Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 7875422, 9 pages
https://doi.org/10.1155/2022/7875422

Research Article

Point Cloud Data Hole Repair Aggregation Algorithm Based on Optimized Neural Network

Siyong Fu

ZTE School of Communication and Information Engineering, Xinyu University, Xinyu, 338024 Jiangxi, China

Correspondence should be addressed to Siyong Fu; 631406080112@mails.cqjtu.edu.cn

Received 24 February 2022; Revised 9 April 2022; Accepted 16 April 2022; Published 9 May 2022

Academic Editor: Deepak Kumar Jain

In order to solve the problem of cost cloud data and hole repair efficiency and accuracy, this article offers a study of integrated cloud network hole algorithm research based on optimal neural network. This paper first introduces a common point cloud hole-filling algorithm, provides a neural network-based point cloud blank filling algorithm, and introduces hotspot problems in a given algorithm, combined with the technology related to high-performance computing, the parallel optimization technology based on OpenMP and CUDA is adopted to accelerate the algorithm accordingly. Experiments have shown that the accuracy of pre- and postoptimization of CUDA-based algorithms varies. As the model and input point cloud data increase, the accuracy of the algorithm decreases slightly. However, when the data increases to 104 orders of magnitude, the rate of accuracy decline meets an inflection point and finally stabilizes to about 96.8% in the environment of 132457 data of monk model. The point cloud hole-filling algorithm based on the optimal neural network given in this article is highly predictable, can well repair incomplete point cloud holes, has good repair effect on point cloud holes, and can obtain high acceleration ratio, which can provide reference for practical engineering application.

1. Introduction

Chinese ancient architecture is an important witness of Chinese history and represents the inheritance and development of Chinese architecture. It is the bounden task of modern people to protect this kind of architecture, that is, to protect Chinese history. With the rapid development of laser surveying and mapping, computer virtual, image processing, 3D modeling, and other technologies, 3D laser scanning technology and virtual reality technology have also been greatly improved. They have been widely popularized and applied in the fields of ancient building reconstruction, mold manufacturing, 3D printing, and other fields and achieved good results [1]. There is no point cloud data due to the model itself being damaged during the 3D point cloud data collection, or the laser scanned line of sight being blocked, which directly affects the modeling quality. Therefore, in order to make the modeling appear smooth, it is necessary to repair the hole. The hole patching algorithm of point cloud data has been improved and developed, but there are many unclosed holes in the measured point cloud data due to the defects of measuring objects and instruments, complex measuring environment, and other factors. 3D reconstruction technology is a subject mixed intersection technology, which is widely used in pattern recognition, virtual reality, and other fields [2]. Among them, three-dimensional reconstruction methods are mainly divided into interpolation-based reconstruction methods and triangular mesh-based reconstruction methods. The research of such methods is becoming more and more mature, but there are still some shortcomings, which makes the effect and speed of reconstruction become a bottleneck. To improve the speed of recovery, the dot cloud design gap needs to be repaired quickly; the existing mature parallel technology can be used to optimize the point cloud hole repair algorithm in parallel and quickly improve the efficiency of hole repair [3, 4].

In the field of machine learning and cognitive science, artificial neural networks are similar in structure and function to the human brain. It is a central neural network system with adaptive nonlinear characteristics composed of a large number of neurons, a mathematical or computational model commonly used to estimate or approximate functions. Artificial neural network can appropriately change its composition structure according to the attributes of input information. It is an adaptive system [5]. The hidden layer is used to complete the corresponding calculation operation; that is, the operation calculation results are transferred to the output layer. The output layer is mainly used to process and display the final calculation results. Because of its unique advantages, neural network has good applications in many fields, especially in classification and recognition. The neural network interprets the data input by the sensor through the machine sensing system, marks the original data, and clusters it. The pattern recognized by neural network is mainly based on the vector of array type. Therefore, before processing the data, we need to convert the data into numerical value and then operate the converted data. Neural network can realize clustering and classification operations, and it can be thought of as a cluster or classification layer at the top of the stored and managed data. For unlabeled data, neural networks can group data according to the similarity of the input sample. Therefore, the deep neural network can be regarded as a very important part of the machine learning system, which also includes reinforcement learning, classification, and regression algorithms [6]. Figure 1 shows the process of optimizing the neural network to restore the 3D mesh surface.

