Design and Implementation of a 3D Dynamic Geometry System

[1]Zheng Liu, [2]Guanhai Liu, [3]Wei Lu, [4]Mao Chen
[1,2,3,4]*National Engineering Research Center of E-learning, Central China Normal University, Wuhan 430079, China*
[1]*Liu.Zheng.jojo@gmail.com*,[2]*L.GuanHai@gmail.com*,[3]*luweilfgl@qq.com*,
[*4]*educhenm@yahoo.com*

Abstract

Dynamic geometry system (DGS) is a computer application that allows the exact on-screen plotting of geometric constructions, and the geometric properties among them remain unchanged when one or several geometric objects are moved. Currently, there are many DGS for plane geometry education, but hardly any DGS are available for space geometry education, especially in China. To aid the teaching and learning of space geometry, we proposed and implemented a three-dimensional DGS named Dragon3D Sketchpad in this paper. Firstly, we made a requirement analysis of the system and presented the system architecture. Then, we designed the interaction and the updating mechanism. At last, we used Microsoft Visual Studio 2008 development tools to implement the system in the Windows environment, and demonstrated the functions by some examples. The results show that geometric objects could be plotted and manipulated easily in three dimensions, the common functions for a DGS such as animation, transformation, locus, iteration and measurement are realized for the spatial objects, and the updating process is very smooth, which means that the Dragon3D Sketchpad is useful in space geometry class.

Keywords*: Dynamic geometry, 3D Dynamic Geometry System, Interaction Mechanism, Update Mechanism*

1. Introduction

Information technology (IT) has a powerful influence on all aspects of our society, from commerce and business to health and entertainment. Obviously, education is not an exception. IT has brought great benefits to the teaching and learning, good examples of which include the wide use of dynamic geometry system (DGS) in geometry education [1]. Dynamic geometry (DG) is a new form of representation of geometry that displays on the computer screen, whose typical characteristics is that when some points, lines or polygons are dragged by mouse or driven by change of parameters, other dependent geometrical objects will change accordingly to maintain the geometric properties imposed initially.

The dynamic feature revealed in DG is of great significance in geometry education [2]. Today, the application of IT in mathematics is largely reflected by the use of DGS, among which The Geometer's Sketchpad(GSP) [3], Super-Smart-Platform (SSP) [4], Geometry Expert [5] and Cinderella [6] are most popular. DGS has proven to be an excellent resource for education [7].

Geometry education, including the plane geometry and space geometry, is one of the difficult and key content of the mathematics curriculum in the secondary school. However, most current DGS mainly target is the teaching and learning of plane geometry. As we know, Cabri3D [12] is the only mature DGS for space education, which is commercial software designed in France and is not available in China. Compared with two-dimensional (2D) DGS, the development of a three-dimensional (3D) DGS is much more difficult and complex in the design and implementation. For instance, as one of the basic operation in DGS, the interaction between the user and the geometric object is much more difficult to handle in three dimensions; moreover, the updating of 3D geometric objects such as polyhedron, sphere and conics, is more complex than that of line, circle and polygon.

Accordingly, in space geometry education, the teachers and students in China have to demonstrate the spatial effect on 2D DGS, or even draw the spatial objects on the blackboard or interactive whiteboard by hand with chalk or whiteboard pen. Drawing a 3D shape in this way become very burdensome, accurate plotting and some spatial-geometric effect such as transparency and lighting effects could not be realized too, not to mention the dynamic transformation and interaction. According to [8, 9, 10], the transition from drawing plane constructs to imagining and manipulating spatial objects is neither natural nor easy. As a result, for the students without enough spatial understanding ability, the learning of space geometry becomes a difficult task without the help of 3D DGS.

For this reason, we design and implement a 3D DGS, named Dragon3D Sketchpad (called Dragon3D for short), to aid the teaching and learning of space geometry. With this system, objects such as plane, polyhedron, pyramids, prism, sphere, cylinder cone and other geometric objects, together with the geometric constraints between them, can be easily constructed, manipulated and viewed in three dimensions.

