Voxel-based Fast Surface Propagation Method of Non-isolated and Sharp Featured Surface Outlier Removal

Student: Yu Chen
Advisor: Prof. Jie Shen

Abstract

In this paper, we propose a new algorithm of removing outlier clusters. It is a voxel-based surface propagation method and can handle non-isolated and sharp featured surface outlier clusters in a fast way. Numerical experiments indicate the effectiveness of the algorithm in terms of accuracy and time efficiency.
Key word: outlier removal, surface propagation

1. Introduction

Outlier removal is an intrinsic problem in modern society. Related techniques have been widely used in a variety of fields: reverse engineering, rapid prototyping, biomedicine, architecture, entertainment industry and etc.

Many studies have been conducted to remove outliers effectively and efficiently. However, most of these methods perform well only in specific situations. Some methods [1, 2] are more suited to handling isolated outliers and some [3] are more suited to handling surface without sharp features. Shen, et al. [4] proposed a surface propagation method combined with minimal variance and normalized histogram which performs well in both non-isolated outlier cluster and sharp edges. But Shen's method encounters an efficiency issue when dealing with large-scale data models.

According to our analysis, non-isolated outlier clusters and sharp featured outlier clusters are two types of the most difficult outlier clusters. As for non-isolated outlier clusters, outlier clusters are so close to main surface that distance-based criteria are not effective to remove outliers. As for sharp featured outlier clusters, it is difficult to preserve these sharp features in a data model since geometric non-smoothness at these features invalidates many analysis arsenals in calculus and differential geometry. In some cases, e.g., in laser scan data models, we might have to handle non-isolated surface with sharp features which becomes the most difficult type of models.

The main objective of this paper is to propose a fast algorithm to remove different especially the most difficult types of outlier cluster in an accurate and efficient way. The rest of the paper is arranged in this manner. We will first discuss some related work in section 2. Then we plan to focus on our new proposed method in section 3. In section 4, numerical experiment will be designed. In section 5, we will discuss the results of the numerical experiment. And in section 6, we will give our

conclusions and future work.

2. Related Works

2.1 Eigen Value Method

Eigen value method is a numerical method based on statistical analysis. We partition data points into voxels in 3-D space. Then we calculate eigen values for each voxel and use the smallest eigen value as the indicator of outliers, as shown in Fig. 1 (a). The larger the indicator is, the more likely the data points in the voxel are outliers. According to [5], we calculate eigen values for each voxel in this way. For each voxel, we first calculate a covariance matrix and then calculate eigen values of this matrix. After getting the indicators, we bin these indicators from the smallest one to the largest one. Data points in voxels whose bin number are smaller than the threshold are treated as true data and the rest of data points are treated as outliers.

The idea of using eigen values to indicate outliers is widely used. We incorporate this idea into our algorithm and treat this method as a baseline in numerical experiments discussed in section 4.

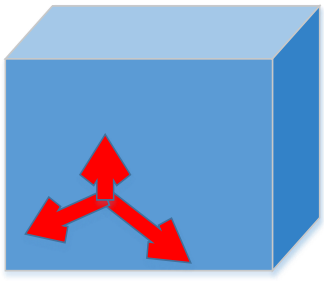
2.2 Connectivity Method

The connectivity approach observes both the density of voxels and the connection relation between voxels. Data points in voxels which are marked as true data are treated as true data and the rest of data points are treated as outliers. The voxels which are marked as true data should satisfy at least one of the following rules (as shown in Fig. 1 (b) and (c)):

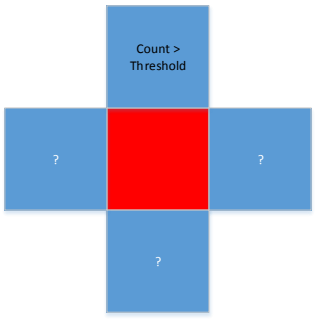
(a) The number of data points in the voxel is larger than the specific threshold.

