Sub-object Vertex Modifying Method of Scene Model in Virtual Training System

Wei-jun CHU¹,∗ and Zi-cheng ZHU²

¹Army Engineering University of PLA, Nanjing, Jiangsu, 210007, China
²China Heavy Truck Group Co., Ltd, Jinan, Shandong, 250116, China

*Corresponding author

Keywords: Vertex Modifying, Scene model, Neighborhood factor, Virtual training system.

Abstract. Vertex Modifying is a key problem in sub-object modeling. Facing local special structures such as convex peaks, pits, step surfaces, slopes, terraces, cliff collapses, landslides, steep cliffs, and gully in scene models, vertex creating process and its definition are illustrated. The modeling method by vertex Modifying and the mutual relation with other sub-objects are analyzed, and the use of repetitive triangulation algorithm and chamfering algorithm based on vertex are further discussed. Vertex control method based on neighborhood factor is expounded, and the influence of its parameters is compared by value and model. The analysis indicates that vertex modification provides flexible modeling method for scene modeling in virtual training system, and the result is anticipative and satisfactory.

Introduction

According to the actual situation of engineering equipment training, the training scene is divided into static and dynamic scenes, among which the static scene includes ground, dry ditch, river, bed, beach head and sky, while the dynamic scene is mainly water. For each scene, corresponding visual model needs to be built, that is, the following visual model needs to be built: ground of different forms, horizontal height and dry gully with height difference, river, bed, sky, vegetation, trees, etc. In order to reduce the computer burden, many insignificant details are replaced with approximate textures. Different from multimedia animation, real-time visual simulation needs to be generated through real-time calculation. Therefore, compared with multimedia animation, when building a visual model, objects should be constructed with as few faces as possible, which can greatly reduce operational overhead and memory consumption. Mesh scene vertexes are the most basic components of a visible scene and are inherited from the CBaseSceneNode class. This class is mainly responsible for maintaining the parent-child relationship of vertexes and defining the data shared by vertexes.

Scene Modeling

The self-developed GTENG engine technology is a complete set of real-time 3D engine for virtual training system based on the Windows operating system, and the underlying interface is for above directx9.0 version. GTENG engine adopts advanced real-time computer graphics algorithm, including computer graphics and optimized physical simulation algorithm set, suitable for a variety of hardware acceleration platform, to achieve scene management, 3D model rendering, dynamics calculation and other simulation functions.

The scene models to be organized in the GTENG engine are simply loose tree structures. All branches of a tree are a vertex. The biggest benefit of using a tree structure is that it's easy to program and take advantage of recursion. All vertexes in the engine inherit from the CBaseSceneNode class, sharing member variables and member functions unique to the vertex. This engine comes in both built-in and external vertexes. Built-in vertexes are classes that the engine has built, such as sky vertexes. When not in use they are not added to the scene manager, and when needed they are created using the functions built into the scene manager and placed in their proper
places. The external vertex is controlled by the user and placed where needed, using functions such as adding and deleting classes. For example, m_pMeshNode, the next level vertex in the built-in scene manager, is added to form a new mesh vertex. These vertexes have their own Pointers or access interfaces that users can use to manipulate the vertexes. The most common operations are changing the location of the vertex, setting its properties, and so on.

**Creation of Sub-object Vertexes**

**Sub-object.** Take 3D terrain modeling as an example. In the modeling process, a lot of work needs to be done to modify the discontinuous part of the model or a local area, so as to obtain the model effect with higher details and complex structure, as shown in figure 1. The transformation processing of such a single or multiple local areas is called sub-object modification. Sub-object of model or mesh can be vertex, edge, spline, face, polygon, patch, path and so on. For example, select an editable spline that represents an elevation contour, and create a mesh surface over the contour. It is also possible to create a terraced representation of the terrain object so that the contour data at each level is a step, similar to the traditional terrain model, as shown in figure 2.

![Figure 1. Visual simulation scene.](image1)

![Figure 2. Sub-object.](image2)

**Vertex Creating.** The most basic elements of 3D terrain mesh model are vertexes, which constitute the control points of the mesh. A vertex can be considered as the smallest entity, which is different from a vertex. Two vertexes form an edge, and on this edge more points can be formed by densifying points (thinning the mesh). Therefore, the vertex essentially contains vertices. The number of vertexes often depends on the degree of refinement and accuracy of the model, and can be reflected in the degree of visualization smoothness, as shown in figure 3.

![Figure 3. Vertexes in the terrain model.](image3)

The position of the created vertex can be precisely controlled, and the created vertex becomes part of the model. The ultimate purpose of the vertex is to create a mesh surface. In contrast, the largest resource for creating a vertex is an existing mesh object. Vertexes can be peeled off the mesh objects by removing faces while retaining them. Typically, refinement of objects or encryption of local structural vertexes requires that vertexes that exist in the mesh objects be copied to the appropriate position so that they are related to the existing vertexes as modeling resources.
This approach is simpler and more accurate than carefully placing an active mesh for each level of the new vertex. For example, define three vertexes in a face, with vertexes n1 and n2 at the periphery of the object, and vertex n near the center of the object. By adding or subtracting a, we get the offset of two peripheral vertexes, Z is 0. It is defined as follows:

```
vert_array=#()       -- This array holds the coordinates of the vertexes
face_array=#()       -- This array is used to hold face definitions
```

```
n1=[radius1*cos(a+width),radius1*sin(a+width),0 ]
n2=[radius1*cos(a-width),radius1*sin(a-width),0 ]
n3=[radius2*cos(a),radius2*sin(a),0 ]
```

But vertexes don't define geometry, they just define where points are in three dimensions. It has no surface or properties of its own and cannot be seen in rendering. If it is not connected to other vertexes through a surface, it becomes an isolated vertex. In addition, when mapping the surface of the model, the vertex position is also used to save the map coordinates. Therefore, when the vertex moves, the corresponding map moves.

