Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2016, Article ID 6340856, 12 pages
http://dx.doi.org/10.1155/2016/6340856

Research Article

Continuous Visible Query for Three-Dimensional Objects in Spatial Databases

Yongshan Liu and Dehan Kong

Department of Information Science and Engineering, Yanshan University, Hebei Street No. 438, Qinhuangdao 066000, China

Correspondence should be addressed to Dehan Kong; kdh0312@163.com

Received 27 April 2016; Revised 8 July 2016; Accepted 11 July 2016

Academic Editor: Romain Aubry

Present research of visible query focuses on points and segments in two-dimensional space, while disfigurements occur during processing of visible query in three-dimensional space. In this paper, Continuous Visible Range Query Based on Control Point (CVRQ-CP) is proposed to solve the visible query in a 3D spatial database. Firstly, the horizontal angle (HA) and Vertical Projection Angle (VPA) for 3D objects in a spatial database were used in the visibility testing method. The HA and VPA in the processing of the continuous visible query created visibility changes, defining and confirming the control point. Finally, the algorithm of Continuous Visible Range Query Based on Control Point (CVRQ-CP) was proposed. Verified by experiments, the CVRQ-CP algorithm correctly deals with the visible query of 3D spatial objects. The CVRQ-CP algorithm has better superior accuracy over present visible queries in 3D spatial databases.

1. Introduction

Continuous query is a kind of common spatial query which is widely used in Location-Based Service (LBS). Some researchers study the continuous query in obstacle space which is named as *visible query*. Visible queries are widely used in the field of assistant decision, navigation, and security monitoring. Although researches into visible query as VkNN [1], CVkNN [2], and VRkNN [3] are numerous, they focus on two-dimensional (2D) space as points and segments. In recent years, three-dimensional space applications such as online games and virtual reality applications have become increasingly common. Present researches are unable to deal with the visibility of three-dimensional (3D) objects correctly because they cannot properly consider altitudinal dimension influence. Meanwhile, methods using double-project in horizontal plane and vertical plane to reduce dimensions are extremely inappropriate and the visible query for 3D objects based on hardware is too expensive to use on a large scale. So there is an urgent need for visible query algorithm in 3D space database. To solve this problem, a method of visibility testing for 3D objects based on horizontal angle (HA) and Vertical Projection Angle (VPA) is proposed and the processing of Continuous Visible Range Query based on this visibility testing method is presented.

To solve the visible query with 3D objects, this paper designs 3D objects models and presents the method of visibility testing based on these models. Because of the visibility changes between nonparallel 3D objects for a query interval in the processing of the Continuous Visible Range Query (CVRQ), the important definition control point is used to reduce computational complexity. We propose an algorithm of the CVRQ Based on Control Point, and its accuracy and validity are validated.

The remainder of the paper is organized as follows: Section 2 reviews related works and defines the problem to be solved. Section 3 proposes the method of visibility testing on 3D objects. Section 4 defines the control point in CVRQ and proposes the processing of CVRQ Based on Control Point (CVRQ-CP). Section 5 provides the accuracy and validity of visibility testing method and performance of CVRQ-CP algorithm by analyzing the experiment results. Section 6 provides the conclusions and future research plans.

2. Related Work and Definition

2.1. Related Work. Initially, algorithms of visibility testing were applied to the field of virtual simulation. Whether the objects in the same scene are visible to each other is confirmed

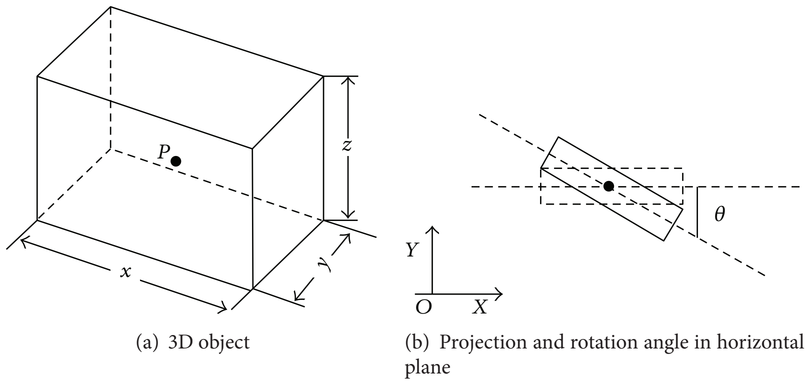


Figure 1: Three-dimensional spatial object in database.

using grid computing. The main algorithms include JANUS [4], DYNTACS [5], ModSAF [6] and the improved algorithm is Bresenham algorithm [7]. These algorithms test visibility based on grid computing and therefore request the scene as an integrated file. This defines the algorithms as load-before-calculate methods. The scale of the scene is greatly limited under this kind of method due to high requirements on the client's hardware and network. In recent years, the spatial database is widely used for virtual scenes but the grid computing pattern is not applied in this background.

Alipour et al. [8] proposed an algorithm using quadratic preprocessing time and space to solve the visibility testing problem. Chen and Wang [9] researched weak visibility queries of line segments in simple polygons. Hershberger [10] constructed the visibility graph in a simple polygon in $O(|E|)$ time, whereas Ghosh and Mount [11] established its construction in $O(n\log n+|E|)$ time for a polygon with holes. Here, $|E|$ is the number of edges in the resulting visibility graph. Nouri Baygi and Ghodsi [12] constructed the visibility graph in a polygonal domain in $O(n^2\log(\sqrt{s}/n)/\sqrt{s}+m_p)$ time. However, these studies did not involve the visibility testing of 3D objects. Currently, there is no appropriate visibility testing algorithm for 3D objects in spatial database. Regarding research of visible query in databases, Zhang et al. [13] proposed an integrated framework that efficiently answers most types of spatial queries in obstacles databases. Nutanong et al. [1] defined a Visible k Nearest Neighbor (VkNN) search with minimum visible distance (MinViDist). The VkNN query incrementally computes the visible neighbors as the search space is enlarged. This brings response time of the query processing up to 35%. Gao et al. [3] introduced a Visible Reverse k Nearest Neighbor (VRkNN) search in a spatial database and proposed an efficient algorithm for VRNN query processing, assuming that P and O are indexed by R-trees. No preprocessing was required, and half-plane property and visibility checks were utilized to prune the search space. Gao et al. [2] developed an efficient continuous visible nearest neighbor (CVNN) algorithm to tackle the CVNN problem by performing a single query for the entire query line segment. Lu et al. [14] used the Voronoi diagram to store and search potential visible objects, improving query speed while increasing the computational complexity. Gu et al. [15] proposed a Continuous k Nearest Neighbor (CkNN) query based on control point to improve CPU efficiency and I/O. Wang et al. [16] achieved Visible k Nearest Neighbor (VkNN) query on moving objects by using periodic queries and R-tree indexes.

All the aforementioned research focuses on 2D spatial databases, and existing algorithms cannot properly deal with the visible query of 3D objects. If we test the visibility of 3D objects by dimension-reduction with double-projection as the present visibility testing method, results will be incorrect due to lack of a proper projection plane for 3D objects with different rotation angles. Thus, this paper addresses the problems of 3D objects visibility testing and query algorithms.