(b) The voxel has at least one neighboring voxel who satisfies rule (a).

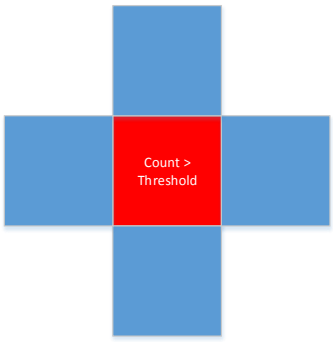
We incorporate this idea into our algorithm and treat this method as a baseline in numerical experiments discussed in section 4.



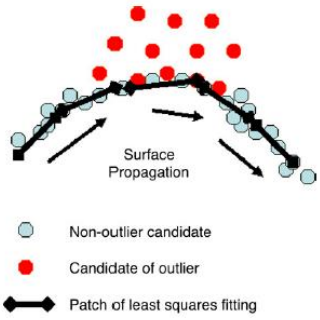
(a) Eigen value method



(b) Connectivity method rule 1



(c) Connectivity method rule 2



(d) Surface propagation method (copy from [4])

Fig. 1. Sketch maps of eigen value method, connectivity method and surface propagation method.

2.3 Surface Propagation

Jie Shen, et al. proposed a unique automatic surface propagation method [4] which performs well even in sharp edges and non-isolated surface clusters. Fig. 1 (d) shows the basic concept of surface propagation. However, it uses kd-tree to support the surface propagation, which affects the efficiency of the method and becomes a noticeable issue for large-scale data models.

To overcome this issue, we propose a voxel-based fast surface propagation method. The details of our method and main differences between Shen's surface propagation and our method will be discussed in section 3. And comparative tests will be conducted in section 4.

3. Technical Approach

In this section, we will talk about the Voxel-Based Fast Surface Propagation (VBFSP) Method in detail. The One unique feature of our propagation is that it is based on voxel, which will be discussed in detail in section 3.1. The whole propagation starts from a selective initial voxel which we will discuss in detail in section 3.3. The driving forces for propagation will be discussed in detail in section 3.2. Finally, we will discuss how to build the whole model in section 3.4.

3.1 Voxel-Based Surface Propagation

We use a voxel as the minimal unit of analysis and display. Specifically, all the numerical analyses are aimed at each single voxel, and each single data point in a voxel is treated as outlier or not by our algorithm at the same time. We adopt this approach for three reasons:

(a) Principle of locality: as for surface outlier removal tasks, data points in a small voxel are highly likely to have similar properties.

(b) A reasonable partition of voxels provide us an effective way to analysis data clusters statistically and locally.

(c) It is cheap to partition data clusters into voxels and work on them.

The propagation is based on voxel and starts from a selective initial voxel. Then we look around the neighboring ring of the current voxel, select the voxels we think are inside the main surface and mark them as true data according to driving force for propagation. Other voxels in the ring are marked as outliers temporary. For each voxel we think are inside the main surface, we look around and do the same thing. The propagation terminates until we have no new voxel to see. After propagation, we treat each data point in the voxels which are marked as outliers or untouched (initial status) as outlier and each data point in the voxels which are marked as true data as true.

Notice that because some voxels will be looked around several times, there are different tricks to take in the propagation. Here, we finally mark the voxel as true data as long as it has been treated as true data for at least one time. But as for these voxels which are treated as true data after being treated as outliers, they are not included into next round of propagation. Tricks may vary depending on specific cases.