The remainder of the paper is organized as follows: Section 2 presents the architecture of the system; Section 3 details the design idea of the two core mechanism. Section 4 shows the overall function of the system by some examples. Finally, the conclusion and the future work are proposed in Section 5.

2. System Architecture

2.1. System Requirement Analysis

To provide the teachers and students with powerful and easy-to-use tool in the space geometry, the following four aspects are essential for the functions of a 3D DGS: (1) Geometric objects plotting. Plotting different geometric objects, including the plane geometric objects and space geometric shapes, is the basic function for a 3D DGS. (2) Interactive manipulating. The geometric objects could be selected, dragged and resized by mouse moving, and could be viewed in different direction. Note that when some parts of the shape are moved, the initially established constraints should remain unchanged. (3) Dynamic geometry. Animation, measurement, locus, tracing and iteration are the basic DG functions, which can be used to demonstrate different dynamic effects to visualize the abstract geometry properties and spatial concepts. (4) Courseware making and display. To maximize the educational potential of a DGS, some supplementary functions such as multimedia resource, text editing, and courseware making need to be integrated into the DGS to aid the teaching.

2.2. System Architecture

Dragon3D has a three-layer architecture, i.e., kernel layer, dynamic geometry layer and application layer, as shown in Figure 1. The main function modules of the architecture are introduced as follows.

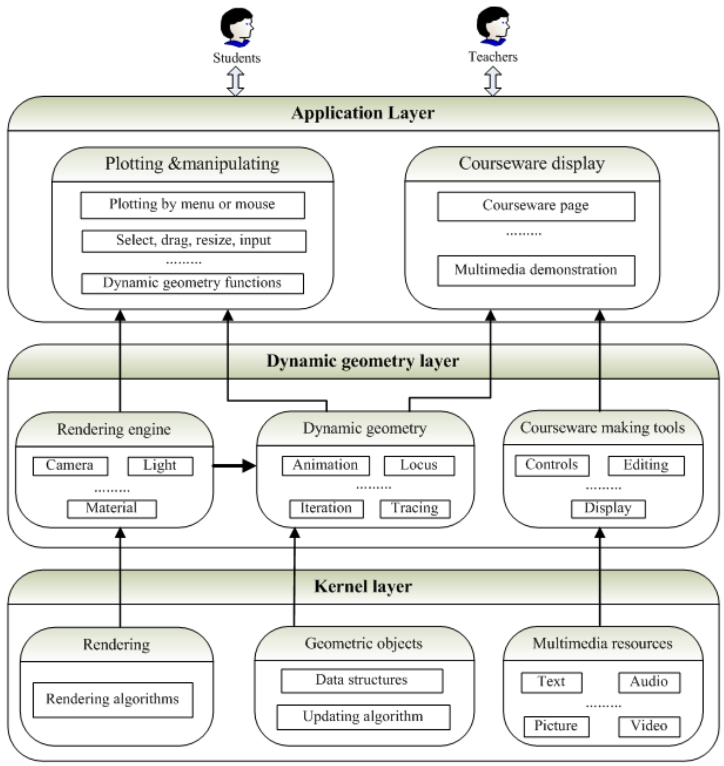

Figure 1. System architecture

2.2.1. Kernel layer

Kernel layer includes the rendering module, geometry objects module and multimedia resource module. The geometry objects and rendering modules mainly offer the algorithms and data structures for geometry objects plotting, updating and rendering, respectively, while the multimedia resource module provides the data and I/O functions of the multimedia resource that commonly used in courseware making.