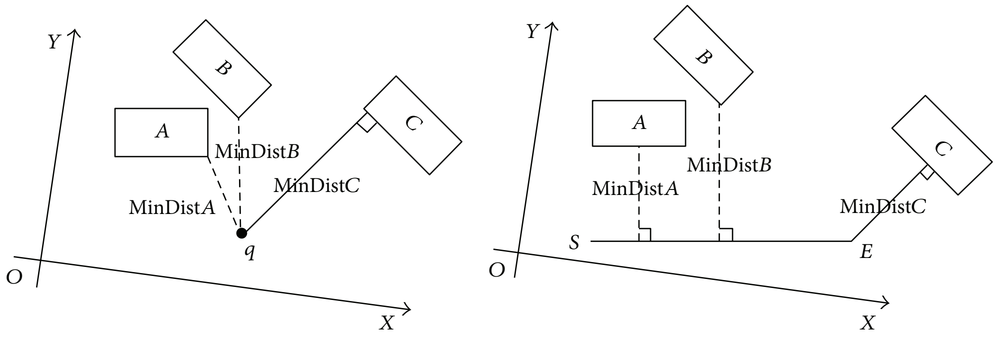
2.2. Definition

Definition 1 (three-dimensional spatial object). A three-dimensional spatial object is a cube $O(P,R,\theta)$ with P as the center of the cube, and $P=(X,Y,Z)$, where X, Y, and Z are the coordinates of P. $R=(L_X,L_Y,L_Z)$, where R is the range of the cube. L_X, L_Y, and L_Z are the ranges in each dimension. θ is the cube's rotation angle in horizontal plane (Figure 1).

In this model, the rotation angle in the vertical plane is zero by default, making the horizontal plane the unified projection plane. With different rotation angles in the horizontal plane, the projections of 3D objects do not have a unified vertical projection plane to reduce dimension through double-projection. Therefore, a new method for visibility testing of 3D objects is needed.

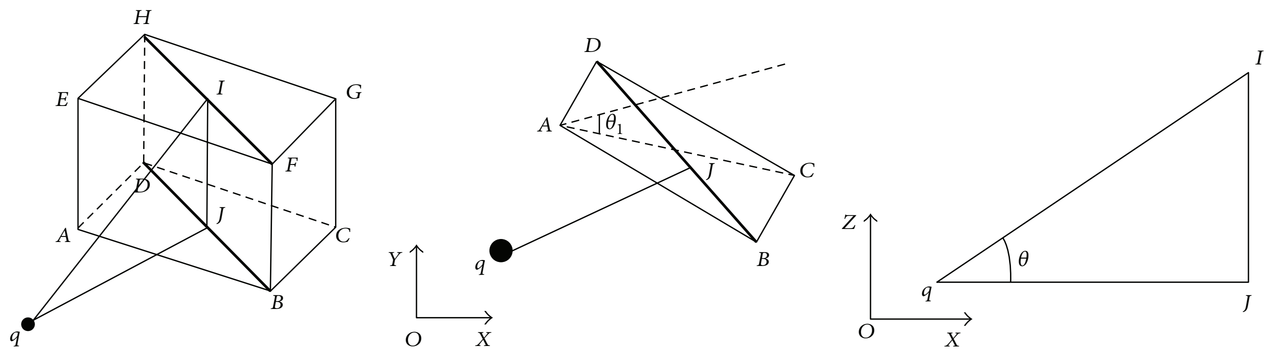
Definition 2 (minimum euclidean distance (MinDist)). If A is the projection of the 3D object in the horizontal plane of query point q, then MinDist is the minimum Euclidean distance between q and A (Figure 2(a)). MinDist between A and a query interval $[S,E]$ is the minimum Euclidean distance between A and any point in $[S,E]$ (Figure 2(b)).

Definition 3 (projection angle, PA). The 3D object is $O(P,R,\theta_1)$ and the query point is q. Rectangle *HDBF* is used to replace $O(P,R,\theta_1)$ during the visibility testing to simplify



(a) MinDist for query point (b) MinDist for query interval

Figure 2: MinDist between a 3D object and query point or query interval.



(a) 3D object $O(A, G, \theta_1)$ (b) Projection of 3D object in horizontal plane (c) Projection angle as θ

Figure 3: Projection in horizontal plane and projection angel in vertical plane.

calculation. At any point J in the segment BD, the highest corresponding point is I (Figure 3(a)). The rectangle $ABCD$ is the projection of $O(P, R, \theta_1)$ in the horizontal plane (Figure 3(b)). Then, φ of $\angle IqJ$ is the projection angle (Figure 3(c)).

The projection angle in the vertical plane is used to describe the influences of height dimension on visibility of 3D objects.

Definition 4 (Visible). The 3D objects are O, an obstacle Ob, and a query point q. The point p ($p \in O$) connects p and q and does not intersect with Ob $\{\exists p \mid p \in O \cap (\overline{pq} \cap Ob = \emptyset)\}$. Then, O is visible to q.

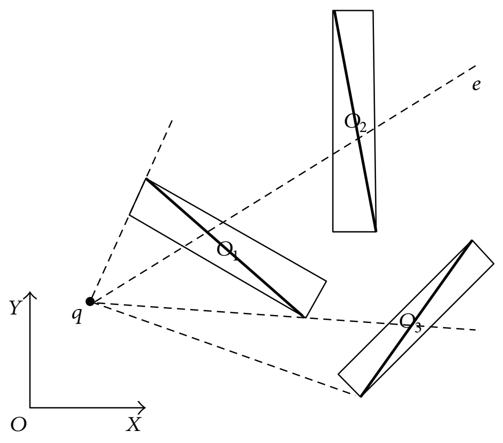
Definition 5 (Continuous Visible Range Query on Three-dimensional Objects). The 3D objects are set as O, an obstacle Ob, and a query interval $q = [S, E]$. CVRQ as $Q([S, E], R)$ is a visibility search with results of this search visible to a subinterval of $[S, E]$ and the MinDist between the result and $[S, E]$ is less than the radius R, and $Q([S, E], R) = \{\forall o \mid o \in O \cap (\overline{oq} \cap Ob = \emptyset) \cap (\text{MinDist}(o, [S, E]) < R)\}$.

By defining these data models, why visible query in 2D could not be directly used in 3D through double-projection is explained.

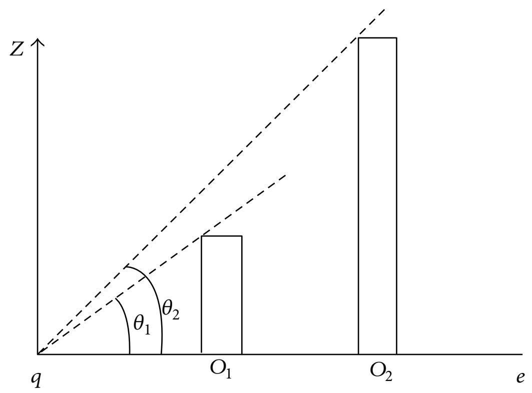
3. Visible Range Query on Three-Dimensional Objects

3.1. Visibility Testing on 3D Objects. The visibility testing on 3D objects is divided into two parts. First, preliminary filter for projection is created in the horizontal plane. Second, visibility testing for the objects, which are completely obscured in the horizontal plane, is conducted.

