Salient object detection in SfM point cloud

Divyansh Agarwal *
divyansh.agarwal@students.iiit.ac.in

Nishit Soni *
hiteshkumar.soni@students.iiit.ac.in

Anoop M. Namboodiri *
anoop@iiit.ac.in

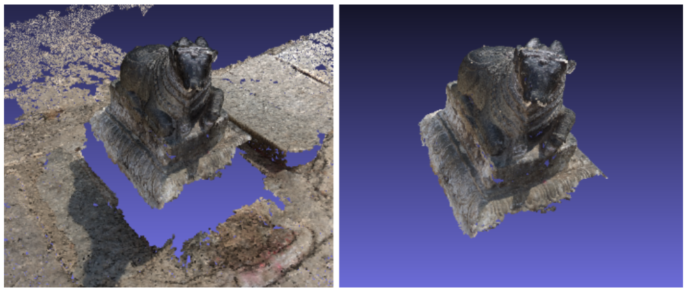


Fig. 1. An example showing salient object being detected in SfM point cloud. The first image shows a raw SfM point cloud with background points that corresponds to the floor around a nandi structure. The second image shows the prominent object.

1

***Abstract*—In this paper we present a max-flow min-cut based salient object detection in 3D point cloud that results from Structure from Motion (SfM) pipeline. The SfM pipeline generates noisy point cloud due to the unwanted scenes captured along with the object in the image dataset of SfM. The background points being sparse and not meaningful, it becomes necessary to remove them. Hence, any further processes (like surface reconstruction) utilizing the cleaned up model will have no hinderance from the noise removed. We present a novel approach where the camera centers are used to segment out the salient object. The algorithm is completely autonomous and does not need any user input. We test our proposed method on Indian historical models reconstructed through SfM. We evaluate the results in terms of selectivity and specificity.**

I. Introduction

In recent years there has been great advancement in the generation of 3D structures from images. The technique is called structure from motion (SfM). SfM is now applied to develop large scale structures like big buildings, historical places and so on. SfM uses images and generates 3D point cloud and estimates the camera centers. It involves extracting features from images, matching them, building feature tracks, estimating the 3D position of the features and finally optimizing the estimates. As it is not possible to capture only the object in images, some of the background also gets reconstructed into a sparse 3D point cloud. The proposed algorithm takes the 3D point cloud resulted from SfM as input and aims to remove these unwanted background points and segregate only the object of interest from the point cloud.

After the object is detected and segregated, surface reconstruction can be applied over the object. The surface reconstruction techniques does not differentiate between connecting parts of the same object or splitting two different objects into separate meshes. With the background points in the point cloud, undesirable mesh is produced. Thus it becomes necessary to apply surface reconstruction only on the object. This justifies the aim of this paper. Figure I depicts the black box explanation of the algorithm. The algorithm takes the 3D point cloud and camera positions as input from SfM pipeline. It outputs the detected object.

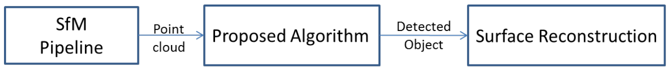


Fig. 2. Black box explanation; the second box represents our algorithm.

The paper is organized as: Section II covers the related work on this area. Section III explains the method. Section IV shows results and evaluation. Section V concludes the paper and discusses future work.

II. Related work

In case of 3D meshes, surface properties have been used to partition mesh into useful segments. [1] uses morphological watershed, an image segmentation technique, to segment 3D surfaces. It uses the consistency of the curvature of surfaces and segments them accordingly. [2] proposes hill-climbing watershed algorithm that identifies regions bounded by contours of negative curvature minimal. [3] shows segmentation in 3D LIDAR point cloud. Point cloud segmentation has been also looked at in [4], [5] and [6]. These works emphasizes on segmentation of object(s) from a point cloud where the other objects (background) are also salient and are of interest. However this paper focuses on a salient object detection from an SfM point cloud where the background points does not represent any prominent object.

III. Method

The SfM pipeline takes a set of images and processes them to provide a 3D point cloud of the object. It also gives the camera position coordinates from where the images were taken. Our algorithm starts by taking this 3D point cloud and the camera positions as input. The algorithm utilizes camera positions to detect the object.