2. Literature Review

In recent years, in-depth study has been well used in image recognition and targeting, which has attracted extensive attention, and has shown good application prospects in other fields. Therefore, it has become a research point of extensive concern in various fields. Neural network in deep learning is the focus of research. When neural network is used to operate the image, its effect is better than traditional methods, the accuracy of image recognition is higher, and the operation process is relatively simple [7]. Trybaa et al. calculate the data points through the weighted average method and take the inverse data of the node distance as the weight function and interpolate it, so as to make the object surface clear. Although this method is quite simple, effective, and widely used, when the amount of point data is large, the amount of calculation will increase and the number of calculation errors will also increase [8]. Gao et al. described a new analysis method for representing irregular surfaces, which involves the summation of quadric surface equations with unknown coefficients. The quadric surface is located at the important point of the whole area to be drawn. It is given to solve the polygonal surface equation based on the polyhedron equation and compare the polygonal surface with the irregular surface [9]. Tu et al. fitted the three-dimensional surface by improving the radial basis function and evaluated the time, storage, and accuracy by using the scattering data interpolation and testing methods, so as to improve the visual observability of the object surface, and the repair speed and accuracy have good effects [10]. Chen et al. studied the application of neutral network algorithm in surface reconstruction of scattered point cloud and implicit surface meshing. It is found that the most suitable neutral network type is the modified version of the growth cell structure. The algorithm randomly extracts a target space, usually a point cloud or an implicit surface, and adjusts the connectivity of the neutral mesh accordingly. The algorithm can better deal with sharp and concave surfaces, and the speed of the algorithm is independent of the size of the input data. It is more suitable for reconstructing surfaces from larger point sets [11]. Lin et al. presented a surface reconstruction algorithm with multiple attributes, which mainly studies multiscale scattered point cloud data. The algorithm uses the balanced binary tree to divide the object space to form several small spatial regions and then uses the radial basis function to calculate the difference, so as to integrate the repaired small regions into a complete three-dimensional model [12]. Martinez et al. found that triangulation is more suitable for cavity repair of 3D model than other methods. The point is that there are certain limitations according to the discrete relationship of cloud and noise information, there is a topologically consistent relationship between the data, and the points are given by the cloud removal algorithm and the noise [13]. Biundini et al. proposed a new method to solve three-dimensional shape deformation, which is the process of maximizing the similarity between two shapes by representing the interpolation of two shapes as the interpolation of one shape. The process incrementally optimizes the objective function and deforms the implicit surface model. The deformable surface is represented as a level set of densely sampled scalars. Since this method adopts the prediction method, there are certain uncertainties and limitations [14]. In order to reconstruct the model, Hsu and Zhuang used coordinate measuring machine or laser scanner to collect point cloud data and then converted the collected data into a model. The reconstruction of model surface was realized by neural network approximation method. Three four-layer neural networks were designed, and the designed neural network was trained by error back propagation method [15]. For the grid method in surface reconstruction, reconstruction, simplification, and stratification of 3 D grids, Duanmu and Xing proposed the algorithm for hierarchical reconstruction of surface grids with large data quantity [16].

Based on this research, this paper proposes a point cloud data hole repair aggregation algorithm based on optimized neural network. The optimized neural network is not only suitable for processing large-scale and noisy data but also the scientific community has proved that the optimization function with a hidden layer has the best approximation ability.

3. Research Methods

3.1. Optimize the Working Principle and Topological Network Structure of Neural Network

3.1.1. Neural Network Overview. The structure of the network contains a large number of nerve cells, and the nerve cells are interconnected. According to the different

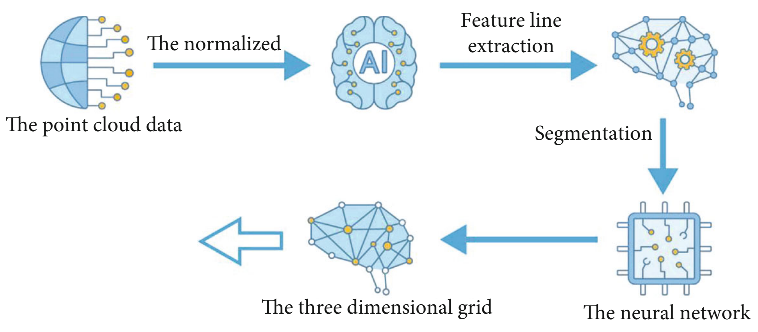

Figure 1: Process of optimizing neural network to reconstruct 3D mesh surface.

connection methods between neurons in the network system structure, it is often divided into the following types:

(1) Feedforward Neural Network. Feedforward neural network is also called forward neural network. The feedback signal will be generated in the training process. Because there is no feedback between the network layers, the data can be transmitted to the output layer during classification; it can only be transmitted forward in one direction, which is referred to as feedforward network for short. The network can be represented by N, the input vector is represented by X, the connection weight vector of each layer in the network is $W_1 \sim W_3$, and the activation function can be represented by $F_1 \sim F_3$. The network output value is $O_1 = F_1(XW_1)$, the output corresponding to the network hidden layer is $O_3 = F_3(F_2(F_1(XW_1)W_2)W_3)$.

