ENHANCING GEOMETRIC EDGE DETAILS IN MVS RECONSTRUCTION

E.K. Stathopoulou[1,2], S. Rigon[1], R. Battisti[1], F. Remondino[1]

[1]3D Optical Metrology (3DOM) unit, Bruno Kessler Foundation (FBK), Trento, Italy
[2]Laboratory of Photogrammetry, School of Rural and Surveying Engineering, National Technical University of Athens, Greece
Email: <estathopoulou><srigon><rbattisti><remondino>@fbk.eu

Commission II - WGII/4

KEY WORDS: multi view stereo, edge, mesh, point cloud, dense reconstruction, surface.

ABSTRACT:
Mesh models generated by multi view stereo (MVS) algorithms often fail to represent in an adequate manner the sharp, natural edge details of the scene. The harsh depth discontinuities of edge regions are eventually a challenging task for dense reconstruction, while vertex displacement during mesh refinement frequently leads to smoothed edges that do not coincide with the fine details of the scene. Meanwhile, 3D edges have been used for scene representation, particularly man-made built environments, which are dominated by regular planar and linear structures. Indeed, 3D edge detection and matching are commonly exploited either to constrain camera pose estimation, or to generate an abstract representation of the most salient parts of the scene, and even to support mesh reconstruction. In this work, we attempt to jointly use 3D edge extraction and MVS mesh generation to promote edge detail preservation in the final result. Salient 3D edges of the scene are reconstructed with state-of-the-art algorithms and integrated in the dense point cloud to be further used in order to support the mesh triangulation step. Experimental results on benchmark dataset sequences using metric and appearance-based measures are performed in order to evaluate our hypothesis.

1. INTRODUCTION

For a given set of images with known orientation parameters (poses), typically as the output of the SfM, multiview stereo methods (MVS) generate a 3D dense point cloud, a triangulated mesh or a volume. Among the various MVS methods for dense reconstruction, depth map merging methods are commonly used in photogrammetry because of their accuracy and scalability. Semi-global matching (Hirschmuller, 2007) and PatchMatch (Bleyer et al., 2011) are within the most widespread methods for depth estimation and dense cloud generation due to their robustness, even though also learning methods have lately been popular (Huang et al., 2018). Mesh representations, on the other hand, can be either generated as part of the photogrammetric workflow or as a standalone task (Nocerino et al., 2020). The first method keeps the photometric consistency criterion active also while wrapping the surface and providing thus a more refined mesh, while the second generates the optimal surface mesh out of a (typically dense) point cloud using i.e. Delaunay Triangulation or Poisson Surface Reconstruction (Kazhdan et al., 2006).
Although such algorithms are mature enough with impressive results, several challenges still exist towards the complete, accurate and detail-preserving 3D reconstruction of scenes. Inadequate image network setups, challenging objects such as textureless or reflective surfaces, occlusions, as well as harsh depth discontinuities may affect the quality of the final reconstruction and the level of preserved details. Crease edges on 3D meshes often do not coincide with natural edges on the object surface. Traditional photogrammetric techniques used vector constraints, the so-called breaklines, to tackle such depth discontinuities and imply geometric constraints during DSM generation (Briese, 2004).
Similar to corner points, edges have been used to support various photogrammetric and computer vision tasks such as image matching (Wang et al., 2009; Wang et al., 2021), camera localization (Hirose and Saito, 2012; Salaün et al., 2017; Miraldo et al., 2018), abstract 3D scene representation (Hofer et al., 2015; 2017), meshing sparse clouds (Bódis-Szomorú et al., 2015; Sugiura et al., 2015) as well as modelling and simplifying the scene (Langlois et al., 2019; Chen et al., 2020; Li and Nan, 2021).

1.1. Aim of the paper

3D edges are usually invariant to significant illumination changes and a robust representation of the most salient parts of the scene. In this study, we investigate whether edges can potentially support detail-preserving MVS reconstruction and generate visually coherent results. Out of the plethora of MVS mesh reconstruction methods, we consider triangulated meshes by 3D point clouds coming from depth map fusion. The scope of this article is bifold: first, two state of the art methods for 3D edge reconstruction are adopted and experimentally compared in terms of performance, usability and practical limitations. Then, we propose an approach for integrating the extracted 3D edge information into the MVS pipeline towards detailed and sharp feature preserving mesh reconstruction and evaluate the potential and the limitations of the method.

2. RELATED WORK

The presented work leverages edge constraints in the MVS reconstruction procedure, hence the respective literature is reviewed.

