PICK CUBES FOR OBJECT PICKING IN VR WORLD

M. Mohamed Sathik [1], G.Sofia [2]
Research and Development Centre,
Bharathiar University, Coimbatore.
[1] Associate Professor in Computer Science, Sadakathullah Appa College, Tirunelveli.
mmdsadiq@gmail.com
[2] Lecturer in Computer Science, Lady Doak College (Autonomous), Madurai.
joesofi@ymail.com

ABSTRACT - *In this paper, we proposed an algorithm which performs object picking in Virtual reality world using cubes. As 'Picking' is an essential interaction technique in graphics applications, Object Picking is performed in VR world using a 3D mouse or an interactive glove to explore and interact with any of the objects. In this paper, cube is used to represent the intersection point of the ray coming from the user input and the object to be picked. By intersecting the objects in the scene with the ray with a pick cube at the intersection point, it is determined which one is picked. An object is selected when this pick cube intersects the object's bounding box. In this paper, object picking is typically performed by bounding box checks, taking the position of the pick cube and the objects in account so that the algorithm is made simpler and the computation time of the algorithm is also reduced. Pick cube can be activated to perform more actions during interactions than other shapes. So the efficiency and the performance of the picking process will be increased which helps the user to play in the virtual world with full enthusiasm.*

KEYWORDS: *Virtual Reality, Object Picking, bounding box, pick cube, Pick distance*

1. INTRODUCTION

Virtual Reality (VR) systems create a 'cyberspace' where it is possible to interact with anything and anyone on a virtual level. Virtual Reality is a generic term associated with computer systems that create a real-time visual, audio and haptic experience [1]. In the past few years, VR has matured considerably. New hardware platforms and software environments have appeared, supported by a young, dynamic, professional work force. The industry is now well established and is leading us towards a truly exciting future. Virtual Reality takes us one step closer to virtual objects by making us a part of the virtual domain. The sensation of immersion reinforces our belief that the virtual domain is real. Virtual reality experience is defined as the one in which the user is effectively immersed in a responsive virtual world[9]. This implies user dynamic control of viewpoint.

We know from experience that when viewing a 3D scene on a computer display, the image is fixed to the screen. No matter how we move our heads, the image remains unaltered, apart from an overall adjustment of perspective. However, if the head is tracked in three-space, and used to control the viewpoint of the displayed scene, something intriguing occurs. The image is no longer frozen to the screen; we are looking into a volume of space behind it. The virtual domain now takes on a different reality. Computer Graphics provides the basis for creating the synthetic images while a Head-Mounted Display (HMD) supplies the user's eyes with the stereoscopic view of the computer generated world.

There are four system elements, which constitute the VR system are Virtual Environment (VE), Computer Environment, VR Technology and the modes of interaction [1]. Virtual Environment covers ideas such as model building, introducing dynamic features, physical constraints, illumination and collision detection. The Computer Environment includes the processor configuration, the I/O channels, the VE database and the real-time operating system. VR technology encompasses the hardware used for head-tracking, image display, sound, haptics and hand-tracking. Modes of interaction involve hand gestures, 3D interfaces and multi-participant systems. Virtual worlds are built with objects called virtual objects [1]. Objects are divided into two groups: static and dynamic. Static objects include objects such as floors, walls, ceilings, and stairs and so on. Dynamic objects include objects such as doors, windows, cupboards, drawers and lids. There will be hierarchical relationship between virtual objects, and it is essential for the sharing of VE between different users. Virtual hands are virtual objects that mimic the movements of the user's real hands. An object can be selected by positioning one of the virtual hands inside it. A glove or a 3Dmouse can be used to pick virtual objects and move them to different positions. The output signals from the VR system carry visual, audio and haptic information about the VE. Visualization of the VE is displayed upon the user's HMD (Figure 1). The sounds will be localized using headphones. The user can wear some sort of glove (Figure 2), holding a reactive joystick, or is attached to some force-feedback device.

