A Parallel Fast Sort Algorithm for Mass 3D Point Clouds of Irregular Model

Liu Yan-ju¹, Zhang Hong-lie², Tao Bai-rui¹ and Li Cheng²

¹Computer Center, Qiqihar University, Qiqihar, China
²College of Computer and Control Engineering, Qiqihar University, Qiqihar, China

*15146692464@163.com

Abstract

According to mass point clouds without explicit topology relation, a parallel fast sort algorithm is proposed in this paper. Morton order is introduced and used to merge one-dimensional data. The mass point clouds of irregular model are generated corresponding address code named Morton code and these points are stored in the octree structure chain. And then a parallel fast sort algorithm based on Euclidean distance is used to sort by CPU and GPU. The k-Nearest Neighbors of point can be located in the chain. The experiment results show that much time is saved and k-Nearest Neighbors of point can be searched directly. This algorithm is simpler than those complex sort methods used on the whole point clouds.

Keywords: mass point clouds, irregular model, fast sort, k-Nearest Neighbors

1. Introduction

With the development of measure device quickly, the point data of irregular model can be obtained more efficient and precise. However, these point clouds have not explicit topology relation each other [1] and any point must be searched in the global scope of the point cloud set [2]. This is the main reason causing speed slow of 3D geometric modeling [3] that point data is measured or traversed in a few megabytes, tens of megabytes of disorder set of point clouds. Therefore, to build the topology relation of point clouds and decrease the searching scope is the key to improve geometrical construction [4] speed for scattered point cloud of the irregular model. In reverse engineering, k-Nearest Neighbors [5] is the base of data simplified [6], normal estimation [7-8], and surface reconstruction. The data structure of k-Nearest Neighbors can reflect the shape information of the point, such as curvature and normal vector. At the same time, the efficiency of k-Nearest Neighbors effects the surface reconstruction directly.

In this paper, Morton order theory [9] is introduced. The mass point clouds of irregular model are generated corresponding address code named Morton code and these points are stored in the octree structure [10]. And then a parallel fast sort algorithm based on Euclidean distance is used to sort and the Morton code of ordered point is stored in one dimensional chain. The k-Nearest Neighbors of point can be located in the chain.
2. Storage Structure Based on Morton Order

In the mathematical analysis and computer science, Morton order is named Z-order that data is converted in multidimensional space to a one-dimensional one and local features of the point cloud data are protected at the same time, looked in figure 1. In three-dimensional space, Morton value of the specified point can be simplified by interleaving to binary representation of coordinate value. Any one-dimensional data structure such as binary tree, B tree, hash table et al. can be adopted to store.

![Figure. 1 Morton Order Curve Iterations Extended to Three Dimensions](image)

2.1. Two-Dimensional Coordinate is Merged to Morton Value

In order to simplified the problem, the two-dimensional point \((x, y)\) is merged to one-dimensional Morton value. In figure 2, set coordinate axis \(0 \leq x \leq 7\), \(0 \leq y \leq 7\), respectively. The values (Decimal and binary form) in x-axis and y-axis are given, and the results of the ranks of interleaving to binary representation are the Morton value. A plane curve will be the Morton value obtained recursively according to numerical order of size connected formed corresponding Morton sequences. In this way, arbitrary dimensional data can be merged into this sequence and the three-dimensional point cloud data into a space curve in this paper.

2.2. Morton Order Generation Algorithm

Point clouds inputted usually take corresponding integer values in each axis. It is can also use fixed-point number representation of numerical between \([0-1]\) in the computer or with the corresponding machine word size that two representations are equivalent. Any given two points can form an overlay the least square. The X and Y coordinates of each point are called xy according to a binary bit interleaving formed by the combination of numerical. The generation algorithm of Morton order is given as following based on [11] that is improved to float data.
Algorithm 1: Morton order generation

Step 1. Input two data p, q
Step 2. Initialization x=0  dim=0  j=0
Step 3. If  j<0
    y=XOR(pj, qj)
Step 4. If  x<y
    x=y
    dim=j
    j++
    return Step 4
Else
    Output  pdim<qdim
Step 5. End.

Algorithm 2: XOR(a,b)

Step 1. Input two data a, b
Step 2. x=mantissa of a
    y=mantissa of b
Step 3. If  x==y
    z=mantissa of a xor b
    z=x-z
    return x
Else
    If y<x
        return x
    else
        return y
Step 4. End.

