3D EDGE DETECTION AND COMPARISON USING FOUR-CHANNEL IMAGES

Thodoris Betsas [1]*, Andreas Georgopoulos [1]

[1] Laboratory of Photogrammetry, School of Rural-Surveying and Geoinformatics Engineering NTUA, Athens Greece;
betsasth@mail.ntua.gr, drag@central.ntua.gr

Technical Commission II

KEY WORDS: 3D Edge Detection, Point Cloud Segmentation, 3D Edge Comparison, SfM, MVS

ABSTRACT:

Point cloud segmentation, is a widespread field of research and it is useful in several research topics and applications such as 3D point cloud analysis, scene understanding, semantic segmentation etc. Architectural vector drawings constitute a valuable platform source for scientists and craftsmen while the production of such drawings is time-consuming because many of the creation steps are done manually. Detecting 3D edges in point clouds could provide useful information for the automation of the creation of 3D architectural vector drawings. Hence, a 3D edge detection method is proposed and evaluated with a proof-of-concept experiment and another one using a professional software. The scope of this effort is twofold, firstly the production of semantically enriched 3D dense point clouds exploiting four-channel images in order to detect 3D edges and secondly the comparison of the detected 3D edges with their corresponding edges in a textured 3D model. Comparing 3D edges in the early step of the 3D dense point cloud production and in the final step of 3D textured mesh, provides useful conclusions of the data used for the automatic creation of 3D drawings. Both of the experiments i.e., the proof-of-concept and using the professional SfM-MVS software were conducted using real world data of cultural heritage objects.

1. INTRODUCTION

The architectural vector drawings are the most commonly used products in several fieldwork cases like construction, conservation, restoration and documentation as well as in many scientific fields like archaeology, architecture and surveying. Although, creating architectural vector drawings, especially in 3D space, constitutes a labor-intensive process, the scientific community has not provided an automated approach for the production of such drawings yet. In fact, the automation of the creation of drawings is a complex task which would encapsulate a plethora of steps such as edge detection in multiple scales, edge vectorization, topology check of the vectorized edges, among others.

Nowadays, 2D-3D architectural drawings using photogrammetry are created manually, especially in the cultural heritage domain in which the objects of interest are commonly characterized by complex surfaces. Useful photogrammetric products for the production of architectural drawings are the orthophotographies usually produced by ortho projecting the textured 3D model created using the conventional photogrammetric pipeline. Afterwards, the orthophotomap is manually vectorized using a CAD environment to produce 2D vector drawings. The production of 3D vector drawings is conducted manually by vectorizing the 3D model or the 3D dense point cloud. In fact, the state-of-the-art process for both the creation of 2D and 3D vector drawings exploiting photogrammetric products, is time-consuming, laborious and it also requires specialists from several scientific fields, e.g., surveyors and architects, during the entire process.

In fact, orthophotos and orthophotomaps are raster data which combine the image's visual information with the ability to perform measurements on them. Additionally, many of the steps required for the creation of orthophotos are performed automatically. Thus, the architectural vector drawings in 2D space can be easily replaced by orthophotos and orthophotomaps because they contain both the visual and measurement information. However, most of the users of 2D architectural vector drawings still prefer the traditional architectural vector drawings than orthophotomaps and thus an automation of the procedure producing them, will be beneficial for the community.

In this effort, a 3D edge detection method is proposed, exploiting manually generated 2D edge semantic information. The main idea of this effort (Figure 1), which is the production of semantically enriched dense point clouds using four-channel images, was evaluated by implementing a simple experiment as proof-of-concept. More precisely, two scripts were developed for the experiment, the first script checks the radiometry of the created four-channel images, with respect to the original RGB images and the original edge semantic information and the second script uses the principles of epipolar geometry for the production of a non-refined 3D semantically enriched point cloud. Moreover, another experiment using professional SfM-MVS software, in combination with four-channel images, was conducted to create an improved semantically enriched dense point cloud, to compensate for the weaknesses of the first experiment. Finally, the 3D edge points were detected by classifying the 3D points into edge and non-edge points with respect to their label value. Apart from the 3D edge detection a 3D edge comparison between the detected 3D edge points and the corresponding 3D edges of a georeferenced textured 3D model was made, with several conclusions regarding to the 3D position and the length of each edge.