Multiple view stereo: MVS algorithms (Furukawa and Ponce, 2009) generate a complete 3D representation of the scene starting from known camera poses. Generally, methods can be point cloud-based, volume-based and mesh-based. Point cloud-based methods use photo-consistency metrics like the Sum of Squared Differences (SSD) or the Normalized Cross-Correlation (NCC) to estimate the depth of the scene pixels and generate dense 3D point clouds (Shen et al., 2013; Schönberger et al., 2016). Such point clouds can be further converted into triangulated meshes using iso-surface extraction e.g. with Poisson (Kazhdan et al., 2006) or Delaunay Triangulation and graph-cut methods (Labatut et al., 2007). Volume-based methods (Curless and Levoy, 1996) discretize the space into voxels or tetrahedra and calculate a 3D volume from which the optimal surface will be extracted using

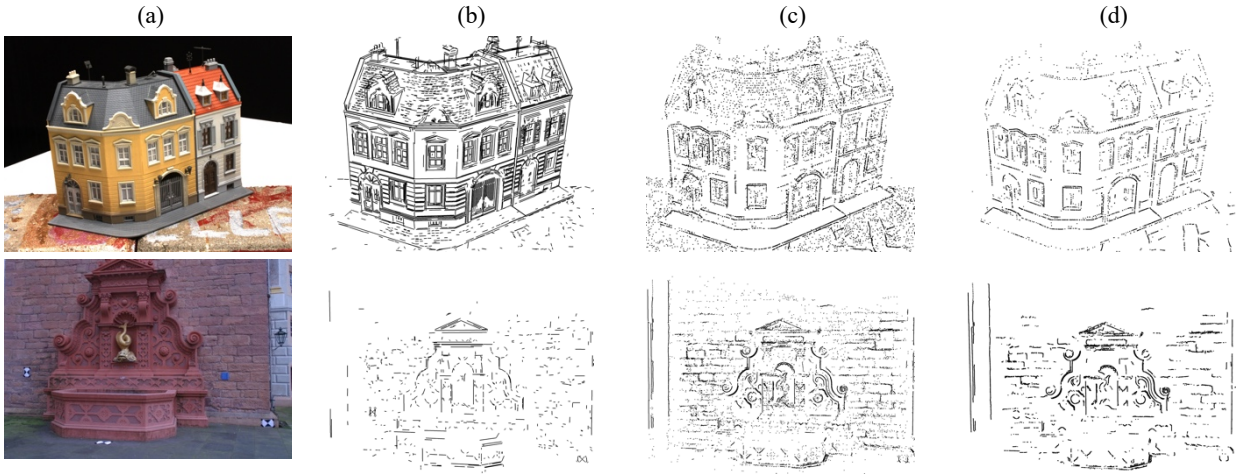

Figure 1. Two examples of extracted 3D edges: Line3D++ (b), EdgeGraph3D - standard output (c) and EdgeGraph3D - isolated edge points (d). Line3D++ generates line segments, whereas EdgeGraph3D by default generates points along the edges fused with the input sparse point cloud.

e.g. graph-cuts (Vogiatzis et al., 2007) or the signed distance function (Newcombe et al., 2011; Werner et al., 2014). Surfaces are produced from volumetric representations using Poisson triangulation (Kazhdan et al., 2006) or the marching-cubes algorithm (Lorensen and Cline, 1987). Mesh-based methods (Vu et al., 2012) use the photo-consistency metrics to refine (i.e. remesh) an existing initial mesh, generated with volume-based methods or point cloud-based methods accompanied with mesh extraction.

In photogrammetry, typically meshes are generated by triangulating dense point clouds coming from depth map fusion methods. Point clouds are converted to meshed surfaces with Poisson triangulation (Kazhdan et al., 2006) or Delaunay Triangulation (Tola et al., 2012; Jancosek and Pajdla, 2014). Delaunay Triangulation is commonly preferred since it adapts to point density and is, thus, more scalable. The final mesh is defined as the boundary between empty and full tetrahedra, typically formulated as a graph-cut problem.

Edge extraction: Traditionally, line segments have been extensively used in the 2D space with simple gradient-based detectors like Sobel (Sobel, 1972) and LoG (Marr, 1980), to most sophisticated solutions as Canny (Canny, 1986), Rothwell (Rothwell et al., 1995), Edison (Meer and Georgescu, 2001) and the linear segment detector LSD (von Gioi et al., 2010). Edge detection directly in the 3D space has also been revised in the literature. Weinmann et al. (2015) classified edges along with corners and planes using features based on eigenvalues in 3D point clouds. Hackel et al. (2016) used eigenvalues and Markov Random Fields (MRF) to create wireframe models. Jain et al. (2010) extracted 3D lines under an optimization formulation, minimizing the reprojection error of the segment end-points and enabled reconstruction under challenging lighting conditions.

3D reconstruction and linear segments: Line segments have been used in image registration tasks in photogrammetry already for a long time (Baillard et al., 1999). In the latest years, linear segment matching has been used in pairwise image matching (Wang et al., 2009; Zhang and Koch, 2014), as well as in SfM (Bertoli and Sturm, 2006; Micusik and Wildenauer, 2018) and SLAM algorithms (Hirose and Saito, 2012; Salaün et al., 2017; Zhou et al., 2019) for pose estimation and mapping or 3D reconstruction purposes (Remondino and Zhang, 2006). At the same time, matched linear segments are coupled with the SfM results as a less computational expensive alternative to the MVS reconstruction as in (Hofer et al., 2015; 2017). Sugiura et al. (2015) performed 2D line matching and reconstructed a so-called 3D "line cloud", and used these edges to extract the mesh using a tetrahedra-carving method. Romanoni and Matteucci (2015) used 3D points belonging to edges and carved a 3D Delaunay Triangulation of sparse points. Bódis-Szomorú et al. (2015) proposed an approach for large scale urban reconstruction for edge preserving mesh reconstruction enforcing the Delaunay Triangulation using CDT (Botsch et al., 2010) on a 2D base mesh. Bignoli et al. (2018) presented an approach to detect both straight and curved edges and support thus the reconstruction of 3D meshes from sparse point clouds. Other approaches combine primitives for regular parts of the scene and meshes for the irregular ones (Lafarge et al., 2010).

