Acta Technica *62* No. 3B/2017, 141–148

Fast K-nearest neighbors searching algorithms for point clouds data of 3D scanning system[1]

Zhang Fan[2,3], Tan Yuegang[2,4], Dai Cong[2,5]

Abstract. The quality of 3D model reverse engineering is affected directly by the processing efficiency of 3D scanning point cloud data. In view of the inefficiency of traditional K-nearest neighbors search algorithm, the enhanced adaptive spatial sphere search algorithm is proposed. The small grid side length is determined by the correlation of point cloud density and the small grid division method is also improved. By using the scaling factor between K value and the point number in the spatial sphere, the expansion radius of spatial sphere is determined. The adaptive spatial sphere expansion method is implemented to obtain the appropriate K domain. The efficiency of the improved algorithm is analyzed by different grid division, different point number in grids and different K values.

Key words. Point clouds data, K-nearest neighbor, small grid, spatial sphere..

1. Introduction

With the rapid development of modern science and technology, reverse engineering plays a more and more role in the manufacturing industry. It has been widely used in the fields of cultural relic's protection, biological medical technology, mold design and manufacture [1]. However, due to the high measurement accuracy, the acquired point clouds data is far more than the actual need for surface reconstruction. The redundant data will consume a lot of storage space, increase the computation amount and decrease the reconstruction precision.

[1]This work was financially supported by the China's Fundamental Research Funds for the Central Universities (WUT: 2017II26GX and 2017IVB087) and the China's Post-doctoral Science Fund (2016M602380).

[2]School of Mechanical and Electrical Engineering, Wuhan University of Technology, Wuhan, 430070,China

[3]Email: fan_dx0202@163.com

[4]Email: ygtan@whut.edu.cn

[5]Email: 876777946@qq.com

Due to the lack of topological geometry information, the feature information, such as curvature, normal vector et al. isall needed. So, the K-nearest neighbors' points should be obtained. The purpose of K-nearest neighbor search is to obtain the K-th nearest point of Euclidean distance which can provide fast neighborhood point search for outlier detection, point cloud simplification and surface reconstruction [2].For general data point sets, the K domain points are obtained by calculating the distance between all points in the point set. The distances are sorted from small to large to get the points corresponding to the first K minimum distances. However, due to the large amount of 3D point cloud data, it will cost lots of time and resources in this way, which will affect the efficiency of the cloud data processing seriously. To solve this problem, the improved K-nearest neighbor search algorithm is proposed.

In literature [3], the K-nearest neighbor points are searched by the dynamic mesh. All points are sorted in the X, Y and Z direction and the index in positive and negative direction of each axis are obtained to construct the point set. But the step length of this algorithm is sensitive. Based on the space partition strategy, the point cloud data space is divided into small girds. Then, the k-th points are searched in the grids. But if there are not enough points, it needs to be expanded to 27 small grids [4]. The grid division method is improved by some scholars. The k-th points are searched in the spatial sphere (space ball grid) [5]. The number of required small grid is reduced, but the expansion of space radius is not ideal and the stability of the algorithm is not high. Combination the advantages of dynamic mesh and small grid, after the division of the small grid, the dynamic mesh is used to get the K nearest points in the small grid. The time complexity is reduced, but the search efficiency is easily affected by the size of small grid [6]. Therefore, on the basis of the small grid division method, an adaptive spatial sphere K nearest neighbors search algorithm is proposed [5]. According to the point cloud data density, the size of the initial spatial sphere is determined. The points inside the cube of the space ball are counted as the candidate points for K nearest neighbor search [7]. But the adaptability is low as the fixed expansion way of the space ball. These methods are applied in the clustering analysis and achieved good results, but the parameter K is easy to be affected by the noise signal.

Due to the huge amount of the point cloud data, it is a very time-consuming processing to obtain the K nearest neighbors for each point. So, it is necessary to adopt a new method to improve the search efficiency. In this paper, the improved K nearest neighbor search algorithm based on the adaptive spatial sphere is studied.