The rest of the paper is structured as follows: Section 2 presents a brief review of the literature. Section 3 describes the proposed approach and section 4 presents the conducted experiments. Section 5 comments on the results of the method. Finally, sec-

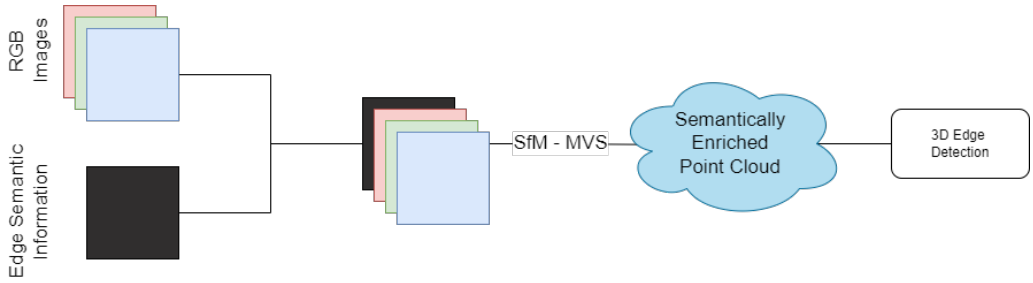


Figure 1. The general idea

tion 6 draws some conclusions for the proposed approach and presents some ideas for future work.

2. RELATED WORK

2.1 2D Edge Detection

Edge detection is a fundamental computer vision problem especially in 2D space and various traditional edge detection operators, such as Sobel (Sobel et al., 1968), Prewitt (Prewitt et al., 1970), Scharr (Kroon, 2009), Kirsh (Kirsch, 1971), Marr-Hildreth (Marr and Hildreth, 1980) and Canny (Canny, 1983, 1986), are proposed in the literature. Moreover, a few methods propose deep learning architectures to detect 2D edges with better results than the results of traditional approaches (Xie and Tu, 2015; Poma et al., 2020; Su et al., 2021). Segmentation techniques aim to cluster the pixels or points into groups with similar characteristics (geometric, spectral) without taking into account semantic meaning (Xie et al., 2020). Image edge detection techniques inspire the development of edge-based 3D point cloud segmentation approaches (Xie et al., 2020).

Several edge-based segmentation methods are presented in the literature (Wani and Arabnia, 2003; Senthilkumaran and Rajesh, 2009; Xie et al., 2020). Although there is a plethora of available automatic image edge detection methods, in this effort the edge semantic information is created manually for accuracy purposes analyzed in Sections 3 and 5. Besides, the scope of this effort is to develop a 3D edge detection method exploiting the 2D edge semantic information and not to identify the best automatic 2D edge detection method.

2.2 3D Edge Detection

Methods presented in the literature detect 3D edges using several techniques e.g., model fitting, normal vectors, analytical geometry etc. To be more specific, Nguatem et al. (2014) use predefined templates of windows and doors in order to detect their 3D boundaries exploiting plane intersection. Mitropoulou and Georgopoulos (2019) firstly segment 3D point clouds into planes and then detect the 3D edge points applying plane intersection. Moreover, Bazazian et al. (2015) firstly find the k-nearest neighbors of 3D points. For each group of 3D points the covariance matrix is calculated. Finally the eigenvalues and eigenvectors of each matrix are examined to detect the sharp 3D edges by deciding whether the point lays on a plane or on an edge. The proposed approach was evaluated on both synthetic and real-world data. Lu et al. (2019) are also exploit the eigenvalues and eigenvectors of the calculated covariance matrix of the points' neighborhood, but the neighborhood is defined using a region growing and region merging iterative approach in contrast to kNN algorithm. To be more specific, the 3D point cloud is segmented into planes, using the region growing and merging method. Afterwards, the 3D points of each fitted plane, are projected onto it, to create images. Finally, a 2D contour detection algorithm is applied and the detected 2D contours are projected back into 3D space. Additionally, Dolapsaki and Georgopoulos (2021), proposed a 3D edge detection method which exploits the relationships of analytical geometry and the properties of planes in combination with digital images. More concretely, the desired edge is firstly detected in a digital image of known exterior orientation, then the plane on which both the 2D edge and the perspective center of the image lay, is defined. Finally, the desired 3D edge points inevitably lie on the same plane and thus are detected.

Detecting edges in 3D space can be performed using 2.5D data e.g., range images, depth images. Bao et al. (2015) proposed an approach which firstly creates range images from a given point cloud, then applies canny operator on them and finally projects the 2D edges into 3D space. Alshawabkeh (2020) was created structured depth images from LiDAR point cloud and was combined them with RGB images to construct RGBD images. The goal of the proposed approach was to detect 3D cracks on the Treasury Monument of ancient Petra in Jordan. This, was achieved by detecting 2D linear features on RGBD images and projecting them in 3D space.