In this study, we consider the combination of edges and 3D reconstruction in a different fashion, as edge information is integrated in the dense point clouds to support and potentially generate more detail-preserving mesh models.

3. METHODOLOGY

3.1 3D Edge extraction

We consider two different state-of-the-art 3D edge extraction methods (Figure 1), namely Line3D++ (Hofer et al., 2016) and EdgeGraph3D (Bignoli et al., 2018) and discuss their results and efficiency.

Edge extraction with Line3D++: Line3D++ (Hofer, 2016) detects and matches 2D line segments across the images and reprojects them in the 3D space, generating an abstract 3D representation of the salient parts of the scene. In more detail, camera poses and the respective sparse point cloud are given as input, obtained by conventional Structure from Motion pipelines like COLMAP (Schönberger et al., 2016a) or OpenMVG (Moulon et al., 2016). Epipolar guidance is used to establish correspondences between linear segments detected on the images. The LSD line segment detector (von Gioi et al., 2008) is used to obtain the line segments. The best hypothesis for the 3D position is selected and overlapping segments from different views are clustered together using graph-clustering to generate a 3D line-cloud. The reconstructed lines can potentially optimize also the SfM result using Ceres solver (Agarwal et al., 2021). The quality of the resulting 3D line cloud depends on several parameters (Figure 2). First, it was proven experimentally that the denser our image network is (higher overlap), the more numerous

the 3D lines are. However, the accuracy of the reconstructed 3D lines, naturally, depends on the accuracy of the SfM pose estimation and the reprojection error. Also, detected 2D edges are prone to outliers due to noise in local gradients and occlusions. Thus, inevitably, some edges fail to reconstruct and duplicates or wrongly reprojected lines are also present. These erroneous reconstructed lines may not affect the abstract representation of the scene alone (Hofer et al., 2016), yet may introduce a significant error in the meshing step.

Figure 2. Line3D++ performance examples for DTU-006 dataset using different visibility threshold values (*v=5,8,11* images) and pixel error *p=1*. The stricter the visibility threshold, the fewer the reconstructed lines in 3D (respectively 2486, 1428, and 860 line segments).

Edge extraction with EdgeGraph3D: EdgeGraph3D algorithm (Bignoli et al., 2018) uses a-priori detected edges coming from standard edge detection algorithms along with the SfM output, i.e. camera poses and sparse point cloud to subsequently project the salient edges in 3D. In contrast with Line3D++, EdgeGraph3D is able to match and reconstruct not only linear edges, but also curved ones in the form of 3D polylines (Figure 3). This is made possible with the use of calculated 2D edge graphs for each image. Practically, to every pixel centre belonging to an edge, a node is assigned. Adjacent edge-pixel nodes are connected in polylines generating the 2D edge graphs. Based on these graphs along with the SfM data and the epipolar constraints, potential edge correspondences are defined, and further validated and reconstructed in 3D on top of the SfM points. In our experiments, we use the Edison algorithm (Meer and Georgescu, 2001) for 2D edge detection. We extended the functionality of the algorithm to export directly edge points and their visibility information without the SfM data.

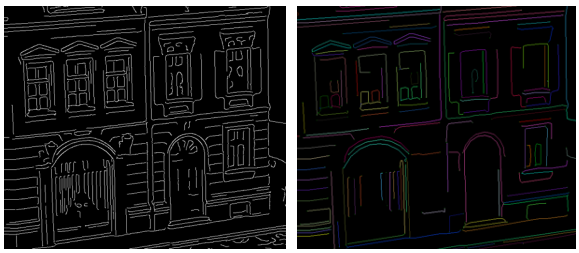

Figure 3. 2D edges generated with Edison edge detection algorithm and given as input to EdgeGraph3D (left), and their respective matched segments using EdgeGraph3D (right).

Our first experiments showed that both algorithms are rather sensitive to the accuracy of the pose estimation and thus often reconstruct noisy 3D lines or edge points. One important parameter that needs to be taken into account in order to optimize the 3D edge reconstruction is the visibility threshold *N*, i.e. from how many images a line or point needs to be visible, in order to be candidate to be reconstructed in 3D. For both algorithms by default the threshold is defined *v=3*, yet in our experiments we tuned the visibility threshold according to the dataset overlap (see Section 4 for more details). For instance, as shown in Figure 1, for the same visibility threshold (*v=3*) in Line3D++, in DTU-006, much more edge lines are reconstructed in comparison with Fountain-P11 due to the large overlap of this specific dataset. Another essential factor is the maximum accepted pixel error *p* for the same line or point across multiple images. By default, this error is 2.5 pixels in Line3D++ (see Section 4 for more details), while in EdgeGraph3D a slightly different approach is adopted with outlier removal based on visibility filtering, with a threshold error of 2.25 pixels.

