applied sciences

MDPI

Article

Virtual Reality Collision Detection Based on Improved Ant Colony Algorithm

Peng Xu and Qingyun Sun *

School of Mechanical and Electronic Engineering, Nanjing Forestry University, Nanjing 210037, China; xupeng1@njfu.edu.cn
* Correspondence: sunqingyun@njfu.edu.cn

Abstract: In order to improve the performance in terms of detecting objects colliding in virtual reality, the ant colony algorithm was used to detect collisions. In the preliminary detection stage, the OBB bounding box and the spherical bounding box were used to detect the collision of objects, and the objects that may collide were selected. In the accurate detection stage, the model was sampled, and the feature pairs were used as the set to be detected for detecting collisions, the collision detection problem of the three-dimensional model was transformed into a nonlinear optimization problem of the distance between the feature pairs in the two-dimensional discrete space. The ant colony algorithm was introduced to solve the problem, and the pheromone concentration and update rules of the ant colony algorithm were optimized to improve the efficiency of the algorithm. The simulation results showed that, compared with the commonly used collision detection algorithms, our algorithm had high accuracy in detecting collisions and was less time-consuming.

Keywords: collision detection; virtual reality; ant algorithm

Citation: Xu, P.; Sun, Q. Virtual Reality Collision Detection Based on Improved Ant Colony Algorithm. *Appl. Sci.* **2023**, *13*, 6366. https://doi.org/10.3390/app13116366

Academic Editors: Dimitris Mourtzis, Cezary Biele, Grzegorz Pochwatko, Wiesław Kopeć and Andrzej Romanowski

Received: 28 April 2023
Revised: 20 May 2023
Accepted: 22 May 2023
Published: 23 May 2023

1. Introduction

Collision detection, also known as interference detection or contact detection, is a classic problem in the field of system simulation, computer graphics and other research fields. The primary task of collision detection is to detect whether there is contact between objects in a virtual scene, that is, to detect whether they occupy an identical space. When they occupy an identical space, it means that there is a collision between the models [1,2]. Since the last century, many researchers have studied the technology of collision detection, but due to the influence of the level of computer hardware, the collision detection algorithms at that time generally could not fulfill the real-time needs of application scenarios. In recent years, with the development of science and technology, real-time collision detection technology has evolved swiftly. At the same time, with the development of computer graphics technology, real-time simulations of intricate scenes have received extensive attention, and the real-time and precision requirements of collision detection algorithms are also rising. There are two main types of collision algorithms: those based on the time domain and those based on the space domain [3].

The traditional collision detection algorithm has several problems when detecting collisions in complex virtual scenes. The number of model patches in complex scenes is large, and the detection time is long and cannot ensure the real-time performance of collision detection. When the parts are moving, the position is constantly changing, and the efficiency of collision detection is low [4]. Therefore, many scholars are constantly researching and optimizing collision detection technology. For example, in order to improve the safety of digital subtraction angiography (DSA) equipment in the treatment of cardiovascular diseases, Hu Anlin combined the oriented bounding box (OBB) algorithm with the GJK algorithm and proposed a hybrid collision detection algorithm based on separation distance [5]. A collision detection system named SOLTD, which was based on an AABB bounding box level developed by Eindhoven University in the Netherlands, used a simplified separation

axis test method to perform overlapping tests between AABBs to improve the efficiency of detecting collisions. Bergen adopted the basic idea of GJK and developed SLTF by combining the incremental culling technology based on axial bounding box sweeping and cutting. The algorithm improved the efficiency of the algorithm by caching the separation axis of the previous frame's object pair and using the coherence between frames to judge the potentially intersecting object pairs [6]. Lin-Canny performed interval subdivision on the polyhedron and constructed the corresponding region for each feature, thereby creating a feature-based collision detection algorithm [7].

In addition to the above research directions, many scholars have introduced an intelligent group algorithm for detecting depth based on bounding volume hierarchy technology: Hui Xue-Wu used a bounding box and a particle swarm optimization algorithm for detecting collisions in virtual reality and used a binary tree structure to store the cross-space features of a bounding box, which effectively improved the accuracy of detecting collisions. However, the particle swarm optimization algorithm is not efficient for searching; therefore, the algorithm is not suitable for large-scale scenarios. Jin Han-Jun proposed an algorithm for detecting collisions between convex polyhedrons based on a genetic algorithm. He mainly discussed the method of calculating the shortest distance in collision detection, that is, the calculation of the shortest distance was summed up as an optimal solution to a nonlinear programming problem with constraints. According to experiments to verify the surface, the genetic algorithm has fast calculation speed and high calculation accuracy, but it is only suitable for detecting collisions between convex polyhedrons. Wang Yi proposed a random collision detection algorithm based on the particle swarm optimization algorithm. This method transforms the problem of detecting objects colliding in three-dimensional space into an optimization problem in a discrete two-dimensional space and then uses the particle swarm optimization algorithm to solve it. The algorithm can deal with detecting collisions between arbitrary polyhedron models while ensuring efficiency and has good versatility. However, it is not guaranteed to find all feature pairs of the collision, and there is room for improvement in terms of speed and accuracy [8–10].

By summarizing the characteristics of the existing collision detection algorithms at this stage, we find that although the research results regarding collision detection algorithms have been relatively rich, they are mainly used to deal with collision detection between the two models. When dealing with the problem of detecting collision in large-scale complex scenes, the current algorithms still have problems and cannot deal with the problem of detecting collisions between objects quickly and accurately. In order to solve the above problems, this paper introduced the ant colony algorithm into the random collision detection algorithm, and combined the bounding volume hierarchy technology to study and design the Sphere-OBB-MACO algorithm. The algorithm is divided into two steps: in the initial detection stage, the OBB bounding box and the spherical bounding box are combined to form a hybrid hierarchical bounding box for detecting collisions, and the bounding boxes that may collide are selected. In the accurate detection stage, the selected bounding boxes are sampled, and the sampled feature pairs are used to form the search space. The improved ant colony algorithm was introduced to solve the optimization problem of the search space. Compared with other collision detection algorithms, this paper uses the OBB bounding box and the spherical bounding box to construct the bounding volume hierarchy instead of a single bounding box for detection in the preliminary detection stage. Although the time consumed in the preliminary detection stage becomes longer, it improves the accuracy of collision detection and reduces the time required for accurate detection. Because of the large number of feature pairs in complex scenes, the ant colony algorithm with higher search efficiency was selected in the accurate detection stage to replace the particle swarm algorithm used by other researchers, and the defects of the ant colony algorithm are improved.