3. METHODOLOGY

Digital images are commonly used for the documentation of monuments and in general the documentation of cultural heritage objects. The state-of-the-art photogrammetric pipeline includes, image alignment, depth maps generation, dense point cloud production, 3D model generation and texturing. Additionally, several steps which clear the point clouds from noise and correct the surface of the 3D model are implemented during a post process procedure. The final product, which is a textured 3D model is useful in many applications such as digital museums, 3D documentation etc.

In this effort, the available RGB images are enriched with an additional to the RGB, channel in which edge semantic information is presented. The edge semantic information is produced manually by annotating the given images in a drawing environment. The four-channel images are passed into a 3D point cloud production algorithms to create a semantically enriched point cloud. Finally the 3D edges are detected by identifying the points with a specific label value. The previously described steps are displayed in Figure 2. The scope of this effort is to contribute to the automation of the production of 2D-3D architectural vector drawings and not to simply detect 3D edges. Thus, a 3D edge comparison between edges in point clouds and

3D meshes is conducted. Besides, the creation of architectural vector drawings automatically is still an open issue. Each step of the proposed approach, is thoroughly reviewed into the following sub sections.

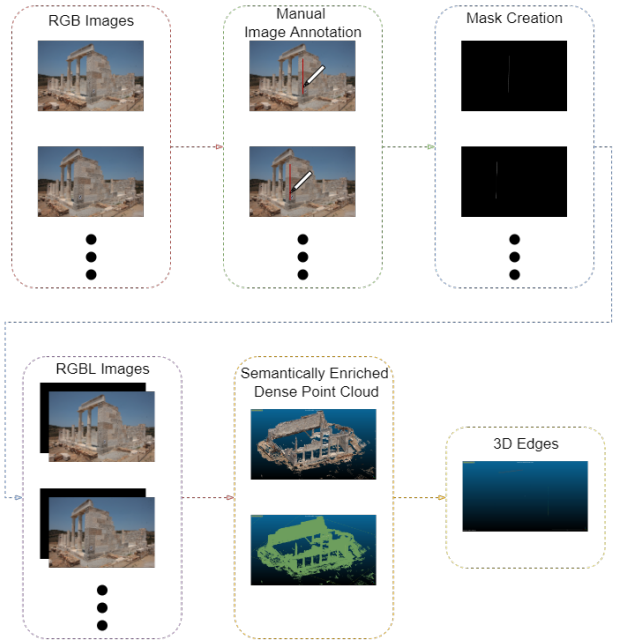
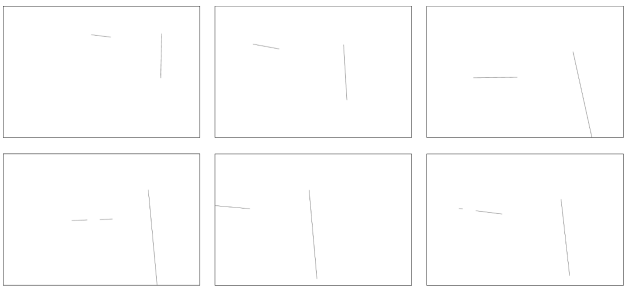


Figure 2. The proposed method

(a) Manually annotated RGB images.



(b) Reversed binary edge maps.

Figure 3. Manually annotated RGB images and binary edge maps

3.1 Labels Creation and Image Enrichment

First and foremost, the edge semantic information must be generated to enrich the source data i.e., the digital images, with an extra channel. In this paper, the edges are defined manually by annotating them on the digital images using red color. Figure 3a depicts six out of the seventy six images used during this investigation. The labels must be one channel images to enrich the RGB images. Thus, the RGB manually annotated images were transformed to binary edge maps, by identifying the red pixels on them. A part of the created edge maps are displayed in Figure 3b. Afterwards, each RGB image was enriched with the corresponding binary edge map using the Python version of the OpenCV (OpenCV, 2022) library and specifically the ”merge” method and it finally stored as pseudo ”.tiff” images.

3.2 3D Point Cloud Creation

Creating semantically enriched point clouds is useful for many applications, such as to improve the SfM-MVS procedure (Stathopoulou et al., 2021). In this paper the semantically enriched point cloud is produced exploiting the four-channel images produced using the RGB images and their edge maps as described in 3.1. Two methods were developed for the production of the semantically enriched point cloud, the ”Triangulation” and the ”Agisoft-Metashape” (Agisoft-Metashape, 2016).