In EdgeGraph3D, points describing an edge do not strictly lie on the exact edge, but rather form a point cloud of irregular distribution and noise around the edge that can be useful for abstract scene representation. Line3D++ on the other hand, generates a clear line cloud and also has the advantage of computational efficiency. Thus, it was preferred over EdgeGraph3D although the latter detected both linear and curved segments (Figure 1).

3.2. Edge-aware mesh reconstruction

Our method uses dense point clouds and edge information in a joint fashion to support the edge-aware mesh reconstruction inspired by Bignoli et al. (2018) and Bódis-Szomorú et al. (2015). Our intuition is that the usage of such edge information, aligned with natural depth discontinuities, may support the meshing algorithms in such a way to highlight well-defined natural geometric edges while preserving details and avoiding over-smoothing. In contrast to most approaches, we do not integrate the edge points with the SfM sparse cloud, but rather leverage them into the MVS pipeline. We consider edge information priorly extracted with Line3D++, as our approach is based on regularly spaced points along the edges and the output of EdgeGraph3D didn't fulfil this requirement.

Line Sampling: Line3D++ extracts linear segments as vectors and outputs information about their start and end points' coordinates in 3D and 2D, along with the visibility information. In our method, we interpolate linearly 3D points along the edges in equal space intervals, based on the average spacing of the input dense cloud (Figure 4). Keeping a regular point spacing will benefit the generation of triangles of similar edge length and face area around the region in the mesh reconstruction step.

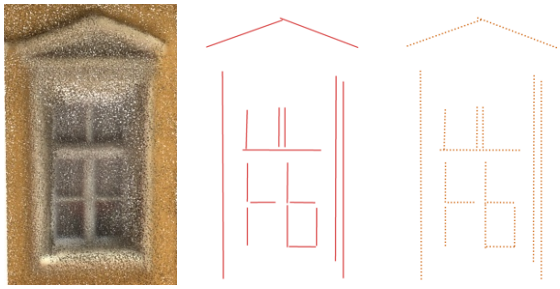

Figure 4. The dense point cloud (left), Line3D++ reconstructed 3D edges (middle) and the respective edge points (right) sampled linearly along them.

Integration to MVS: 3D edge points and their respective visibility information are integrated into the dense point cloud generated by the PatchMatch MVS algorithm (Bleyer et al., 2011; Shen, 2013) as implemented in the OpenMVS library (Cernea, 2021). Hence, we generate a final dense cloud merging the dense cloud points coming from the depth map fusion plus the 3D points along the edges. In order to eliminate the noise and the effects of potential errors, we filter out dense cloud points within a buffer of *m* times the average spacing before meshing, depending on the dataset. Mesh reconstruction is performed

using Delaunay tetrahedralization as implemented in OpenMVS and built upon the CGAL library (2021).
Iterative consistency checks are made for every point of the input cloud and finally the surface is extracted using a graph-cut method. Since such methods produce inevitably a significant amount of non-manifold vertices, an essential post processing step follows to repair the local topology of the mesh. Using the VCG library (2021) as integrated in OpenMVS, standard spike and spurious removal, hole filling and mesh smoothing is performed. Vertex repositioning follows, applied only to vertices belonging to edges, in order to keep the original position of the edge points.
The mesh reconstruction implemented in OpenMVS was preferred over others, as it has been proven robust enough in previous works (Nocerino et al., 2020). In our experiments though, we do not perform the final mesh refinement minimizing the photometric error based on point visibility, but rather focus on the standard mesh reconstruction.

4. EXPERIMENTS AND EVALUATION

4.1 Datasets

We perform our experiments on benchmark datasets for which ground truth (GT) 3D information is provided. In particular:
Fountain-P11 (Strecha et al., 2008): 11 high resolution images (3072 x 2048 pixel) with known camera poses from the EPFL benchmark. GT 3D mesh is provided for evaluation, as a result of a laser scanning acquisition.
ETH3D-Façade (Schöps et al., 2017): 76 high resolution (6048 x 4032 pixel) sequence from the ETH benchmark. GT point cloud acquired with laser scanning is available for evaluation.
DTU-006 (Aanaes et al., 2014): 49 medium resolution images (1600 x 1200) of known poses from the DTU benchmark under seven varying illumination conditions. For our experiments we used the middle exposure images. GT point clouds for evaluation are acquired with a structure light scanner.

4.2 Parameter settings

Line3D++ pixel error and minimum visibility threshold were tuned appropriately for each dataset, taking into consideration criteria as overlap between images, image resolution, pixel size and level of fine details that need to be reconstructed. Specifically, $p = 2.5$ was chosen for all datasets and $v = 3,5,7$ for Fountain-P11, ETH3D-Facade and DTU-006, respectively. For dense reconstruction, images were resized to max 3200 pixels and during the mesh reconstruction no decimation was performed. For the dense point filtering, we used $m = 1.5,2,3$ times the average spacing for Fountain-P11, ETH3D-Facade and DTU-006, taking into consideration the image resolution and noise of every sequence.