2.2.2. Dynamic geometry layer

Dynamic geometry layer consists of the rendering engine, the dynamic geometry modules and the courseware making module, which provides a bridge connecting the kernel layer and the user interaction and application layer. Rendering engine module is used to manage the camera, light, material and texture, etc., which determines to a great degree the three-dimensional rendering effect. The dynamic geometry functions such as animation, locus, and measurement are provided by the dynamic geometry module, which is mainly based on the geometry objects-related data structures and algorithms in the kernel layer. Courseware making module offers the commonly used tools to make the courseware, for instance, the text editing tool, the button use to set the property of the animation, and the audio and video play tools.

2.2.3. Application Layer

This layer provides a graphic user interface to deal with the user's operation, which consists of two modules. By using the plotting and manipulating module, the user could conveniently use the menu or toolbar or mouse to plot the geometry object, and manipulate it by selecting, dragging and resizing and other dynamic geometry functions. The courseware display module could be used to produce and display the courseware.

3. Design of Core technology

As an interactive geometry system, interaction between user and geometric objects, and maintaining the initial geometric constraints during interaction, are the two most important and difficult aspects in system design. The implementation mechanism of them in Dragon3D is introduced as follows.

3.1. Interaction mechanism

Interaction in 3D DGS need to solve two problems: the first one is how to pick a geometry object in space; the second one is how to drive it move in three dimensions by the 2D mouse move on the computer screen.

3.1.1. The picking of geometry object

The picking of a geometry object in three dimensions could be realized by a ray tracing algorithm, which is illustrated in Figure2. The ray tracing algorithm is processed as the following steps:

Step1 : When the user clicks on the screen by mouse, the coordinate of the point could be described by a vector [x, y], corresponding to the horizontal and vertical coordinate in the screen coordinate.

Step2 : Transformation matrix is used to convert the point from screen coordinates to world coordinates, i.e., $[x, y] \rightarrow [x', y', z']$.

Step3 : By the position of point in world coordinates and the position of the camera, a ray could be constructed in the 3D space, and all the geometric objects intersecting with the ray could be found and kept in list L.

Step4 : Return the geometry object in L which is the closest one from the camera.

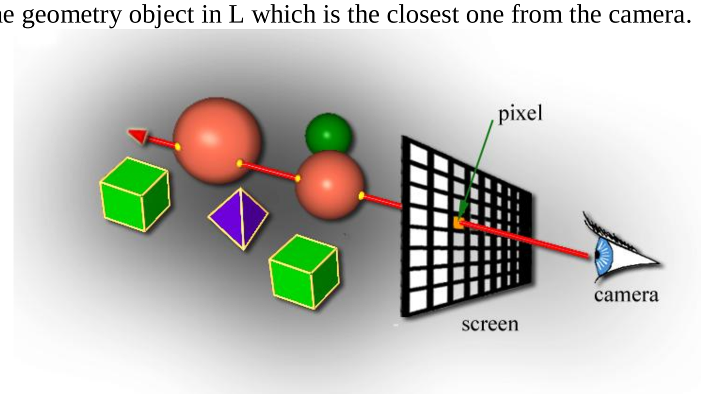

Figure 2. The demonstration of ray tracing algorithm

3.1.2. The dragging of geometry object

Picking a geometric object is only the first step for manipulating. After selecting a geometric object, a user usually want to drag or resize the object by mouse move to see it from different direction or size. However, as we know, when the mouse moves on the screen from point A to point B, it is impossible to drive the object move in three dimensions by the 2D vector $\overrightarrow{AB}$. As mentioned above, a point on the screen could be converted into a line by ray tracing algorithm, then point A and point B could correspond to two lines. If we can find a reference plane for the selected object, then the two lines would intersect with the reference plane and two intersecting point A', B' could be obtained. Obviously, $\overrightarrow{A'B'}$ is a 3D vector, move the object according to which is feasible and reasonable. Then, the key of the dragging problem is to find a reference plane for a geometric object. The corresponding method is described below.