During the visibility testing in the horizontal projection plane, all 3D objects are projected to the horizontal plane of the query point. The projection of 3D object is described as a segment to simplify calculations (Figure 4(a)). In the first step for visibility testing, 3D objects O_1, O_2, and O_3 are projected to the horizontal plane of query point q. 2D visibility testing of the projections filtered out visible objects. As shown in Figure 4(a), the projections of O_1 and O_3 are visible to q, and therefore, the 3D objects O_1 and O_3 are naturally visible to q. The projection of O_2 is invisible to q in the horizontal plane because it is covered by the projection of O_1, resulting in 3D object O_2 requiring further visibility testing in the vertical plane (the second step for testing). As illustrated in Figure 4(b), if the projection angle of O_2 is larger than O_1, O_2 is visible to q even if its projection is completely obscured in the horizontal plane. This reflects the principle

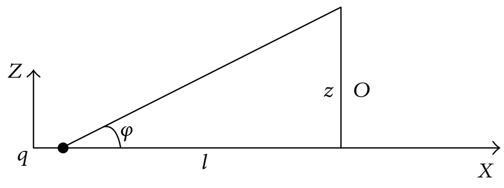


(a) Visibility testing for projections in horizontal plane

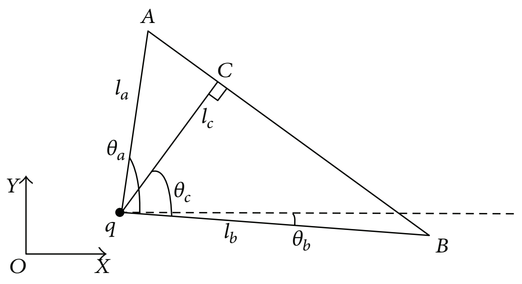


(b) Influence of projection angle for visibility

FIGURE 4: Visibility testing on 3D object.



(a) Calculation of the projection angle



(b) Distance between query point and different points in projection of object

FIGURE 5: Change of distance and projection angle.

that a farther object could not affect the visibility of a closer object. Therefore, to avoid the impact of inserted objects, visibility testing of 3D objects is done in order by MinDist.

The projection angle φ of 3D objects acts as a variable in this second step of visibility testing. The projection angle is related to the distance l between the query point q and the projection of the 3D object in the level plane of query point (Figure 5(a)). The projection angle φ_p is calculated as Formula (1), where z is the height of the 3D object and l_p is the distance between the query point q and the corresponding point p. l_p is calculated as Formula (2), where (x_p, y_p) is the coordinates of point p. The horizontal angle θ_p between p and q is calculated as Formula (3). As shown in Figure 5(b), point C has the minimum distance between object AB and query point q. Therefore, φ_C is the maximum projection angle between AB and q. Then, the coordinate system $\theta - \varphi$ based on θ_p and φ_p is built to describe the visibility of the 3D object using these three formulas. If the projection angle of O_1 is bigger than O_2 in the same horizontal angle, then O_1 is not obscured by O_2. Otherwise, O_1 is invisible:

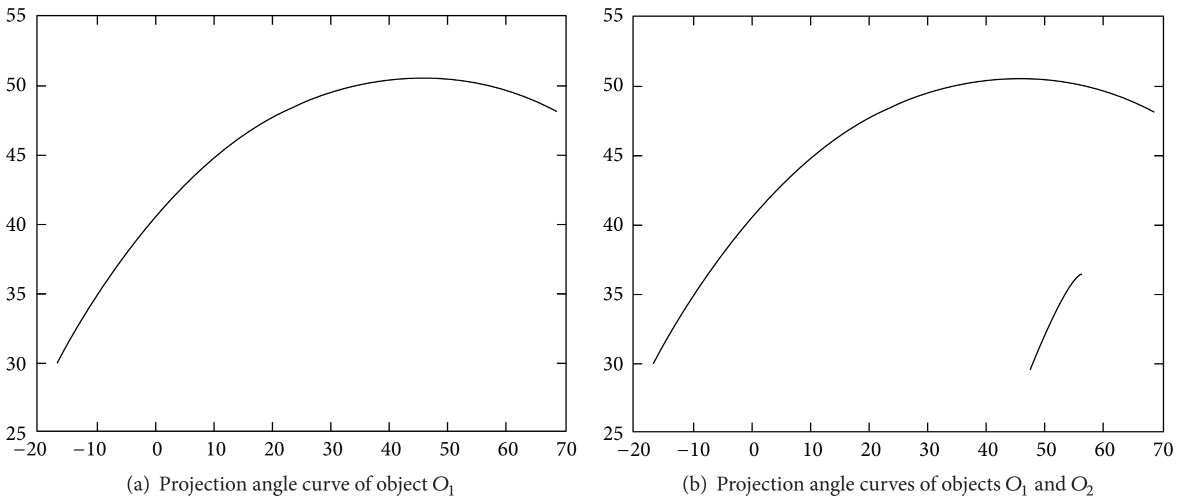
$$\varphi_p = \arctan \frac{z}{l_p}, \tag{1}$$

$$l_p = \sqrt{\left(x_p - x_q\right)^2 + \left(y_p - y_q\right)^2}, \tag{2}$$

$$\theta_p = \arctan \frac{y_p - y_q}{x_p - x_q}. \tag{3}$$

The function curve of the projection angle as calculated using Formulas (1) and (2) is shown in Figure 6(a). The x-axis is θ and the y-axis is φ, and the curve is of object O. In the same way, the curves of O_1 and O_2 are illustrated in Figure 6(b). The visibility of 3D objects is calculated by comparing the projection angle in the same horizontal angle. The curves for objects O_1 and O_2 in Figure 4(a) are shown in Figure 6(b) and $\varphi_{O_2} < \varphi_{O_1}$. Therefore, O_2 is obstructed by O_1 all the time, and O_2 is invisible to query point q.

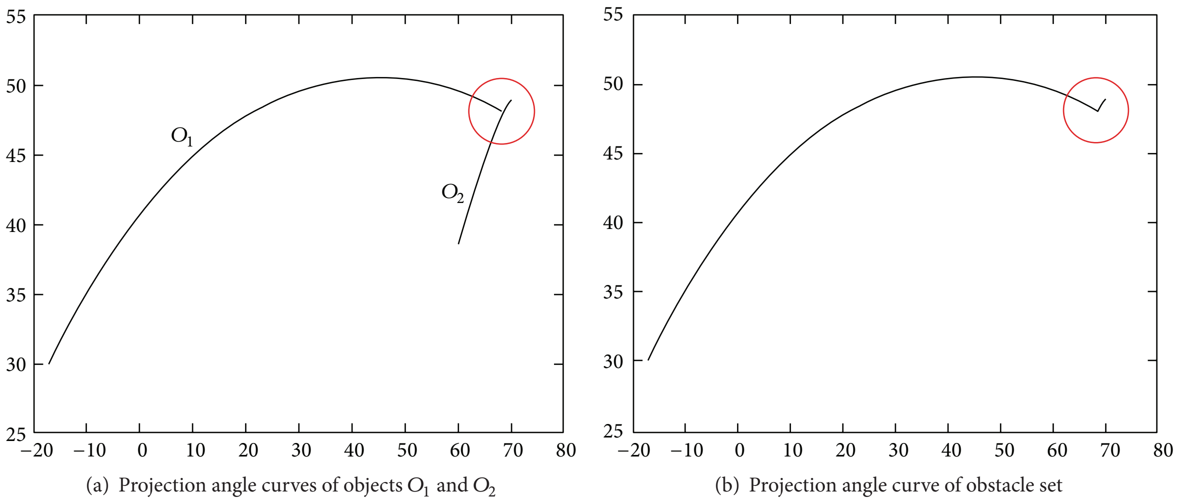
3.2. Visible Range Query in 3D Objects. According to the visibility testing method based on the horizontal and projection angles, the algorithm of Visible Range Query (VRQ) in 3D object is proposed. In VRQ, the first step is to use a range query to determine involved objects. Then, the primary object set is sorted using MinDist. Finally, the visibility of the primary objects is calculated one by one. If an object is visible, its added to the result set, and the obstacle set is updated. Otherwise, it must be covered by other objects in the obstacle set. This process is looped until the entire primary set is completely processed.