2. Construction of the Bounding Volume Hierarchy

As a logical structure, the bounding volume hierarchy is also a hierarchical structure, which has the following two characteristics: the root node of the tree has no precursor, and all nodes except the root node only have one precursor; all nodes in the tree can have 0 or more successors. For a balanced m-ary tree with n nodes, the time spent traversing a tree is $f(m) = m^2 log_m n$. When $m = 2$, $f(m)$ is the smallest. In summary, the tree structure of this study was a binary tree. In order to construct it conveniently, this study chose the top-down method with the highest perfection to construct the binary tree. The core problems of the top-down construction method were how to determine the split plane and maximize the adjacent geometric elements in the space into the same subspace, and a split point and a split axis can determine the split plane. For the splitting axis, according to calculations of the variance of the triangular surface axis, the axis with the largest variance is the splitting axis. As the splitting point, we can take the average value of the coordinates of the projection points on the splitting axis of the midpoint of all the triangular primitives in the bounding box.

The compact spherical bounding box is the weakest, but the structure is simple, which can reduce the time required for detecting collisions. The structure of the OBB bounding box is complex, but it has excellent compactness, which can improve the accuracy of detecting collisions. Therefore, this study chose the spherical bounding box with the weakest tightness but the simplest structure, and the OBB bounding box with a complex calculation but excellent tightness to construct the hybrid hierarchical bounding box, which combined the advantages of the two bounding boxes as much as possible to improve the efficiency of detecting collisions. Figure 1 shows the process of constructing the bounding volume hierarchy.

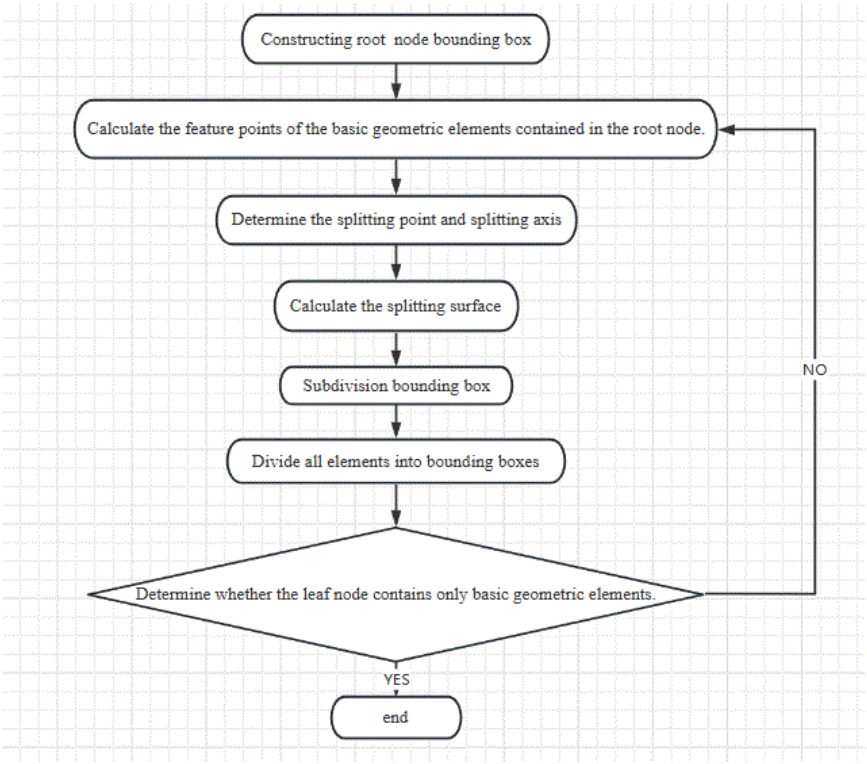

Figure 1. Construction process of the bounding volume hierarchy.

Optimization of the Traditional Method of Constructing an OBB Bounding Box

The traditional method of constructing an OBB bounding box is based on the triangular patch information of the geometric model, using the statistics of the mean and the unitized covariance matrix. The specific steps are as follows:

(1) The vertex coordinates of the triangular patch are added and divided by the number to obtain the mean value, and then the covariance of the vertex's coordinate vector is calculated, the mathematical model is derived from Yu Mei's article [11].

$$m = \frac{\sum_{i=0}^{n}\left(p^i + q^i + r^i\right)}{3n} \tag{1}$$

$$C_{jk} = \frac{\sum_{i=o}^{n}\left(\overline{p_j^i}\cdot\overline{p_k^i} + \overline{q_j^i}\cdot\overline{q_k^i} + \overline{r_j^i}\cdot\overline{r_k^i}\right)}{3n} \tag{2}$$

$$\overline{p^i} = p^i - m$$

$$\overline{q^i} = q^i - m$$

$$\overline{r^i} = r^i - m$$

(2) Three eigenvectors of the covariance matrix C are obtained, which are orthogonal to each other. In order to determine the three axial coordinates of the OBB bounding box, the feature vector is unitized. The difference between the maximum and minimum projection values on the coordinate axis is the size of the OBB bounding box.

Because m in Equations (1) and (2) is only a simple average of the coordinates, the coordinate's position will shift when the model's structure is not uniform. Therefore, this study used the centroid to determine the center of the OBB bounding box. The specific steps are as follows:

Firstly, the local coordinate system of the triangle is established, where the static moment of the triangle on the X axis and the Y axis is:

$$S_{x_i} = \int_{A_i} y dA$$

$$S_{y_i} = \int_{A_i} x dA$$

A_i : the area of the *i th* triangle.
dA : the microarea at this point.
The centroid position of the triangle is:

$$x_i = \frac{S_{y_i}}{A_i}, y_i = \frac{S_{x_i}}{A_i}$$

(3) The coordinates obtained above are in the local coordinate system and need to be converted into global coordinates. Finally, the combined centroid coordinates are

$$X_c = \frac{\sum_{i=1}^{n} A_i \cdot X_{c_i}}{\sum_{i=1}^{n} A_i}$$

$$Y_c = \frac{\sum_{i=1}^{n} A_i \cdot y_{c_i}}{\sum_{i=1}^{n} A_i} \tag{3}$$

$$Z_c = \frac{\sum_{i=1}^{n} A_i \cdot Z_{c_i}}{\sum_{i=1}^{n} A_i}$$

By including Equation (3) into Equation (2), the improved covariance matrix can be obtained.

3. An Improved Ant Colony Algorithm Based on Random Collision Detection

3.1. The Random Collision Detection Algorithm

If the two models are regarded as two feature sets, the problem of detecting collisions between the models can also be regarded as the process of judging whether there is at least a pair of feature pairs between the models that meets the threshold condition of a collision. In this way, the collision detection problem is transformed into an optimization problem in a discrete space composed of two object features [12,13].