Step1 : Search all the parent nodes of the object to find whether all parent nodes have reference plane. If not, it means that the object can’t be dragged, return;

Step2 : Search all reference plane of the parent nodes to find whether they have the same normal (parallel or coincide). If not, return; otherwise save the normal;

Step3 : Through the point-normal equation of the plane we can get the reference plane by following formula: $\vec{n} \bullet (X - P) = 0$ (the normal achieved at step 2 and the point is the origin pixel world coordinates).

3.2. Updating mechanism

3.2.1. Data structure

Directed acyclic graph (DAG) is used as the data structure of updating mechanism because of the information of DAG only flows in one direction, which is similar to GSLP [6]. The DAG consists of nodes and arcs. The nodes represent a geometry object or some constraints, and the arcs represent the dependant. So if it is an arc between two nodes, it means that one node depends on another, in other words there are parent-child relationship between them. As shown in Figure 3, a circle is constructed of three points, and a line is tangent with the circle, the corresponding DAG is shown by the right part of Figure 3.

Updating process starts whenever a free or a semi-free object is moved. The updating changes the state of a node, and then the changes further passes to the nodes depend directly or indirectly on it. In other words, updating will propagate to all the dependent elements. To speed up the update process, a pre-computed partial ordering of all its dependent nodes for each node is obtained by DAG. The node initiating the update uses this ordering to make sure that every dependent node will update once and only after all its parent nodes have been updated. So whenever an object is changed, we just have to iterate through the list of dependent elements and update each of the algorithms only once.

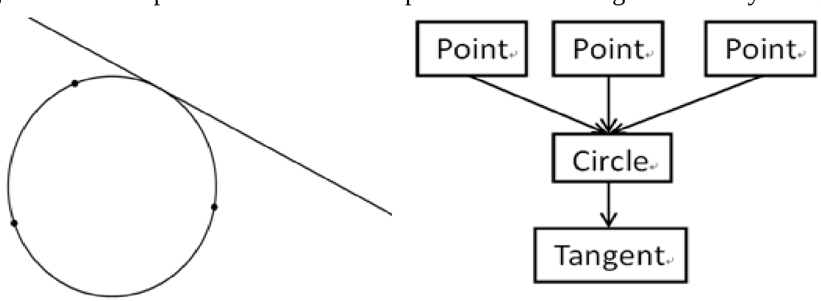

Figure 3. A circle on three points and one of its tangents

3.2.2. Updating process

Based on DAG, the updating process could be demonstrated by the time loop in Figure 4. Time loop is like the event concept in window system and some event could make a updating happen. Assuming that the user pick a free or semi-free object on the screen and drag it, then each motion event generates a cascade of operations, the object itself and all objects depending on it must update at once, and the series of operations will repeat each time whenever the object is moved. Each iteration process is represented by one time loop.

