ability to work in complex geometry settings and running time. As the comparison study, three most common hole filling algorithms: MSL, GG and RFR were implemented and tested on a number of real built infrastructure scenes as the case studies. Number of generated 3D points for filling the gaps, proper distribution of points on covered surfaces and running time are three major comparison metrics has been taken into account. Results indicate that in general PML outperforms other algorithms on both flat and curved surfaces.

KEYWORDS: *Built infrastructure, triangulation, gap, Point Cloud Data, surface reconstruction*.

1. INTRODUCTION

Spatial sensing of built environments is an active field of practice and research in civil engineering with a wide range of applications including: 3D as built documentation of civil infrastructure, progress monitoring of construction projects, effective design of job site layout, structural damage assessment and quality control of construction products. Results of spatial sensing of built infrastructure are ultimately presented in the form of point cloud data (PCD) (Brilakis et al., 2011)). Two major approaches are utilized for reconstructing built environments and generating PCD: using active sensors (e.g. laser scanners) and using passive sensors (e.g. digital camera). It is well known that using laser scanners results in generating denser PCD with higher qualities however, neither of the approaches is able to generate nearly perfect PCD. Some areas might be poorly reconstructed and there are always gaps or holes on the surfaces of PCD. This phenomenon usually happens due to a number of reasons including accessibility issues and occlusions (for laser scanners) and insufficient coverage of the scene, occlusions, lack of sufficient number of textures on the surfaces of elements and possible errors in calibration and optimization algorithms (for image/video based techniques).

One possible solution to this issue would be generating the PCD, identifying the missing/poorly reconstructed areas and trying to re-scan the scene to acquire extra 3D points. This solution is not always feasible since there might be changes in the original scene over the time. In addition, re-scanning the scene is not helpful if gaps/hole occurred due to reasons such as lack of texture or computational errors. As the result, adding a post processing step for cleaning generated PCD and filling gaps/holes is crucial.

For most applications, gaps on surfaces of PCD are not appropriate. Depending on the size/location, gaps may cause two problems: They negatively impact the quality of PCD when it comes to visual representations. PCD

Citation: Rashidi, A., Brilakis, I. & Vela, P. (2013). Built infrastructure point cloud data cleaning: an overview of gap filling algorithms. In: N. Dawood and M. Kassem (Eds.), Proceedings of the 13th International Conference on Construction Applications of Virtual Reality, 30-31 October 2013, London, UK.

entail several gaps and poorly reconstructed areas are not eye catching. The second problem is related to proper extraction of geometric information from PCD. Gaps, holes and incomplete areas might be located on edges and intersections of elements and those locations are extremely significant for conducting length measurements and extracting geometric information of the scene (Figure 1).

Fig. 1: It is not possible to extract the geometric information of poorly reconstructed areas.

To tackle the issue, a number of algorithms are proposed for filling the gaps on smooth surfaces. This paper provides an overview of existing algorithms, as well as advantages and shortcoming of these methods for real applications in the area of civil infrastructure. To these end, three of the most popular algorithms are implemented and tested on different real cases of civil infrastructure. The rest of the paper is organized as follows:

A brief overview of current algorithms for filling the gaps, as well as major characteristics of each algorithm is presented in section 2. Sections 3 and 4 briefly summarize our approach for implementing the three most common algorithms for filling the holes and explain the procedure for conducting experiments on real built infrastructure datasets to evaluate the performance of the above mentioned algorithms. Results of the comparison study are summarized in section 5, and finally conclusions and plans for future research are presented in section 6.

2. FILLING THE GAPS ON SURFACES OF POINT CLOUD DATA

The process of hole-filling on PCD can be divided into two steps: 1) determining locations of gaps/holes and 2) smoothly covering the hole and reconstructing missing parts using available data (Wang and Oliveira (2002)). For several applications, holes/gaps on surfaces of built infrastructure PCD present simple topology, i.e. the holes are located on flat surfaces. The situation is more challenging when holes are located on intersection of different surfaces, or geometrically complex surfaces. Existing algorithms for filling holes/gaps are classified into three major categories:

i) Algebraic methods which reconstruct the missing surfaces by fitting appropriate functions based on sets of neighbor points as the input. This approach works very well for holes with simple structure located on flat/curved surfaces but is not able to successfully cover the holes located on more geometry complex or twisted surfaces.

ii) Computational geometry: Under this class of algorithms, different computational approaches such as region growing and Delaunay triangulation and usually leave under sampled areas.

iii) Implicit functions: this group of algorithms uses various implicit functions for reconstructing missing parts and covering holes. This category is the most popular approach for filling gaps/holes however; these groups of algorithms are not able to reconstruct the missing parts in the case that surfaces have boundaries.