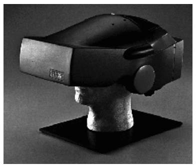


Figure 1. Head Mounted Display

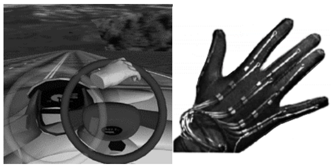


Figure 2. Data Glove

There are two kinds of Interaction in the VR world. One is Immersive Interaction and the other one is Non-Immersive. In Immersive VR system the user is immersed in the VE through the use of HMD[10]. Some of the operations in this system are **Flying,** through which Navigation in the VE is achieved, **Teleporting** which involves identifying a new location in the VE to which the operating system automatically transports the user without having to travel, **Object Picking** in which a 3D mouse or an interactive glove can be used to explore and interact with any of the objects that have been assigned dynamic properties, **Gesture Recognition** in which the user's hand position and finger joint angles are used to give commands through an accepted vocabulary of signs. In Non-Immersive VR Systems, the user will navigate through the environment using a joystick or spaceball. The workstation will be operating like an interactive television.

2. OBJECT PICKING

Selection of specific objects is essential to interactivity. Without an ability to select objects, you cannot manipulate them. 'Picking' is an important interaction technique in graphics applications. Without question, Picking is indispensable. The Picking behaviour enables selecting objects in the scene. A pick in 3D is usually carried out as Ray Pick.[11] The virtual camera position and the 2D mouse pointer on the image plane define the ray. By intersecting the objects in the scene with the ray, it is determined which one is picked. Not only objects but also their topological elements, i.e. faces, edges, vertices, can be picked. A glove or a 3Dmouse can be used to pick virtual objects and move them to different positions [2]. Object Picking is performed in VR world using a 3D mouse [4] or an interactive glove to explore and interact with any of the objects. The user opens a door by moving the icon of the 3D mouse towards the position of the door's handle. When a collision is detected, the user confirms the selection of the door by activating a button on the mouse. In virtual environments, Object picking is typically performed by bounding box checks or collision detection, taking the position of a virtual hand and the objects in account.

Since 3D Picking is a costly task, especially when performed on a precise 3D CAD model, hence an algorithm is needed that accelerates Picking. In the field of 3D CAD it is important that the pick generates precise information using the accurate CAD model and supports the identification of topological entities such as faces, edges and vertices. Thus, a 'real' 3D pick requires the 'is-inside-model check'. To reduce these costly operations on the precise CAD model, we introduce an algorithm that performs multi-level bounding box checks and uses coherence during user interactions. This algorithm is gaining its performance by reducing these checks dramatically compared to the naive approach where the checks are carried out with each 3D cursor movement.

Use Object picking to identify the objects on the screen that appear near the cursor. To use Object picking, the software must be structured so that the picture can be regenerated on the screen whenever picking is required. As there was nothing special about the notation used for 2D shape picking, the same approach can be used for 3D object picking [1].

3. METHODS

This section presents a brief description of the methods used for object picking using cubes in VR world.

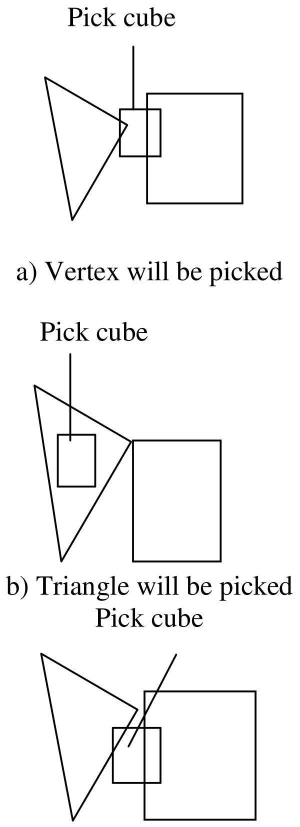
3.1 Pick Cube

The most intuitive and easy way for the user to select an object is to simply reach out until his hand intersects the desired object [3]. After some trial, we decided that the most precise way to support this was to put a tracker on the forefinger of each Pinch Glove and define a small, box (any shape) around it. An object is selected when this box intersects the object's bounding box. The size of the box can be adjusted to allow for a larger or smaller 'error margin'. Here we have considered the shape of the box as cube which is referred here as pick cube.