3.2 Driving Forces for Surface Propagation

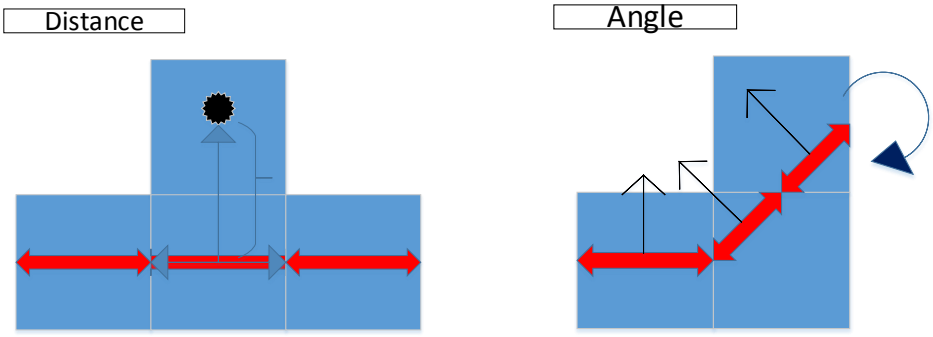
The driving force is crucial for surface propagation. We use it to determine whether a voxel should be included into the main surface. The key principles to select driving force for surface propagation include:

(a) The driving force should be direct or indirect measurements or statistics.
(b) The driving force should imply the levels of intimacy between voxels.
(c) The driving force should imply the difference between true data and outliers.

We define two kinds of driving forces in our method: distance and angle. As for distance, as we can see in Fig. 2 (a), it indicates the Euclidean distance between the centroid of data points in the neighboring voxel and the plain determined by the two larger eigen vectors in the current voxel. It is apriori assumed that the smaller the distance is, the larger the probability that the neighboring voxel is actually inside the main surface is. In other words, in most cases, outliers have larger distances than true data do.

As for angle, as we can see in Fig. 2 (b), it indicates the included angle between the two plains each determined by the two larger eigen vectors in the current and neighboring voxel, respectively. It is apriori assumed that the smaller the angle is, the larger the probability that the neighboring voxel is actually inside the main surface is. In other words, in most cases, outliers have larger angles than true data do.

Notice that each single driving force mentioned above is not effective enough to handle all cases. For example, in non-isolated surface outliers removal scenario which is regarded as the most difficult case, outliers are often quite close to the main surface, so the distance approach is likely to fail. Similarly, in some scenarios, the angle approach is not effective enough to tell true data and outliers. The most straightforward way is to combine these two criterions so that we can handle most different cases.



(a) Distance for VBFSP method (b) Angle for VBFSP method

Fig 2. Sketch maps of VBFSP method

3.3 Initial Voxel of Surface Propagation

The initial voxel is another crucial factor for surface propagation. If we select an incorrect initial voxel, the propagation cannot help us to remove outliers. The key principle to select initial voxel of surface propagation is to find the voxel which is most likely to be inside the main surface. Here, we adapt connectivity flags and histogram bins of eigen values to determine the hlevel variable for each voxel which indicates the probability that a voxel is inside the main surface. Then we select the voxel with the highest probability as an initial voxel.

We have tried to select initial voxel manually, and found that a good initial voxel means a lot for the final result. It indicates that how to select a good initial voxel for surface propagation is really a good topic which deserves a further study in future work.

3.4 Building the Model

Model Graph:

Fig. 3 model graph of VBFSP method

The algorithm of VBFSP method is shown as follows:
Algorithm 1:
Name: Voxel-Based Fast Surface Propagation
Input: data_points
Output: true data & outliers

1. *VoxelSet = GetVoxelSet(data_points);*

```
2.  InitialVoxel = GetInitialVoxel(VoxelSet);
3.  Add InitialVoxel in MainSurface;
4.  Loop:
5.      foreach current_voxel in MainSurface:
6.          if current_voxel is marked as untouched or unprocessed:
7.              Mark current_voxel as true_data;
8.          Foreach neighboring_voxel in GetNeighboringVoxels(current_voxel):
9.              if DrivingFroceJudgement(neighboring_voxel, current_voxel) == true:
10.                 if neighboring_voxel is marked as untouched:
11.                     Add neighboring_voxel in FrontWave;
12.                     Mark neighboring_voxel as unprocessed;
13.                 else if neighboring_voxel is marked as unprocessed or true_data:
14.                     donothing;
15.                 else:
16.                     Mark neighboring_voxel as true_data; // It varies on specific cases
17.             else:
18.                 if neighrboring_voxel is marked as untouched:
19.                     Mark neighboring_voxel as outlier;
20.     if number of voxel in FrontWave is zero:
21.         Jump out of loop;
22.     Copy FrontWave to MainSurface;
23. All data points in voxel marked as true_data are true data;
24. All data points in voxel marked as outliers or untouched are outliers;
```