General properties of different algorithms suggested by researchers are presented in table 1:

Table 1: An overview of existing algorithms for filling gaps/holes on PCD

Category	algorithm	Reference	Input data	Holes with boundaries?	Geometric complex surfaces?
Computational geometry	Voronoi-Based surface reconstruction	Amenta et al. (1998)	meshes	yes	no
	Delaunay triangulation	Edelsbrunner and Muche (1994)	points	no	no
	Projection-based approach for region growing	Gopi and Krishnan(2000)	meshes	yes	no
	Graph-based approach for region growing	Mencl (1995)	meshes	yes	no
Algebraic methods	Ball Pivoting Approach	Bernardini et al. (1999)	points	yes	yes
	Generalized implicit functions approach	Scarloff and Pentland (1991)	meshes	yes	no
	Dynamic implicit functions approach	Terzopoulos and Metaxas (1991)	points	yes	no
Implicit methods	Volumetric Diffusion	Davis et al. (2002)	Partial meshes	yes	yes
	Moving Least Square Method	Wang and Oliveira (2007)	Points and meshes	yes	no
	Compactly supported radial basis functions	Morse et al. (2001)	points	yes	yes

To evaluate the performance of gap filling algorithms, we focus on three most popular methods vastly used in computer graphic domain:

2.1 Moving Least Square Method (Suggested by Wang and Oliveira (2007))

A brief description of Moving Least Square method is presented in this section (Wang and Oliveira (2007)):

In order to fill holes, extra 3D points should be added to the un-sampled regions. For this end, the algorithm first identifies hole's boundaries and their vicinities. For each hole, the algorithm fits a plane through the vicinity points and, for each corresponding point, calculates the distance of the new point to this plane as well as its projection onto the plane. This set of distances forms a height field around the hole which is then applied for surface fitting. Following this procedure, the problem of reconstructing holes in 3D is converted to a simpler interpolation problem. Once a surface has been fitted to the height field using MLS, new points for filling the hole can be obtained by re-sampling the fitted surface. The basic version of the algorithm is presented as the following step-by-step algorithm:

1. Generate the triangle mesh from the input point cloud;

2. Repeat

    3. Automatically identify a whole boundary and its vicinity;

    4. Compute a reference plane for the hole vicinity;

    5. Calculate the distances between the vicinity points and the above mentioned plane;

    6. Fit a surface through this height field using MLS;

7. Cover the hole by re-sampling the fitted surface;

8. until no holes exist.

2.2 The Ball-Pivoting Algorithm (Suggested by Bernardini et al. (1999))

The Ball-Paving Algorithm is based on a pretty simple principle. Assume that the manifold M is the three dimensional surface of an element of the scene and S is a point sampling of M. If we consider S dense enough, we can assume that a ball with ρ radius is not able to pass through the surface without touching sample points. We can also place the ρ-ball in contact with three sample points. That being said, we can keep the ball in contact with two sample points and pivot the ball until it touches the third point. We can pivot around each edge of the hole boundary so the ball contacts form new triangles. The set of triangles which are formed while the ball is traversing on the hole's area continuously cover the surface of the hole. The procedure is also depicted in figure 2 where the pivoting ball touches three vertices of the triangle $\tau = (\sigma_i, \sigma_j, \sigma_0)$whose normal is n. The z axis is perpendicular to the page surface and points towards the viewer and m is the origin of the coordinate system. The circle $s_{ij0}$is located at the intersection of the ρ-ball with z=0. During the pivoting procedure, the ρ-ball touches two edges with endpointsσ_i, σ_j and the ball center represents a circular trajectory γ whose center is m and the radius$\|c_{ij0} - m\|$. During the pivoting stage, the ball hits a new data point,σ_k. If we considerss_k as the intersection of aρ-sphere centered at σ_k with z=0. The centerc_k of the pivoting ball when it hits σ_k is the intersection of γ with s_k located on the negative half-plane of oriented line l_k. More details can be found at Bernardini et al. (1999).

