Automatic Navigation in 3D Geological Virtual Environments

M.Sc. Oltion Fociro

M.Sc. Jeton Pekmezi

Faculty of Geology and Mining, Polytechnic University of Tirana, Albania
oltion.fociro@fgjm.edu.al, jpekmezi@gmail.com

Doi:10.5901/mjss.2015.v6n1s1p513

Abstract

With the increased need for information in the field of geology, it became increasingly necessary to use technological tools, such as Virtual 3D modeling. What does it mean geological modeling? Geological modeling or geomodelling is an applied science which creates computer representations of certain parts of the earth's crust based on geophysical and geological surveys carried out under the ground surface. Geological data typically characterized by three dimensionality, multi-z, and multi-level [1]. These three essential features, although coexistent not necessarily constitute strong point processing geological. They should actually be the result of the full integration of the various components that determine spatial geological structures. Three dimensional modeling of geological bodies is an important feature of geosciences and one of the key technological tools for building digital mining. In recent years, the increasing demand of 3D geological simulation information on mining, geology and environment in the development of modeling and 3D visualization of geosciences is very fast.

Keywords: *3D automatic navigation, 3D geological models, 3D navigation, 3D geological environment loader, Pixel shader, Vertex shader, Programming, Server-Client*

1. Introduction

Virtual camera control is a fundamental activity in almost all three-dimensional virtual environments and geological especially. Placement, orientation and movement of the camera directly affect the users' familiarity with the virtual environment, defining which objects you will see in the virtual scene, and the perception of spatial relationships between them. For example, in a virtual environment geological movement or a non-optimal camera positioning can affect normal geological processing (e.g. because some of the geological layers are not visible).

In the majority of applications, the camera control is carried out manually, but it can cause problems for non-expert users or to make camera control more difficult. One of the reasons for these problems is the fact that users do not have knowledge of the virtual environment, leading to difficulties in obtaining geological information. Thus arises the need for automated positioning and movement of the virtual camera. Most of the algorithms currently used to manage the movement of a virtual camera techniques based on "Path Planning" and derived from robotics algorithms for tracking 3D objects. These approaches, however, face difficulties should be taken into account when some elements in the scene (e.g. visualization of geological layers), or when required complex features that cameras should respect (e.g. visualization of some facilities throughout camera movement).

Regarding the generation of static cameras, are several algorithms proposed in the literature (e.g. [5]) able to calculate the position and orientation of a virtual camera in order to display the image from the camera itself, satisfies a set of constraints camera type (e.g. size, position and angle of the objects in the scene). The problem is solved by these algorithms in the literature called "` Virtual Camera Composition (VCC) '".

2. The Occlusion Problem

One of the most fundamental problems affecting the virtual shooting of a scene is getting a clear view of the subjects that are in it. This statement raises obvious problems, some of which are very interesting about automatic camera control in a three dimensional environment. For example, in many video games the camera should follow the avatar controlled by user keeping a predetermined distance and orientation by avoid or minimize occlusion. Often the algorithms used are not generating appropriate framing or insignificant changes unexpected performances by confusing the scene player or the browser.

Before looking at how movement affects occlusion the camera, the problem should be examined. First of all, a basic quality of confinement is it depends only on the position of the camera and not its orientation. Apart addition, the region of space in which an object is occluded, which will now call occluded space, may be discontinued or otherwise consisting of separate regions.

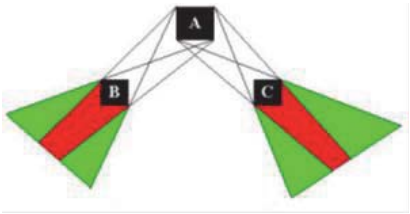
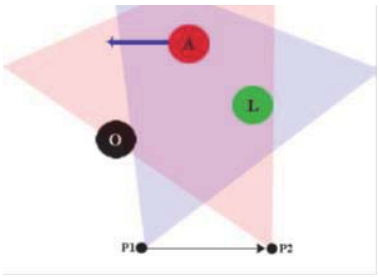


Figure.1. Diagram in two dimensions in which is shown an example of occlusion.