(a) Projection angle curve of object O_1

(b) Projection angle curves of objects O_1 and O_2

FIGURE 6: Curve of projection angle.



(a) Projection angle curves of objects O_1 and O_2

(b) Projection angle curve of obstacle set

FIGURE 7: Curve of obstacle set.

The obstacle set creates an obstacle curve in the $\theta - \varphi$ system (Figure 7(a)). The point in the curve is the maximum projection angle in the corresponding horizontal angle. For example, O_2 is inserted into the obstacle set, and the obstacle curve is updated (Figure 7(b)).

According to these steps, the VCR algorithm is proposed as shown in Algorithm 1.

In line (1), the range query obtains the primary set of objects. Then, the set is sorted by MinDist. In line (3), the visibility testing of the objects is processed through the loop. In line (4), the primary filter is done in horizontal plane. If the projection of the object is not covered then the object is visible. Otherwise, further filtering based on the projection angle is conducted for more accurate visibility testing. Next, the projection angle curve in the $\theta - \varphi$ system is calculated, the resulting curve is compared to the obstacle curve, and the visibility of the object is calculated. After processing all objects, the result set is returned in line (11).

4. Continuous Visible Range Query

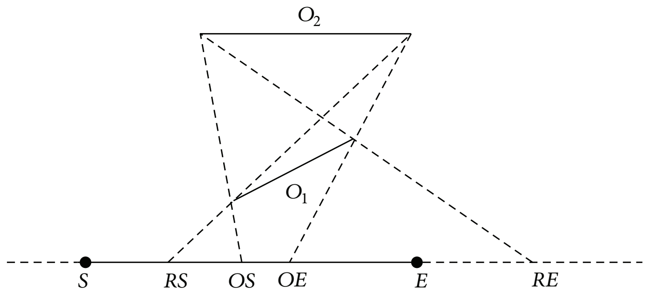
4.1. Continuous Visible Range Query. Continuous Visible Range Query (CVRQ) is a range query and a visible query. The primary set of 3D objects is obtained by the range query, and the resulting set is calculated from the primary set through visibility testing.

Let us take a CVRQ with radius R as an example. In step one, the entire query interval is divided into a group of segments, and the range query for each segment determines the primary set of 3D objects. Then, the primary set is sorted and visible objects are obtained by visibility testing.

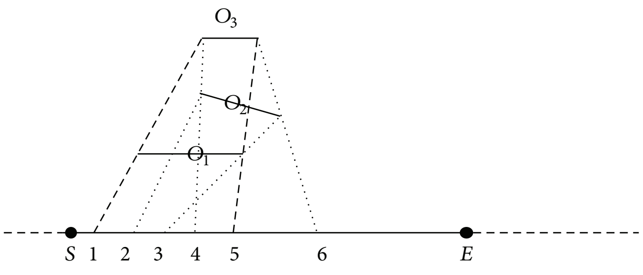
In the CVRQ, the visibility testing is a segment between the objects and query interval (Figure 8(a)). The visibility testing of segments is divided into two parts: one in a horizontal plane, and the other in a vertical plane. Only the projection angle of objects completely covered in the horizontal plane needs to be calculated in visibility testing.

```
Input: ObjectSet OS; QueryPoint q; Range r
Output: ResultSet RS
(1) Object = RangeQuery(q, r, OS); //do the range query, object.MinDist ≤ r
(2) Sorting(Object);
(3) for (every element in Object) do
(4)   if Boolean(object.i, obstructSet(θ_max, θ_min)) then
(5)     Add object.i(object.i, θ_imax, θ_imin) into obstructSet;
(6)     Add object.i into RS;
(7)   else
(8)     Boolean(object.i, obstructSet(θ_imax, θ_imin))
(9)     if object.i visible then
(10)      Add object.i into RS;
(11)        Add object.i(object.i, θ_imax, θ_imin) into obstructSet;
(12)      end if
(13)    end if
(14) end for
(15) Return RS;
```

Algorithm 1: Visible range query.



(a) Visibility testing of segment



(b) Visible area

Figure 8: Visibility testing of query interval.

Table 1: Visible area of inserted object.

Object	Obstacle	VA	IA
O_1	$\emptyset$	$[S, E]$	$\emptyset$
O_2	O_1	$[S, 2] \cup [3, E]$	$[2, 3]$
O_3	O_1, O_2	$[S, 1] \cup [6, E]$	$[1, 6]$

As an example to introduce the visibility testing of segments, segment $[S, E]$ is the query interval (Figure 8(a)). The visible areas results are shown in Table 1. O_1 and O_2 are the projections of 3D objects in the horizontal plane. Since $\text{MinDist}\, O_1 < \text{MinDist}\, O_2$, O_1 is in the obstacle set when O_2 is inserted. Based on the terminals of O_1 and O_2, $[S, E]$ is divided into three subintervals: $[S, OS]$, $[OS, OE]$, and $[OE, E]$. $[S, OS] \cap [OE, E]$ is the Visible Area (VA) and is described as $\text{VA}(O_2, O_1) = [S, OS] \cap [OE, E]$, and $[OS, OE]$ is the Invisible Area (IA) and is described as $\text{IA}(O_2, O_1) = [OS, OE]$. The VA of object is the complement set of the IA as shown in Formula (4). O_1 and O_2 are in the obstacle set before inserting O_3 (Figure 8(b)). The IA of object to the query interval is the union set of all IAs between object and obstacle objects as shown in Formula (5). $\text{IA}(O_3, O_1) = [1, 4]$ and $\text{IA}(O_3, O_2) = [4, 6]$, so $\text{IA}(O_3) = [1, 4] \cap [4, 6] = [1, 6]$. Then, $\text{VA}(O_3) = [S, E] - \text{IA}(O_3) = [S, 1] \cap [6, E]$ as shown in Formula (6):

$$[S, E] = \text{VA}\left(O_i, O_j\right) + \text{IA}\left(O_i, O_j\right), \tag{4}$$

$$\text{IA}\left(O_i\right) = \bigcup_{j}^{i-1} \text{IA}\left(O_i, O_j\right), \tag{5}$$

$$\text{VA}\left(O_i\right) = [S, E] - \text{IA}\left(O_i\right) = [S, E] - \bigcup_{j}^{i-1} \text{IA}\left(O_i, O_j\right). \tag{6}$$

4.2. Point Change and Control Point. There is a distinctive visibility change during continuous visible query on 3D objects. The distances between the query point and 3D objects constantly change, along with the projection angles. This may cause the obstacle relationships between objects to also change. For example, when we walk along the street, a building may disappear behind another building. In this paper, the *PointChange* and *ControlPoint* are defined as follows.

Definition 6 (Point Change (PC)). Point Change is an end point of the projection of 3D object O_2. If the visibility of

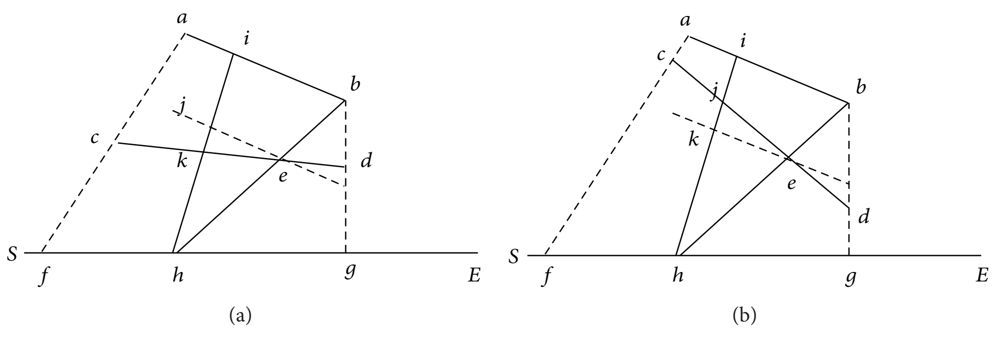


Figure 9: Point Change.