Random collision detection is a collision detection algorithm for approximate response detection. The algorithm performs feature sampling on the model, and the sampled feature pairs form a search space. Random collision detection transforms the collision detection problem of the three-dimensional model into a discrete two-dimensional space. The optimization problem in terms of space can be expressed by the following formula:

$$minf(x) = f(x_1, x_2, x_3, \ldots, x_n) \tag{4}$$

$$a_i < x_i < b_i; i = 1, 2, 3, \ldots, n$$

The feature point pair (a_i, b_i) in the two-dimensional space is the solution of the equation above, and the task of random collision detection is to search for the pairs of feature points that may collide. If the two models collide, there must be at least one pair of feature pairs between the two models, and the distance between the feature pairs must be within the distance threshold of a collision. D represents the distance between feature pairs and δ is the collision distance threshold.

$$minD \leq \delta \tag{5}$$

In the space of virtual reality, there may be multiple collision points when two models collide, so there may be multiple feature pairs, which is a multi-peak optimization problem. Since the object may be moving at all times, the position of the optimal solution and the fitness of the solution space are constantly changing [14–17]. If the task of collision detection is regarded as a whole from the first moment, then this solution space is a complex and constantly changing multi-peak environment.

3.2. Random Collision Detection in the 3D Model

For three-dimensional models, selecting points as the feature pairs of random collision detection can improve the operational efficiency of the algorithm. Therefore, the combinations of feature points of the model are usually selected as the feature pairs to be detected in random collision detection. In practical applications, because the three-dimensional model is a three-dimensional structure, its position is generally determined by the vertex coordinates of the model, so the distance between the models will generally be transformed into the distance between the vertex coordinates or geometric primitives. Therefore, the collision distance threshold (δ) mentioned above was used to judge the distance between the three-dimensional models [18–21].

The nonlinear optimization problem requires an objective function to be selected as the fitness function to be optimized. For collision detection, precise detection is described as the test of a junction between patches. Therefore, the centroid of the triangular patch of the three-dimensional model was selected as the feature point of the sampling algorithm. When the centroids' spacing is lower than a certain value, the patches intersect, and the model intersects. Let the two centroid coordinates be (x_1, y_1, z_1), (x_2, y_2, z_2). The centroid spacing is as follows:

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \tag{6}$$

When the distance between the centroids is less than the collision distance threshold, collision occurs between the models.

In the stage of accurate collision detection, the geometric primitives that may collide in the preliminary detection stage are selected for feature sampling, and the sampled feature points are used as the set to be detected for detecting a collision. The three-dimensional model to be detected in the space is mapped to the two-dimensional space. The problem of detecting a collision in the three-dimensional model is transformed into a nonlinear optimization problem of the distance between the feature pairs in the discrete two-dimensional space. An intelligent algorithm is needed to solve this problem.

Intelligent algorithms are inspired by the laws of nature. According to their principles, they imitate algorithms for solving problems, such as genetic algorithms, simulated annealing algorithms and artificial neural network algorithms. There are many intelligent algorithms. To be applied to the field of collision detection, we first need to understand the problems that the optimization algorithm can solve and analyze the essence of the collision problem and then design and improve the optimization method that can be applied to detect collisions.

Researchers have generally introduced the particle swarm optimization algorithm to solve optimization problems, but because of the low search efficiency of the particle swarm optimization algorithm, it cannot be applied to complex search spaces. The robustness of the algorithm is not good, it can easily miss the interference elements and the spatial adjacency of the points is not easy to determine. Therefore, this study used the ant colony algorithm to solve the optimization problem. The ant colony algorithm uses distributed computing in the search process, and multiple individuals perform parallel computations at the same time, which greatly improves the computing power and operating efficiency of the algorithm [22,23].

3.3. The Traditional Ant Colony Algorithm

The ant colony algorithm is a probabilistic algorithm proposed by the Italian scholars Dorigo et al. to find an optimal path. Ants leave pheromones every time during the round-trip process of finding food. The shorter the distance, the more ants return and leave more pheromones, which will attract more ants to leave more pheromones. Over time, all ants move to the path with the most pheromones [24].

The basic ant colony algorithm is described as follows. The mathematical model is derived from M. Dorigo's article [25]. The probability of ants moving from one city to another city is:

$$P_{ij}^{k}(t) = \frac{\left|\tau_{ij}(t)\right|^{\alpha} \cdot \left|\eta_{ij}(t)\right|^{\beta}}{\sum_{s \epsilon J_K(i)} \left|\tau_{is}(t)\right|^{\alpha} \cdot \left|\eta_{is}(t)\right|^{\beta}} \quad j \epsilon J_k(i) \tag{7}$$

The initial amount the pheromone $\tau_{ij}(0) = c$, $J_K(i)$ represents the set of cities that the ants choose next, η_{ij} represents the expected number of ants moving from i to j, and the pheromone's heuristic factor α represents the degree of influence of the amount of information on the selected path. The ants are affected by the pheromone when selecting the path. This kind of wizard is the expected heuristic factor β. When an ant completes a round trip, the pheromone is expressed as follows:

$$\tau_{ij}(t+1) = (1-p) \cdot \tau_{ij}(t) + \Delta\tau_{ij} \tag{8}$$

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$$

ρ represents the evaporation coefficient of the pheromone on the path, $1-\rho$ represents the persistence coefficient of the pheromone, $\Delta\tau_{ij}$ represents the increase in the pheromone in this iteration process and $\Delta\tau_{ij}^{k}$ represents the amount of the pheromone left by the ants during this iteration.

$$\Delta\tau_{ij}^{k} = \frac{Q}{L_K}$$

where Q is a pheromone constant, which represents the total amount of pheromone released by ants once in a cycle; L_K represents the length of the path that the ants have traveled during this tour.

3.4. Optimization of the Ant Colony Algorithm

The traditional ant colony algorithm has two defects. As a large space search algorithm, the ant colony algorithm will perform a blind search in the early stages to generate a large number of invalid paths, and the convergence speed is slow; after all the ants have completed a pheromone update, the concentration of pheromones left on the shorter paths is higher than that on the longer paths. Under the action of positive feedback, ants will be attracted to pass through the path with more pheromones. However, if the algorithm has not found the optimal solution but rather the suboptimal solution at the beginning, the suboptimal solution will soon hold the absolute advantage and fall into the local optimal solution. In order to solve these problems, this study improved the ant colony algorithm [26,27] as follows.

(1) When selecting the next node, the ant is required to compute the transition probability of all the adjacent nodes linked to the current node and then select the route with the highest probability. The main factors affecting the probability of a transition are the factor of the pheromone's influence α and the expectation heuristic factor β. The pheromone's concentration and volatilization status in the ant colony algorithm are different in different stages: a low pheromone concentration and easy volatilization in the early stage; in the later stages, the concentration of pheromones is high and difficult to volatilize, but the traditional ant colony algorithm does not distinguish the ant colony's state. The values of the parameters α, β are fixed, and the ant colony algorithm is more likely to fall into the local optimal solution in this case [28].

Therefore, the algorithm needs to dynamically modify the value of α, β during iteration, and these take different values at different stages. In the initial stage, most ants do not find a reasonable path. At this time, the random search ability of ants should be enhanced to reduce the value of α, β; in the mature stage, most ants will find a better solution. In order to improve the convergence speed of the algorithm, the value of α, β should be increased:

$$\alpha' = \alpha \frac{N_c}{n} \tag{9}$$

$$\beta' = \beta \frac{N_c}{n} \tag{10}$$

α', β' are the values of the optimized factor of the pheromone's influence and the expected heuristic factor, n is a constant value, and N_c is the value of the current iteration.