(2) Feedback Neural Network. A feedback neural network is a neural network with its own internal or global response from output to input. In this type of network, each neuron feeds back its output signal to other neurons as input signal. After working for a period of time, the network can reach a relatively stable state. Compared with feedforward neural network, it has higher complexity [17, 18].

3.1.2. Topology and Network Structure. The neural network topology is a three-layer network with one hidden layer. The function responds locally to the input signal. If the input signal x of the function is close to the center of the function, the nodes of the hidden layer produce a larger output. You can see that this network has local proximity capabilities [19]. The value r of hidden layer neurons can be obtained by the following formula:

$$R(\|D\|) = e^{\left(-\|D\|^2/2\delta^2\right)}. \tag{1}$$

3.2. Neural Network Point Cloud Data and Hole Repair Algorithm

3.2.1. Working Mechanism of Neural Network Applied to Point Cloud Data Hole Repair. Neural network is a forward network composed of three layers, which can realize the approximation to any function. Its commonly used classical function $f(x)$ has the following types:

(1) Gaussian function (Gaussian function)

$$f(x) = e^{-t^2/\delta^2} \tag{2}$$

(2) Reflected sigmoidal function (abnormal S-type function)

$$f(x) = \frac{1}{e^{t^2/\delta^2}} \tag{3}$$

(3) Inverse multiquadric (inverse distortion correction function)

$$f(x) = \frac{1}{\left(t^2 + \delta^2\right)^{\alpha}}, \quad \alpha > 0 \tag{4}$$

Through the analysis of several functions, it is found that Gaussian function is more suitable for the hidden layer transformation function in this study, and its expression is as follows:

$$\phi_j(X) = e^{-}\frac{\|X - c_j E\|}{2\delta^2}, \quad j = 1, 2, \cdots, h. \tag{5}$$

In the formula, φ_j is the output of the j-th unit of the hidden layer; $X = (x_1, x_2, \cdots, x_T)$ is the input vector; $\|.\|$ represents a norm, usually taken as

$$\|X - c_j E\| = \left(X - c_j E\right)^T \left(X - c_j E\right). \tag{6}$$

c_j is the center of the j Gaussian unit in the hidden layer; E is $n \times 1$ unit vector; δ_j is the radius.

The output of a neural network can be expressed as follows.

$$y_k = \sum_{j=1}^{h} \omega_{kj}\phi_j(X), \quad k = 1, 2, \cdots, m, \tag{7}$$

written in matrix form

$$Y = W\Phi, \tag{8}$$

where $Y = (y_1, y_2, \cdots, y_m)^T$ is the output vector and $\Phi = (\phi_1(X), \phi_2(X), \cdots, \phi_h(X))^T$ is the output of hidden layer.

3.2.2. Implementation of Point Cloud Data and Hole Repair Algorithm of Neural Network. Main steps of point cloud filtering and hole repair using neural network are the following:

(1) Collection and normalization of prediction data and training data: silent point cloud data is extracted from raw cloud data loaded in MATLAB R2016a, part of the data is randomly selected as the prediction data, the rest is the training data, and then, the two parts of the data are normalized between [0,1]

(2) Train the neural network: initialize the parameters, use the corresponding command to create the training function traingdm, and use traingdm () to train the network until the model accuracy requirements are met

(3) Use a neural network to predict output from predictive data

(4) The neural network filters the point cloud data. Input the original experimental data containing noise into the trained system for normalization and inverse normalization. Finally, replace the corresponding original coordinates with the predicted output coordinates of all points after the training

(5) After the system training, input (x_i, y_i) at the second hole to predict z_k. Finally, take all (x_i, y_i, z_k) of the prediction output as the coordinates of the point cloud at the hole

3.3. Design and Study of the Parallel Algorithm

3.3.1. Point Cloud Hole Patching Algorithm Based on OpenMP. When using OpenMP parallel technology based on multicore CPU to optimize the point cloud hole repair algorithm in parallel, it mainly was aimed at selecting the center point of the basis function in the algorithm: octree data structure initialization, CNN convolution neural network convolution operation, and CNN convolutional neural network pooling operation; CNN convolutional neural network performs parallel optimization for full connection operation and other parts. The specific parallel flow diagram of the parallelizable part of the parallel algorithm based on OpenMP is shown in Figure 2.