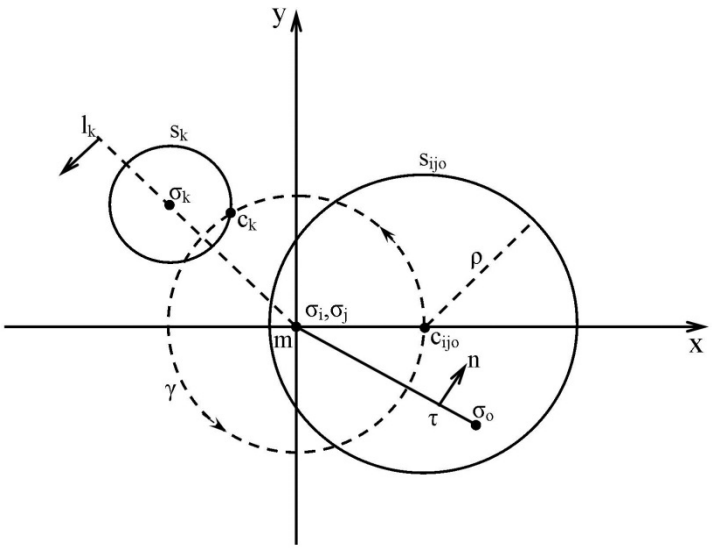

Fig. 2: Pivot Ball Algorithm

2.3 Volumetric Diffusion Method (Suggested by Davis et al. (2002))

As the first step of this method, the surface is first converted into a volumetric representation which is a spaced 3D grid of values of a clamed sign distance function represented as ds(x) (Davis et al. (2002)). This function is merely defined in a narrow band near the observed surface, and it is positive inside the surface and negative outside. Since the units of the function are not important, it is defined to be in the range from -1 to 1 over the width of the band. The thickness of the band does not have a significant impact on the result; so based on the authors' recommendation, 5voxels on either side of the observed surface were considered. As suggested by the authors, the VRIP algorithm was implemented to build this function directly from the PCD. At the same time an associated weight function w(s) was defined, which ranges from 0 to 1 and measures the confidence in the values of ds. In most areas w(s) = 1, but it typically decreases near boundaries of the observed surface, where noise increases.

The goal of the volumetric diffusion algorithm is to extend ds to a function that is defined over the entire volume, though in practice it is only required to compute d near the surface (in fact only near holes in the surface). This can be done by diffusing the values of ds outward from the observed surface into adjoining undefined areas. In

particular, the diffused function propagates inward across the holes, eventually spanning them. Once diffusion is complete, the zero set of this function is the desired hole-free surface. More information about the diffusion method can be found at (Davis et al. (2002)).

3. COMPARISON MATRICS

Built infrastructure elements possess some unique characteristics, which should be taken into account when it comes to evaluating the performance of algorithms for filling/covering the gaps. In many cases, infrastructure scenes consist of elements with flat surfaces, e.g. surface of brick walls, rectangular concrete columns, plywood sheets and gypsum panels. However, curved surfaces also might be found among other types of built infrastructure elements. Circular concrete columns and arches are examples of these curved surfaces. The focus of this paper is on simple flat surfaces and gaps/holes with more complex geometry, e.g. those located on curved surfaces or intersection of several elements, are not within the current scope of our work. It is necessary to mention that by flat surfaces, we mean regular flat surfaces found in built environments such as surfaces of concrete and masonry walls; despite the fact that some of these elements are not mathematically defined as perfect planar surfaces and there might be small curves involved.

To evaluate the performance of the algorithms, two major criteria were considered: 1) running time of the algorithm and 2) their capabilities in finding gaps and properly covering them. The second property is quantitatively measured by using two matrices: number of generated 3D points for filling gaps and uniformity in distribution of the generated 3D points on the covered surfaces of holes. To evaluate the uniform distribution of added 3D points for covering the hole's surfaces, the following approach has been implemented: First the surface of different elements of built infrastructure case studies are partitioned into small square regions (2×2cm in this study). The surface is considered as successfully reconstructed if for each square, there is a corresponding 3D in the improved PCD after filling gaps. The percentage of coverage or completeness is then calculated as the ratio of the number of successfully reconstructed squares over total number of squares.