Algorithm 2:
Name: GetInitialVoxel
Input: Voxel Set
Output: initial voxel

```
1.  foreach voxel in Voxel Set:
2.      if GetConnectivityFlag(voxel) is true:
3.          error = CalcSmallestEgienValue(voxel);
4.          hlevel = Bin(error);
5.      else:
6.          hlevel = MaxBinNum – 1;
7.      Select a single voxel with smallest hlevel as initial voxel;
```

Algorithm 3:
Name: DrivingFroceJudgement
Input: neighboring_voxel & current_voxel
Output: true or false

```
1.  Calculate eigen vectors for the two voxels;
2.  Calculate the two plains for the two voxel;
```

3. *Calculate the distance between the centroid of data points in neighboring_voxel and plain in current_voxel;*
4. *Calculate the included angle between the two plains;*
5. *DrivingForce = ratio*Normalized(distance) + (1-ratio)*Normalized(angle);*
6. *if DrivingForce < Threshold:*
7. *return true;*
8. *else:*
9. *return false;*

3.5 Unique Features

Some unique features of our VBFSP method compared with Shen's surface propagation method include:

(a) The VBFSP method is based on voxels instead of kd-tree, which makes it less expensive and more efficient than Shen's method.

(b) The VBFSP method combines not only the distance factor but also the angle factor, which makes it competent to different types of outlier clusters.

(c) The VBFSP method starts from one initial voxel and runs one round of propagation while Shen's surface propagation method starts from a number of initial points and runs several rounds of propagation. The experiment results show that as long as the driving force for surface propagation is effective and the initial voxel is good, one round of propagation is enough.

In summary, the VBFSP method simplifies the complexity of surface propagation while keeping a high performance of removing outliers which are difficult to deal with.

4. Numerical Experiment

4.1 Test Cases

We compared eigen values method, connectivity method, Shen's surface propagation method and our VBFSP method on two data models. Both data model 1 (Fig. 4 (a)) and data model 2 (Fig. 4 (b)) have sharp edges and non-isolated surface outlier clusters, which makes them extremely difficult to handle. In fact, data model 2 is more difficult than data model 1 to deal with because data model 2 has the non-isolated surface outlier clusters in the sharp edge area while data model 1 does not combine the two difficulties. Results and discussions about the experiment will be shown in section 5.

(a) Data model 1 (b) Data model 2

Fig 4. Data models which have non-isolated and sharp featured surface outlier clusters

4.2 Parameter Setting

The setting of parameters varies on different models. We adjusted the setting according to the performance and display the setting we exploited in our experiment as follows. Notice that among all the parameters, m influences both the efficiencies and performances of algorithms, and the rest of parameters only influence performances.

Table 1

Parameter setting of eigen value method

Method\parameters	m (#voxel in each axis)	HISTOGNUM (#histograms)	Threshold of Histogram
Eigen value method	100 (50 for data model 2)	30	15

Table 2

Parameter setting of connectivity method

Method\parameters	m (#voxel in each axis)	HISTOGNUM (#histograms)	Threshold of Histogram	Count threshold of connectivity
Connectivity method	100 (50 for data model 2)	30	15	2

Table 3

Parameter setting of VBFSP method

Method\parameters	m (#voxel in each axis)	HISTOGNUM (#histograms)	Threshold of Histogram	Count threshold of connectivity
VBFSP method	100 (50 for data model 2)	30	15	2

Method\parameters	Dist_angl_ratio	Scale	Angle_t	Dist_angl_Threshold
VBFSP method (cont'd)	0.6 (0.68 for data model 2)	0	0	0.6 (0.355 for data model 2)