Triangulation algorithm uses the python version of the Opencv library for scene reconstruction. To be more specific, the ”Triangulation” algorithm simplifies the complex steps, of an SfM-MVS algorithm, into the standard two-image epipolar geometry problem. The major steps of the proposed algorithm are (i) Extract images' features using one of the available algorithms i.e., Akaze (Alcantarilla and Solutions, 2011), SIFT (Lowe, 2004), SURF (Bay et al., 2006) or ORB (Rublee et al., 2011), (ii) choose the image pairs and match each of them using a Flann based matcher (Muja and Lowe, 2009). Image pairs selection is implemented with respect to the number of the given images, for instance for five images (1, ..., 5) ten image pairs are constructed i.e., (1-2, 1-3, 1-4, 1-5, 2-3, 2-4, 2-5, 3-4, 3-5 and 4-5). Then, (iii) calculate the essential matrix exploiting the detected points and the camera matrix or the fundamental matrix, if the camera matrix is not available. The camera matrix is calculated using the exif data of each image, from which the focal length and the principal point, are retrieved. If the principal point is not available, it is defined as the center of the image. Afterwards, (iv) calculate the rotation and translation matrix of one image of the image pair, with respect to the other image. Finally, (v) calculate the projection matrix, and generate a semantically enriched point cloud for each image pair i.e., ten different point clouds.

Producing point clouds using a simple triangulation process leads to several problems. A thorough review of the quality of the point clouds, using the ”Triangulation” algorithm is presented in Section 5. The most important problem is the production of a non-refined point cloud since the bundle adjustment step is omitted. Additionally, the ”Triangulation” algorithm is computationally inefficient. Thus, professional SfM-MVS e.g., Agisoft-Metashape, was combined with the four-channel images, for the production of a semantically enriched dense point cloud. Agisoft-Metashape has the ability to create semantically enriched point clouds by default since it can process multispectral image. Additionally, Agisoft-Metashape is one of the leading photogrammetric software so it guarantees that the semantically enriched point cloud is produced exploiting state-of-the-art techniques. The semantically enriched dense point clouds produced using the ”Triangulation” algorithm as well as the Agisoft-Metashape software are depicted in Section 4, in which both experiments are presented.

3.3 3D Edge Points Classification

During this effort, 3D edge detection, is conducted using a binary classification procedure. Producing semantically enriched dense point clouds means that except for the geometric and color information, there is also semantic information into the saved file. In this effort, edge semantic information is passed into the final file, which is stored in various formats such as ".ply", ".txt" and ".pts", after the color values. Each 3D point is associated with a label value ranging from 0 to 255 which represents the non-edge and edge points, respectively. Finally, 3D edge points are detected by separating the 3D points with label value greater than an empirical threshold value. In fact, the edge maps used as the forth channel into RGBL images, contain only 0 and 255 values, but the produced ASCII files of the semantically enriched point clouds contain label values ranging from 0 to 255. On the one hand, for the "Triangulation" algorithm the reason for the aforementioned range should be the annotation process in which the radiometry of the points close to the annotated edges may be affected. On the other hand, the Agisoft-Metashape may produce such labels i.e., in the range between 0 and 255, because images' dimensions change during the process to robustly handle large number of images. Also, the manual image annotation process affects the implementation using the Agisoft-Metashape as using the "Triangulation" algorithm. Thus, we define an empirical threshold value, to classify the points over the threshold as edge points and all the rest as non-edge points.

3.4 3D Edge Comparison

Several products are available in 3D space to detect 3D edges like sparse point clouds, dense point clouds, 3D models and textured 3D models. All of them can be georeferenced. In this effort, a comparison between 3D edges detected in a georeferenced dense point cloud and on a georeferenced textured 3D model, is conducted. Also, sparse point cloud was excluded from the comparison, since we would like a dense representation of the edges i.e., the edges should contain as many 3D points as possible. Apart from that, the textureless 3D model was also omitted since texture is a fundamental characteristic to visually detect edges on 3D models. Both, the 3D dense point cloud and the textured 3D model were georeferenced using the same control and test points, in order to make the validation process accurate. The textured 3D model was generated using a FARO 330X terrestrial laser scanner for the creation of the object geometry in combination with digital images, which were aligned using the Agisoft-Metashape pro software v.1.4., for the creation of object texture. The flowchart of the method presented in this paper is depicted in Figure 4