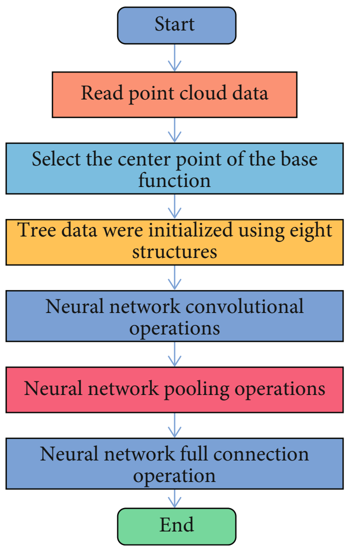
Start
Read point cloud data
Select the center point of the base function
Tree data were initialized using eight structures
Neural network convolutional operations
Neural network pooling operations
Neural network full connection operation
End

Figure 2: Flow chart of point cloud hole repair algorithm based on multicore OpenMP.

The point cloud hole repair algorithm uses the scattered point cloud data points in the point cloud model as the basis of the research, and the amount of data generated is usually used, hundreds of millions, and each data has a certain independence for a very large number of data points can be combined with the corresponding parallel technology. For a very large number of data points, the corresponding parallel technology can be combined to perform parallel operations on a large number of data points. Read the point cloud data from the entry point cloud model and use the octal data. This structure stores the point cloud data collected in the point cloud model, and the stored information mainly includes the coordinates of each data point in the point cloud model, number of octet layers and suboctet pointers, and operation of octree data structural elements.

3.3.2. CUDA-Based Point Cloud Hole-Filling Algorithm. The programming model based on GPU (Compute Unified Device Architecture, CUDA) has strong floating-point operation ability. When the amount of calculation is large, the calculation advantage of GPU is more obvious, and it has the advantages of better multithreaded calculation ability and less transmission delay.

Pay attention to the following points when using CUDA programming: initialize the video memory at the GPU end. When transmitting a large amount of data, the transmission delay between GPU and CPU becomes higher and higher, which reduces the transmission efficiency of data. In order to transmit data quickly and effectively, the transmission frequency between GPU and CPU can be reduced [20].

This chapter adopts CUDA parallel technology based on GPU to optimize the point cloud hole repair algorithm accordingly. When using convolutional neural network to repair the point cloud hole, there are three main parts with large computational overhead, and parallel optimization is carried out for the parts with hot spots (for the process of

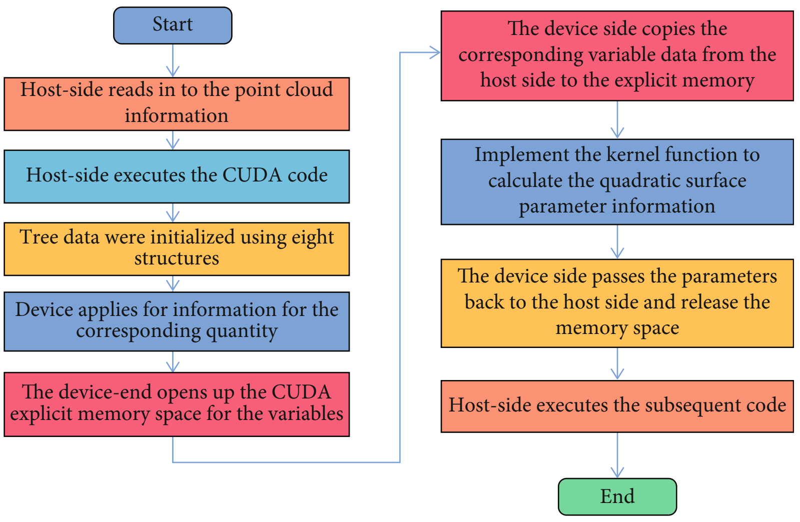
Start
Host-side reads in to the point cloud information
Host-side executes the CUDA code
Tree data were initialized using eight structures
Device applies for information for the corresponding quantity
The device-end opens up the CUDA explicit memory space for the variables
The device side copies the corresponding variable data from the host side to the explicit memory
Implement the kernel function to calculate the quadratic surface parameter information
The device side passes the parameters back to the host side and release the memory space
Host-side executes the subsequent code
End

Figure 3: Flow chart of parallel computing quadric surface parameters based on CUAD.