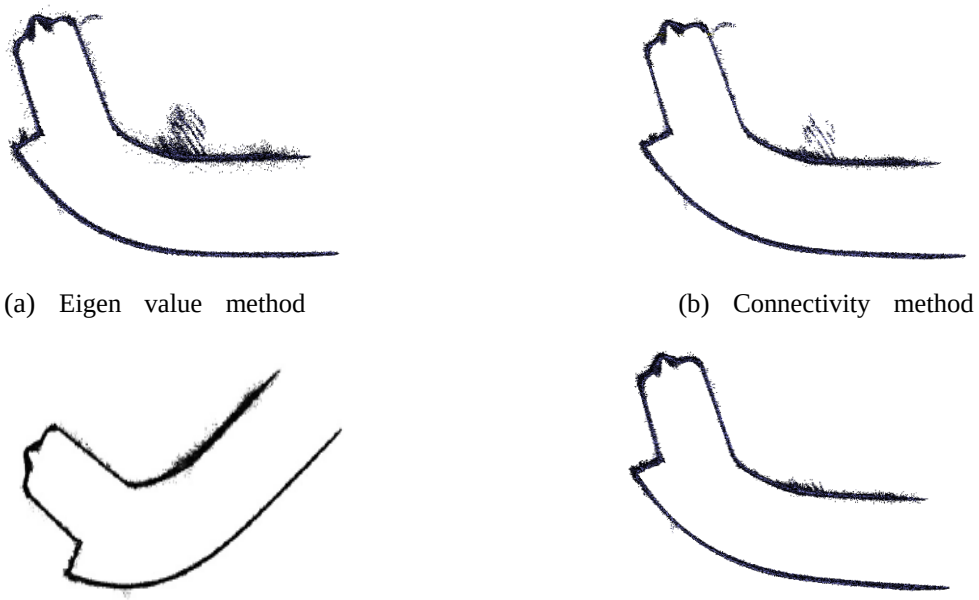
4.3 Computer Used

The numerical experiment conducted in this paper was implemented in Visual Studio and tested on a Dell PC with 2.2 GHz Intel CPUs and 8 GB internal storage. To show the efficiencies of all the tested algorithms intuitively, no parallel computing techniques were exploited.

4.4 Evaluation Criteria

We use execution time as the evaluation criterion of time efficiency. To evaluate the performance of algorithms, we compare the raw data models and the processed data models after outlier removal.

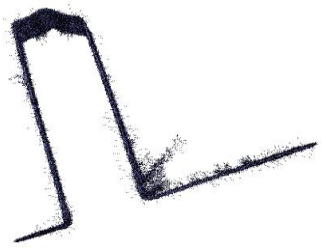
5. Results & Discussions



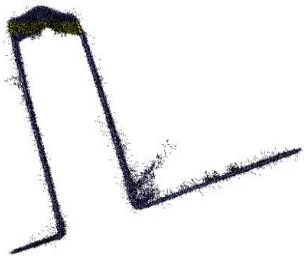
(a) Eigen value method (b) Connectivity method

(c) Shen's method (d) VBFSP method

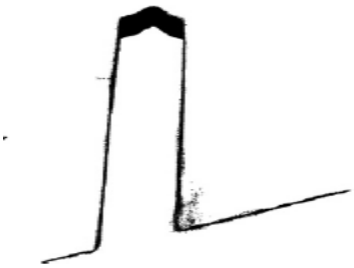
Fig 5. Data model 1 (Fig. 4 (a)) processed by eigen value method, connectivity method, Shen's method and our VBFSP method.



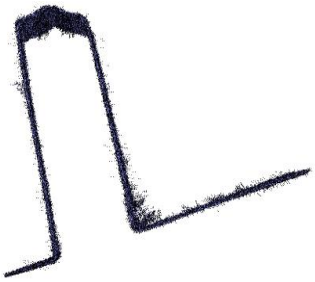
(a) Eigen value method



(b) Connectivity method



(c) Shen's method



(d) VBFSP method

Fig 6. Data model 2 (Fig. 4 (b)) processed by eigen value method, connectivity method, Shen's method and our VBFSP method.