O_2 projection as ab changes in the obstacle area $\mathrm{OA}(O_2, O_1)$, then when the query point moves from Point S to point E in the query interval, PC is the earliest, or latest, visible point in projection of O_2.

Definition 7 (control point (CP)). Control point is a point in obstacle area $\mathrm{OA}(O_2, O_1)$ that divides the $\mathrm{OA}(O_2, O_1)$ into two different parts. In one part, O_2 is visible, and in the other part it is invisible because of O_1.

When b is just visible to the query point in the query interval, $hb/he = Z_{ab}/Z_{cd}$, where Z is the height dimension of the 3D object (Figure 9(a)). Since je is parallel to ab, $hj/hi = Z_{ab}/Z_{cd}$. Therefore, $\varphi_{hk} > \varphi_{hi}$ due to $hj > hk$. The other points in O_2 are invisible, so point b is the Point Change. Similarly, point a is PC in Figure 9(b).

Conclusively, the PC is related to the relationship between objects' rotation angles in horizontal plane. When $\omega_{O_1} > \omega_{O_2}$, the PC is the point which is the closest point in the projection of 3D object in the horizontal plane. Otherwise, the PC is the farthest point.

The trace of query is a query interval $[S, E]$ during Continuous Visible Range Query (CVRQ) on 3D objects. The projection angle between the object and the query interval is a continuous variable, and the visibility of object may change. For example, O_1 is invisible to query point S due to O_2, but with travel from point S to point E, the distance between O_1 and the query interval is shortened while the distance between O_2 and the query interval is lengthened. Therefore, the projection angle between O_1 and the query interval decreases while the projection angle between O_2 and the query interval increases. However, the visible relationship between O_2 and the query interval may change if φ_{O_1} is larger than φ_{O_2} at a certain point in the query interval. That point is a control point (CP) to O_2. Since it divided the query interval into two subintervals for O_2 by O_1, it is described as $\mathrm{CP}(O_2, O_1)$:

$$\frac{l_{ic}}{l_{im}} = \frac{Z_{O_2}}{Z_{O_1}}. \tag{7}$$

Query interval $[S, E]$ and projections of objects O_1 and O_2 are shown in Figure 10. Since $\omega_{O_1} > \omega_{O_2}$, point c is the Point Change in O_2 to O_1. For any point n in the query interval, O_2 will be invisible because of O_1 at some point. If a straight line which goes through point c crosses O_1 at m and crosses $[S, E]$ at n so that $\varphi_{O_2} = \varphi_{O_1}$, then the relationship between φ_{O_2} and φ_{O_1} is the same as shown in Figure 11 and Formula (7). Ultimately, the control point depends on the Point Change.

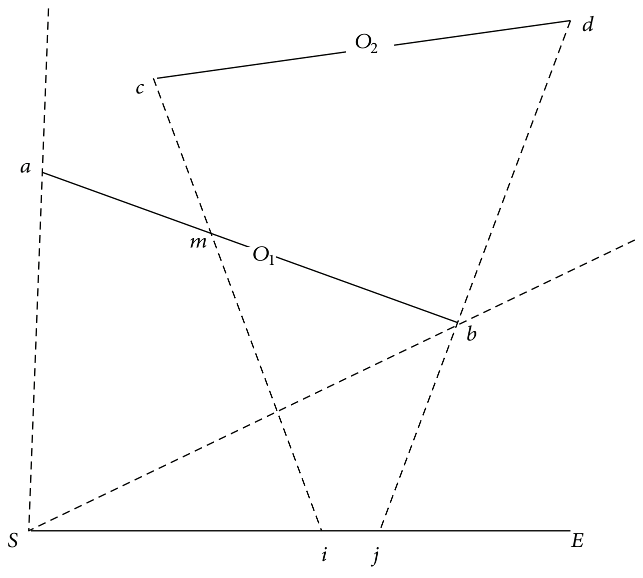


Figure 10: Confirm the control point by Point Change.

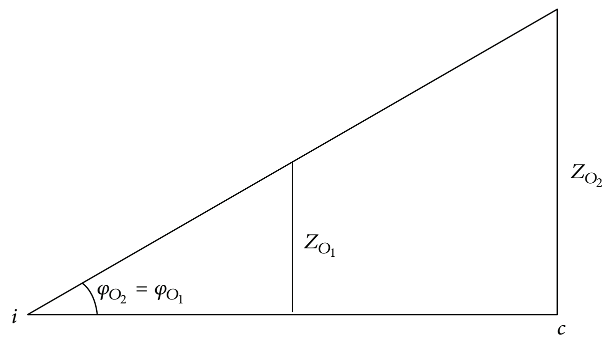


Figure 11: The projection angle of change point and Point Change.

As shown in Figure 12, point a is the Point Change (PC) of object O_2, L_1 represents object O_1, and L_2 is the query interval. L_3 is vertical to L_2. The coordinate of d is calculated

TABLE 2: Result of CVRQ-CP in Figure 14.

Object	Result set	Obstacle set
O_1	$\{O_1, [S, E]\}$	$\{O_1, [S, E]\}$
O_2	$\{[S, b]\ \{O_1, O_2\}\}\ \{[b, a]\ \{O_1\}\}\ \{[a, E]\ \{O_1, O_2\}\}$	$\{O_1, [S, E]\}\ \{O_2, [S, b]\ [a, E]\}$
O_3	$\{[S, b]\ \{O_1, O_2\}\}\ \{[b, a]\ \{O_1\}\}$ $\{[a, h]\ \{O_1, O_2\}\}\ \{[h, E]\ \{O_1, O_2, O_3\}\}$	$\{O_1, [S, E]\}\ \{O_2, [S, b]\ [a, E]\}$ $\{O_3, [h, E]\}$
O_4	$\{[S, b]\ \{O_1, O_2\}\}\ \{[b, a]\ \{O_1\}\}\ \{[a, h]\ \{O_1, O_2\}\}$ $\{[h, f]\ \{O_1, O_2, O_3, O_4\}\}\ \{[f, E]\ \{O_1, O_2, O_3\}\}$	$\{O_1, [S, E]\}\ \{O_2, [S, b]\ [a, E]\}$ $\{O_3, [h, E]\}\ \{O_4, [h, f]\}$

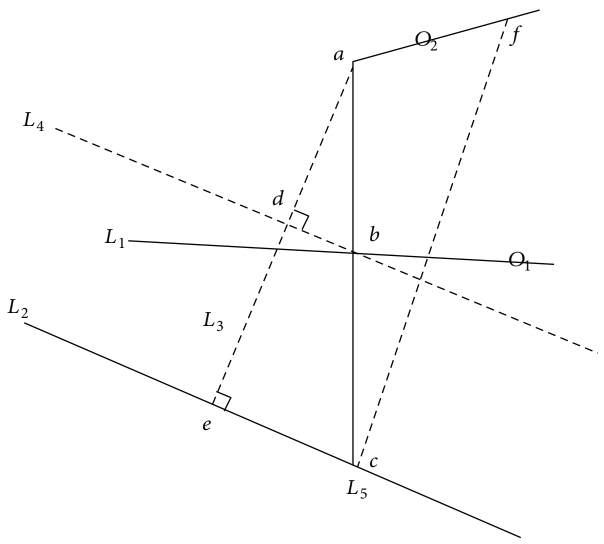


FIGURE 12: Coordinate calculation of control point.

using Formula (7). L_4 is vertical to L_3 and crosses L_5 at point b. L_5 which is the line through points a and b crosses L_2 at point c. Point c is the control point as $\mathrm{CP}(O_2, O_1)$. The CP is unique as in Theorem 8 proof.