A. Overview

- The world ground plane is approximated.
- The 3D point cloud and camera centers are projected on this plane.

1
*Center for Visual Information Technology, IIIT-Hyderabad (http://cvit.iiit.ac.in)

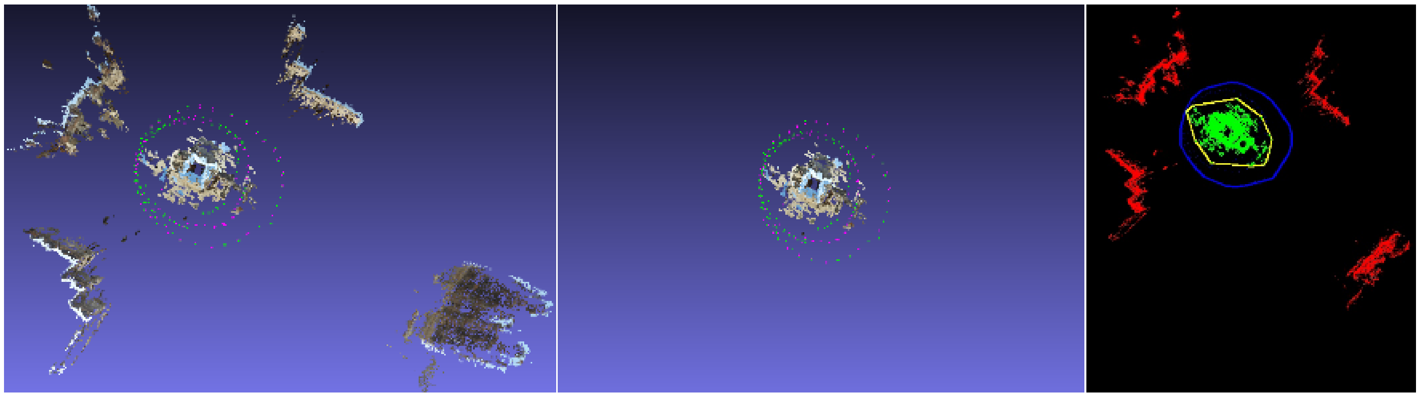


Fig. 3. In the first two images, top view of the monument along with the camera centers(green and pink colored points) is shown. (a) Monument before noise removal. (b) Monument after noise removal by the proposed algorithm, background points corresponding the surrounding monuments are removed. (c) shows the point cloud projected on the ground plane. The green and red dots represent the object and background points respectively. The blue and yellow boundaries represent $C1$ and $C2$ respectively.

- Convex hull of the camera centers is constructed and the inliers and the outliers are labeled accordingly.
- A graph is built having the vertices as the projected 2D model points.
- Energies are assigned to edges.
- Min-cut max-flow algorithm is applied and the object is detected.

B. *Estimate the Ground Plane*

The ground plane is defined as the plane that shares the same normal as that of the world ground plane. Camera center positions are used to estimate the ground plane. PCA is applied on the camera centers to get 3 eigenvectors and their corresponding eigenvalues. The vector corresponding to the smallest eigenvalue is assumed to be the normal vector to the world ground plane. This assumption is justified as there will be little variation in the heights of the camera centers compared to their spread around the model. The spread of camera positions along the X and Y directions of the world ground plane will be more than that in the Z direction. Hence, the plane containing the other two eigenvectors is the plane parallel to the world ground plane.

The above method is centered on the assumption that the camera centers will be surrounding the object of interest. When the camera centers are concentrated only on one side of the model, the ground plane estimation is undesirable. For example, the model shown in figure 4 resulted in a vertical ground plane. Thus, the proposed algorithm excludes such models in its ambit.

C. *Project points onto the ground plane and find convex hull*

The 3D point cloud and the camera centers are projected on the estimated ground plane. A convex hull $C1$ is defined from the projected camera centers. The projected model points enclosed by $C1$ would be the probable object points. The last image of Figure 3 shows the ground plane where the points are projected. The blue boundary shows the convex hull $C1$. The red and green points are the 2D model points corresponding to the background and object respectively.