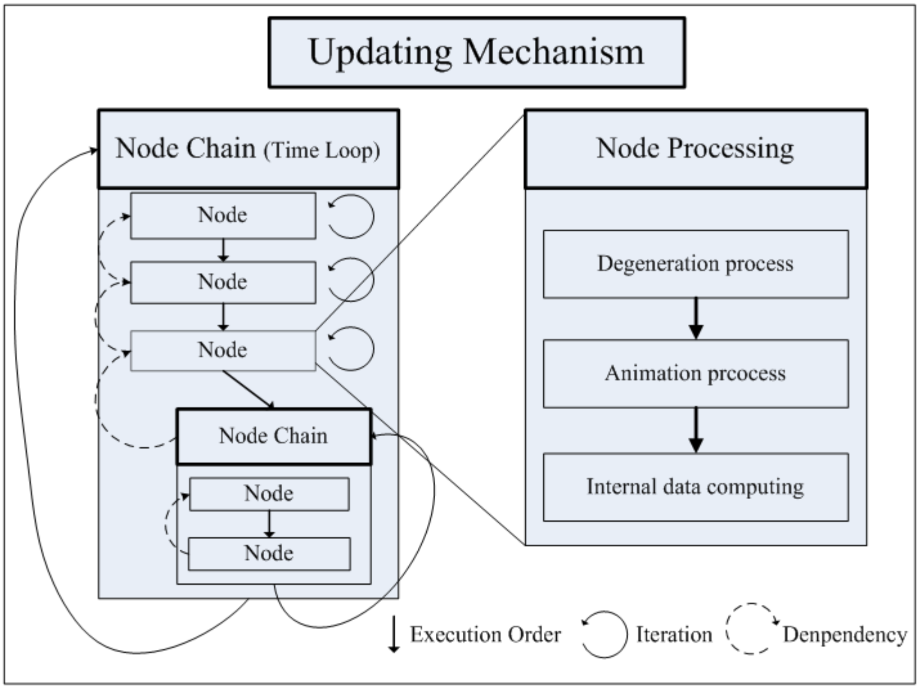

Figure 4. The demonstration of updating mechanism

An updating process is empowered to decide whether it has to be re-executed or not and the whole updating is repeated until a required result is achieved, i.e., iteration reaches the bottom of the dependent node. Node processing represents a computation process whenever the updating happens, which consists of three main parts: degeneration process, animation process and internal data computing. The degeneration process deals with the degeneration of nodes by user's interaction, the animation process deal with the computing of the new position of a node and all its dependent nodes when a animation happens, and the internal data computing deal with the parameters changing by the user's interaction.

4. Examples

Based on the above mentioned system architecture and module design, we used C++ language on the Microsoft Visual Studio 2008 development platform to implement the Dragon3D in the Windows environment, and the 3D rendering is based on the OpenGL API. The functions of Dragon3D mainly are geometry object plotting, constraint plotting, transformation, measurement, animation, locus, tracing, iteration, algebraic expression calculation, and courseware making. Due to space limitation, only four most representative examples are given here to demonstrate the functions of Dragon3D.

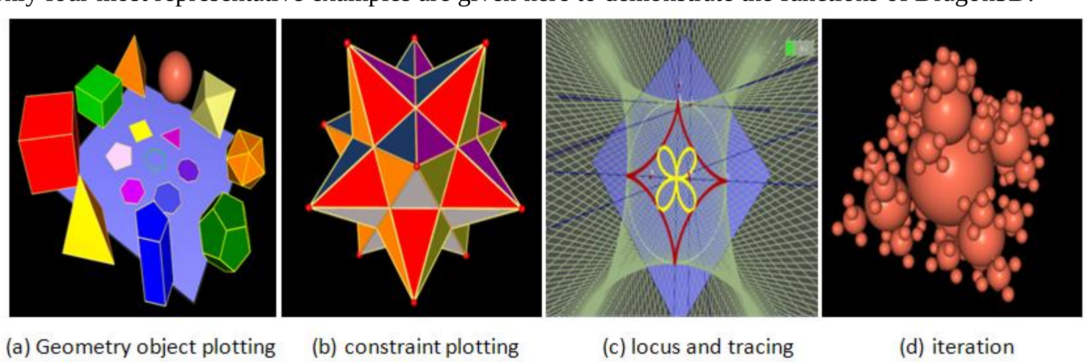

Figure 5. The function demonstration of Dragon3D

- **Geometry object plotting :** As shown in Figure 5.(a), all the commonly used geometric objects in secondary school could be constructed through Dragon3D.
- **Geometry constraint plotting :** One of the key functions of DGS is the constraint plotting, i.e., the geometric relationships (such as parallel, midpoint, perpendicular, etc) among the geometric elements are determined after plotting, and the relationships remain unchanged when the geometric object is moving. The figure in Figure 5.(b) shows that very complex geometric shape could be plotted by constraint plotting of Dragon3D.
- **Locus and tracing** : Locus and tracing is used to observe the trajectory path of geometry object's motion by dragging or animation, which is of great educational significance. Figure 5.(c) gives a beautiful figure resulted from tracing.
- **Iteration :** Iteration refers to the act of performing some mathematical process such as a computation, algorithm, or construction on some initial value, and then repeats the process on the result until a given step is met. Iteration is a touchstone to test the update efficiency of a DGS. From the shape in, we can see that the updating process is very smooth.