For example, in Figure.1. is shown scene with three objects (represented in two dimensions). Areas in which object is completely occluded are shown in red, those in the occlusion which is partially shown in green. Space in the A is occluded which is divided into two unrelated parts.

In the evolution of a dynamic virtual scene the moving objects can edit the occlusion spaces enabling the camera to be found in a space of occlusion. For this reason, the camera should move on and find a new position and orientation respect to the limitation of not confinement, in every moment of time t. This means that the facilities of the scene should be present in the image produced by the new camera.



Figur.2. The diagram in two dimensions in which is shown an example of movement camera.

In figure (Fig. 3.3) shows a two-dimensional scene in which the present three buildings, two of which are the objects of interest in the scene (green building and red). In the case in which the red object moves according to The translation vector reported then the camera moves from position P1 to position P2 by and changed its orientation. The new camera always keeps involved objects of interest, avoiding occlusion one of them.

Finally, occlusion depends only on the position of the camera, not the orientation 30.

3. Description of the Algorithm Created

The problem of camera movement is directly related facilities should incorporate this camera. It is therefore chosen to create a system that generates a movement where every new position of non-occlusion of the objects observed with interest the scene and also some limitations provided by the user.

CamLib Solver Library (Virtual Camera Library) (P. Burelli, L. Di Gaspero, A. and R. Ranon, 2008 [5]) is able to calculate a camera, to comply with the restrictions set by the user, in the visualization which (View Volume) are objects of interest (usually a landmark and an avatar).

Given that CamLib Solver is very slow to use in real-time system that uses must do this as often as possible as it can slow down the rendering process. The library has a response time between 0.1 and several seconds depending on the number of objects to be included, they have limitations and virtual environment. This means that the "frame rate" is between 10 and less than 1 frame per second.

- The general idea of the algorithm can be summarized in this way on two main points: calculation of a camera through CamLib Solver, giving as input a set of constraints that belong to some particular objects of interest

(no occlusion, presence in the image, the distance from the camera), including the user's avatar.

- calculating, for each frame, small modifications to the camera parameters (position and orientation), in such a way that:
  - The objects of interest are inside "view volume" of the camera;
  - objects of interest are not occluded from other objects

In other words, the new camera should respect the original camera settings.

If the new configuration of the camera does not respect one of the limits will then generate a new camera through CamLib Solver (thus performing a cut film) and the above mentioned process will restart from scratch.

It should be noted that this algorithm does not provide a solution to every situation: for example, CamLib Solver may not be able to generate a camera that observes all given constraints. To prevent this problem, the idea is to apply until CamLib Solver generate a camera that involves at least avatar and then execute our algorithm starting from camera and considering only objects that are visible in the generated image. So in this way, tolerated visibility not ever been the object of interest, but does not tolerate viewership of avatar: if CamLib Solver library generates a camera that involves no avatar, avatar until then executed again be visible in the image produced. In the following algorithm will only justify the venues involved on the image produced.

In addition, the idea is to generate a new configuration of the camera in accordance with the evolution of dynamic scene that is in accordance with the movement of one (or more than one) of the objects of interest. To facilitate the work will be divided logically calculated in two separate parts, the calculation of the camera and the other to calculate the orientation.

To find a potential position can generate a predetermined number of positions and then evaluate each of them. To evaluate each position estimate a "quality factor" (real number) for each object of interest. So for each position will be the Quality factor as are the objects of interest. Once evaluated positions then proceed with selecting the best position to place the camera.

Another key component in the system is the orientation of the camera. That's because a good orientation is able to avoid logging. For example, in a three-dimensional scene camera should involve two objects of interest. Preceding the distance between two objects that allow both to be in Volume View, where the camera is oriented toward one of the objects and the other is outside the View Volume should generate a cut. To avoid this situation should be included if "n" cameras will focus objects to the midpoint of objects. This technique is a solution that is more likely to engage all objects of interest.

Once determined the position and orientation of the camera that moved and oriented on the basis of the calculated data and the process repeats.

4. Test

For each trajectory and for each scene, we made some tests by changing the number of objects of interest that the system must manage.