2. Point cloud data processing flow

The working flow of the 3D scanning system includes two parts: calibration and scanning. In the calibration process, based on the feature points of the world coordinates and image coordinates in the checkerboard, the internal and external matrix parameters of the camera is calculated. In the scanning step, based on the calibration parameters and the 2D image coordinates, the point cloud data is obtained from the laser profile of the scanning mode surface. The rotating platform is actuated by the motor from 0 to 360 degrees, and the extracted data is combined to form the 3D point cloud model. The data processing flow is shown in Fig. 1.

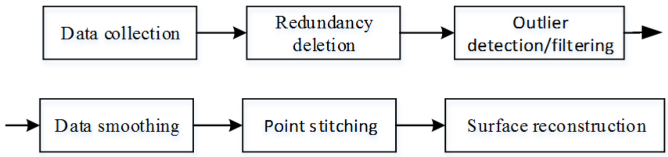

Fig. 1. 3D scanning data processing flow

3. Fast K-nearest neighbors search algorithm

3.1. Point cloud data grid division

The total number of the point cloud data in the laser scanning system is N, the coordinate interval of the X, Y, Z axis are: $[x_{\text{min}}, x_{\text{max}}]$, $[y_{\text{min}}, y_{\text{max}}]$ and $[z_{\text{min}}, z_{\text{max}}]$. Then, the length, width and height of the smallest cuboid bounding box of the point cloud data is shown in formula (1). If the point cloud is evenly distributed, the side of the small grid for each point in the cube is in formula (2)

$$\begin{cases} x_{\text{width}} = x_{\text{max}} - x_{\text{min}} \\ y_{\text{width}} = y_{\text{max}} - y_{\text{min}} \\ z_{\text{width}} = z_{\text{max}} - z_{\text{min}} \end{cases}, \tag{1}$$

$$L_{\text{orginal}} = \sqrt[3]{\frac{x_{\text{width}} \times y_{\text{width}} \times z_{\text{width}}}{N}}. \tag{2}$$

All point data are assigned to the corresponding small grid according to the coordinates, the number of small grids is N_{p}. The actual density of the point cloud is

$$\rho = \frac{N}{N_{\text{p}} * L^3_{\text{oginal}}}. \tag{3}$$

The point number in the small grid is k_{n}, the length of the cube surrounding is

$$L_{\text{final}} = \sqrt[3]{\frac{k_{\text{n}}}{\rho}}. \tag{4}$$

For the point (x, y, z), the indexes of the axis x, y, z in the small grids are

$$\begin{cases} i = floor((x - x_{\text{min}})/L_{\text{final}}) \\ j = floor((y - y_{\text{min}})/L_{\text{final}}) \\ k = floor((z - z_{\text{min}})/L_{\text{final}}) \end{cases}. \tag{5}$$

3.2. Implementation of the K-nearest neighbor algorithm

3.2.1. Initial radius of space ball After the small grid division, the serial number is determined by calculating the index of x, y, z directions in the corresponding small grid. Then, the ordinal number of all the points in the small grid is obtained.

$$r = 0.5 * L_{\text{final}} * \sqrt[3]{\frac{k}{N_{\text{c}}}}\,. \tag{6}$$

3.2.2. Candidate points of the K-nearest neighbor Taking the search point as the point center, the initial range of dynamic ball is determined by radius and the center. The points in the ball are the candidate points.

3.2.3. K-nearest neighbor search

1. Calculate the initial radius of the space ball and obtain the index of all cube small grids covered by the spatial sphere.
2. Calculate the distance between the point to be searched and every point in the covered and not-marked small grid. Then, the small grid will be marked.
3. The point number in the covered small grid is N_{w}. If $N_{\text{w}} < k$, the radius of the space ball should be increased a, so the new radius is: $r \cdot a$. Then, continue the steps 1-3 until $N_{\text{w}} > k$. The threshold b is used to avoid the infinite cycle when $a - 1 \approx 0$. If $a < b$, then $a = b$. If $N_{\text{w}} > k$, continue the next step.
4. Traversal the distances between the point to be researched and the N_{w} candidate points. Search the points in the space ball; if the distance is less than r, the number of the points in the ball is N_{s}. If $N_{\text{s}} < k$, the updated radius is $r \cdot \alpha$. Then, loop performs step 1 to 4 until $N_{\text{s}} > k$.
5. When the point number N_{s} inside the space ball are greater than k, the N_{s} points are ordered from small to large. The front k points are the K-nearest neighbor points. Also, the marked small grid will be removed.