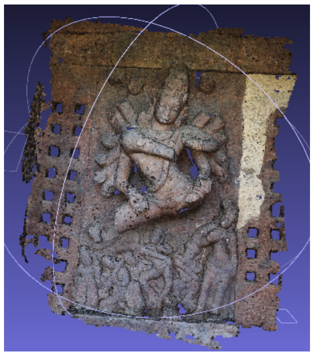


Fig. 4. Nataraj Model; camera centers focused only on one side of the monument and not enclosing the object.

D. *Build the graph and apply min-cut*

Approximate k nearest neighbour algorithm is applied to construct the graph of the projected points. Experimentally the value of k is chosen to be 7. Each 2D projected point is connected to its 7 neighbours. Then the object vertices are segmented from the graph by applying min-cut max-flow algorithm by Vladimir Kolmogorov [14].

1) Smooth energy: Smooth energy is assigned to the edges connecting two vertices. Higher the smooth energy, higher is the probability of the edge not getting cut i.e the vertices are part of the same class (either object or background). This is because two points belonging to the same class (object/background) will be close to each other. Hence, this energy is defined as :

$$S(v1, v2) = \frac{1}{dist(v1, v2)}$$

$$dist(v1, v2) = \sqrt{(x1 - x2)^2 + (y1 - y2)^2)}$$

Here $v1$ and $v2$ are any two projected model points. $(x1, y1)$ and $(x2, y2)$ are the 2D coordinates of $v1$ and $v2$. $dist(v1, v2)$ is the euclidean distance between $v1$ and $v2$. $s(v1, v2)$ is the smooth energy assigned to the edge between $v1$ and $v2$.

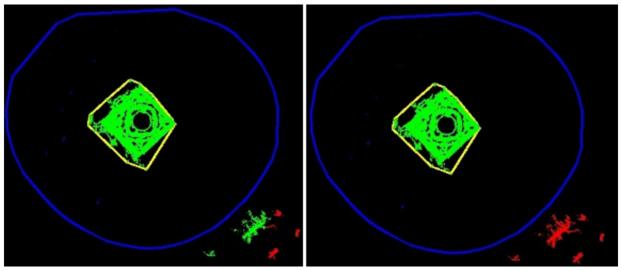


Fig. 5. Left and right images shows the case where distance is considered from $C1$ and $C2$ respectively. In the first case, the background points got included in the object because of proximity to $C1$. This is rectified in the second image by considering distance from $C2$ in the data energy function.

2) Data energy: Data energy is assigned to the edges connecting source/sink to every other vertex. Before defining the data energy, another convex hull $C2$ is defined from the projected model points lying inside $C1$.

$$D\_sink(v) = \begin{cases} 10 & \text{if v lies inside } C2; \\ 10 * e^{-x^2} & \text{if v lies outside } C2. \end{cases}$$

$$D\_source(v) = 10 - D\_sink(v)$$

Here $D\_sink(v)/D\_source(v)$ is the data energy assigned to the edge connecting sink/source and vertex v. x is the distance of v from $C2$ rather than $C1$. If it was taken from $C1$ then similar data energies would be assigned to the outliers at similar distance from $C1$ irrespective of their proximity from the actual object. This is illustrated in figure 5.

Assignment of these energies ensures that :

- Closer points get higher probability of belonging to the same class (object/background).
- Points lying inside $C2$ get higher chance of getting classified as object points. Their probability decreases rapidly as distance from the boundary of $C2$ increases.

IV. Results and Evaluation

A. Dataset and code

The dataset consists of 3D point clouds of Indian historical monuments. The point cloud is obtained using VisualSFM by Changchang Wu ([7], [8], [9], [10], [11], [12], [13]) on different image datasets. The proposed algorithm is implemented in C++. We use openCV library for approximate k nearest neighbour algorithm. Vladimir Kolmogorovi's implementation of min-cut max-flow algorithm is used [14].