Number of objects of interest comes step by step decreasing. Started with three objects of interest (red, blue, and yellow), then the two (red and blue), a (red), and in the end only avatar (in the third stage avatar used only with the addition of some restrictions). Objects of interest are static, ie not moving, and more their number is limited because CamLib Solver Library administers a small number of objects of interest. As can be seen in the images of scenes, objects of interest in red and blue have a little distance between them and the object of interest is far away objects yellow interest of red and blue.

In the first two scenes are added some restrictions to ensure the presence of the image produced every object of interest and avatar, while in the third phase besides visualization restrictions are also placed two other limitations: first limiting distance from the camera avatar and the second a vertical angle.

Figure. 3. Test scenes

Tests conducted have shown that the system allows to make a move the camera based on the evolution of the scene or rather the movement of the avatar. The results obtained allow that the method used is very effective in generating a trajectory visualization camera observing the object of interest and the image produced avatar. Also the tests performed show that the system is not very efficient with increasing interest objects in the scene or its complexity.

5. Conclusion

Camera Control has always been an important tool for communicating information in a virtual environment. Currently inside three dimensional interactive applications of camera movement is created based on the user's direct control or through simple algorithms that try to follow a target. In literature, proposed more sophisticated approaches that take into account the type cinematic features in the image created by the camera, but have not yet found practical application. In this thesis proposed a solution based on the management and prevention of occlusions for the problem of camera motion for dynamic scenes. The proposed solution was implemented and tested successfully.

The main objective of this system is to facilitate the users in the process of controlling the camera in a three dimensional environment. More natural approach to this kind of problem is to generate a position and orientation of the camera for each frame based on the evolution of objects of interest.

User (or application), this type of approach, specifying the objects of interest in the scene and the system tries to visualize all objects of interest in the image produced for each frame. However, the problem becomes more complex with increasing interest objects to be managed.

References

Arnav Jhala (2006). Darshak - An Intelligent Cinematic Camera Planning System. Proceedings of the 20th National Conference on Artificial Intelligence (AAAI 2006), Boston, MA. Liquid Narrative Group, Department of Computer Science North Carolina State University 890 Oval Dr, Raleigh, NC - 27606.

Alexander Hornung, Gerhard Lakemeyer, Georg Trogemann (2003). An Autonomous Real-Time Camera Agent for Interactive Narratives and Games. Intelligent virtual agents. International workshop N.4, Kloster Irsee. Laboratory for Mixed Realities, Academy for Media Arts Cologne Am Coloneum 1, 50829 Kholn, Department of Computer Science V, Aachen University of Technology, Ahornstr. 55, 52056 Aachen Germany.

Bourne Owen, Sattar Abdul (2005). Applying Constraint Weighting to Autonomous Camera Control. Arti_cial Intelligence and Interactive Digital Entertainment. Institute for Integrated and Intelligent Systems, Grifith University PMB50 Gold Coast Mail Centre, QLD 9726.

CHE De-fua, WU Li-xina ,YIN Zuo-rub (2008). Multi-scale spatial modeling for geological body based on GTP model. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. Vol. XXXVII. Part B2. Beijing

E. Marchand, N. Courty (2002). Controlling a camera in a virtual environment. The Visual Computer Journal 18. IRISA - INRIA Rennes, Campus universitaire de Beaulieu, 35042 Rennes Cedex, France.

Fred Charles, Jean-Luc Lugrin, Marc Cavazza, Steven J. Mead (2002). Real-Time camera control for interactive storytelling. Game On, 2002. School of Computing and Mathematics University of Teesside Middlesbrough, TS1 3BA, United Kingdom.

Paolo Burelli, Luca Di Gaspero, Oltion Fociro, Roberto Ranon (2008). Virtual Camera Composition with Particle Swarm Optimization. Proceedings of the 9th international symposium on Smart Graphics. University of Udine, via delle Scienze 206, 33100, Udine, Italy.

Pickering H. Jonathan (2002). Intelligent Camera Planning for Computer Graphics. PhD Thesis, Department of Computer Science, University of York York, YO10 5DD, UK.

Steven M. Drucker, David Zeltzer (1994). Intelligent Camera Control in a Virtual Environment. Proceedings of Graphics Interface. MIT Media Lab, MIT Research Laboratory for Electronics Massachusetts Institute of Technology Cambridge, MA. 02139, USA.