4.3 Evaluation Metrics

The literature on the geometric quality of meshes and preserved edges is quite weak, despite for industrial meshing (Stimpson et al., 2007). In our experiments, appearance-based metrics were chosen for the evaluation since our approach aims to add fine edge details in the final mesh reconstruction. More particularly, we use geometric features based on the combination of the eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$ of the covariance tensor computed within a local neighbourhood of a point, as used by (Weinmann et al., 2015; Hackel at al., 2016) as implemented in CloudCompare (2021). More particularly, the metrics that were experimentally found to be of significance for highlighting the fine details, were surface variation and normal change rate. Local surface variation is defined as:

$$C_\lambda = \frac{\lambda_3}{(\lambda_1 + \lambda_2 + \lambda_3)}.$$

Similar to surface variation, normal change rate represents the curvature variation within a local neighbourhood radius. This kernel size k was decided taking into consideration the density of the vertices. Results are shown as colour maps to highlight the high frequency details and noise (Figure 6-7-8).
Using the proposed approach, more evident details are observed in the qualitative comparison of the mesh models, especially where accurate 3D edges were reconstructed (Figure 5). Inevitably, potential errors in edge reconstruction may add some noisy triangles, hence a more robust 3D edge reconstruction and refinement step may be needed. However, RMS error to the GT model is similar between the standard mesh and the mesh generated using our approach (Table 1). But using the appearance-based metrics, we observe more evident fine details using the proposed approach.

Dataset		RMS (mm)
Fountain-P11	with edges	10.761
	standard	10.756
ETH3D-Facade	with edges	16.440
	standard	16.415
DTU-006	with edges	0.313
	standard	0.309

Table 1. RMS errors for the three datasets.

5. CONCLUSIONS

An ideally reconstructed mesh in MVS scenarios should be smooth but also detail-preserving, especially around the natural crease edges. In this paper we presented an approach to leverage edge information in the standard MVS mesh reconstruction pipeline. Edges are detected and reconstructed in 3D using the approach of Hofer et al. (2016) enwrapped in the Line3D++ algorithm, while the performance and usability of EdgeGraph3D is also evaluated for our dense MVS scenarios. This presented edge-enhancing MVS methodology provide for more detailed-preserving meshes based on the appearance-based metrics used for the evaluation. Among the potential limitations of the method we address that the quality of the reconstructed mesh highly depends on the accuracy of the extracted 3D edges.

ACKNOWLEDGEMENTS

The authors would like to thank Dan Cernea for his valuable advice regarding OpenMVS library.

REFERENCES

Aanæs, H., Jensen, R.R., Vogiatzis, G., Tola, E. and Dahl, A.B., 2016. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, *120*(2), pp.153-168.

Agarwal, S., Mierle, K., and others, 2021. Ceres solver. http://ceres-solver.org

Baillard, C., Schmid, C., Zisserman, A. and Fitzgibbon, A., 1999. Automatic line matching and 3D reconstruction of buildings from

Figure 5. GT mesh and clouds with overlaid 3D edges in red (left) standard MVS mesh (centre) mesh with edge details (right). Fountain-P11 (upper row), ETH3D-Facade (middle row), DTU-006 (lower row).

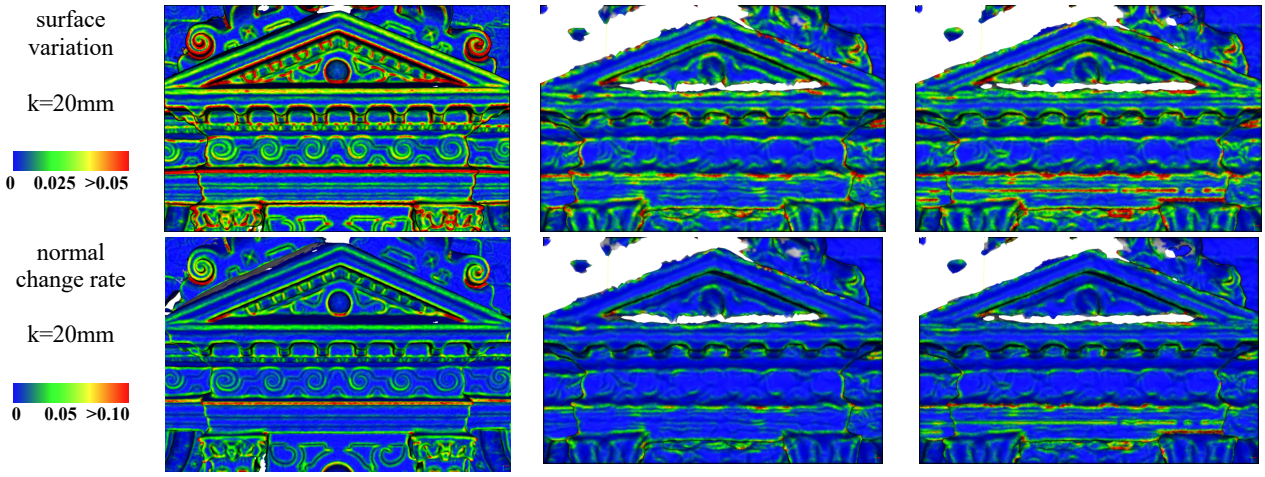

Figure 6. Fountain-P11: surface variation and normal change rate for GT (left), standard mesh (centre), mesh with edges (right).