4. COMPARISON RESULTS

To evaluate the performance of the three major algorithms for filling holes/gaps, a C# based prototype was implemented. It was written in Visual Studio 2010 using Windows Presentation Foundation (WPF) and publicly available libraries such as Open CV 2.0 (wrapped by EmguCV) for access to computer vision tools and DirectX 10 for the graphic display of results (Dai et al., (2013)). To conduct the experiments, we selected five built infrastructure scenes with flat surfaces as our case studies. The selected case studies include concrete stairs (case#1), a masonry wall (case#2), a brick wall (case#3), a rectangular concrete column (case#4) and a plywood panel (case #5). Each scene was videotaped from multiple viewpoints to minimize occlusions. An off-the-shelf Canon Vix-731 ia HF S100 was utilized for data collection purposes. The captured video clips were processed and the PCD for each built infrastructure scene was generated following the procedures explained in Rashidi et al. (2013). Three major hole filling algorithms are then implemented on the generated PCD to cover the existing gaps. Percentages of completeness and number of points before and after implementing the algorithms as well as running time for each algorithm are measured. Snapshots of the generated point clouds before and after using algorithms are depicted in figures 4 and 5:

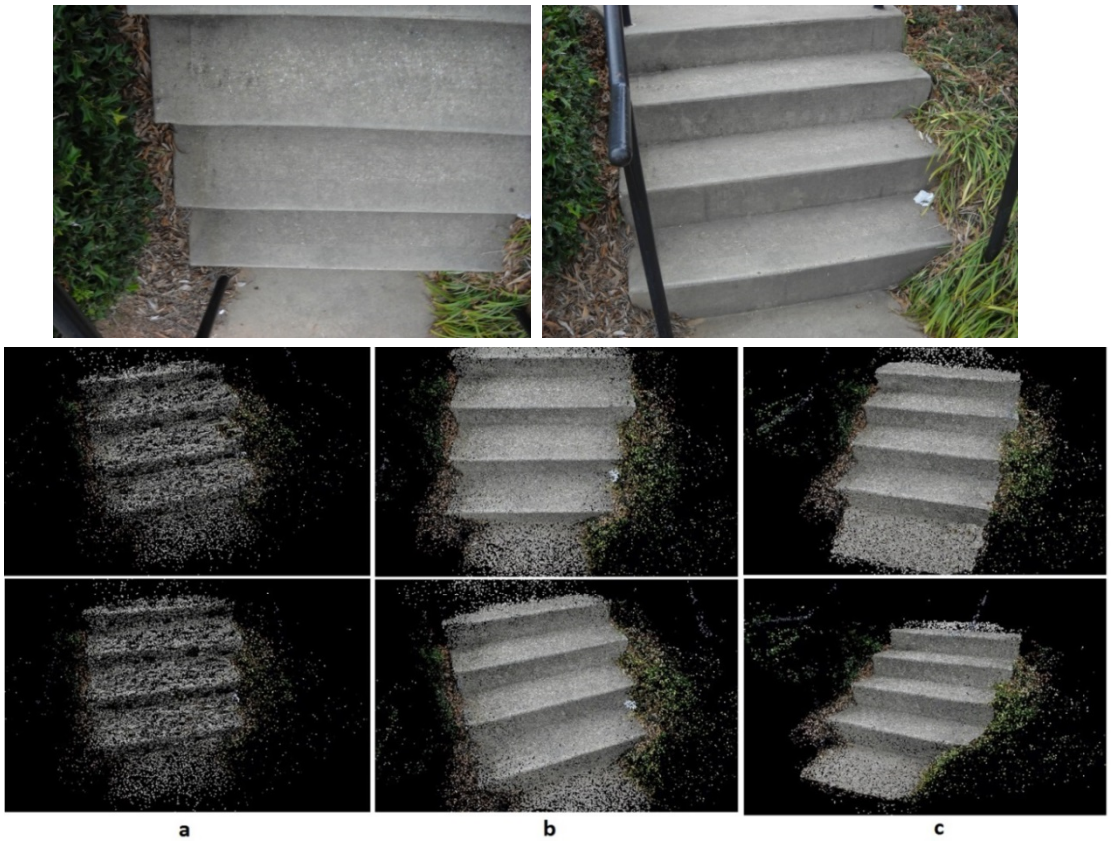

Fig. 4: sample snapshots of results of implementing algorithms on the concrete stairs dataset: a) generated PCD b) results of using VDM c) results of using MLS method