A pick in 3D is usually carried out as Ray Pick [5]. The virtual camera position and the 2D mouse pointer on the image plane define the ray. In this paper, cube is used to represent the intersection point of the ray coming from the user input and the object to be picked. So at the cursor position we can be able to see the pick cube while we are picking the object. By intersecting the objects in the scene with the ray with a pick cube at the intersection point, it is determined which one is picked. Not only objects but also their topological elements, i.e. faces, edges, vertices, can be picked. An object is selected when this pick cube intersects the object's bounding box.

3.2 Pick Heuristic

Since the pick cube can intersect several topological entities of one or different object(s), there need to be a rule to show which entity is selected in such cases. Simply selecting the nearest one usually does not conform to the user's expectation and makes it nearly impossible to pick elements with a smaller extent, i.e. vertices with respect to edges and edges with respect to faces. Thus, a pick priority is assigned to the different entity types: vertices are favored against edges and edges against faces. If for instance the pick cube intersects an edge and contains a vertex, the vertex is returned from the pick procedure even if a point that belongs to the edge is nearer to the cursor (pick cube) center (see Fig 3.a, 3.c). If the pick cube contains several topological elements of the same type, the closest one is chosen (see Figure 3.b). Since this default behavior does not always fit to the user's intention, there is an additional pick filter, which allows the user to specify which topological element type should be considered by the pick procedure.



a) Vertex will be picked

b) Triangle will be picked

c) Edge of the rectangle will be picked

Figure 3. Pick Priorities

'**Pick Distance**' is the shortest distance between the center of the pick cube and the nearest cube edge.

3.3 Bounding Box

Three-dimensional bounding volume is a ubiquitous tool of accelerating various geometric calculations including intersection detection.[12] In virtual environments, 3D picking is typically performed by bounding box checks or collision detection, taking the position of a virtual hand and the objects in account. As a basic idea for reducing the collision detection complexity, preliminary collision tests can be performed using simplified bounding volumes that contain the complex objects to be tested [6].

In checking whether two objects intersect or not, bounding volumes make this intersection test more efficient, especially when the objects do not intersect most of the time. A bounding volume approximates an object by another simpler object that contains the original. Because bounding volumes are chosen to have much simpler topology and geometry than the original objects, checking the intersection between bounding volumes can be performed with a lower computational cost.

Bounding Volume is a 3D object that encloses an object. Different types of bounding volumes may be considered, each of them having their own strong points and weaknesses. There are two types of Bounding Volumes. They are

(i) Bounding Boxes
(ii) Bounding Spheres.

The bounding boxes are usually axis-oriented, described by two opposite corner vertices, and the bounding spheres are described by the center and the radius.

A Bounding Box for an object is just a rectangular box in three dimensional space, with sides parallel to the coordinate planes, that contains (or surrounds) the object as shown in Figure 4. A bounding box is an orthogonal, rectangular volume [7] that bounds an object.

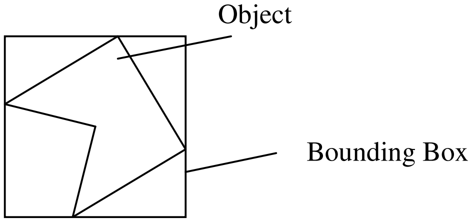


Figure 4.Bounding Box

More complicated bounding volumes may be considered for efficient bounding when a small number of bounding primitives are required. Such volumes use more parameters in their description, allowing a wider range of shapes in optimizing their filling efficiency (reducing their "empty space" so their collisions become more representative of the collisions of the contained objects) and trading away some of their computational simplicity (more complicated shape description). The choice is highly dependent of the shape of the objects to be bounded. For elongated objects, possible solutions include bounding ellipsoids and cylinders.