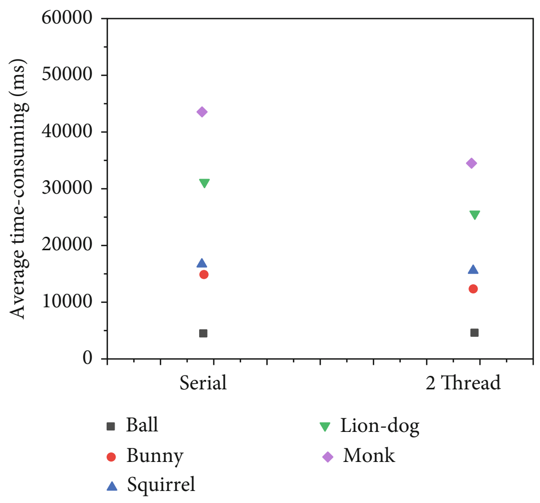

Figure 4: Broken line diagram of running time of different point cloud models under different threads of platform 1.

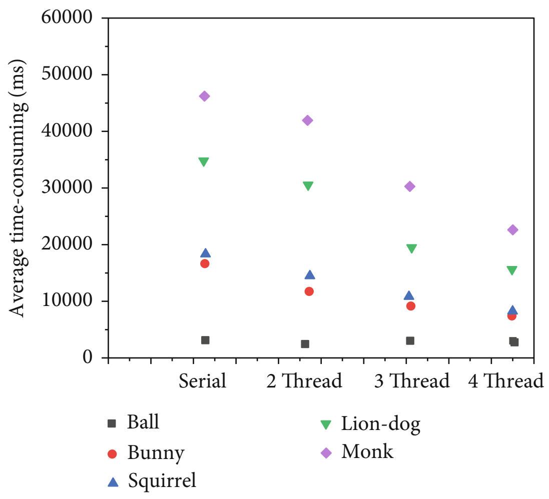

Figure 5: Broken line diagram of running time of different point cloud models under different threads of platform 2.

using CUDA to calculate surface parameter information in parallel, see Figure 3).

4. Result Discussion

4.1. Analysis of Parallel Optimization Results Based on OpenMP. The previous section analyzes and compares the serial average running time of algorithms on different platforms through experiments. In this section, OpenMP based on CPU is used to optimize the hot issues in the algorithm.

4.1.1. Average Running Time Analysis. Under platform 1, for different point cloud models, the average running time of the algorithm allocated different number of parallel threads is analyzed. The number of threads allocated by the algorithm is different. The number of circuits allocated by the algorithm is different, and the algorithm's run time is also different. The specific run time of the algorithm is shown in Figure 4.

According to the above figure, when OpenMP is used for optimization, for the same point cloud model, the more threads the algorithm allocates, the less the average running time of the algorithm. For different point cloud models, the average run time of the algorithm also increases as the amount of point cloud data contained in the point cloud model increases. For the ball model, due to the small amount of point cloud data contained in the model, when using multithreading to execute the model, the running cost of threads is large. Therefore, as the number of circuits increases, so does the algorithm's run time. When the point cloud model contains more point cloud data or the number of threads allocated in the algorithm is more, the shorter the algorithm, the lower the algorithm's performance.

In addition, this paper makes a corresponding comparative analysis on the algorithm running time of different point cloud models to allocate different number of threads under platform 2. The specific experimental results are shown in Figure 5.

It can be seen from the above figure that when OpenMP is used for optimization, when a single point cloud model contains a large amount of point cloud data in a point cloud

Table 1: Statistical table of running time of different point cloud models under different number of parallel threads of platform 3.

Model name Thread number	Serial	2 threads	3 threads	4 threads	5 threads	6 threads
Ball	65	72	78	81	90	103
Bunny	8277	7150	4406	3261	1040	1056
Squirrel	10875	6428	5162	4025	2316	1204
Lion-dog	10283	15603	12470	8207	4502	3500
Monk	17650	20320	15180	11058	6014	4821

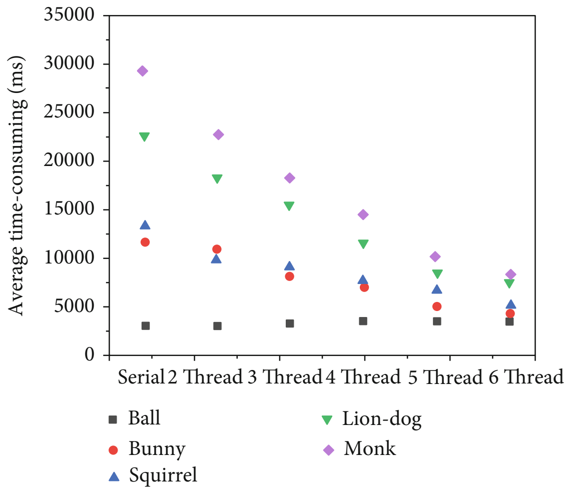

Figure 6: Broken line diagram of running time of different point cloud models under different threads of platform 3.

Table 2: Comparison of the number of point clouds before and after algorithm optimization.