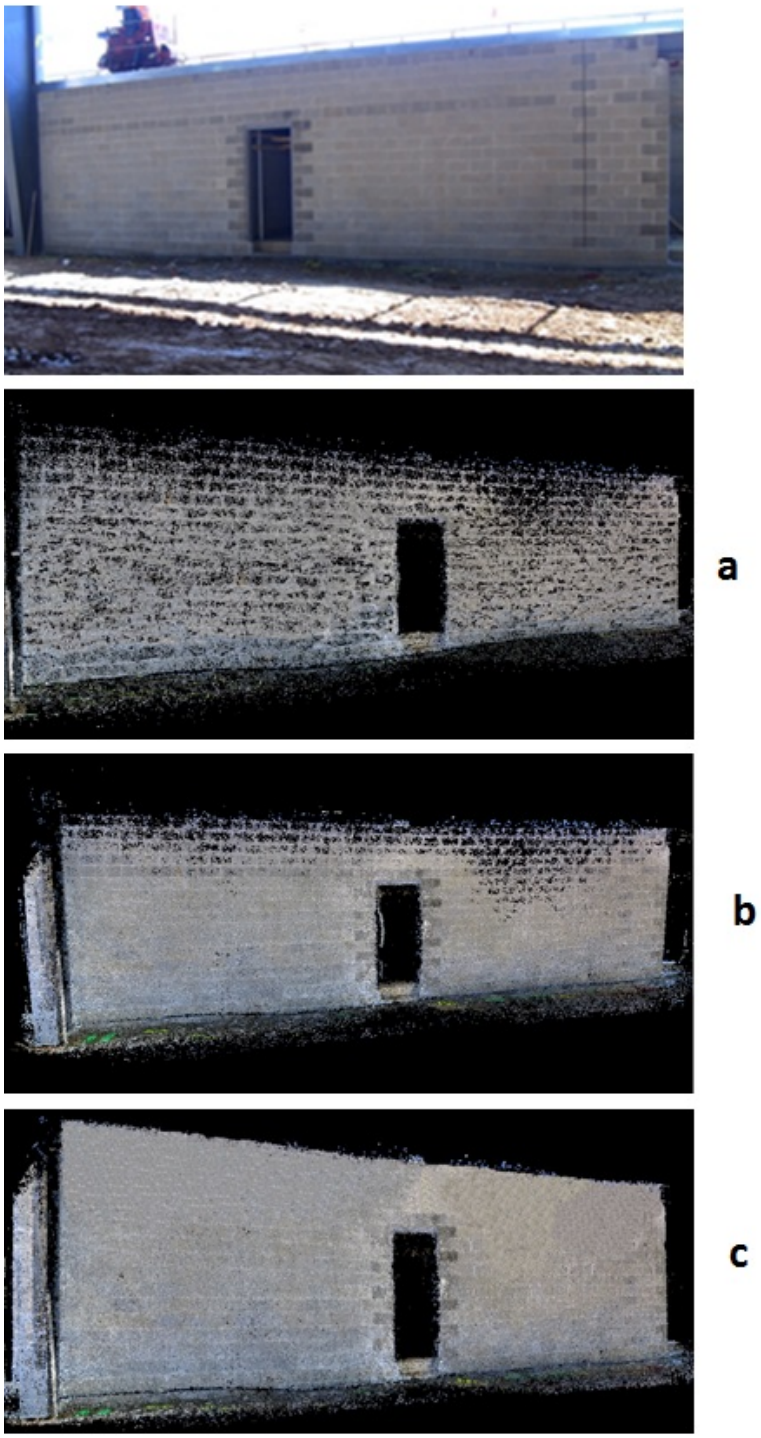

Fig. 5: Sample snapshots of results of implementing algorithms on the concrete stairs dataset: a) generated PCD b) results of using BPA c) results of using MLS method.

The summery of the comparison results are presented in table 2. As shown in the table, MLS outperforms other algorithms in terms of successfully filling the gaps; however, compared to the other two algorithms the processing is slower.

.Table 2: Summary of comparison results for 5 different datasets.

Dataset #	Method	Number of points before implementing the algorithm	Number of points after implementing the algorithm	Completeness before implementing the algorithm (%)	Completeness after implementing the algorithm (%)	Run time (second)
1	MLS*	459’412	512’307	82.74	95.15	39
	VDM**	459’412	483’251	82.74	91.32	37
	BPA***	459’412	491’775	82.74	89.43	21
2	MLS	2’578’209	2’876’292	79.21	88.10	51
	VDM	2’578’209	2’793’251	79.21	84.49	47
	BPA	2’578’209	2’655’439	79.21	81.98	37
3	MLS	782’891	841’257	73.65	86.21	35
	VDM	782’891	804’219	73.65	84.54	33
	BPA	782’891	812’593	73.65	80.22	28
4	MLS	1’021’544	1’261’004	85.74	93.24	42
	VDM	1’021’544	1’110’736	85.74	92.73	38
	BPA	1’021’544	1’008’703	85.74	88.41	33
5	MLS	678’325	697’841	86.29	94.91	36
	VDM	678’325	692’610	86.29	91.65	32
	BPA	678’325	683’541	86.29	92.81	26