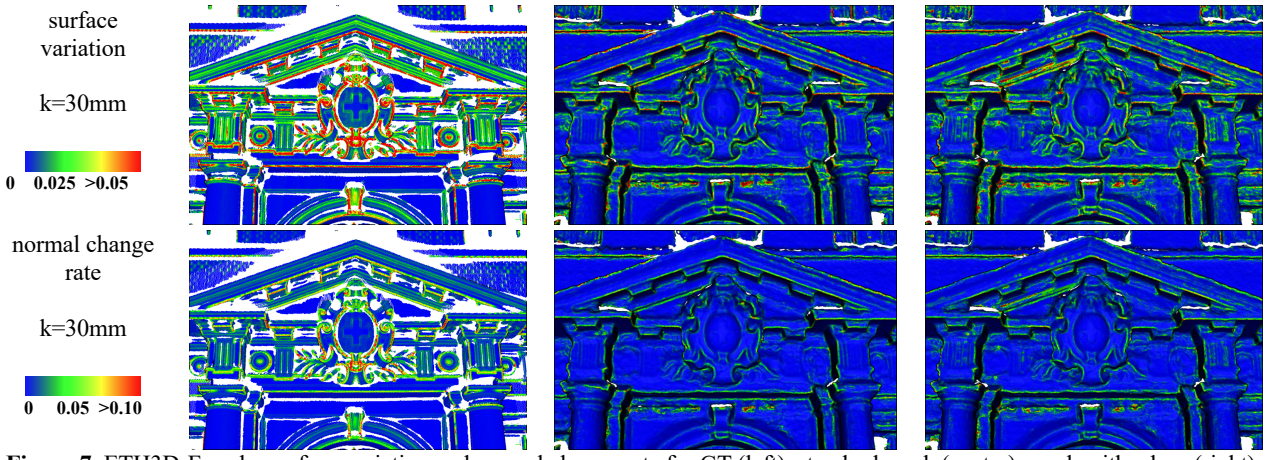

Figure 7. ETH3D-Façade: surface variation and normal change rate for GT (left), standard mesh (centre), mesh with edges (right).

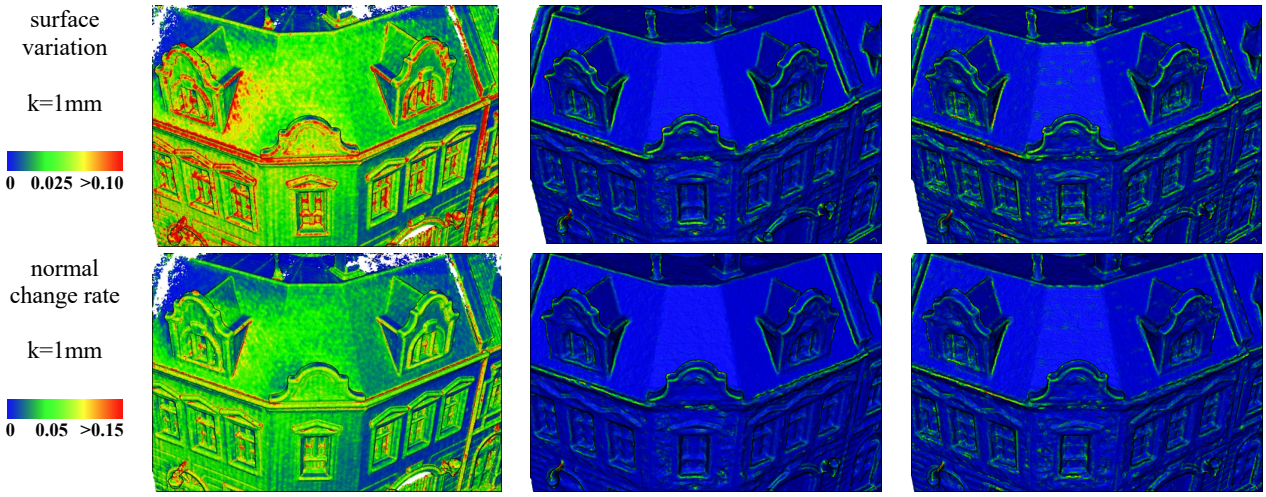

Figure 8. DTU-006: surface variation and normal change rate for GT (left), standard mesh (centre), mesh with edges (right).

multiple views. In *ISPRS Conference on Automatic Extraction of GIS Objects from Digital Imagery*, *Vol. 32*, pp. 69-80.

Bartoli, A. and Sturm, P., 2005. Structure-from-motion using lines: Representation, triangulation, and bundle adjustment. *Computer Vision and Image Understanding*, *100*(3), pp.416-441.