(2) The volatilization factor ρ represents the degree of volatilization or the reduction in the concentration of pheromones. When the factor is too large, the degree of pheromone volatilization will be accelerated, and the convergence speed of the algorithm will be reduced. If the factor is too small, this will reduce the speed of pheromone volatilization, resulting in too many pheromones and the algorithm falling into the local optimal solution. The value of the factor of volatility in the traditional ant colony algorithm is fixed, so it needs to be improved to make it change dynamically. The algorithm should make the volatile factor larger in the early stage, so that the algorithm has a strong global search ability; in the mature stage, the volatilization factor should be gradually reduced to accelerate the convergence speed of the algorithm:

$$\rho = \frac{N}{N_c + N} \tag{11}$$

The volatilization coefficient decreases linearly with an increase in the number of iterations. N is the total number of iterations, and N_c is the value of the current iteration.

(3) The reciprocal of the distance between the two nodes is the heuristic factor η. The shorter the distance between the two nodes, the larger the heuristic factor, and the higher the probability that the ants will choose the path. However, this relationship can easily make the ants fall into the local optimal solution. In this study, the heuristic factor was improved, and the influence of the position of the starting point and the end point on the probability of an ant selecting it is included in the formula as follows:

$$\eta_{ij} = \frac{1}{d_{oj}+d_{ij}+d_{js}} \tag{12}$$

where d_{oj} is the distance from the starting point to node, and d_{js} is the distance from the node to the end point.

(4) The pheromone update rules greatly affect the efficiency of the ant colony in finding the optimal path, and all the ants in the traditional ant colony algorithm simply update their pheromones after reaching the end point, and the advantages of ants walking through shorter paths are not strongly reflected. Therefore, this study improved the pheromone update strategy and added a reward and punishment mechanism, strengthening the pheromone of an excellent path and weakening the pheromone of a poor path.

Under the assumption that the optimal evaluation value is bt and the initial value is 0, the formula is as follows:

$$bt = \begin{cases} bt & , bt(i) \geq bt \\ bt(i), & bt(i) < bt \end{cases} \tag{13}$$

where $bt(i)$ is the local optimal evaluation of the ith generation of ants. When $bt(i)$ is not better than bt, the optimal solution remains unchanged as bt. When $bt(i)$ is better than bt, the optimal solution is updated and $bt(i)$ becomes the optimal solution.

The pheromone is superimposed on the path that is superior to bt, and the pheromone on the path that is not superior to the optimal solution bt is volatilized. The formula is as follows:

$$\tau_{ij}(t+n) = (1-p)\cdot\tau_{ij} + \Delta\tau_{ij} + \Delta R \tag{14}$$

$$\Delta R = \begin{cases} \theta\cdot\Delta\tau_{ij} & , bt(i) \geq bt \\ \mu\cdot\Delta\tau_{ij} & , bt(i) < bt \end{cases} \tag{15}$$

$$\theta = 0.5, \mu = -0.5$$

4. Flow of the Algorithm

In the preliminary detection stage, the spherical and OBB bounding boxes are selected, the binary tree is selected as the tree structure and the Bounding Volume Hierarchy is constructed in a top-down manner. The depth-first traversal method is used to traverse the tree's structure, and the triangular primitive pairs without a collision are quickly removed. In the accurate detection stage, the bounding box with a collision after preliminary detection is processed by random sampling to form a search space composed of feature pairs. The optimized ant colony algorithm is introduced to solve the optimization problem of the search space, as shown in Figure 2. The specific steps are as follows:

(1) Establish a hierarchical bounding box for the target part to be detected and traverse the hierarchical bounding box to see whether the leaf nodes intersect.
(2) If there is a collision between the leaf nodes, feature sampling is performed on the primitive patches between the collision nodes to form a two-dimensional discrete search space.
(3) Initialize the ant population M, the pheromone intensity, visibility, population location and number of iterations of the algorithm.
(4) Calculate the optimal solution of each ant and the current global optimal solution and select the optimal solution after a comparison.

(5) To meet the final requirements, output the best individual that meets the collision threshold.
(6) Conduct the next test.

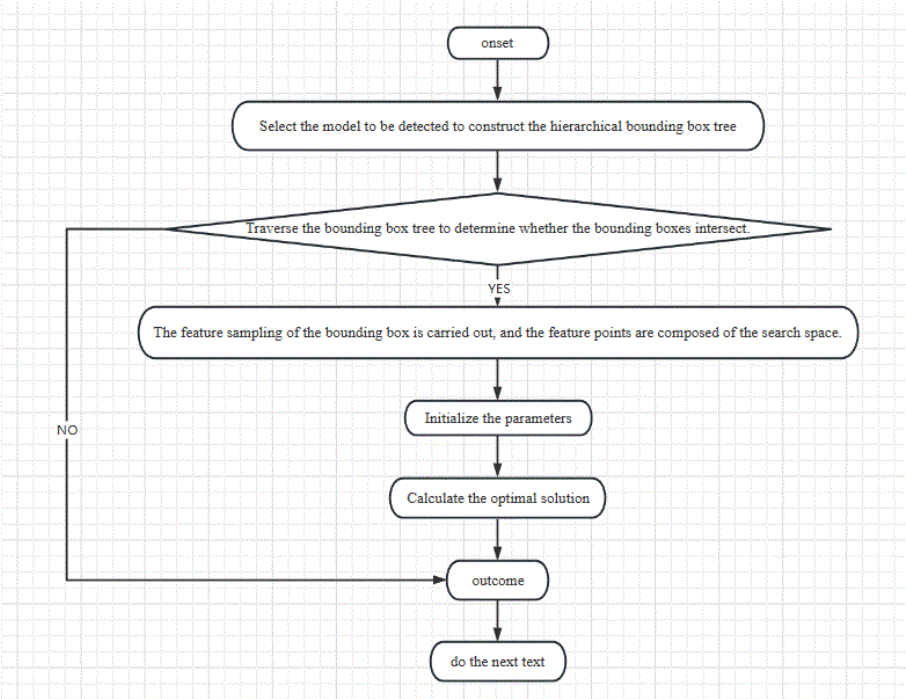

Figure 2. Collision detection algorithm flow chart.

5. Experiment and Results

5.1. Experiment

In order to test the performance of the algorithm proposed in this study for detecting collisions in virtual reality, simulation experiments were carried out. The experimental environment was a PC (version 2019; Visual Studio, Seattle, WA, USA). The OpenGL graphics library and C++ were implemented on VS2019. The simulation model was a three-dimensional model of the equipment in a veneer production line. The number of models was 90, and the average number of vertices in the object model was 750. Firstly, the performance of the sphere-OBB-MACO algorithm in terms of detecting collisions under different feature pairs was simulated. Secondly, the performance of Sphere-OBB-MACO algorithm in terms of detecting collisions with different ant colony sizes was compared, so that the appropriate ant colony size could be selected. Finally, the performance of the most commonly used virtual reality collision detection algorithms and the proposed algorithm was compared. The number of simulations for detecting collisions was 300.