3.4 Picking procedure

Generally, there are two ways to implement Picking with a pick cube: either one checks for collisions between the cube and the topological entities' geometry or one calculates the point of the topological entity's geometry that is next to the center of the pick cube and checks if the distance between the nearest point and cube center is smaller than the current pick distance. Only the second approach promises exact results, but requires time-consuming nearest-point calculations. To minimize those calculations, a multi-level bounding box check is introduced and described in the following.

First of all the bounding boxes around each object and all its topological elements' geometry are determined and stored. If the user initiates a pick, the procedure starts with checking the pick cube against the object bounding boxes. This is done by enlarging the bounding box with the pick distance of the pick cube and checking if the center of the pick cube is inside the enlarged bounding box. This trick is used to avoid the calculation whether the pick cube intersects or touches the bounding box. Using Pick cube avoids small errors due to the difference in enlarged bounding box and the Minkowski sum [8] between the bounding box and the Pick cube as shown in Figure 5.b.

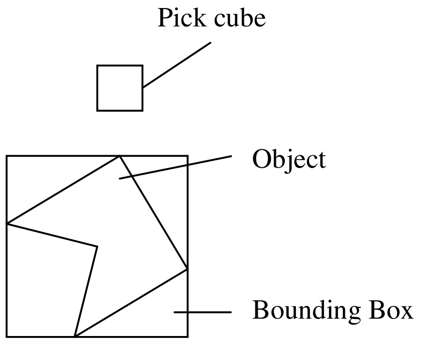


Figure 5.a. Object, bounding box and pick cube

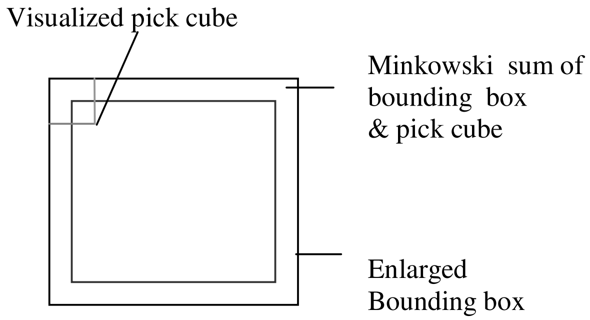


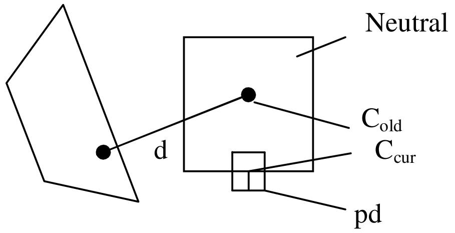
Figure 5.b. Minkowski sum between bounding box and pick cube Vs. Enlarged Bounding Box

If the center of the cube is inside the enlarged object-bounding box, all object faces are checked. Again, the procedure starts with enlarging the face bounding boxes by the pick distance and checks if the cursor center is inside the enlarged box. Actually, the enlargement is done component-wise in the bounding box check to speed up the process. If the center of the cube is inside the bounding box, the nearest point of the face geometry is calculated. The vertices are checked by simply comparing their position with the cursor position and the pick distance. Finally, if all these checks failed, an is-inside check is performed to see if the cursor is in the interior of an object without intersecting any of its topological elements.

In principle, a Pick action has to be performed with every cursor movement, to detect if the cursor has to be snapped to an object. In the above-described Pick procedure, the nearest point calculation is the crucial time-consuming factor. Thus, the number of these calculations has to be further minimized beyond the level already reached with the bounding box checks to enable real-time snapping on moderately complex and precise models.

To minimize the nearest point calculations, the multi-level bounding box approach is enhanced by storing the distance between the nearest point of the topological element and the corresponding cursor position. This way, for further cursor movements it can be decided if a re-calculation of the nearest point is needed or not. A re-calculation is unnecessary if the cursor moved less than the stored distance minus the current pick distance as illustrated in Figure 6.

Each time a nearest point calculation takes place, the cursor position and the distance to the nearest point is stored with the topological element currently regarded.