Bignoli, A., Romanoni, A., Matteucci, M. and di Milano, P., 2018. Multi-View Stereo 3D Edge Reconstruction. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 867-875. IEEE.

Bleyer, M., Rhemann, C. and Rother, C., 2011. PatchMatch Stereo-Stereo Matching with Slanted Support Windows. In *BMVC*, *Vol. 11*, pp. 1-11.

Bódis-Szomorú, A., Riemenschneider, H. and Van Gool, L., 2015. Superpixel meshes for fast edge-preserving surface reconstruction. In *Proc. IEEE CVPR*, pp. 2011-2020.

Botsch, M., Kobbelt, L., Pauly, M., Alliez, P. and Lévy, B., 2010. *Polygon mesh processing*. CRC press.

Briese, C., 2004. Three-dimensional modelling of breaklines from airborne laser scanner data. *ISPRS Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., Vol. 35(B3),* pp. 12-23.

Canny, J., 1986. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (*6*), pp.679-698.

Cernea, D., 2020. OpenMVS: Multi-view stereo reconstruction library.

CGAL - Computational Geometry Algorithms Library, 2021. https://www.cgal.org

Chen, Z., Zheng, X. and Guan, T., 2020. Structure-Preserving Mesh Simplification. *KSII Transactions on Internet and Information Systems (TIIS)*, *14*(11), pp.4463-4482.

CloudCompare (version 2.11.1) [GPL software]. (2021). Retrieved from http://www.cloudcompare.org/

Curless, B. and Levoy, M., 1996. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer Graphics and Interactive Techniques*, pp. 303-312.

Furukawa, Y. and Ponce, J., 2009. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. on PAMI*, *32*(8), pp.1362-1376.

Von Gioi, R.G., Jakubowicz, J., Morel, J.M. and Randall, G., 2008. LSD: A fast line segment detector with a false detection control. *IEEE Trans. on PAMI*, Vol. 32(4), pp.722-732.

Hackel, T., Wegner, J.D. and Schindler, K., 2016. Contour detection in unstructured 3d point clouds. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1610-1618.

Hirschmuller, H., 2007. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *30*(2), pp.328-341.

Hirose, K., and Saito, H., 2012. Fast line description for line-based SLAM. Proc. 23rd *BMVC.*

Hofer, M., Maurer, M. and Bischof, H., 2014. Improving sparse 3D models for man-made environments using line-based 3D reconstruction. In *2nd International Conference on 3D Vision*, *Vol. 1*, pp. 535-542. IEEE.

Hofer, M., Maurer, M. and Bischof, H., 2015. Line3d: Efficient 3d scene abstraction for the built environment. In *German Conference on Pattern Recognition*, pp. 237-248. Springer, Cham.

Hofer, M., Maurer, M. and Bischof, H., 2017. Efficient 3D scene abstraction using line segments. *Computer Vision and Image Understanding*, *157*, pp.167-178.

Huang, P.H., Matzen, K., Kopf, J., Ahuja, N. and Huang, J.B., 2018. Deepmvs: Learning multi-view stereopsis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2821-2830.

Jain, A., Kurz, C., Thormählen, T. and Seidel, H.P., 2010. Exploiting global connectivity constraints for reconstruction of 3D line segments from images. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1586-1593. IEEE.

Jancosek, M. and Pajdla, T., 2014. Exploiting visibility information in surface reconstruction to preserve weakly supported surfaces. *International Scholarly Research Notices*, *2014*.

Kazhdan, M., Bolitho, M. and Hoppe, H., 2006. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics Symposium on Geometry Processing, Vol. 7*.

Labatut, P., Pons, J.P. and Keriven, R., 2007. Efficient multi-view reconstruction of large-scale scenes using interest points, delaunay triangulation and graph cuts. In *2007 IEEE 11th International Conference on Computer Vision*, pp. 1-8. IEEE.

Lafarge, F., Keriven, R., Brédif, M. and Hiep, V.H., 2010. Hybrid multi-view reconstruction by jump-diffusion. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 350-357. IEEE.

Langlois, P.A., Boulch, A. and Marlet, R., 2019. Surface reconstruction from 3d line segments. In *2019 International Conference on 3D Vision (3DV)*, pp. 553-563. IEEE.

Li, M. and Nan, L., 2021. Feature-preserving 3D mesh simplification for urban buildings. *ISPRS Journal of Photogrammetry and Remote Sensing*, *173*, pp.135-150.

Lorensen, W.E. and Cline, H.E., 1987. Marching cubes: A high resolution 3D surface construction algorithm. *ACM Siggraph Computer Graphics*, *21*(4), pp.163-169.

Marr, D. and Hildreth, E., 1980. Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, *207*(1167), pp.187-217.

Meer, P. and Georgescu, B., 2001. Edge detection with embedded confidence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(12), pp.1351-1365.

Micusik, B. and Wildenauer, H., 2017. Structure from motion with line segments under relaxed endpoint constraints. *International Journal of Computer Vision*, *124*(1), pp.65-79.