Model name	Number of serial point clouds	Number of platform 1 points	Number of platform 2 points	Number of platform 3 points
Ball	104	101	100	98
Bunny	26751	26070	26004	25842
Squirrel	42080	41068	40856	40706
Lion-dog	100623	107308	107076	107065
Monk	132457	130310	128510	128304

Table 3: Accuracy analysis of OpenMP based algorithm before and after optimization.

Model name	Number of serial point clouds	Accuracy rate of 1 point number of platforms (%)	Accuracy of platform 2 points (%)	Accuracy of platform 3 points (%)
Ball	104	97.4	96.4	96.0
Bunny	26751	97.1	97.0	96.5
Squirrel	42080	97.0	96.6	96.3
Lion-dog	100623	96.8	96.6	96.5
Monk	132457	96.7	96.2	96.0

model, the more threads allocated in the algorithm, the less the average running time of the algorithm. When the point cloud model contains more point cloud data, the more threads are allocated, the shorter the running time of the algorithm, and the execution time shows a downward trend. Finally, this paper compares and analyzes the running time of the algorithm when platform 3 allocates different number of threads. The specific experimental results are shown in Table 1 and Figure 6.

As can be seen from the figure and table above, the different models of point clouds on platform 3 increase the number of circuits and the algorithm run time. For the same number of circuits, the higher the point cloud data contained in the point cloud model, the shorter the average algorithm run time. The trend is more pronounced. It can be seen that the average operating time of the algorithm is clearly related to the amount of data contained in the point cloud model and the number of circuits.

4.1.2. Comparison of Algorithm Accuracy under Different Platforms. This section mainly analyzes and compares the accuracy of the algorithm based on OpenMP before and after optimization. This paper compares the number of preoptimization point cloud data and the postoptimization point cloud data on different platforms to determine the accuracy of the algorithm before and after optimization. The specific situation of the number of point cloud data before and after optimization is shown in Table 2, and the accuracy before and after algorithm optimization based on OpenMP is shown in Table 3.

According to Tables 2 and 3, the change of the number of point clouds is within the normal error range and has a certain controllability. Before and after the optimization of the algorithm, the number of point cloud data in the point cloud model does not change much but increases or decreases slightly accordingly. When the platform is more complex, the change range of the number of point clouds increases slightly, but it is generally within the error range. Therefore, the accuracy of the algorithm does not change much before and after optimization.

4.2. Analysis of Parallel Optimization Results Based on CUDA. In this paper, NVIDIA Tesla c2070 graphics card is used for experimental analysis, and the parallel efficiency and speedup ratio of the algorithm is compared through experiments. Different point cloud models are tested under platform 3. The specific test results are shown in Table 4.

Table 4: Statistics of running time and acceleration ratio of GPU-based algorithm under platform 3.

Model name	Serial time	Parallel time	Acceleration ratio
Ball	65	31	1.70
Bunny	8277	1400	2.64
Squirrel	10875	2316	2.10
Lion-dog	10283	3852	3.20
Monk	17650	4634	4.00

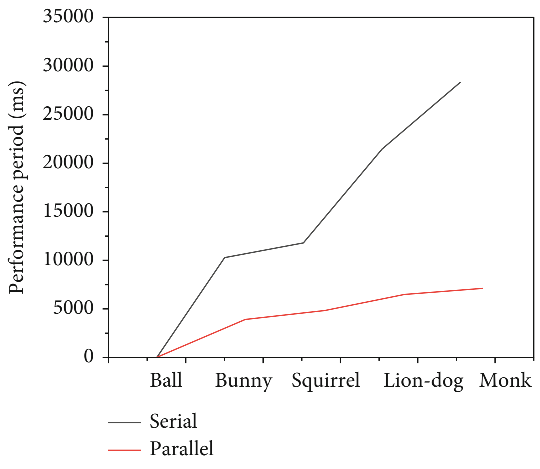

Figure 7: Running time histogram of each point cloud model based on CUDA under platform 3.

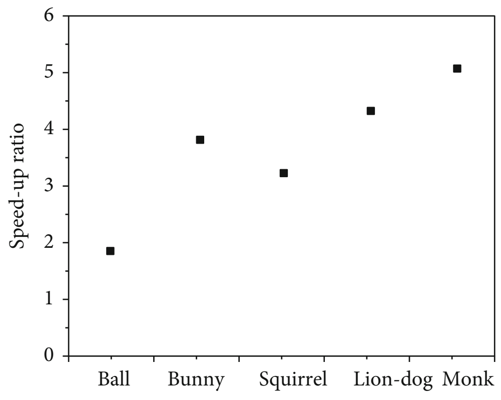

Figure 8: Linear graph of the acceleration ratio of each point cloud model on platform 3.