5. Conclusion

In this paper, we presented the design and implementation of Dragon3D, a 3D DGS for space geometry education in China. The architecture of the system and the design of core technology are presented in detail, and some examples are given to demonstrate the function of the system, which shows that Dragon3D could give students a tangible, visual way to learn geometry, and it could also help the teaching of space geometry by visual and dynamic illustration of abstract geometric concepts and spatial geometry shapes.

In the next step, the major work is to research about the automated reasoning of space geometry and integrate it into Dragon3D, so that it could meet the full range of requirement of teaching and learning of space geometry.

6. Acknowledgment

This work was supported by National Natural Science Foundation of China (No. 60903023), Specialized Research Fund for the Doctoral Program of Higher Education (No. 200805111011) and The Major National Science and Technology Special Projects High-Nuclear-Based Project of China (No.2010ZX01045-001-005-3).

7. References

[1] K. Ruthven, S. Hennessy, "Constructions of dynamic geometry: A study of the interpretative flexibility of educational system in classroom practice", Computers and Education, vol. 51, no. 1, pp. 297-317, 2008.

[2] B. Ubuz, I. Üstün, A. K. Ebraç, "Effect of dynamic geometry environment on immediate and retention level achievements of seventh grade students", Eurasian Journal of Educational Research, vol.35, pp. 147-164, 2009.

[3] N. Jackiw, "The Geometer's Sketchpad", CA: Key Curriculum Press, USA, 1995

[4] J. Z. Zhang, C. Z. Li, "An Introduction to Logical Animation", In Proceeding of the 6th Computer Algebra and Geometric Algebra with Applications, pp. 418-428, 2004.

[5] X. S. Gao, C. C. Zhu, S. C. Chou, "Geometry expert", Nine Chapters Publishers, Taiwan, 1998

[6] U. Kortenkamp, Foundations of Dynamic Geometry, Ph.d Thesis, Swiss Federal Institute of Technology Zurich, 1999.

[7] F. Botana, J. L. Valcarce, "A dynamic-symbolic interface for geometric theorem discovery", Computer and Education, vol. 38, no. 3, pp. 21-25, 2002.

[8] A. Gutierrez, "Visualization in 3-dimensional geometry: In search of a framework", In Proceeding of the 20th conference of the international group for the psychology of mathematics education, pp. 3-19, 1992.

[9] M. Battista, “On Greeno's Environmental/Model View of Conceptual Domains: A Spatial/Geometric Perspective”, Journal for Research in Mathematics Education, vol. 25, no. 1, pp. 86-94, 1994.
[10] H. Hauptman, “Enhancement of spatial thinking with Virtual Spaces 1.0”, Computer and Education, vol. 54, no. 1, pp. 123-135, 2010.
[11] S. Isotani, L. O. Brandã, “An algorithm for automatic checking of exercises in a dynamic geometry system: iGeom”, Computer and Education, vol. 51, no. 3, pp. 1283-1303, 2008.
[12] E. Bainville, J. M. Laborde, Cabri3D v.1.0.3, Cabrilog, http://www.cabri.com
[13] M. H. Su, P. T. Yu, “A Directable and Designable Course Recording System”, JCIT: Journal of Convergence Information Technology, vol. 6, no. 3, pp. 234-243, 2011.
[14] Y. Zhao, L. Y. Zhang, “Dynamics Modeling and Simulation of Information Resources Allocation of R&D Cooperation in China”, JDCTA: International Journal of Digital Content Technology and its Application, vol. 5, no. 2, pp. 21-33, 2011.