C_{old}=old cursor position
C_{cur}=current cursor position
d=stored distance
pd=pick distance
neutral Zone = the zone in which the cursor can move without a recalculation of the nearest point

Figure 6. Concept of neutral zone to minimize nearest-point calculations.

Edges are handled similarly to faces, only on the vertex level the bounding box check is not needed, the vertex position is directly compared with the pick cube position.

3.5 Algorithm in Pseudocode :

3.5.1 To Pick an object

```
Procedure Pick()
 {
     for each cursor movement
       for each visible object
            if cursor is inside enlarged object bounding box
                 for each face
             if cursor is inside enlarged face bounding box
                  if ($C_{old}$-$C_{new}$ >=stored distance-pick distance)
                          {
                             call NearestPoint();
                             store distance and current position;
                           }
         }
end Pick;
```

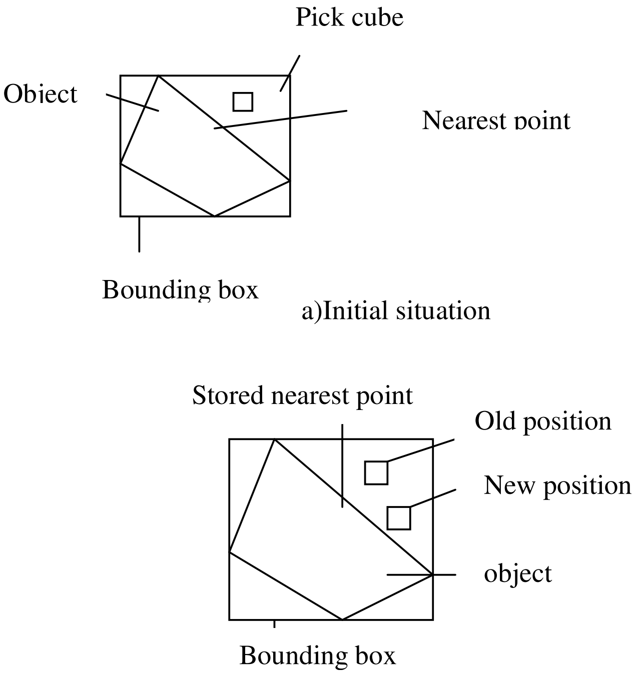
3.5.2 To calculate the Nearest Point

```
//bpx[ ], bpy[ ], bpz[ ] – Array of boundary points of the object.
//cx, cy, cz – Centre of the Cube.
//sd – Shortest distance
//dist – Distance between the center of the cube and the array point.
//nx, ny, nz – Nearest point
Procedure NearestPoint()
  {       Input bpx[ ], bpy[ ], bpz[ ];
       Assign sd= $\mathrm{sqrt}((cx-bpx[0])^2 + (cy-bpy[0])^2 + (cz-bpz[0])^2 )$;
       For each value in the array index i from 1 to the final point
       {
          dist= sqrt($(cx-bpx[i])^2 + (cy-bpy[i])^2 + (cz-bpz[i])^2$ );
          if dist < sd
             {
                Assign sd = dist;
                Assign nx=bpx[i];
                     Assign ny=bpy[i];
                     Assign nz=bpz[i];
       }
  return nx, ny, nz;
 }
end NearestPoint;
```