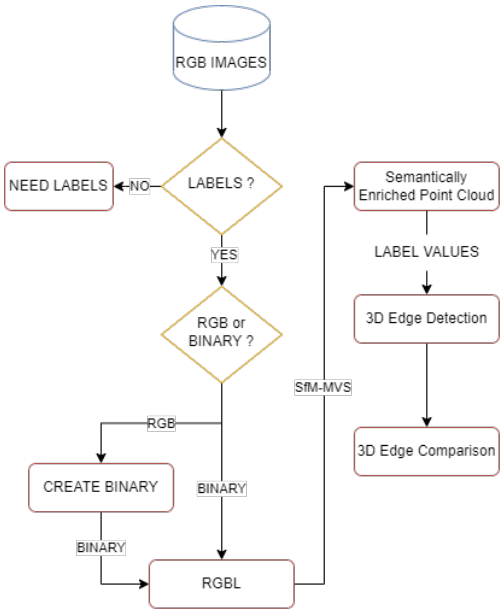


Figure 4. Flowchart of the proposed approach

4. RESULTS

One set of data used during this effort was the RGB images and the 3D model produced during two postgraduate theses ((Giannakoula, 2018; Stefanou, 2018)) which were documenting the Temple of Demeter in Naxos. The documentation was conducted using professional digital cameras and a Faro Laser Scanner, from which the textured 3D Model was created. The detection of 3D edges with the proposed approach exploits the available RGB images while the comparison between the detected 3D edges exploit the generated textured 3D Model. The second set of data, which are two aerial RGB images, captured in 2019 during a summer field course in the ancient Kymissala in Rhodes, organized by the Laboratory of Photogrammetry SRSGE NTUA. The two RGB images were used in order to check the performance of the "Triangulation" algorithm using aerial images instead of terrestrial, because the performance of the algorithm using the latter was not satisfying. A sample of images from both datasets, is depicted in Figure 5

Figure 5. RGB images from Temple of Demeter (First row) and summer field course (Second Row).

4.1 Semantically Enriched Point Cloud Creation

In this section, the results from both the "Triangulation" and the "Agisoft-Metashape" methods, are presented. Figure 6 presents the semantically enriched point clouds produced by the "Triangulation" algorithm. The first row depicts two 3D point clouds created using the two aerial images. The difference between the point clouds is that the first one was created using the Akaze while the second one using the Sift, feature extraction method. Additionally, the second row presents the semantically enriched 3D point cloud using two terrestrial images of The temple of Demeter dataset.

The semantic information is passed into the 3D point cloud as an extra value to the position and color values. Thus, the ASCII file contains seven values instead of six (Figure 7).

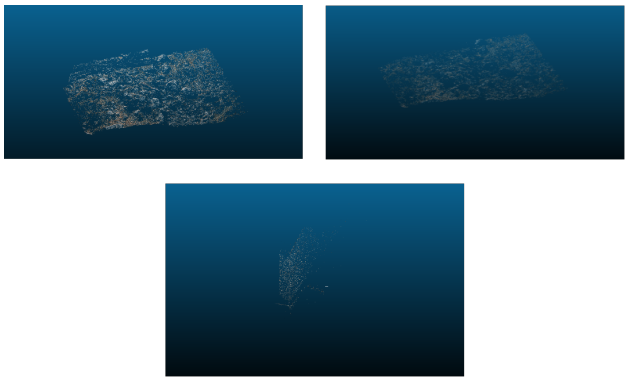


Figure 6. Semantically enriched 3D point clouds created using the Triangulation algorithm. First row presents the point clouds created exploiting the aerial images while the second row exploiting the terrestrial images

```
-0.2705, -5.7712, -1.0233, 132, 116, 103, 255
-0.3084, -5.9694, 0.6245, 160, 151, 146, 0
-0.3127, -6.0133, 0.8886, 102, 89, 81, 0
-0.3165, -6.0197, 1.0661, 115, 105, 96, 0
-0.312, -6.0024, 0.8776, 141, 134, 128, 0
-0.3122, -6.0088, 0.9045, 114, 101, 95, 0
-0.3498, -6.1129, 2.3592, 118, 99, 85, 0
-0.3151, -6.0231, 1.0388, 167, 159, 146, 0
-0.3191, -6.0625, 1.2817, 164, 164, 152, 0
-0.2789, -5.8154, -0.6174, 194, 187, 171, 0
-0.302, -5.9441, 0.4165, 173, 165, 154, 0
-0.2837, -5.8492, -0.3386, 180, 171, 156, 1
```

Figure 7. ASCII file produced using the Triangulation algorithm

It is obvious that the 3D semantically enriched point clouds produced by the triangulation algorithm were not of sufficient quality especially using the terrestrial images. Besides, the scope of the ”Triangulation” algorithm was to evaluate the general idea under the simplest conditions. Thus, the Agisoft-Metashape photogrammetric software was exploited in combination with the constructed four-channel images. In this experiment, seventy six RGBL images were used. Figure 8 depicts the semantically enriched 3D dense point cloud generated using the Agisoft-Metashape software. The first row depicts the same view with different colors i.e., RGB and Scalar. The 3D edges can be observed using the scalar visualization. The second row depicts the same as the first one, but from a closer distance.