(1) Wang Yu believes that the number of feature pairs is an important factor affecting the performance of collision detection algorithms and has done a series of experiments to verify his conjecture [29]. The performance of the sphere-OBB-MACO algorithm in terms of detecting collisions under different feature pairs was simulated. Five feature pairs were selected to detect collisions using the sphere-OBB-MACO algorithm. The two objects with the longest motion time were selected for detection. Table 1 shows the time of collision detection.

From Table 1, it can be seen that for detecting a collision between two objects, the number of different sampling features of the selected two objects was directly related to the number of feature pairs under the same accuracy in detecting collisions. At the same detection rate, a greater number of sampling feature pairs for two objects means that more feature quantities need to be iterated, so the process is more time-consuming. For example, when the number of feature pairs is 5000^2, it takes more time than when the number of feature pairs is 1000^2. However, the large quantity of features can better reflect the spatial

position attributes of the two objects, which is more conducive to accurate detection of a collision. With the same number of feature pairs, the longer the iterative optimization time of the sphere-OBB-MACO algorithm is, the more likely it is to achieve higher accuracy in detecting collisions.

Table 1. Detection time of sphere-OBB-MACO algorithm with different feature pairs.

	Duration/ms			
Number of Characteristic Pairs	**Detection Rate 20%**	**Detection Rate 40%**	**Detection Rate 60%**	**Detection Rate 80%**
1000^2	5.1	8.4	10.6	13.4
2000^2	7.0	11.1	12.9	16.4
3000^2	9.2	13.4	16.1	20.8
4000^2	12.3	15.6	18.5	22.0
5000^2	14.1	17.2	20.5	27.3

In the following, the detection collision logarithm A and the actual collision logarithm B under different detection rates were compared to verify the influence of the number of feature pairs on the performance of the algorithm for detecting collisions in virtual reality.

It can be seen from Table 2 that there is a certain gap between the logarithm of collisions obtained by the prediction of collisions based on the number of sampled features and the logarithm of actual collisions of the object in virtual reality. When the number of features is greater, the features involved in training are closer to the actual features of the object, and the logarithm of detected collisions is closer to the actual value. When the number of feature pairs is 1000^2, although the sphere-OBB-MACO algorithm could detect 80% of the collisions with 1000 feature pairs, these collisions only accounted for 30.2% of the actual total number of collisions. In the process of evaluating the collisions of the objects, it may cause misjudgment. Therefore, the number of feature pairs sampled by objects in virtual reality has a significant impact on the accuracy of detecting collisions between objects.

Table 2. The collision detection ratio of sphere-OBB-MACO algorithm with different feature pairs.

	(A/B) %			
Number of Characteristic Pairs	**Detection Rate 20%**	**Detection Rate 40%**	**Detection Rate 60%**	**Detection Rate 80%**
1000^2	7.3	15.2	22.1	30.2
2000^2	10	21.2	34.3	45.6
3000^2	13.3	27.8	39.6	51.2
4000^2	16.2	35.9	55.1	62.7
5000^2	20.0	44.1	65.4	85.3

If we combine the results of Tables 1 and 2, the number of feature pairs of objects sampled in virtual reality had a significant impact on the collision detection time of the sphere-OBB-MACO algorithm and the accuracy of the algorithm in detecting collisions. Too high a sampling number will inevitably lead to a decrease in real-time detection, and too low a sampling number will lead to a decrease in the rate of detection. Therefore, in actual operation, in order to ensure the accurate detection of collisions, an appropriate number of sampling feature pairs should be selected.

(2) In order to verify the impact of the ant colony's size on the performance of the collision algorithm, different ant colony sizes were selected, and the sphere-OBB-MACO algorithm was used for training on collisions of the equipment of a veneer production line.

It can be seen from Figure 3 that different sizes of the ant colony had a great influence on the detection rate of the algorithm, and the detection rate maintained the law of decreasing first and then increasing. When the number of feature pairs was 1000^2, the

highest detection rate could be obtained if the number of ant colonies was 40. When the number of feature pairs was 2000^2 and 3000^2, the highest detection rate could be obtained if the number of ant colonies was 50. The highest detection rate could be obtained when the number of feature pairs was 4000^2 and 5000^2 if the number of ant colonies was 60.

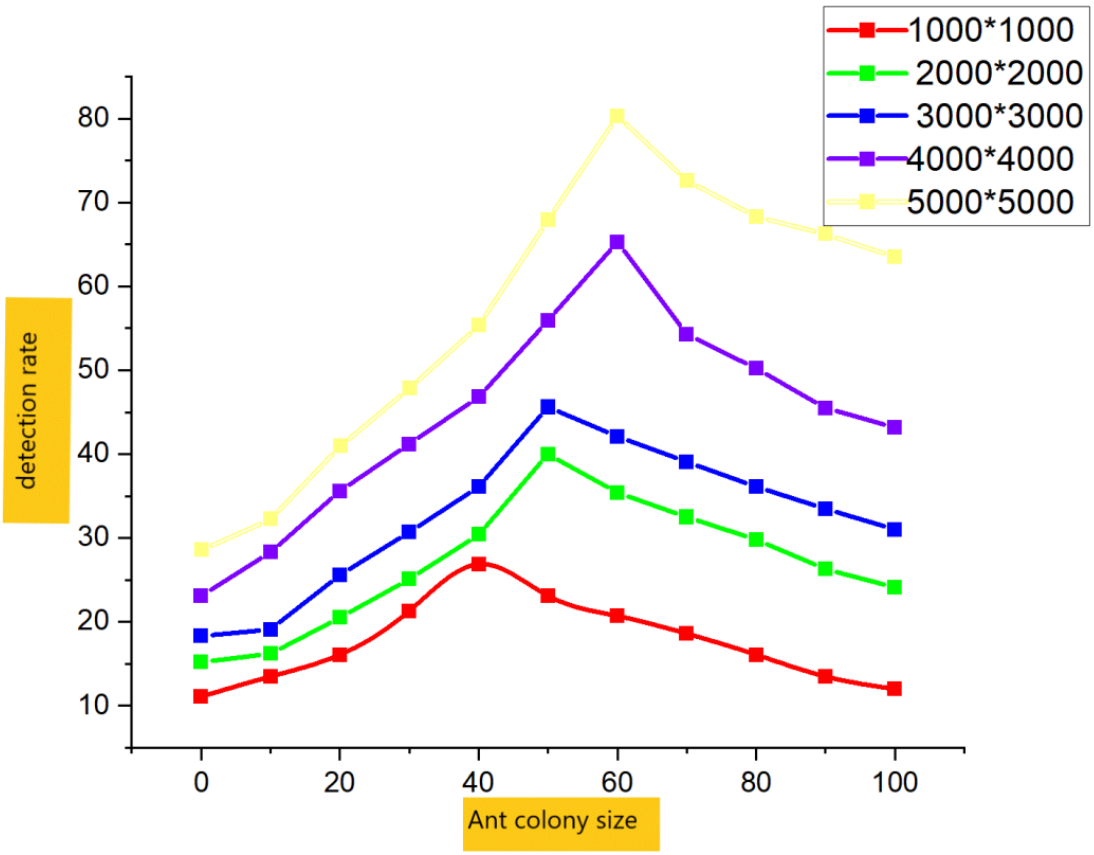

Figure 3. Collision detection rate of different ant colony size.