4. RESULTS

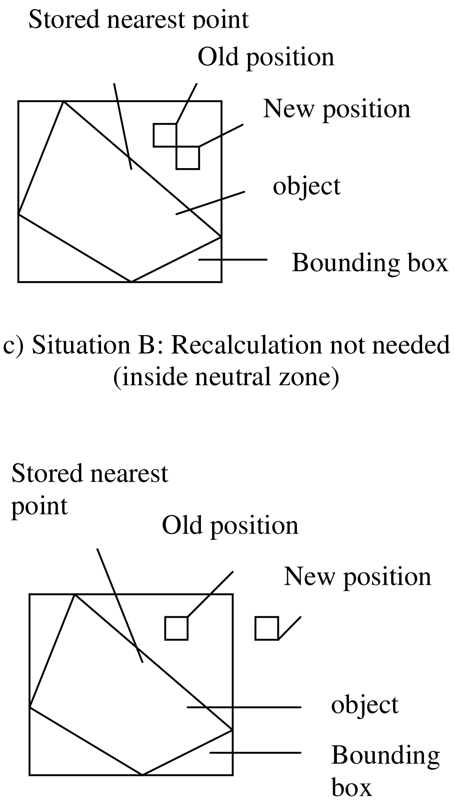
4.1 Pick performance

Different situations to show what happens under certain conditions are given in Figure 7. It depicts the former situation and show alternatives for the current situation. On Figure 7.b the cursor has left the neutral zone (its distance from the former position exceeded the stored distance minus the pick distance). Hence recalculation is needed. On Figure 7.c the cursor is still in the neutral zone. Due to that it simply cannot intersect with the face and a recalculation is also not necessary. In Figure 7.d the cursor has left the bounding box (of the face and the object), the object is no longer considered as a potential intersector. In order to find the nearest point, the boundary points of the object should be found. Calculate the distance between the center of the cube and the boundary points. Then the distances should be compared and the shortest distance is picked up. The boundary point, which produces the shortest distance, is the nearest point.



a)Initial situation

b) Situation A: Recalculation needed

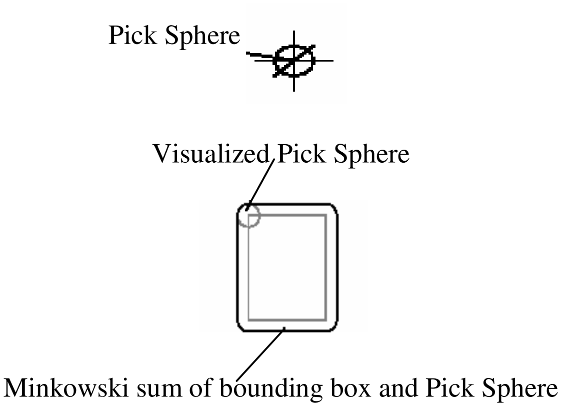


c) Situation B: Recalculation not needed
(inside neutral zone)

d)Situation C: Recalculation not needed
(outside bounding box)

Figure 7. Different situations for the pick cube position and the impact on the face calculations.

Using Pick Sphere leads to a small error because the enlarged bounding box is bigger than the Minkowski sum [8] between the bounding box and the Pick Sphere (see Figure 8). This error can be avoided by using Pick Cube.



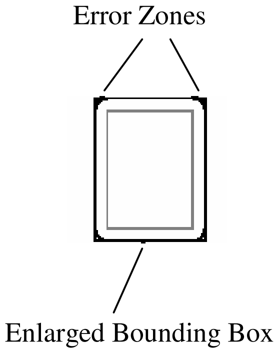


Figure 8. Minkowski sum between bounding box and pick sphere vs. enlarged bounding box

Cubes are easy to recompute as objects move making them well suited to dynamic environments.[13] At each time step as the object moves we recompute the cube as shown in Figure 9.

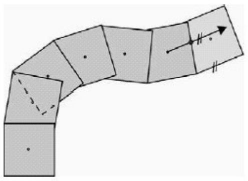


Figure 9 Cube Movement

Transform the center using one vector matrix multiplication. Compute the minimum and maximum x, y and z coordinates by subtracting and adding the radius from the coordinates of the center. Step involves only one vector matrix multiplication. Step needs six arithmetic operations additions and subtractions.