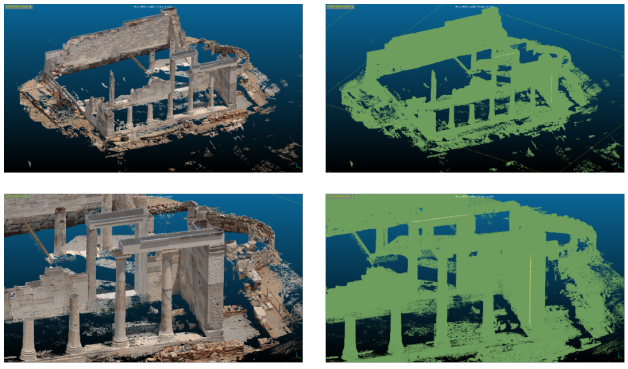


Figure 8. Semantically enriched dense 3D point cloud created using the Agisoft-Metashape software.

The edge semantic information is passed into the 3D point cloud in a way similar to the ”Triangulation” algorithm. Figure 9 depicts the ASCII file produced using the Agisoft-Metashape software.

```
1011.4719 1016.7871 103.1458 93 83 76 0
1011.5060 1016.8475 103.1418 144 136 125 250
1011.4520 1016.7606 103.1304 95 87 81 0
1011.5056 1016.8498 103.1305 138 131 122 209
1011.5043 1016.8515 103.1179 121 117 110 126
1011.4748 1016.8705 103.1089 129 113 105 26
1011.4329 1016.8993 103.0139 147 130 119 0
```

Figure 9. ASCII file produced using the Agisoft-Metashape software

Finally 3D edge detection is performed by classifying the 3D points into edge and non-edge regarding their label value. Observing the ASCII files of the semantically enriched 3D point clouds the label value ranges from 0 to 255, for several reasons as described in section 3.3. Figure 10 presents an experiment conducted to specify the empirical threshold value described in section 3.3. The classification is performed using six different threshold values i.e., greater than 0, 50, 80, 100, 230 and equal to 255, to examine the detected 3D edge points under each condition. By increasing the threshold value, the detected 3D edge points seems to be closely to the desired 3D edges. A threshold value around 100, it visually seems to be the best one because after that value many visually correct 3D points are excluded, resulting to shorter edges (Figure 10).

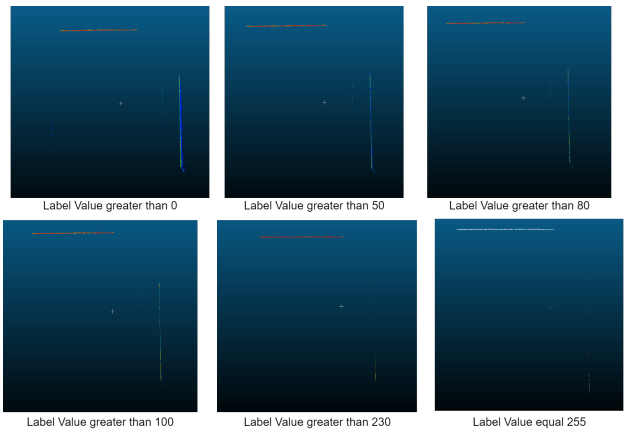


Figure 10. Label values threshold experiment

4.2 3D Edge Comparison

The comparison of the detected 3D edges, was performed regarding (i) the 3D edge start point and end point and (ii) the 3D distance. To be more specific, there are four 3D edges (two horizontal and two vertical), two of them are detected from the 3D model (one horizontal and one vertical) and the other two are detected using the proposed approach (one horizontal and one vertical). The start and end points were clearly defined in all point clouds. The horizontal and vertical edges which were collected from the 3D Model are the same as those which were collected by the proposed approach, in 3D space. Thus, the comparison between them can be conducted. Additionally, the accuracy of the 3D Model and the 3D dense point cloud is estimates to approx. 0.008m.

4.2.1 3D start point and 3D end point: A series of measurements were performed using the Cloud-Compare (CloudCompare, 2003) software. To be more precise, the textured 3D

model was inserted in the Cloud-Compare and then, the 3D start point and end point of each edge were picked multiple times using the "point picking" tool. In fact, we got several "observations" of the start and end point of each edge from the 3D model. The same steps were applied using the detected, by the proposed approach, 3D edges. Then, an average value of each set of observations were calculated for the 3D model and the 3D edge detection approach for each edge and were subtracted to each other according to X, Y and Z dimension (Table 1).