**Modification of Vertex**

**Vertex Modifying.** Vertex modifying is the deepest level of adjustment by editing the smallest constituent elements of the mesh surface. A large number of operations in computer graphics are to modify vertexes and simultaneously push and pull the surface according to the connection relationship between vertexes. The relative position of vertexes are changed to meet different modeling need, and affect the topological structure of objects. The vertex topology represents the interconnection between different vertexes in the spatial entity, and any change in the shape of the model will lead to the rearrangement of the vertexes.

The vertex modifying is similar to moving, rotating and scaling the faces or boundaries. Because mesh modifying always affects the location of vertexes. The faces and boundaries of the mesh object follow the moving vertex to a new location specified by the vertex. When an element is scaled, the vertex position is actually scaled. When the face rotates, the vertex position is also rotated, and the face changes direction according to the new position of the vertex. Taking this into account when doing any mesh modifying will make the results more predictable and the algorithm easier to implement.

One or more vertices are removed, and then a repetitive triangulation algorithm is applied to the mesh to piece and keep the surface intact, as shown in figure 4. If a simple method of removing points is used, polygons that depend on those vertices will also be removed, creating a hole in the mesh, which may also cause the mesh shape to change and produce non-ideal polygons.

![Figure 4. Repetitive trigonometry algorithm after removing a vertex.](image)

**Use of Chamfering Algorithm.** When multiple selected vertexes are chamfered, all edges connected to the original vertex will generate a new vertex, and each chamfering vertex will be
effectively replaced by a new face which will connect all new vertexes. These new points are exactly the chamfering distance from each edge of the original vertex to the new point, as shown in figure 5. The new chamfer faces are created with the material ID of one of the adjacent faces and the smoothing group as the intersection group of all adjacent smoothing groups. For example, if an angle of a cube is chamfered, the vertices of the outer corners will be replaced by triangular surfaces, whose vertexes are on the three sides that connect to the original outer corners. The outer sides are rearranged and split to use the three new vertexes, and a new triangle is created at the corner.

Figure 5. Vertex chamfer.

Vertex Control Method Based on Neighborhood Factor

The correct determination of the vertexes to be adjusted is the basis of effective vertex processing. Single or multiple vertexes can be selected, or they can be combined into a selection set. For a complex model, there are thousands of vertexes, and the premise of adjusting the structure of different parts of the part model is to reasonably determine different vertex sets. The newly selected vertexes can be added to the vertex selection set as required, or some selected vertexes can be removed from the set. When selecting vertexes in a view that affect each other, you can hide some vertexes or collections, making it easy to select from the remaining vertexes.

The method of neighborhood factor control is to determine a smaller set of vertexes to affect a larger range.

The influence of neighborhood factor be decided by $damp$, $pinch$, and $bubble$ three parameters.

$$P_f = (D, P, B)$$

Where $D$($damp$) controls the attenuation trend and range of selection set, $P$($pinch$) controls the extruding trend of selection set, and $B$($bubble$) controls the doming trend of selection set. The select value of neighborhood factor is shown in table 1. The form of control curves and model changing are different according to neighborhood factor, as shown in figure 6.

Table 1. The select value of neighborhood factor.

| serial number | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---------------|---|---|---|---|---|---|---|---|
| Damp          | 30| 60| 60| 60| 60| 60| 60| 60|
| Pinch         | 0 | 0 | 1 | 1 | 2 | 0 | 0 | 2 |
| Bubble        | 0 | 0 | 0 | 1 | 1 | 1 | 2 | 2 |
Conclusion

The flexible application of sub-object modeling technology can facilitate the 3D modeling of complex topographic structures. Local special structures such as convex peaks, pits, step surfaces, slopes, terraces, cliff collapses, landslides, steep cliffs, and gully can all be conveniently constructed, and the modeling authenticity and efficiency are greatly improved to achieve satisfactory results, as shown in figure 7. Different from general modeling methods, 3D terrain modeling must be modified with vertex control and other advanced methods, so as to achieve a better effect of special surface shape. The model can use multiple mapping channels, also can add atmosphere, fog, particle system lamp special effects in the scene.

References

[1] Chu Weijun, He Xiaohui, Zhu Zicheng, Rsearch on the Semi-physical Simulation Maintenance Training System of One Engineering Equipment. 2019 International Conference on Modeling, Analysis, Simulation Technologies and Applications (MASTA2019). Advances in Intelligent Systems Research, Volume 168(2019): 34-39.

[2] Zhang Min. 3D Terrain Based on Unity 3D. Journal of Hunan Industry Polytechnic, 2014.06.

[3] Yang Yang, Pan Mao, Wu Gengyu, Liu Yin, Li Kuixing. Sealed Geological Block Modeling in 3D Geological Structural Model. Journal of Computer-Aided Design & Computer Graphics, 2015.10: 1929-1935.
[4] Pei Yanyun, Chen Feixiang. Mesh simplification algorithm based on non-smoothness. Computer Engineering and Applications, No.14 (2013): 174-177.

[5] Sa LiangKun, Zhang DeTong. Polygon Topology Generation Algorithm Research Based On 3d Modeling. Proceedings of 2011 IEEE 2nd International Conference on Software Engineering and Service Science, 2011.07.

[6] Zhang Jing. Research of the Algorithms for Three-dimensional Terrain in Virtual Scene. North University of China, 2012.06.

[7] Huang Xinyuan, Zhang Ruoqing, Gu mei. Inside 3D Studio, Tsinghua University Press, Beijing, 1997.