B. Evaluation Criteria

For evaluation, we compare our output with the ground truth. We calculate confusion matrix for each output. The two classes are : object and background. We define positive test as the point being classified as object. The evaluation is done on the basis of selectivity and specificity. Selectivity is the probability of an object point being classified as object and specificity is the probability of a background point being classified as background.

We manually segment the object of interest from the SfM point cloud of monuments to generate the ground truth. We also manually define an immediate surrounding (wherever applicable) of the object that can neither be truly classified as object nor background. This is referred as proximity region. It helps to provide context to the object.

C. Results

Table I shows the evaluation of our results. As seen in this table, the selectivity of all the results is 1.0 because all object points are detected. However some of the specificities do not look promising. This is because the detected object has some non-object points also. But these non-object points basically represent the proximity area around the object (as defined in section IV-B) and so does not necessarily represent the background. Looking at the proximity : background ratio values in table I, it can be noticed that most of the non-object points in the detected object correspond to the proximity area rather than pure background. Table II shows the output images along with the ground truth and raw point cloud.

V. Conclusion

The selectivity of all the outputs is 1.0. This implies that none of the object points remain undetected. Also the average of the specificities comes out to be 0.77. This implies that the background is reasonably well segmented out leaving the object of interest and some background points in the scene. The little background which gets included in the segmented object is generally the proximity area (ground/floor) as seen in table I. This can be improved by designing a ground elimination algorithm.

The min-cut is applied on the 2D vertices on the 2D ground plane (XY plane). Their variation along Z direction is not considered. So when the points are projected on the ground plane, the points that correspond to the artifacts (sky/tree) above the object are also projected along the object and are segmented as object. This needs to be addressed. A possible strategy is to make use of the height (along Z direction) and colour information to eliminate such artifacts.

The models like the one shown in figure 4, where the ground plane is not along the world ground plane, are not addressed by our algorithm. An alternate way to estimate the ground plane is required. This may require manual intervention to identify such a scenario.

References

[1] Mangan A. P. and Whitaker R. T. (1999) Partitioning 3D Surface Meshes Using Watershed Segmentation. IEEE Transactions on Visualization and Computer Graphics. Vol. 5, 1999. pp. 308-321.

[2] A.F.Koschan, M.A.Abidi and D.L.Page, "Perception-based 3d triangle mesh segmentation using fast marching watersheds," 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. II-27-II-32, Jun. 2003.

[3] B. Douillard, J. Underwood, N. Kuntz, V. Vlaskine, A.Quadros, P. Morton, and A. Frenkel, "On the Segmentation of 3D LIDAR Point Clouds", IEEE International Conference on Robotics and Automation, 2011.

Model no.	Points	Selectivity	Specificity	proximity : background area ratio in detected object	Algorithm running time
1	449400	1.0	0.70	83:16	0m42.923s
2	1586300	1.0	0.66	60:39	2m8.384s
3	1009360	1.0	0.62	75:24	1m15.873s
4	356282	1.0	0.67	93:6	0m34.870s
5	534464	1.0	0.88	93:6	0m36.766s
6	1023430	1.0	0.90	- (no immediate proximity exists)	1m18.577s
7	906480	1.0	0.79	- (no immediate proximity exists)	1m5.356s
8	358727	1.0	1.0	- (no immediate proximity exists)	0m26.026s