Horizontal Edge			
3D Model	X (m)	Y (m)	Z (m)
Start Point	1012.755	1011.590	106.326
End Point	1010.670	1012.950	106.337
3D Edge Detection	X (m)	Y (m)	Z (m)
Start Point	1012.746	1011.574	106.33
End Point	1010.662	1012.933	106.332
Absolute Differences			
Start Point	**0.008**	**0.016**	**0.004**
End Point	**0.008**	**0.017**	**0.006**

Vertical Edge			
3D Model	X (m)	Y (m)	Z (m)
Start Point	1011.492	1016.868	102.937
End Point	1011.490	1016.860	106.063
3D Edge Detection	X (m)	Y (m)	Z (m)
Start Point	1011.499	1016.849	102.914
End Point	1011.488	1016.851	106.054
Absolute Differences (m)			
Start Point	**0.007**	**0.019**	**0.023**
End Point	**0.002**	**0.010**	**0.009**

Table 1. Start point and end point comparison

4.2.2 Length: Apart from the 3D coordinates of the start and end points, multiple observations of the length of each edge were collected. Then, the average length and the difference between the average length using the 3D model and the 3D edge detection approach, were also calculated. Each average length and the absolute difference of them are presented in Table 2.

Horizontal	3D Model	3D Edge Detection
	3D Euclidean Distance (m)	
Average Value	2.500	2.484
Difference	**0.016**	

Vertical	3D Model	3D Edge Detection
	3D Euclidean Distance (m)	
Average Value	3.147	3.154
Difference	**0.007**	

Table 2. Length of each edge as the average 3D euclidean distance between a set of start and end points.

5. DISCUSSION OF RESULTS

The presented results are visually evaluated regarding each point cloud creation method. More concretely, the "Triangulation" algorithm produces a non-refined 3D point cloud for each image pair, which obviously is computationally expensive. Additionally, it struggles to produce point clouds using terrestrial images as depicted in Figure 6. However, the scope was to evaluate the general idea and to investigate it using a simple experiment, which was achieved. On the other hand, the proposed method is combined with a professional photogrammetric pipeline i.e., Agisoft-Metashape, thus the semantically enriched 3D dense point cloud could be of state-of-the-art quality. In this effort the 3D point clouds were not post processed to achieve the best results. Additionally, the creation procedure, of the four-channel images, associates correctly the pixels between the RGB channels and the label channel since the detected 3D edge points are in the correct 3D position.

The 3D edge comparison approach was performed taking into account a hypothetical 3D vectorization procedure. To be more specific, we assume that we want to 3D vectorize the desired edge using either the detected 3D edge or the 3D model. Hence, multiple observation using both data sources were conducted for each edge. The absolute differences between each case are depicted in Table 1. No safe conclusion can be derived for the accuracy of the proposed approached due to the limited experiments. However, the absolute differences are into the general desired range i.e., are not observed extreme differences and are close to the accuracy of the source data i.e., 0.008 (m).

6. CONCLUSIONS

In this paper, a first implementation of 3D edge detection method which exploits four-channel images, is presented. The proposed approach was evaluated on real-world data of cultural heritage assets. The proposed approach has several drawbacks which could be a starting point for future work and research. To be more specific, automated 2D edge detection, traditional and learning, algorithms could be included to the proposed approach. Additionally, open source SfM-MVS photogrammetric pipelines like VisualSfM and CMVS-PMVS (VisualSfM, 2022), Meshroom (Meshroom-AliceVision, 2022) and OpenSfM (Mapillary-OpenSfM, 2022) could be tested using four-channel images for the production of semantically enriched point clouds in order for the proposed approach to be available to anyone. Of course, a 3D vectorization procedure should be integrated into the proposed method to automate the production of 3D architectural vector drawings. Additionally, a comparison using a large amount of 3D edges should be conducted to correctly evaluate the proposed approach. However, the proposed approach is a simple yet efficient pipeline because it exploits the SfM-MVS advantages e.g., bundle adjustment, to detect 3D edges with the best possible accuracy depending on the 2D edge semantic information. Moreover, the proposed method does not introduce additional errors, as it does not involve further mathematical calculations to the SfM-MVS standard ones.

References

Agisoft-Metashape, 2016. Discover intelligent photogrammetry with Metashape. http://www.agisoft.com/. Accessed: 2022-10-26.

Alcantarilla, P. F., Solutions, T., 2011. Fast explicit diffusion for accelerated features in nonlinear scale spaces. *IEEE Trans. Patt. Anal. Mach. Intell*, 34(7), 1281–1298.

Alshawabkeh, Y., 2020. Linear feature extraction from point cloud using color information. 8(1), 28. https://heritagesciencejournal.springeropen.com/articles/10.1186/s40494-020-00371-6.

Bao, T., Zhao, J., Xu, M., 2015. Step edge detection method for 3D point clouds based on 2D range images. 126(20), 2706–2710. https://linkinghub.elsevier.com/retrieve/pii/S0030402615005586.