(3) We selected the equipment of a veneer production line for the problem of detecting collisions. The number of feature pairs was 5000^2, the ant colony size m = 60, the initial pheromone influence factor and the expected heuristic factor $\alpha = \beta = 1$, the maximum number of iterations N = 100 and the set value n = 50. Three algorithms, namely sphere-OBB (the hierarchical bounding box algorithm), sphere-OBB-ACO and sphere-OBB-MACO, were compared and analyzed, and the final results are shown in Figure 4.

From Figure 4, it can be seen that all three algorithms achieved stable results after 30 ms. Among them, the detection rate of the sphere-OBB-MACO algorithm was the highest (about 0.85) and sphere-OBB was the worst (about 0.65).

The experimental results showed that the optimized sphere-OBB-MACO collision detection algorithm was better than the traditional bounding box algorithm and the traditional random collision detection algorithm.

(4) In the fourth experiment, Smagulovak's BOX-LSTM algorithm [30], Shen Xue Li's BOX-PSO algorithm [31], JIN Yan-Xia's BOX-DNN algorithm [32] were selected for comparison with the algorithm designed in this paper so as to judge whether the algorithm designed in this paper was superior to the algorithms designed by other researchers.

According to Figure 5, the four collision detection algorithms all obtained stable detection results within 55 ms. Among them, the detection rate of the sphere-OBB-MACO algorithm was the highest (about 0.85), followed by the BOX-DNN algorithm (about 0.82) and the BOX-PSO algorithm (about 0.8); BOX-LSTM was the worst (about 0.76). From

the perspective of the convergence time of detection, the BOX-LSTM algorithm had the shortest detection time of about 42 ms, while the BOX-DNN was the most time-consuming (about 50 ms), and the BOX-PSO and sphere-OBB-MACO algorithms were in the middle (about 45 ms).

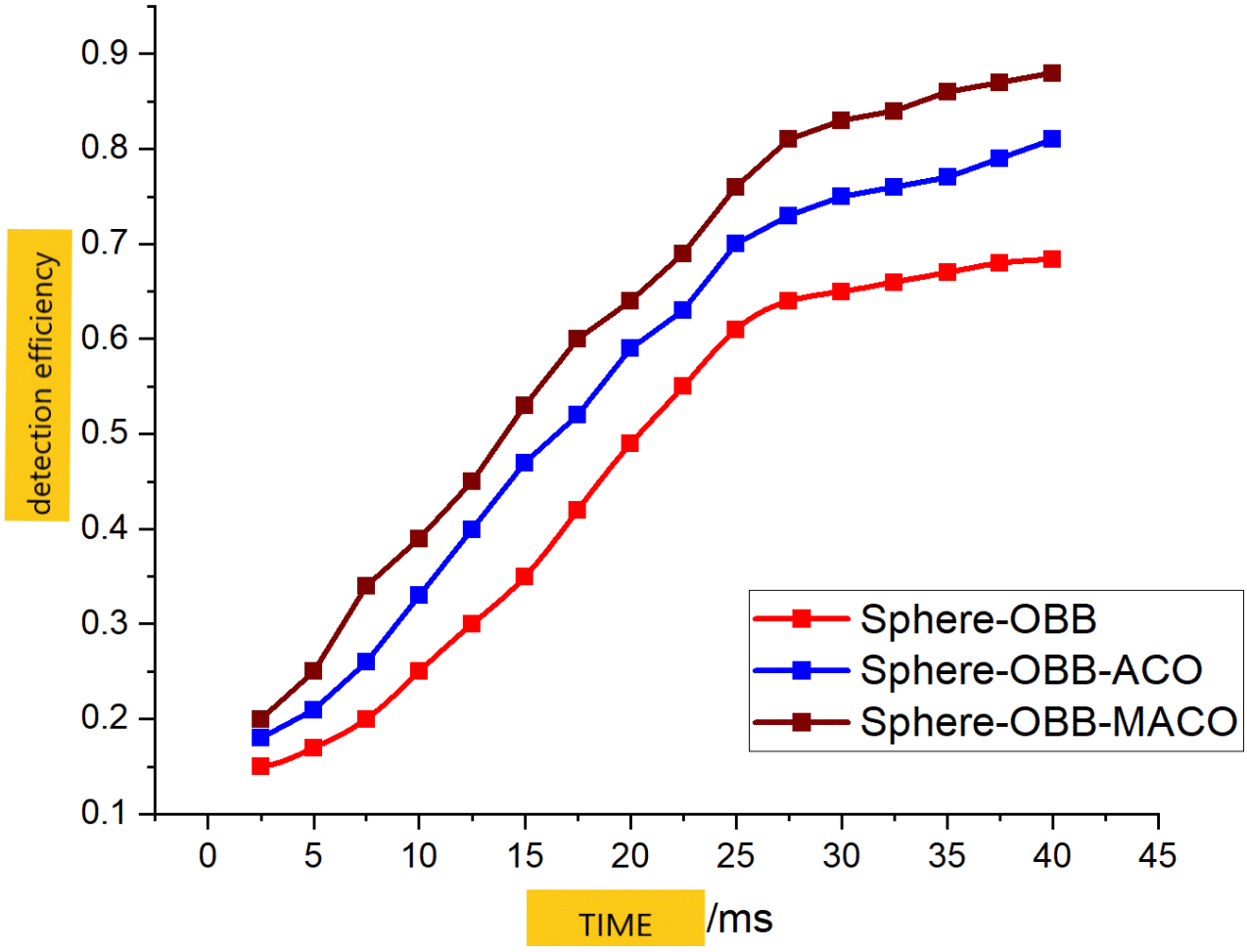

Figure 4. The detection efficiency of three algorithms.