Table 4

Execution time and number of removed outliers in data model 1 for three methods.

Method	# of reserved points	# of removed outliers	Execution time (milisec)
Eigen Value	187652	6450	7982
Connectivity	183626	10476	8320
Shen's method	N/A	N/A	N/A
VBFSP method	187809	6293	9359

Execution time and number of removed outliers in data model 2 for three methods.

Method	# of reserved points	# of removed outliers	Execution time (milisec)
Eigen Value	89911	617	4681
Connectivity	89642	886	3422
Shen's method	N/A	N/A	N/A
VBFSP method	86176	4352	3911

Notice: Time Efficiency of Shen's method for these two data models are not available yet. But the execution time of Shen's method should be much larger than that of VBFSP method.

Fig. 5 is a special case in which there are some clustered noise data inside a space bounded by a concave surface. The outlier clusters are so close to the main surface that many existing methods

(e.g., eigen value method and connectivity method) fail to remove outliers, as shown in Fig. 5 (a) and (b). Shen's method and our VBFSP method perform well in this case since they have a stronger ability of handling non-isolated outlier clusters as we mentioned in section 2 and 3.

Fig. 6 is another typical case where non-isolated outlier clusters exist around a sharp corner. This case is more difficult than the above one since it combines non-isolated outlier clusters and sharp featured surface. Similarly to what happens in the above case, Shen's method and our VBFSP method perform much better than eigen value method and connectivity method.

Notice that the performance of our VBFSP method is very close to that of Shen's method, but our VBFSP method is much more efficient than Shen's method. Since our propagation is voxel-based, its efficiency is close to eigen value method and connectivity method, as shown in Table 4. However, Shen's method is based on kd-tree which is much more time-consuming.

6. Conclusion & Future Work

In summary, our algorithm is compared favorably to existing methods since it can remove outliers even in the most difficult cases and it is not so time consuming compared with Shen's method which can also remove difficult outliers. The unique contribution of this paper is to propose a voxel-based surface propagation approach and a liner combination of distance and angle as the driving force of propagation which can deal with complex types of outlier clusters.

In the future work, we plan to focus on:

(a) Looking for more effective ways to select initial voxel of surface propagation. Notice that initial voxel is crucial for propagation and we are still not able to find the most suitable initial voxel automatically, we should do more work on that in the future work.
(b) Looking for other effective driving forces of surface propagation. Notice that driving force is the power of propagation and determines the effects of propagation, finding more effective driving forces can help us remove more complex types of outlier clusters.

7. References

[1] Fleishman S, Drori I, Cohen-Or D. Bilateral mesh denoising. In: Proceedings of the 30th annual conference on computer graphics and interactive techniques. 2003. p. 950_3.

[2] Xie H, McDonnell KT, Qin H. Surface reconstruction of noisy and defective data sets. IEEE Visualization 2004 2004;259_66.

[3] Schall O, Belyaev A, Seidel H. Robust filtering of noisy scattered point data. In: Eurographics symposium on point-based graphics. 2005. Pauly, M. and Zwicker, M.

[4] Jie Shen, David Yoon, et al. Spectral moving removal of non-isolated surface outlier clusters. Computer-Aided Design. 2008.09.

[5] William H. Press, Saul Teukolsky, et al. Numerical Recipes in C. Cambridge University Press. 1992.

8. Appendix

As for VBFSP method, since we use the results of eigen value method and connectivity method as the criterion of selecting an initial voxel of surface propagation, so the parts of the original code related to the above two are contributing to the final result. However, these parts are optional for our method, we can replace them with anything which can help select a good initial voxel. It means that we can regard them as some tools in our toolkit for selecting an initial voxel. Of course, the part of the original code related to the propagation procedure is the core part.