In the algorithm1, set p, q is two data of point clouds that the data can be decimal. pj is the jth coordinate of p. When The index is equal, XOR (a,b) function is used for comparing two number mantissa coordinates and XOR. The returning values of these two functions are integer format.
2.3. Storage Structure

Point clouds of irregular model are stored in text file. The structure of Point_3D is defined as following:

```c
Struct Point_3D{
    double x;
    double y;
    double z;
    SNormal normal;
};
```

`normal` is normal vector of point. If the value of normal is absent, normal is equal to zero.

Every Morton value of data computed by the algorithm of Morton order generation is stored in Morton_3D.

```c
Struct Morton_3D{
    Long Morton;
    Point_3D *p;
};
```

This paper adopts linear octree storage structure that leaf nodes are retained only. In each leaf node, there are a point cloud coordinate position and the corresponding Morton value. The octree structure is defined as following.

```c
typedef struct {
    Point_3D p[8];
    double val[8];
} GRIDCELL;
```

```c
typedef class OctreeNode *Ocnode;
```

```c
class OctreeNode
{
    private:
        Point_3D min; /* bounds */
        Point_3D max;
        double value[8];
        int density;
        char at_max_depth;
        char not_fully_divided;
        CPtrList vex;
        CPtrList normal;
        Ocnode *children;
    public:
        OctreeNode * make_octree(Point_3D min, Point_3D max );
        void isoface(Octree* o);
        int subdivide_octree( int min_depth, int max_depth, Octree* o );
        int octree_needs_to_be_split(OctreeNode * o );
        void marchingcube(int depth, OctreeNode *o);
        … …
}
```
### 3. Parallel Fast Sort Algorithm for Irregular Model

In the application of overall performance, the large amount of data sorting phase [12] will become the bottleneck. Recursive-segmentation-sequence-sorted is used in the algorithm. The point bellowing center point data is moved the point of the left position, while one higher than the center point is moved to the right. It does not stop until the entire sequence is sorted.

In each partition, the center point is selected iteratively. Therefore, the new sequences created can be sorted independently. Then each thread block is assigned to the corresponding sub sequences, but the thread block is required to work with the sub sequence together. So the algorithm is divided into two stages. The algorithm is described as following.

**Algorithm 3: Parallel fast sort algorithm for irregular model**

- **Step 1.** Input set of point clouds
- **Step 2.** Calculate the Euclidean distance of each point to the origin
- **Step 3.** Point clouds are divided into n subsets according to the distance
- **Step 4.** If n>0
  - If CPU is free
    - Select a set from the wait queue
    - n=n-1
    - CPU is set busy
    - Data in subset is sorted by CPU
    - CPU is set free
    - return Step 4
  - else
    - if GPU is free
      - Select a set from the wait queue
      - n=n-1
      - GPU is set busy
      - Data in subset is sorted by GPU
      - GPU is set
      - CPU is set free
      - Return Step 4
- Else

### 4. Results

The purpose of the experiments is to evaluate the performance of our proposed parallel fast sort algorithm by denture model. The time complexity of this algorithm is $O(n \log(n))$ and space complexity of this one is $2n+c$ which $c$ is constant.

The number of models is showed in Table 1. The algorithm is compared by GPU sorting [13] and global radix sort algorithm [14]. In AMD Phenom II 2.99GHz processor, display card is NVIDIA 8600GTS and NVIDIA 8800GTX. The results are shown in Figure 3.

<table>
<thead>
<tr>
<th>Model</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cutting</td>
<td>3714</td>
</tr>
<tr>
<td>Canine</td>
<td>4677</td>
</tr>
</tbody>
</table>

Table. 1 Number of Model Point Clouds
Because CPU and GPU are all scheduled at the same time during GPU fast sorting, sorting speed is fast. The following is the four kinds of models is run in the NVIDIA 8800GTX and NVIDIA 8800GTX in Figure 3. And percentage of denture models sorted on the CPU and GPU by GPU-quick sort is shown in Figure 4. As we can see, the ability of GPU quick sorting is super than one of CPU. GPU has the advantage in both processing quantity and processing speed.

![Figure 3 Comparison of Sorting for Four Dentures](image)

![Figure 4 Percentage of Denture Models Sorted on the CPU and GPU by GPU-Quick sort](image)

## 5. Conclusions

The irregular model is complex object that some part is smooth and some parts are irregular including sharp features. In order to improve the speed of sorting, we present a parallel fast sort algorithm for mass 3D point clouds of irregular model. CPU and GPU are all taken part in the sorting. This method is simpler than those complex methods used on the whole point clouds directly. Further work concerns k-Nearest Neighbors travel algorithm applying to other complex model.

### Acknowledgments

This work was jointly supported by the National Natural Science Foundation of China (Grant No. 61403222), Natural Science Foundation of Heilongjiang Province (Grant No. F201332, F201439, F201440).
References