5.2. Analysis of the Results

As described in the previous section, it was verified by experiments that the algorithm designed in this study was able to detect collisions in complex scenes and was also superior to other algorithms. In the first experiment, five feature pairs were selected to test the algorithm. It can be seen from the results that as the number of feature pairs increased, the time was longer. When the efficiency of detection was 80%, the algorithm had a maximum difference of 13.9 ms. However, after comparing the logarithm of detected collisions with the logarithm of actual collisions, it was found that the greater the number of feature pairs, the higher the accuracy of detecting collisions. In the second experiment, different ant colony sizes and sampling feature pairs were selected for collision testing. The results showed that the ant colony size had a significant effect on the rate of detecting collisions, and the detection rate had the shape of first rising and then falling. When the number of feature pairs was 1000^2, the algorithm achieved the highest detection rate when the ant colony's size was 40. When the number of feature pairs was 2000^2 and 3000^2, the algorithm achieved the highest detection rate when the ant colony's size was 50. When the number of feature pairs was 4000^2 and 5000^2, the algorithm achieved the highest detection rate when the ant colony's size was 60. In the third experiment, the algorithm designed in this study was compared with two traditional algorithms. The results showed

that the algorithm designed in this study was always better than the traditional collision detection algorithms. The accuracy of detection by the traditional hierarchical bounding box algorithm (sphere-OBB) could only reach 0.65. The accuracy of detection by the ant colony algorithm sphere-OBB-ACO before optimization and the optimized ant colony algorithm sphere-OBB-MACO was 0.8 and 0.85, respectively. In the fourth experiment, the algorithm developed in this study was compared with three published algorithms. The three algorithms are similar to those of this study. They all introduced intelligent algorithms to solve optimization problems. The results showed that the algorithm from this study had the highest accuracy for detecting collisions in complex scenes.

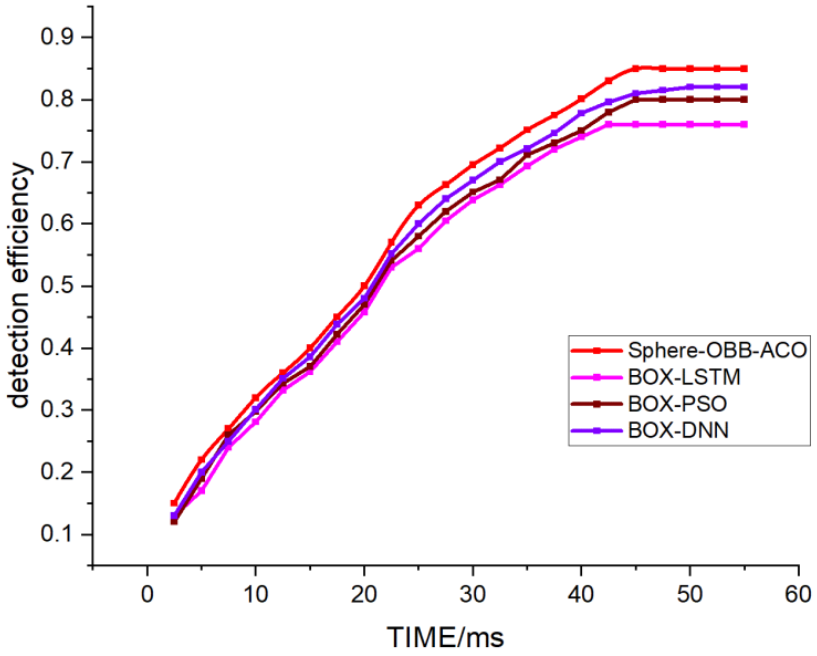

Figure 5. The detection efficiency of four algorithms.

From the experimental results, it can be seen that it is rational to introduce intelligent algorithms to deal with the problem of detecting collisions, as these can effectively improve the efficiency of detection. However, due to the limitations of the intelligent algorithm itself, researchers need to optimize the intelligent algorithm, so as to further improve the efficiency of the collision detection algorithm. The results of the fourth experiment showed that the ant colony algorithm was more suitable for dealing with the collision detection problem of complex scenes. Therefore, when dealing with the collision detection problem under different conditions, researchers need to flexibly use different algorithms to deal with the problem.

6. Conclusions

This study designed, implemented and verified a new collision detection algorithm with good performance, namely sphere-OBB-MACO, which is suitable for detecting collisions in complex scenes. The detection process of the algorithm is divided into two steps. In the preliminary detection stage, the spherical and OBB bounding boxes are selected, the binary tree is selected as the tree structure and the bounding volume hierarchy is constructed in a top-down manner. The tree's structure is traversed by the depth-first traversal method to quickly remove the triangular primitive pairs without collisions. In the accurate detection stage, the bounding boxes with collisions after preliminary detection are processed by the random sampling method, and the feature pairs are used to form the search space. The ant colony algorithm is introduced to solve the optimization problem of the search space. To overcome the problems of a slow convergence speed and the local

optimal solution in the ant colony algorithm, the pheromone update rule of the ant colony algorithm has been optimized.

In the fifth section, the superiority of the algorithm designed in this study was verified. The sphere-OBB-MACO algorithm designed in this study was more efficient in detecting collisions than traditional algorithms and some published algorithms in large-scale complex scenes, but it had some limitations as follows:

(1) Detecting collisions of flexible objects. The improved algorithm in this study is limited to rigid objects, but the detection of collisions between flexible objects is more difficult.
(2) In a simple scenario, due to the small number of models and the simple structure, particle swarm optimization could be used to solve the optimization problem. For detecting collisions between convex polyhedrons, a faster genetic algorithm should be used to solve the optimization problem.

Although this study has made some improvements to the collision detection algorithm and achieved certain results in terms of efficiency, there are still many problems that need to be further studied in the future as follows:

(1) Optimization of the complexity of the hierarchical tree's structure space. The sphere-OBB structure requires more storage space than a single bounding box structure. In the future, the storage space of the hierarchical tree needs to be optimized.
(2) Optimization of ant colony algorithm. In this study, the pheromone update rules and the correlation coefficients of the ant colony algorithm were improved to optimize the ant colony algorithm. In the future, other intelligent algorithms could be introduced to optimize the ant colony algorithm.
(3) In the future, the acceleration of computer hardware could be studied to further improve the efficiency of detection.

Author Contributions: Conceptualization, P.X.; methodology, P.X.; software, Q.S.; validation, P.X.; formal analysis, P.X.; investigation, Q.S.; resources, Q.S.; data curation, P.X.; writing—original draft preparation, P.X.; writing—review and editing, P.X.; visualization, P.X.; supervision, Q.S.; project administration, Q.S.; funding acquisition, Q.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhang, C.; Mu, C.; Wang, Y.; Li, J.; Liu, Z. Collision detection for six-DOF serial robots force/position hybrid control based on continuous friction model. *Meas. Control.* **2023**, *56*, 571–582. [CrossRef]
2. Song, S.; Lan, L.; Yao, J.; Guo, X. Continuous collision detection with medial axis transform for rigid body simulation. *Commun. Inf. Syst.* **2022**, *22*, 53–78. [CrossRef]
3. Wu, D.; Yu, Z.; Adili, A.; Zhao, F. A Self-Collision Detection Algorithm of a Dual-Manipulator System Based on GJK and Deep Learning. *Sensors* **2023**, *23*, 523. [CrossRef] [PubMed]
4. Dyllong, E.; Luther, W. The GJK Distance Algorithm: An Interval Version for Incremental Motions. *Numer. Algorithms* **2004**, *37*, 127–136. [CrossRef]
5. Hu, A.; He, Y. Research on hybrid collision detection algorithm based on separation distance. *J. Phys. Conf. Ser.* **2022**, *2258*, 012011. [CrossRef]
6. Meister, D.; Ogaki, S.; Benthin, C.; Doyle, M.J.; Guthe, M.; Bittner, J. A Survey on Bounding Volume Hierarchies for Ray Tracing. *Comput. Graph. Forum* **2021**, *2*, 683–712. [CrossRef]
7. Gandotra, S.; Pungotra, H.; Moudgil, P.K. Representation of model for efficient collision detection in virtual reality environment. *Int. J. Precis. Technol.* **2020**, *9*, 335. [CrossRef]
8. Jin, Y.; Geng, J.; He, Z.; Lv, C.; Zhao, T. A capsule-based collision detection approach of irregular objects in virtual maintenance. *Assem. Autom.* **2021**, *41*, 89–105. [CrossRef]