The K-nearest neighbor search of the point cloud data can be completed by traversing all the points processed by step 1–5.

4. Experiment and analysis

The experimental platform is: Core i5 CPU, 3.20 GHz, 4 GB Memory, OS: 64 bit Windows 7. The point cloud model is shown in Fig.2.

Experiment 1: By this algorithm, not only the candidate point selection and the space ball expansion method are improved, but also the small grid division way is improved. The differences between the traditional small grid division method and the improved algorithm are compared. Using the model 10 % re-sampling in Fig. 2(c) and the neighbor value $K = 10$, the experimental results are shown in Table 1. The

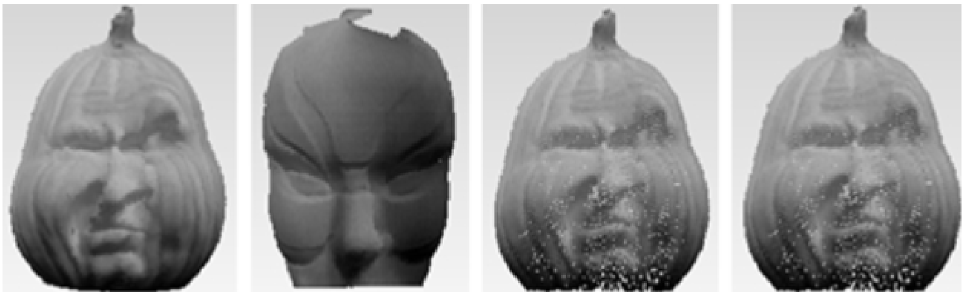

Fig. 2. Point cloud model: (a) $N = 231623$, (b) $N = 110358$, (c) Resample 10 % $N = 23163$, (d) Resample 20 % $N = 46325$

Table 1. Algorithm efficiency comparison between different grid divisions

Grid division method	Point number in the small grid	Side length of the grid	K-nearest neighbor search time (ms)
Improved algorithm	0.5	1.378	1.581
	1	1.736	1.628
	2	2.187	1.761
	4	2.756	1.939
	8	3.472	2.105
	16	4.376	2.176
Traditional algorithm	0.5	2.716	1.915
	1	3.422	2.090
	2	4.312	2.138
	4	5.432	2.240
	8	6.844	2.708
	16	8.623	3.634

small grid side length is different with the equal number of points in the grid. So, the side length of the small cube grid is affected by different division methods. The K-nearest neighbor time is close to each as long as the side length is similar.

Experiment 2: The K-nearest neighbor search efficiency is compared in different K values, different point numbers in the small grids and different models. The testing models are shown in Fig. 2(a)-(d). In order to avoid the influence of different small grid division methods, the same improved grid division algorithm is used. The K-nearest search time of each point is shown in Table 2. From the table, the stability of the proposed algorithm is better than the traditional space ball K-nearest algorithm in the number of K_{n}. The time consuming amplification for searching the K-nearest points of the improved algorithm is smaller than the traditional method.