Theorem 8 (unique control point). *According to the PC, the control point could be calculated and there is only one CP.*

Proof. A vertical line crosses point a and meets L_2 at point e (Figure 12). The resulting line is L_3. According to $ab/ac = Z_{cd}/Z_{ab}$, to get the only point d to satisfy the formula $ad/ae = Z_{cd}/Z_{ab}$, the vertical line L_4 crosses point d and meets line L_1 at point b. Point b is unique because L_4 is not parallel to L_1. Then, line L_5 crosses the points a and b and meets L_2 at point c, and $ab/ac = Z_{cd}/Z_{ab}$. If there are two CPs in the query interval, L_1 meets L_4 at two points. This is impossible, and therefore, point c is the only control point. □

If O_1 is parallel to O_2, the objects have the same rotation angle in the horizontal plane, and the visible relationship between O_1 and O_2 will not change in the obstacle area. As shown in Figure 13, $[S, J]$ is the OA of object O_2 by O_1. At any point n in the OA, a straight line crosses O_1 at b and crosses O_2 at d. The rate of projection angles of these two points is invariable (Formula (8)), and the visible relationship will not change in the OA. In these calculations, Z is the height dimension of the object:

$$\frac{\tan\varphi_{d,n}}{\tan\varphi_{b,n}} = \frac{Z_{O_2}/dn}{Z_{O_1}/bn} = \frac{Z_{O_2}/cn}{Z_{O_1}/qn} = \frac{\tan\varphi_{c,n}}{\tan\varphi_{q,n}}. \quad (8)$$

The obstacle area is divided into two subintervals with different visibility by the control point in the CVRQ.

4.3. Continuous Visible Range Query Based on Control Point (CVRQ-CP). In Figure 14, the projections of 3D objects O_1, O_2, O_3, and O_4 are sorted by MinDist. The height dimensions of the objects are 4, 5, 3, and 7, respectively. $\mathrm{CP}(O_2, O_1)$ is point b. The results of visibility testing are shown in Table 2.

When O_1 is inserted, the obstacle set is empty, so O_1 is visible to the entire query interval, and O_1 is put in both the result and the obstacle sets. Next, O_2 is inserted by order. The VA and IA are obtained through visibility testing, and $\mathrm{CP}(O_2, O_1)$ is calculated. The VA and IA of object O_2 are updated. Then, all objects are inserted one by one in order, and visibility testing is conducted. Finally, the result set is reconstructed and summarized.

The algorithm of Continuous Visible Range Query Based on Control Point is proposed as shown in Algorithm 2.

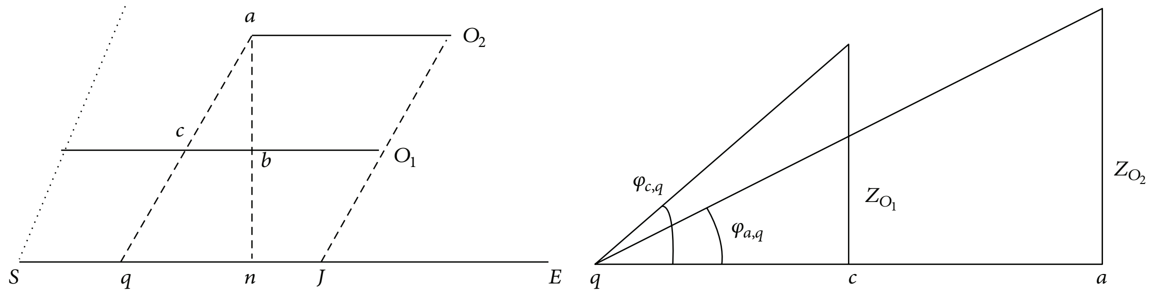
The range query based on the query interval $[S, E]$ in object OS is done in line (1), and results are sorted by MinDist in line (2) to avoid the influence of inserted objects. In line (3), the visibility testing of obstacle objects is calculated in order. During testing, the control point is calculated if the insert object is not parallel to the obstacle object. If the CP is in the IA, then it needs to be recorded in the result set. In line (13), the result set is reconfigured to summarize the visible objects for each subinterval of the query interval.

5. Experimental Evaluations

This section compares performance of three algorithms of visible query: (1) Basic-Algorithm-one (Basic I); (2) Basic-Algorithm-two (Basic II), a double-projection algorithm that separately processes the projections of 3D objects in the horizontal and vertical planes and describes visibility by their intersection; (3) the CVRQ Based on Control Point algorithm, proposed in this paper as CVRQ-CP. Performance evaluation was conducted on an Intel(R) Core(TM) i5 machine with 4 GB memory, running Microsoft Windows XP Professional Edition. The program was written in C++

```
Input: QueryArea SE; ObjectSet OS; Range R;
Output: ResultSet RS
(1)  Ob = rangeQuery(SE, R, OS); //do the range query, $MinDist(object.i, SE) \leq R$
(2)  Sorting(Ob);
(3)  for (every element i in Ob) do
(4)      for (every element j in ObsertObject) do
(5)          getVisibleArea;
(6)          if Boolean(i parallel j) then
(7)          else
(8)              getControlPoint and update VisibleArea;
(9)          end if
(10)     end for
(11)     if i is visible then
(12)         add i to ObsertObject and RS;
(13)     end if
(14) end for
(15) RS = reorganized(RS);
(15) Return RS;
```

Algorithm 2: Continuous visible range query based on control point.



(a) Horizontal plane (b) Projection angle

Figure 13: Visibility relationship between two parallel objects.

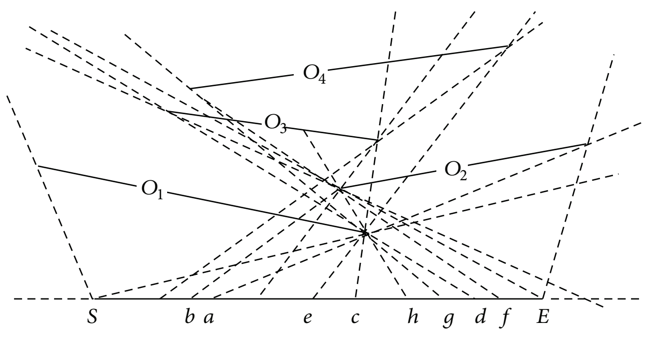


Figure 14: Continuous Visible Range Query Based on Control Point.

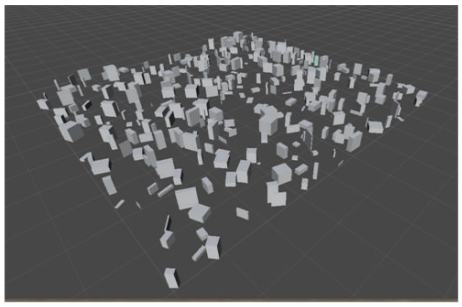


Figure 15: Dataset III contains 500 3D objects.

in Visual Studio.net 2010 development environment, called algorithm subroutine written in MATLAB R2012b.