Bay, H., Tuytelaars, T., Gool, L. V., 2006. Surf: Speeded up robust features. *European conference on computer vision*, Springer, 404–417.

Bazazian, D., Casas, J. R., Ruiz-Hidalgo, J., 2015. Fast and robust edge extraction in unorganized point clouds. *2015 international conference on digital image computing: techniques and applications (DICTA)*, IEEE, 1–8.

Canny, J. F., 1983. Finding edges and lines in images. Technical report, MASSACHUSETTS INST OF TECH CAMBRIDGE ARTIFICIAL INTELLIGENCE LAB.

Canny, J. F., 1986. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, 679–698.

Cloud-Compare, 2003. 3D point cloud and mesh processing software. https://www.danielgm.net/cc/. Accessed: 2022-10-26.

Dolapsaki, M. M., Georgopoulos, A., 2021. Edge Detection in 3D Point Clouds Using Digital Images. 10(4), 229. Publisher: MDPI.

Giannakoula, X., 2018. Geometrical Documentation of monuments using modern technologies, an application on the Temple of Demeter in Naxos. (In Greek).

Kirsch, R. A., 1971. Computer determination of the constituent structure of biological images. *Computers and biomedical research*, 4(3), 315–328.

Kroon, D., 2009. Numerical optimization of kernel based image derivatives. *Short Paper University Twente*, 3.

Lowe, D. G., 2004. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2), 91–110.

Lu, X., Liu, Y., Li, K., 2019. Fast 3D line segment detection from unorganized point cloud.

Mapillary-OpenSfM, 2022. An open-source Structure from Motion library that lets you build 3D models from images. https://opensfm.org/. Accessed: 2022-10-04.

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

Meshroom-AliceVision, 2022. Photogrammetric Computer Vision Framework. https://alicevision.org/. Accessed: 2022-10-26.

Mitropoulou, A., Georgopoulos, A., 2019. AN AUTOMATED PROCESS TO DETECT EDGES IN UNORGANIZED POINT CLOUDS. 4.

Muja, M., Lowe, D., 2009. Flann-fast library for approximate nearest neighbors user manual. *Computer Science Department, University of British Columbia, Vancouver, BC, Canada*, 5.

Nguatem, W., Drauschke, M., Mayer, H., 2014. Localization of Windows and Doors in 3d Point Clouds of Facades. II-3, 87–94. https://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/II-3/87/2014/.

OpenCV, 2022. Open Source Computer Vision Library. https://opencv.org/. Accessed: 2022-10-04.

Poma, X. S., Riba, E., Sappa, A., 2020. Dense extreme inception network: Towards a robust cnn model for edge detection. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 1923–1932.

Prewitt, J. M. et al., 1970. Object enhancement and extraction. *Picture processing and Psychopictorics*, 10(1), 15–19.

Rublee, E., Rabaud, V., Konolige, K., Bradski, G., 2011. Orb: An efficient alternative to sift or surf. *2011 International conference on computer vision*, Ieee, 2564–2571.

Senthilkumaran, N., Rajesh, R., 2009. Image segmentation-a survey of soft computing approaches. *2009 International Conference on Advances in Recent Technologies in Communication and Computing*, IEEE, 844–846.

Sobel, I., Feldman, G. et al., 1968. A 3x3 isotropic gradient operator for image processing. *a talk at the Stanford Artificial Project in*, 271–272.

Stathopoulou, E. K., Battisti, R., Cernea, D., Remondino, F., Georgopoulos, A., 2021. Semantically derived geometric constraints for MVS reconstruction of textureless areas. 13(6), 1053. Publisher: MDPI.

Stefanou, A. B., 2018. The contribution of new technologies to archeology: the case of the classic church of Demeter in Sagri Naxos. (In Greek).

Su, Z., Liu, W., Yu, Z., Hu, D., Liao, Q., Tian, Q., Pietikäinen, M., Liu, L., 2021. Pixel difference networks for efficient edge detection. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5117–5127.

Visual-SfM, 2022. VisualSFM : A Visual Structure from Motion System. http://ccwu.me/vsfm/index.html/. Accessed: 2022-10-26.

Wani, M. A., Arabnia, H. R., 2003. Parallel edge-region-based segmentation algorithm targeted at reconfigurable multiring network. *The Journal of Supercomputing*, 25(1), 43–62.

Xie, S., Tu, Z., 2015. Holistically-Nested Edge Detection. 9.

Xie, Y., Tian, J., Zhu, X. X., 2020. Linking points with labels in 3D: A review of point cloud semantic segmentation. *IEEE Geoscience and Remote Sensing Magazine*, 8(4), 38–59.