4.2.1. Average Running Time Analysis. In this section, the running time of different point cloud models on platform 3 is analyzed, and the optimized experimental results are compared and analyzed, as shown in Figure 7.

According to Figure 7, the running time of different point cloud models running on platform 3 after parallel optimization is significantly lower than that of serial algorithm. Therefore, parallel optimization based on GPU has certain feasibility and effectiveness.

4.2.2. Acceleration Ratio Analysis. This section mainly analyzes and compares the acceleration ratio of CUDA-based point cloud model running on platform 3. The experimental results are shown in Figure 8.

According to Figure 8, under the same experimental platform, when using CUDA for parallel optimization, the more point cloud data in the point cloud model, the higher the acceleration ratio of the algorithm.

Table 5: Accuracy analysis of CUDA-based algorithm before and after optimization.

Model name	Number of serial point clouds	Accuracy rate of 1 point number of platforms (%)	Accuracy of platform 2 points (%)	Accuracy of platform 3 points (%)
Ball	104	98.4	98.0	97.4
Bunny	26751	97.8	97.6	97.5
Squirrel	42180	97.7	97.3	97.0
Lion-dog	100623	97.5	97.2	97.0
Monk	132457	96.8	96.6	96.4

Table 6: Comparison of the number of point clouds before and after algorithm optimization.

Model name	Number of serial point clouds	Number of platform 1 points	Number of platform 2 points	Number of platform 3 points
Ball	104	103	102	101
Bunny	26751	26334	26260	26221
Squirrel	42080	41442	41228	41015
Lion-dog	100623	108073	107741	107520
Monk	132457	130442	130155	128868

4.2.3. Comparison of Algorithm Accuracy under Different Platforms. This section mainly analyzes and compares the accuracy before and after CUDA-based algorithm optimization. Among them are the main points of analysis of serial algorithms and number of cloud data points and cloud models downloaded on platform 1, platform 2, and platform 3 using CUDA parallel optimization number of point cloud data, as shown in Table 5.

Before and after optimizing the algorithm, the number of point cloud data in the point cloud model does not change much, as Table 6 shows that the point cloud data for that model may be missing to optimize the object model. It can be seen from Table 5 that the number of point cloud data in the point cloud model before and after optimization of the algorithm does not change much, because when optimizing the object model, the point cloud data of the model may be missing due to human reasons, but the point cloud data has only slight increase or less, and there is no widespread or massive data loss. As can be seen from Table 5, there are clear changes in the accuracy of the CUDA-based algorithm before and after optimization. Since the model contains more dot cloud data, the algorithm's accuracy decreases slightly, but the data increases in the order of 104 points, and the accuracy rate decreases to about 96.8% in a monkey model environment with a data size of 132,457 when a bending point occurs. In addition, the accuracy of algorithms that work on different platforms varies for the same point cloud model. As the complexity of the platform increases, the accuracy of the algorithm decreases slightly, but in the point cloud model, the precision of point cloud data or different action programs and algorithms the accuracy has hardly changed. Therefore, the accuracy of the algorithm before and after the optimization is almost the same, and there is not much difference.

5. Conclusion

For the scattered point cloud data collected in this document, the octree data structure was decomposed into the entire point cloud data, and then, the hierarchical operation of the point cloud model was completed. The two-dimensional coordinates in three directions are obtained through spatial mapping. Use neural networks to unravel the hole features of point cloud models. According to the extracted hole features, the corresponding sparse representation classification model is constructed, and the residuals of each category are calculated to generate the boundary of the point cloud hole, obtain the pure hole area, and then complete the filling of the polygon hole. When using a multilayer CNN convulsive neural network to fill a point cloud gap, the efficiency of the algorithm is low due to factors such as the size of the network and the complex operational process. The point cloud hole compensation algorithm, which was aimed at solving a large number of scattered cloud data computational problems, optimizes a combination of commonly used parallel computing technologies, such as CPU acceleration and GPU acceleration technology, to quickly reduce downtime, improve algorithm performance, and increase algorithm efficiency.

Data Availability

No data were used to support this study.

Conflicts of Interest

The author declares that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

This work was supported by the Science and Technology Research Project of Jiangxi Education Department of China (GJJ212303) and the Industry University Research Innovation Fund of Chinese Universities (2019ITA01051).

References

[1] M. Siranec, M. Hger, and A. Otcenasova, "Advanced power line diagnostics using point cloud data—possible applications and limits," *Remote Sensing*, vol. 13, no. 10, pp. 1880–1887, 2021.