Four datasets were created using the Spatial Object Data Generator and normally distributed in three-dimensional spaces. Table 3 lists all the parameters of datasets in the experiments. Dataset III contains 500 objects, as shown in Figure 15. The entire experiment is divided into two parts. The first part of the experiment is carried out in Datasets I, II, and III. The purpose of this part is to verify the correctness and performance of the 3D visibility testing method which is proposed in this paper. The second part of the experiment is carried out in Dataset IV to verify the influence of the query radius and the query interval length on the performance of CVRQ-CP.

Table 3: Datasets in the experiment.

Name	Objects	Space	Index	Distribution
Dataset I	100	1000 × 1000 × 500	Null	Uniform
Dataset II	250	1000 × 1000 × 500	Null	Uniform
Dataset III	500	1000 × 1000 × 500	Null	Uniform
Dataset IV	5000	10000 × 10000 × 500	Quadtree	Uniform

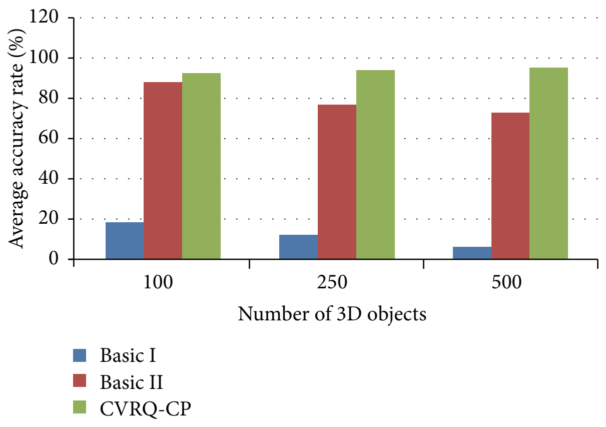


Figure 16: Accuracy rate.

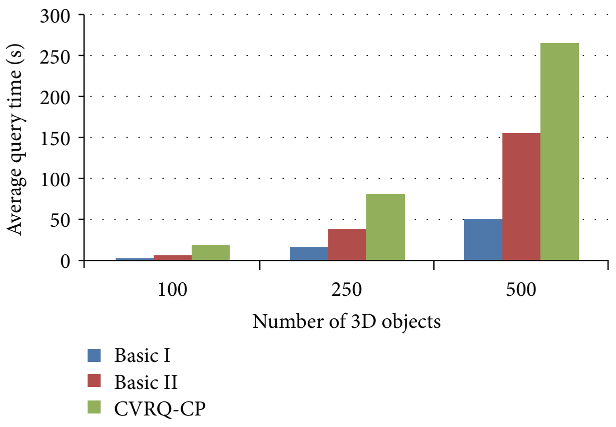


Figure 17: Response time.

In the first part of the experiment, in order to prove the influence of the quantity and density of object on the accuracy and response time of the query, 500 queries were done for each of the three algorithms in Datasets I, II, and III. To verify the accuracy of the three algorithms, we upload the objects in result list of the visible query in 3DMax and verify the correctness by manual inspection. The radius of range query is 250 and the query length is 0 in these experiments.

Accuracy results are shown in Figure 16. Obviously, the CVRQ-CP algorithm is more accurate than the other two algorithms. In Basic I, the height dimension of the objects is not considered, and the result set of visible query underwent little change across the three different scales. Therefore, the average accuracy rate decreased as the number of 3D objects increased. In Basic II, projections in the vertical plane were unable to describe the relationship between two objects due to rotation in the horizontal plane, and therefore, the average accuracy rate decreased as the number of 3D objects increased. However, since Basic II considers the influence of height, it is more accurate than Basic I. Conversely, for the CVRQ-CP algorithm, the projection of a 3D object is considered a segment. Although the accuracy is not 100%, the rate of accuracy is much higher than in Basic I and Basic II.

Response time results are shown in Figure 17. Each object in query range must be read and calculated in all three algorithms. This results in reduced query speed. The number of objects in query range has obvious influence on the response time. The same algorithm with same parameters has distinctly different response time in three different datasets. The only reason is that the number of objects in query range is different. The response time of the visible query increases as the number of obstacle objects increases. Meanwhile, compared in the same dataset, Basic I is fastest while the CVRQ-CP algorithm is slowest. In Basic I, visibility testing is only required once, in a horizontal plane. Meanwhile, the same calculation must be executed twice in Basic II, in both the horizontal and the vertical planes. On the other hand, CVRQ-CP algorithm includes projection angle calculation. Each visible object added to the obstacle set participates in the visibility testing of the subsequent object.

Table 4: Parameter ranges and default values.

Parameter	Range	Default value
Radius	250, 500, 750	500
Query length	500, 1000, 2000	1000

Since the Continuous Visible Range Query is regarded as a range query for a segment, the radius and query length greatly influence the visible query. The number of relevant objects increases with increasing radius under the same query length. Furthermore, response time dramatically increases with an increasing number of relevant objects due to the iteration in visibility testing. Each object in obstacle set may create a subinterval needing calculation.

The second part of experiment is to prove the influence of the query length and radius on the response time and I/O performance. This part of the experiment is done in Dataset IV. Dataset IV contains 5000 objects, as shown in Figure 18. Table 4 lists all the parameters that are considered in the experiments, with numbers in bold representing default settings. In each experiment, only one parameter is changed in order to evaluate its impact on the performance, while all the other parameters are fixed at their defaults. We run 200

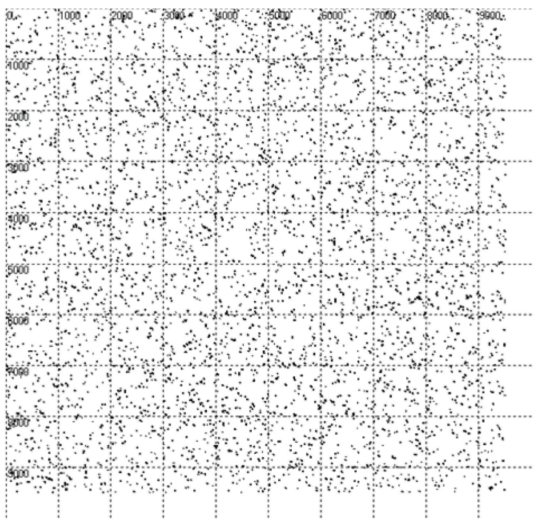


FIGURE 18: Dataset IV contains 5000 3D objects indexed by quadtree.

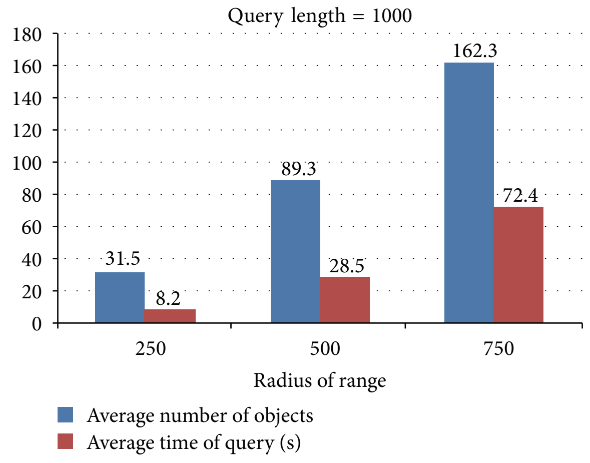


FIGURE 19: Influence of radius in CVRQ-CP.