Table 2. Algorithm efficiency with different k_n, K and model

Alg.	Model	k	Point number in the small grid k_n						
			1	3	5	7	9	11	13
Improved algorithm	3.2(a) N=231623	10	0.490	0.478	0.491	0.516	0.540	0.558	0.576
		20	0.648	0.606	0.628	0.667	0.685	0.704	0.729
		50	1.289	1.093	1.084	1.118	1.127	1.158	1.181
	3.2(b) N=110358	10	0.424	0.401	0.413	0.421	0.433	0.450	0.451
		20	0.651	0.578	0.580	0.596	0.606	0.618	0.629
		50	1.188	0.990	0.988	0.978	0.989	1.036	1.012
	3.2(c) N=23163	10	0.432	0.396	0.413	0.422	0.445	0.496	0.472
		20	0.663	0.591	0.594	0.604	0.620	0.641	0.658
		50	1.276	1.033	1.028	1.016	1.039	1.045	1.053
	3.2(d) N=46325	10	0.440	0.423	0.436	0.454	0.462	0.485	0.491
		20	0.666	0.589	0.601	0.605	0.627	0.638	0.652
		50	1.249	1.032	1.042	1.018	1.046	1.032	1.071
Traditional algorithm	3.2(a) N=231623	10	1.881	2.163	2.352	2.387	2.504	2.598	2.665
		20	2.654	3.103	3.331	3.449	3.552	3.638	3.731
		50	6.762	6.384	6.978	7.319	7.796	7.933	8.107
	3.2(b) N=110358	10	1.595	1.791	1.847	1.778	1.796	1.831	1.852
		20	2.689	3.078	3.291	3.470	3.527	3.586	3.610
		50	6.064	6.165	6.646	6.980	7.412	7.691	7.820
	3.2(c) N=23163	10	1.521	1.751	1.920	1.951	2.048	2.056	2.068
		20	2.505	2.866	3.078	3.221	3.403	3.531	3.586
		50	6.177	5.728	6.122	6.522	6.771	7.058	7.258
	3.2(d) N=46325	10	1.544	1.831	1.923	1.997	2.063	2.075	2.127
		20	2.593	2.962	3.214	3.414	3.568	3.642	3.738
		50	6.171	5.903	6.262	6.743	7.008	7.221	7.398

At the same K value and the same point number in the grid, the time consuming is different with different model or different sampling rates. But the difference is small between different models. In the same model, the same K value and the same point number in the small grid, the improved algorithm is better than the traditional space ball K-nearest neighbor method in the searching time and efficiency.

5. Summary

Aiming at the efficiency of 3D scanning point cloud data processing, though the experimental test of the improved K-nearest neighbor algorithm, the conclusions are:

1. The density of the point cloud data is larger than that of the original grid division. So, if the point number is equal, the side length of the small grid is smaller than the traditional algorithm. Under the smaller side length, the average K-nearest neighbor search time will be reduced greatly.

2. The points inside the small grid of cubes covered by a space ball are used as the candidate data. The calculation and sorting of redundant points are reduced. The fixed radius increasing ratio method is improved. The expansion ratio of the space ball is determined by the ratio of the value K and the point number in the space ball. The number of expansion and the processing workload are reduced. So, the efficiency of K-nearest neighbor algorithm is improved.

References

[1] J. E. Rangel Diaz, J. A. Ambrosio Prieto, L. M. Patiño Garcia: *3D digitalization: Reverse engineering-towards the industrial development.* IEEE International Congress of Engineering Mechatronics and Automation (CIIMA), 22–24 October 2014, Cartagena, Colombia, IEEE Conference Publications (2014), 1–5.

[2] G. Zhu, C. Ma, X. Feng: *The K-nearest neighbor fast searching algorithm of scattered data.* IEEE International Conference on Future Information Technology and Management Engineering, 9–10 October 2010, Changzhou, China, IEEE Conference Publications *1* (2010), 125–128.

[3] X. Yuan, L. Guo, J. Wang, X. Zhao: *Efficient K-nearest neighbors searching algorithms for unorganized cloud points.* IEEE World Congress on Intelligent Control and Automation, 25–27 June 2008, Chongqing, China, IEEE Conference Publications (2008), 8506–8510.

[4] M. Connor, P. Kumar: *Fast construction of K-nearest neighbor graphs for point clouds.* IEEE Transactions on Visualization & Computer Graphics *16* (2010), No. 4, 599–608.

[5] A. G. Jivani: *The novel K-nearest neighbor algorithm.* IEEE International Conference on Computer Communication and Informatics, 4–6 Januar 2013, Coimbatore, India, IEEE Conference Publications (2013), 1–4.

[6] Y. Park, H. Hwang, S. G. Lee: *A fast K-nearest neighbor search using query-specific signature selection.* ACM International Conference on Information and Knowledge Management, 18–23 October 2015, Melbourne, Australia, ACM Proceeding NY, USA (2015), 1883–1886.

[7] B. Q. Shi, J. Liang, Q. Liu: *Adaptive simplification of point cloud using K-means clustering.* Computer-Aided Design *43* (2011), No. 8, 910–922.

Received September 12, 2017