9. Mavrovouniotis, M.; Yang, S. Ant algorithms with immigrants schemes for the dynamic vehicle routing problem. *Inf. Sci.* **2015**, *294*, 456–477. [CrossRef]
10. Adery, L.H.; Ichinose, M.; Torregrossa, L.J.; Wade, J.; Nichols, H.; Bekele, E.; Bian, D.; Gizdic, A.; Granholm, E.; Sarkar, N.; et al. The acceptability and feasibility of a novel virtual reality based social skills training game for schizophrenia: Preliminary findings. *Psychiatry Res.* **2018**, *270*, 496–502. [CrossRef]
11. Ding, X.J. Research on Collision Detection Algorithm Based on OBB. *Appl. Mech. Mater.* **2013**, *2755*, 433–435. [CrossRef]
12. Xiong, Y.M.; Chen, Y.M.; Chen, Y.H. Research on Bounding Box-Tree Algorithm for Collision Detection. *Adv. Mater. Res.* **2011**, *186*, 645–649. [CrossRef]
13. Shang, Y.; Wang, H.; Qin, W.; Wang, Q.; Liu, H.; Yin, Y.; Song, Z.; Meng, Z. Design and Test of Obstacle Detection and Harvester Pre-Collision System Based on 2D Lidar. *Agronomy* **2023**, *13*, 388. [CrossRef]
14. Hwang, A.D.; Peli, E.; Jung, J.H. Development of Virtual Reality Walking Collision Detection Test on Head-mounted display. In Proceedings of the SPIE—The International Society for Optical Engineering, San Francisco, CA, USA, 30 January–2 February 2023; p. 12449.
15. Zhang, X.; Liu, J. Research on Collision Detection Algorithm for Human-SRL Collaborative Motion Planning. *J. Phys. Conf. Ser.* **2022**, *2402*, 012021. [CrossRef]
16. Hao, T.; Liu, Y.; Gong, X.; Kong, D.; Wang, J. Visibility Detection of 3D Objects and Visual K-Nearest Neighbor Query Based on Convex Hull Model. *Math. Probl. Eng.* **2022**, *2022*, 8302974. [CrossRef]
17. Hariyono, J.; Kurnianggoro, L.; Jo, K.H. Analysis of Pedestrian Collision Risk using Fuzzy Inference Model. In Proceedings of the 16th International Conference on Control, Automation and Systems (ICCAS), Gyeongju, Kore, 16–19 October 2016. 제어로봇시스템학회 국제학술대회 논문집.
18. Huang, Z.; Yang, X.; Min, J.; Wang, H.; Wei, P. Collision detection algorithm on abrasive belt grinding blisk based on improved octree segmentation. *Int. J. Adv. Manuf. Technol.* **2021**, *118*, 11–12. [CrossRef]
19. Cheng, S.; Qu, H. Key Issues of Real-time Collision Detection in Virtual Reality. *Int. J. Front. Eng. Technol.* **2021**, *3*, 43–49.
20. Hu, Y.; Yan, Z.; Yin, Z.; Du, Z. Collision detection based on octree for virtual surgery system. *IOP Conf. Ser. Mater. Sci. Eng.* **2020**, *7*, 072107. [CrossRef]
21. Wang, M.; Mao, Z.; Ma, Y.; Cao, J. Stable and efficient collision detection scheme for hip-surgery training system. *Clust. Comput. J. Netw. Softw. Tools Appl.* **2019**, *22*, 8769–8781. [CrossRef]
22. Kim, M.; Sung, N.J.; Kim, S.J.; Choi, Y.J.; Hong, M. Parallel cloth simulation with effective collision detection for interactive AR application. *Multimed. Tools Appl.* **2019**, *78*, 4851–4868. [CrossRef]
23. Giang, T.; O'sullivan, C. Approximate collision response using closest feature maps. *Comput. Graph.* **2006**, *30*, 423–431. [CrossRef]
24. Wong, W.; Liu, W.; Bennamoun, M. Tree-Traversing Ant Algorithm for term clustering based on featureless similarities. *Data Min. Knowl. Discov.* **2007**, *15*, 349–381. [CrossRef]
25. Dorigo, V.M. Maniezzo, Distributed Optimization by Ant Colonies. In Proceedings of the 1st European Conference on Artificial Life, Paris, France, 11–13 December 1991; Elsevier Publishing: Amsterdam, The Netherlands, 1991; pp. 134–142.
26. Su, Y.; Bai, Z.; Xie, D. The optimizing resource allocation and task scheduling based on cloud computing and Ant Colony Optimization Algorithm. *J. Ambient. Intell. Humaniz. Comput.* **2021**. [CrossRef]
27. Zhang, H.; Gao, Y. Solving TSP based on an Improved Ant Colony Optimization Algorithm. *J. Phys. Conf. Ser.* **2021**, *1982*, 012061. [CrossRef]
28. Gao, W. New Ant Colony Optimization Algorithm for the Traveling Salesman Problem. *Int. J. Comput. Intell. Syst.* **2020**, *13*, 44–55. [CrossRef]
29. Yu, W.; Linjuan, M.; Fuquan, Z. Virtual reality collision detection method based on quantum ant colony algorithm. *J. Nanjing Univ. Sci. Technol.* **2022**, *46*, 735–741. [CrossRef]
30. Smagulova, K.; James, A.P. Asurvey on LSTM memristive neural network architectures and applications. *Eur. Phys. J. Spec. Top.* **2019**, *228*, 2312–2324. [CrossRef]
31. Shen, X.; Zhang, J.S. Research of collision detection algorithm based on particle swarm optimization. *Int. Conf. Comput. Des. Appl.* **2010**, *1*, V1-60–V1-63.
32. Jin, Y.; Cheng, Q.; Zhang, J. Fusion of DNN and AABB-circular bounding box self-collision detection. *Chin. J. Image Graph.* **2020**, 1674–1683.