TABLE I. QUANTITATIVE ANALYSIS OF RESULTS

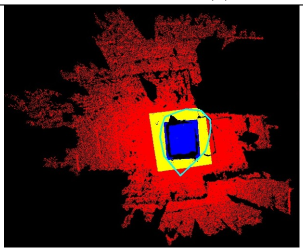
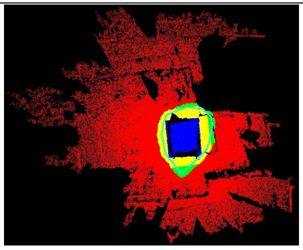
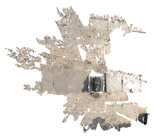
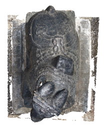
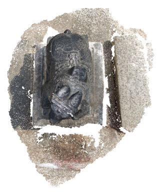
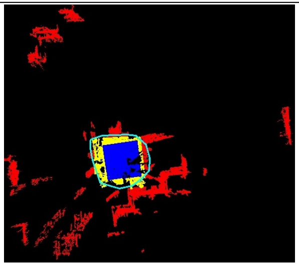
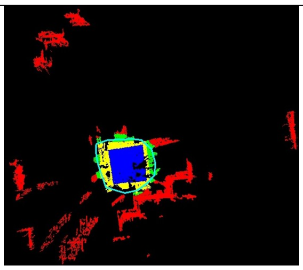
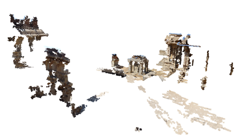
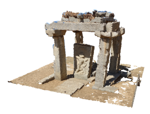
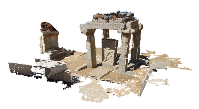
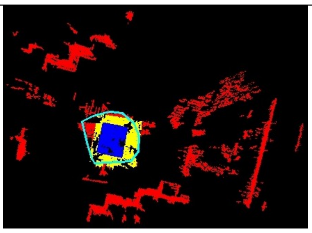
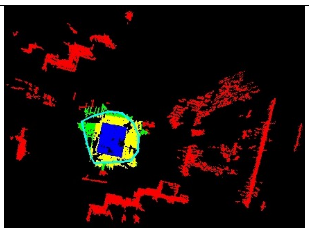
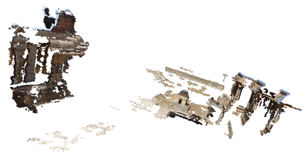
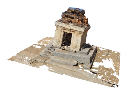
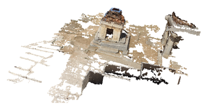
Ground truth (a)	Output (b)	Raw point cloud (c)	Ground truth (d)	Output (e)
				
				
				

TABLE II. RESULT

The below images shows results for model1(row1) , model2(row2) and model3(row3). The first two images (a,b) in each row are ground plane projections of the 3D point cloud. In these images, the blue boundary represents the hull C1. The first image (a) is the ground truth. Blue, yellow and red points represent the object, proximity area (as defined in section IV-B) and background respectively. The second image (b) is the detected object. The detected object is shown in combination of blue(object), yellow(context) and green(background) points. Rest of the red points are detected as non-object points. Images (c),(d) and (e) represent the 3D point cloud corresponding to the raw model, ground truth and the detected object respectively.

[4] H. Woo et al. ”A new segmentation method for point cloud data” International Journal of Machine Tools & Manufacture, 42 (2002), pp. 167178

[5] M. Johnson-Roberson, J. Bohg, M. Bjorkman, and D. Kragic, Attentioin based active 3d point cloud segmentation, IROS 2010, 2010

[6] A. Golovinskiy and T. Funkhouser, ”Min-cut based segmentation of point clouds, ” in IEEE Workshop on Search in 3D and Video (S3DV) at ICCV, Sep. 2009.

[7] Changchang Wu, ”SiftGPU: A GPU implementation of Scale Invaraint Feature Transform (SIFT)”, http://cs.unc.edu/ ccwu/siftgpu, 2007

[8] Changchang Wu, Sameer Agarwal, Brian Curless, and Steven M. Seitz, ”Multicore Bundle Adjustment”, CVPR 2011

[9] Changchang Wu, ”VisualSFM: A Visual Structure from Motion System”, http://homes.cs.washington.edu/ ccwu/vsfm/, 2011

[10] Changchang Wu, ”Towards Linear-time Incremental Structure From Motion”, 3DV 2013.

[11] Changchang Wu, Sameer Agarwal, Brian Curless, Steven M. Seitz, ”Schematic Surface Reconstruction”, CVPR 2012

[12] Changchang Wu, Jan-Michael Frahm, Marc Pollefeys, ”Detecting Large Repetitive Structures with Salient Boundaries”, ECCV 2010

[13] Changchang Wu, Jan-Michael Frahm, Marc Pollefeys, ”Repetition-based Dense Single-View Reconstruction”, CVPR 2011

[14] An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision
Yuri Boykov and Vladimir Kolmogorov
In IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)
September 2004