[2] M. Eslami and M. Saadatseresht, "Imagery network fine registration by reference point cloud data based on the tie points and planes," *Sensors*, vol. 21, no. 1, pp. 317–324, 2021.

[3] S. K. Tyagi, A. Mukherjee, Q. Boyang, and D. K. Jain, "Computing resource optimization of big data in optical cloud radio access networked industrial Internet of things," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 11, pp. 7734–7742, 2021.

[4] L. Mattheuwsen and M. Vergauwen, "Manhole cover detection on rasterized mobile mapping point cloud data using transfer learned fully convolutional neural networks," *Remote Sensing*, vol. 12, no. 22, pp. 3820–3826, 2020.

[5] G. Park, Y. J. Lee, and S. Sung, "Study of integrated navigation system based on point cloud data and map for environment under bridge," *Transactions of the Korean Institute of Electrical Engineers*, vol. 69, no. 12, pp. 1970–1976, 2020.

[6] Y. H. Jin, I. T. Hwang, and W. H. Lee, "A mobile augmented reality system for the real-time visualization of pipes in point cloud data with a depth sensor," *Electronics*, vol. 9, no. 5, pp. 836–841, 2020.

[7] Y. Xu, S. Arai, F. Tokuda, and K. Kosuge, "A convolutional neural network for point cloud instance segmentation in cluttered scene trained by synthetic data without color," *IEEE Access*, vol. 8, no. 99, pp. 70262–70269, 2020.

[8] P. Trybaa, J. Blachowski, R. Baej, and R. Zimroz, "Damage detection based on 3d point cloud data processing from laser scanning of conveyor belt surface," *Remote Sensing*, vol. 13, no. 1, pp. 55–61, 2021.

[9] Y. Gao, C. Ping, L. Wang, and B. Wang, "A simplification method for point cloud of t-profile steel plate for shipbuilding," *Algorithms*, vol. 14, no. 7, pp. 202–205, 2021.

[10] X. Tu, C. Xu, S. Liu, S. Lin, and R. Li, "Lidar point cloud recognition and visualization with deep learning for overhead contact inspection," *Sensors*, vol. 20, no. 21, p. 6387, 2020.

[11] W. Chen, X. Li, H. Ge, L. Wang, and Y. Zhang, "Trajectory planning for spray painting robot based on point cloud slicing technique," *Electronics*, vol. 9, no. 6, pp. 908–912, 2020.

[12] W. Lin, W. Fan, H. Liu, Y. Xu, and J. Wu, "Classification of handheld laser scanning tree point cloud based on different knn algorithms and random forest algorithm," *Forests*, vol. 12, no. 3, pp. 292–296, 2021.

[13] J. Martinez, G. Albeaino, M. Gheisari, W. Volkmann, and L. F. Alarcon, "Uas point cloud accuracy assessment using structure from motion-based photogrammetry and ppk georeferencing technique for building surveying applications," *Journal of Computing in Civil Engineering*, vol. 35, no. 1, pp. 05020004–05020015, 2021.

[14] I. Z. Biundini, M. F. Pinto, A. G. Melo, A. Marcato, and M. Aguiar, "A framework for coverage path planning optimization based on point cloud for structural inspection," *Sensors*, vol. 21, no. 2, pp. 570–573, 2021.

[15] P. H. Hsu and Z. Y. Zhuang, "Incorporating handcrafted features into deep learning for point cloud classification," *Remote Sensing*, vol. 12, no. 22, pp. 3713–3728, 2020.

[16] J. Duanmu and Y. Xing, "Annular neighboring points distribution analysis: a novel pls stem point cloud preprocessing algorithm for dbh estimation," *Remote Sensing*, vol. 12, no. 5, pp. 808–813, 2020.

[17] X. Wu, "Context-aware cloud service selection model for mobile cloud computing environments," *Wireless*

Communications and Mobile Computing, vol. 2018, Article ID 3105278, 14 pages, 2018.

[18] L. Ni, X. Sun, X. Li, and J. Zhang, "GCWOAS2: multiobjective task scheduling strategy based on Gaussian cloud-whale optimization in cloud computing," *Computational Intelligence and Neuroscience*, vol. 2021, Article ID 5546758, 17 pages, 2021.

[19] M. Ramezani Mayiami, M. Hajimirsadeghi, K. Skretting, X. Dong, R. S. Blum, and H. V. Poor, "Bayesian topology learning and noise removal from network data," *Discover Internet of Things*, vol. 1, no. 1, p. 11, 2021.

[20] D. J. Lee, "Transfer learning based semantic segmentation for 3d object detection from point cloud," *Sensors*, vol. 21, no. 12, pp. 3964–3972, 2021.