5. CONCLUSION

We found fast picking is crucial for task-performance in VR-like systems, which employ 3D input devices. In this paper an algorithm is presented that enables Picking of an object of a precise 3D model. The algorithm gains its performance by multi-level bounding box checks and by taking succeeding situations into account, i.e. by using coherence. Thus, it is possible to minimize the costly nearest point calculations. With this algorithm it is possible to pick an object with a 3D cursor seamlessly along the surface of moderately complex VR Envirnment. Results show that the run-time behaviour of the algorithm is nearly constant, i.e. independent from model complexity. In this paper, as cube is used to represent the intersection point of the ray coming from the user input and the object to be picked the errors that may occur when using other pick shapes are eliminated. It is also easy to recompute the movement which reduces the recalculation process. Thus the performance of the picking process increase while using the pick cube. Instead of using the collision deduction methods, bounding box method is used in our algorithm so that the algorithm is made simpler and the computation time of the algorithm is also reduced accordingly. Thus this algorithm is too simple, more efficient and it helps in easy interaction with the virtual world, hence it makes the Virtual world user friendly. In the future, fast picking can be a basis for controlling a force feed-back device. Instead of uploading a haptic scenegraph, it could be possible to just position the force feedback device based on the picking information.

REFERENCES

[1] John Vince, Virtual Reality Systems, Published by Pearson Education, 2002.

[2] Kiyoshi Kiyokawa, Haruo Takemura, Yoshiaki Katayama, Hidehiko Iwasa and Naokazu Yokoya, VLEGO: A Simple Two-handed Modeling Environment Based on Toy Blocks, Nara Institute of Science and Technology (NAIST), JAPAN.

[3] Rogier van de Pol, William Ribarsky, Larry Hodges and Frits Post, Interaction in Semi-Immersive Large Display Environments.

[4] Fadel, G.M., Crane, D., Dooley, L., Geist, R. " Support Structure Visualization in A Virtual Environment" Presented at the sixth International Rapid Prototyping Conference, Dayton, OH, 1995.

[5] Weimar, The ***P***rojector-based, ***D***esktop ***R***each-***I***n ***V***irtual ***E***nvironment, Eurographics Symposium on Virtual Environments/ Immersive Projection Technology , July 2007

[6] M. Moore, J. Wilhelms, Collision Detection and Response for Computer Animation, 1988.

[7] K. Shimada, J. Cagan and S. Yin, Geometric Representations for Intersection Detection in Intelligent Packaging, 1998.

[8] Rolf Klein, Thomas Kamphans, The Minkowski Sum of two arbitrary polygons , Institut für Informatik , 2001.

[9] Frederick P. Brooks, Jr ., What's Real About Virtual Reality?, IEEE Computer Graphics and Applications, 1999.

[10] Cecilia Sik Lanyi, Zoltan Geiszt, Peter Karolyi, Adam Tilingerand Viktor Magyar, Virtual Reality in Special Needs Early Education, The International Journal of Virtual Reality, 2006.

[11] Andre Stork, An Algorithm for Fast Picking and Snapping using a 3D Input Device and 3D Cursor, Fraunhofer Institut fur Graphische Datenverarbeitung, Darmstadt, Germany.

[12] Chin-shyurng fahn, Jui-lung wang, Collision detection among Polyhedral objects, Journal of Information Science and Engineering, 1999

[13] B. de Vries and H.H. Achten, DDDoolz: designing with modularMasses,Elsevier Science, 2002

AUTHORS

Dr.M.Mohamed Sathik received his Ph.D degree, M.Tech. and M.Phil., degree in Computer Science from Manonmaniam Sundaranar University, Tirunelveli, India, Master in Business Administration from Alagappa University, Karaikudi, India and M.S in Counseling And Psycho Therapy from Institute Of Psychotherapy and Management Mumbai, India. Currently he is working as Associate Professor in the Department of Computer Science, Sadakathullah Appa College, Tirunelveli.. His experience started from 1987. He has guided more than 30 research scholars and also published two books. He is a member of curriculum development committee of various universities and autonomous colleges of Tamilnadu and his area of specialization are Virtual Reality, Image Processing and Sensor Networks.

G. Sofia received her M.C.A degree from Manonmaniam Sundaranar University, Tirunelveli, M.Phil., degree in Computer Science from Mother Teresa Women's University, Kodaikanal and pursuing part-time Ph.D in computer science in Bharathiar University, Coimbatore. Currently she is working as a Lecturer in the Department of Computer Science, Lady Doak College, Madurai. Under the guidance of Dr.M.Mohamed Sathik, she has presented many papers in National and International conferences. Her area of specialization are Image Processing and Virtual Reality.