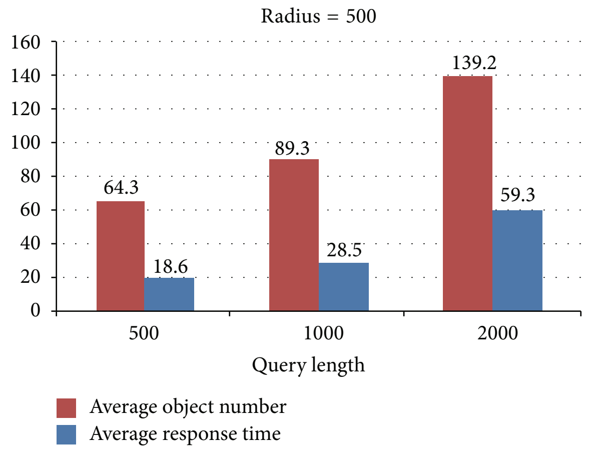


FIGURE 20: Influence of query length in CVRQ-CP.

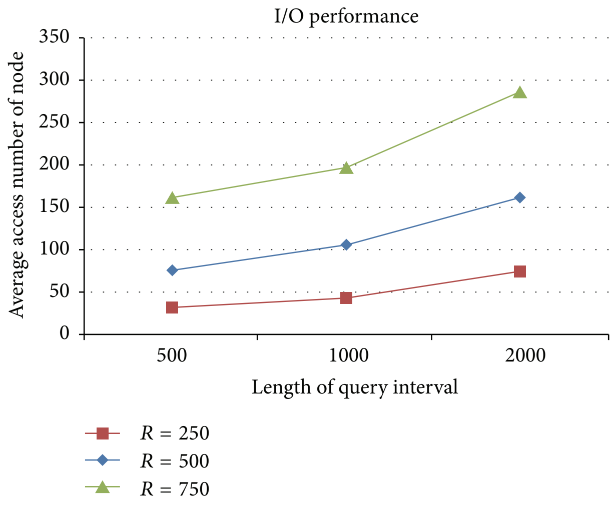


FIGURE 21: Access number of nodes of different query interval lengths and radius.

queries for each experiment, and the average performance is reported.

Response time results under different radius are shown in Figure 19. The number of relevant objects increases with increasing radius under the same query length. Furthermore, response time dramatically increases with an increasing number of relevant objects due to the iteration in visibility testing. Each object in obstacle set may create a subinterval needing calculation.

Response time results under different query length are shown in Figure 20. The number of relevant objects increases with increasing query length under the same radius. Furthermore, response time dramatically increases with an increasing number of relevant objects due to the iteration in visibility testing. Each object in obstacle set may create a subinterval needing calculation.

The average access number of node increased with the visible query radius and the radius of range. The result of I/O is shown in Figure 21. Conclusively, the visible query radius has more significant impact than the query interval length. As a result, the radius is very important in the visible query application. If the radius is too small, the VR scene will be too small to simulate a realistic view of the human to reduce the reality. On the other hand, a large radius may render response time too long to effect the application of visible query.

6. Conclusions

The visible query for three-dimensional objects plays an important role in large-scale virtual reality scenes, such as in online games and virtual cities. The algorithm Continuous Visible Range Query Based on Control Point (CVRQ-CP) proposed in this paper addresses the lack of a specific algorithm for visible query for three-dimensional objects. The visibility testing based on projection angles in the vertical plane describes the visibility of 3D objects, while the control point describes the distinctive visibility change caused by changing distances during continuous visible query for 3D objects. Experimentation showed CVRQ-CP accuracy as superior to present algorithms.

However, some deficiencies require further research. For example, we kept the query interval in a horizontal plane, preventing the CVRQ-CP algorithm from dealing with some situations. We recommend further study of CVRQ-CP processing to solve these problems and optimize I/O operation in 3D spatial databases.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant no. 61572422 and the Natural Science Foundation of Hebei Province, China, under Grant no. F2013203187.

References

[1] S. Nutanong, E. Tanin, and R. Zhang, "Visible nearest neighbor queries," in *Advances in Databases: Concepts, Systems and Applications*, R. Kotagiri, P. R. Krishna, M. Mohania, and E. Nantajeewarawat, Eds., vol. 4443 of *Lecture Notes in Computer Science*, pp. 876–883, Springer, Berlin, Germany, 2007.

[2] Y. Gao, B. Zheng, G. Chen, Q. Li, and X. Guo, "Continuous visible nearest neighbor query processing in spatial databases," *VLDB Journal*, vol. 20, no. 3, pp. 371–396, 2011.

[3] Y. Gao, B. Zheng, G. Chen, W.-C. Lee, K. C. K. Lee, and Q. Li, "Visible reverse k-nearest neighbor query processing in spatial databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1314–1327, 2009.

[4] D. Champion, K. G. Pankratz, and L. A. Fatale, *The Effects of Different Line-of-Sight Algorithms and Terrain Elevation Representations on Combat Simulations*, US Army TRADOC Analysis Center, White Sands Missile Range, NM, USA, 1995.

[5] D. L. Henderson, *Modterrain: A Proposed Standard for Terrain Representation in Entity Level Simulation*, Naval Postgraduate School, Monterey, Calif, USA, 1999.

[6] G. Schaufler, J. Dorsey, X. Decoret, and F. X. Sillion, "Conservative volumetric visibility with occluder fusion," in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '00)*, pp. 229–238, ACM Press/Addison-Wesley, 2000.

[7] E. Angel and D. Morrison, "Speeding up Bresenham's algorithm," *IEEE Computer Graphics & Applications*, vol. 11, no. 6, pp. 16–17, 1991.

[8] S. Alipour, M. Ghodsi, A. Zarei, and M. Pourreza, "Visibility testing and counting," *Information Processing Letters*, vol. 115, no. 9, pp. 649–654, 2015.

[9] D. Z. Chen and H. Wang, "Weak visibility queries of line segments in simple polygons," *Computational Geometry. Theory and Applications*, vol. 48, no. 6, pp. 443–452, 2015.

[10] J. Hershberger, "Finding the visibility graph of a simple polygon in time proportional to its size," in *Proceedings of the Annual Symposium on Computational Geometry*, pp. 11–20, Waterloo, Canada, June 1987.

[11] S. K. Ghosh and D. M. Mount, "An output-sensitive algorithm for computing visibility graphs," *SIAM Journal on Computing*, vol. 20, no. 5, pp. 888–910, 1991.

[12] M. Nouri Baygi and M. Ghodsi, "Space/query-time tradeoff for computing the visibility polygon," *Computational Geometry: Theory and Applications*, vol. 46, no. 3, pp. 371–381, 2013.

[13] J. Zhang, D. Papadias, K. Mouratidis, and M. Zhu, "Spatial queries in the presence of obstacles," in *Advances in Database Technology—EDBT 2004*, E. Bertino, S. Christodoulakis, and D. Plexousakis, Eds., vol. 2992 of *Lecture Notes in Computer Science*, pp. 366–384, Springer, Berlin, Germany, 2004.

[14] L. Lu, C. Yang, W. Wang, and J. Zhang, "Voronoi-based potentially visible set and visibility query algorithms," in *Proceedings of the 8th International Symposium on Voronoi Diagrams in Science and Engineering (ISVD '11)*, pp. 234–240, Qingdao, China, June 2011.

[15] Y. Gu, X.-N. Yu, and G. Yu, "Method for continuous reverse k-nearest neighbor queries in obstructed spatial databases," *Journal of Software*, vol. 25, no. 8, pp. 1806–1816, 2014 (Chinese).

[16] Y. Wang, R. Zhang, C. Xu, J. Qi, Y. Gu, and G. Yu, "Continuous visible k nearest neighbor query on moving objects," *Information Systems*, vol. 44, pp. 1–21, 2014.