Miraldo, P., Dias, T. and Ramalingam, S., 2018. A minimal closed-form solution for multi-perspective pose estimation using points and lines. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 474-490.

Moulon, P., Monasse, P., Perrot, R. and Marlet, R., 2016. OpenMVG: Open multiple view geometry. In *International Workshop on Reproducible Research in Pattern Recognition*, pp. 60-74. Springer, Cham.

Newcombe, R.A., Lovegrove, S.J. and Davison, A.J., 2011. DTAM: Dense tracking and mapping in real-time. In *2011 International Conference on Computer Vision*, pp. 2320-2327. IEEE.

Nocerino, E., Stathopoulou, E.K., Rigon, S. and Remondino, F., 2020. Surface reconstruction assessment in photogrammetric applications. *Sensors*, *20*(20), p.5863.

Remondino, F., Zhang, L., 2006: Surface Reconstruction Algorithms for Detailed Close-Range Object Modelling, *ISPRS Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, Vol. 36(3).

Romanoni, A. and Matteucci, M., 2015. Incremental reconstruction of urban environments by edge-points delaunay triangulation. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4473-4479. IEEE.

Rothwell, C.A., Mundy, J.L., Hoffman, W. and Nguyen, V.D., 1995. Driving vision by topology. In *Proc. of International Symposium on Computer Vision-ISCV*, pp. 395-400. IEEE.

Salaün, Y., Marlet, R. and Monasse, P., 2017. Line-based robust SfM with little image overlap. In *2017 International Conference on 3D Vision (3DV)*, pp. 195-204. IEEE.

Schönberger, J.L. and Frahm, J.M., 2016. Structure-from-motion revisited. In *Proceedings of the IEEE CVPR,* pp. 4104-4113.

Schönberger, J.L., Zheng, E., Frahm, J.M. and Pollefeys, M., 2016. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision*, pp. 501-518. Springer, Cham.

Schöps, T., Schonberger, J.L., Galliani, S., Sattler, T., Schindler, K., Pollefeys, M. and Geiger, A., 2017. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3260-3269.

Shen, S., 2013. Accurate multiple view 3d reconstruction using patch-based stereo for large-scale scenes. *IEEE transactions on image processing*, *22*(5), pp.1901-1914.

Sobel, I., 1972. *Camera models and machine perception* (No. CS Technion report CS0016). Computer Science Department, Technion.

Stimpson, C. J. , Merchant St, N., Ernst, C. D., Merchant St, N., Pébay, P. P., Thompson, D., 2007. The Verdict Library Reference Manual. SAND: 1751. Sandia National Laboratories.

Strecha, C., Von Hansen, W., Van Gool, L., Fua, P. and Thoennessen, U., 2008. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *2008 IEEE conference on computer vision and pattern recognition*, pp. 1-8. IEEE.

Sugiura, T., Torii, A. and Okutomi, M., 2015. 3D surface reconstruction from point-and-line cloud. In *2015 International Conference on 3D Vision*, pp. 264-272. IEEE.

Tola, E., Strecha, C. and Fua, P., 2012. Efficient large-scale multi-view stereo for ultra high-resolution image sets. *Machine Vision and Applications*, *23*(5), pp.903-920.

VGL - Visualization and Computer Graphics Library, 2021. http://vcg.isti.cnr.it/vcglib/

Vogiatzis, G., Torr, P.H.S., and Cipolla, R. Multi-view stereo via volumetric graph-cuts. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Diego, CA, USA, 2005, *Vol 2*, pp. 391-398.

Vu, H.H., Labatut, P., Pons, J.P. and Keriven, R., 2012. High accuracy and visibility-consistent dense multiview stereo. *IEEE Trans. PAMI*, *34*(5), pp.889-901.

Wang, Z., Wu, F. and Hu, Z., 2009. MSLD: A robust descriptor for line matching. *Pattern Recognition*, *42*(5), pp.941-953.

Wang, J., Zhu, Q., Liu, S. and Wang, W., 2021. Robust line feature matching based on pairwise geometric constraints and matching redundancy. *ISPRS Journal of Photogrammetry and Remote Sensing*, *172*, pp.41-58.

Werner, D., Al-Hamadi, A.; Werner, P. Truncated signed distance function: Experiments on voxel size. In *International Conference Image Analysis and Recognition*; Springer: Cham, Switzerland, 2014; pp. 357–364.

Weinmann, M., Urban, S., Hinz, S., Jutzi, B. and Mallet, C., 2015. Distinctive 2D and 3D features for automated large-scale scene analysis in urban areas. *Computers & Graphics*, *49*, pp.47-57.

Zhang, L. and Koch, R., 2012. Line matching using appearance similarities and geometric constraints. In *Joint DAGM (German Association for Pattern Recognition) and OAGM Symposium*, pp. 236-245. Springer, Berlin, Heidelberg.

Zhou, H., Fan, H., Peng, K., Fan, W., Zhou, D. and Liu, Y., 2019. Monocular Visual Odometry Initialization with Points and Line Segments. *IEEE Access*, *7*, pp.73120-73130.