*MLS: Moving Least Square method

** VDM: Volumetric Diffusion Method

*** BPA: Ball-Pivoting Algorithm

5. SUMMARY AND CONCLUSION

PCD data generated by processing video/images are usually imperfect and there are holes/gaps poorly reconstructed areas. Multiple reasons such as insufficient coverage of images/video frames, texture-less surfaces and possible errors in 3D reconstruction algorithms might cause this issue. Holes/gaps on surfaces negatively affect the visual quality of PCD. In addition, poorly constructed areas might cause problems while extracting geometric information/length measurements from PCD. As the result, post processing algorithms for fixing the problem and covering holes and gaps are required.

This paper presented an overview of existing algorithms for filling gaps on smooth surfaces of built infrastructure. The performance of three major algorithms in this field; i.e., Moving Least Square, Volumetric Diffusion Method and Ball-Pivoting Algorithm were quantitatively evaluated using five built infrastructure scenes as case studies. The scope of current paper was flat surfaces. The results showed that overall Moving Least Square method outperforms the other two algorithms in terms of quality of the generated covers for filling gaps and reconstructing missing parts. As the next phase of the research, the authors intend to evaluate the performance of the algorithms on more challenging cases, e.g. curved surfaces and geometrical complex holes.

6. REFERENCES

Amenta N., Bern M. and Kamvysselis M. (1998).A new voronoi-based surface reconstruction algorithm, *Proceedings of the SIGGRAPH'98*, 415-421.

Brilakis I., Fathi H. and Rashidi A. (2011). Progressive 3D Reconstruction of Infrastructure with Videogrammetry, *Journal of Automation in Construction*, Vol. 20, No. 7, 884-895.

Bernardini F., Mittleman J., Rushmeier H., Silva C. and Taubin, G. (1999). The Ball-Pivoting Algorithm for surface reconstruction, *IEEE Transactions on Visualization and Computer Graphics*, Vol. 5, No.4, 349-359.

Dai F., Rashidi A., Brilakis I. and Vela P. (2013).Comparison of Image- and Time-of-Flight-Based Technologies for 3D Reconstruction of Infrastructure,*ASCE Journal of Construction Engineering and Management*, Vol. 139, No. 1, 69-79.

Davis J., Marschner S., Garr M. and Levoy M. (2002).Filling holes in complex surfaces using volumetric diffusion, *Proceedings of the First international Symposium on 3D Data Processing*, *Visualization, Transmission*, 428-438.

Edelsbrunner H. andMuche E.P. (1994). Three-dimensional alpha shapes, *ACM Transactions on Graphics*, Vol. 13, No. 1, 43-72.

Gopi M. and Krishnan S. (2000). A fast and efficient projection based approach for surface reconstruction, *International Journal of High Performance Computer Graphics*, *Multimedia and Visualization*, Vol. 1, No. 1, 1-12.

Mencl R. (1995).A graph-based approach to surface reconstruction, *Proceedings of EUROGRAPHICS'95*, *Computer Graphics Forum*, Vol. 14,No. 3, 445-456.

Morse B. S., Yoo T. S., Chen D. T., Rheingans P. and Subramanian K. R. (2001). Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions,*Proceedings of the International Conference on Shape Modeling & Applications*, IEEE Computer Society Press, 89-98.

Rashidi A., Dai F., Brilakis I. and Vela P. (2013).Optimized Selection of Key Frames for Monocular Videogrammetric Surveying of Civil Infrastructure,*Journal of Advanced Engineering Informatics*,Vol.27, No.2, 270-282.

Scarloff S. andPentland A. (1991).Generalized implicit functions for computer graphics, P*roceedings of the SIGGRAPH'91*, 247-250.

Terzopoulos D. and Metaxas D. (1991). Dynamic 3D models with local and global deformations: deformable superquadrics, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, No. 7, 703-714.

Wang J. and Oliveira, M. M. (2002). Improved scene reconstruction from range images, *Proceedings of the EUROGRAPHICS'02*, 521-530.

Wang J. and Oliveira M.M. (2007).Filling holes on locally smooth surfaces reconstructed from point clouds, *Image and Vision Computing*, Vol. 25, No. 1, 103-113.