SOFTWARE APPLICATION FOR EXPLORING A VIRTUAL SOLAR SYSTEM

Arnold Konrad SIPOSS, Cosmin ŞTIRBU, Florentina Enescu
University of Pitesti, Romania
konrad.siposs@gmail.com, cosmin.stirbu@upit.ro, enescu_flor@yahoo.com

Keywords: educational software, C# functions, .Net Framework, OpenGL

Abstract: *This paper presents a virtual environment where you have the ability to move around the Sun, stars, planets and natural satellites. This is educational software that uses the OpenGL library and C# programming language. The environment consists in a virtual space, centered on the Sun. Orbiting the Sun are the Planets of the Solar System that are located at correct relative distances. In the System Planets orbits are implemented which are drawn using trigonometric*

equations..

1. INTRODUCTION

In present there are many software applications for the virtual simulation of the solar system, but we preferred to create our own simple and efficient software that can be easily used. The study for creating my application took me to a model created by NASA. Their application named "Eyes on the Solar System" contains data and characteristics of objects in our solar system and missions made in space by astronauts, but the less good part is that the application is hard to use due to its complexity, that is why I tried to keep my application simple and accurate (Fig. 1).

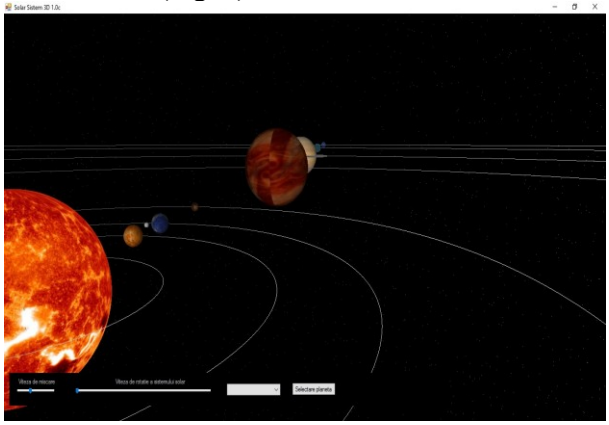

Fig. 1 Application interface

The Application was created using C# programming language to exploit Virtual Solar System. Taking into account that the Solar System is made up of stars, planets, satellites and the most important star in our system the sun, the application will be for information, documentation and education.

This software consists of several items related to the formation of the solar system, so we can follow in detail a few important aspects of application: Universe, Solar System, the Sun, Planets, Moons and Stars.

The solar system consists of a central star, the Sun and all the heavenly bodies smaller traveling continuously around it. Smaller bodies include eight planets: Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, and Neptune, who are blinded by more than 140 satellites. (Only Mercury and Venus have no moons). In addition, the solar system contains asteroid and millions billions of rocky ice comets. All these objects are kept together in a group of Sun's gravity. [1]

The sun is a star, a hot ball of incandescent gas in the center of our solar system. Its influence extends far beyond the orbit of our

planet. Without power and intense heat of the Sun, there would be no life on Earth. And although it is special for us, there are billions of stars like our sun scattered across the Milky Way galaxy. [2]

A planet is a celestial body of considerable mass orbiting a star that does not produce energy by nuclear fusion. For this reason, the planets are much cooler than stars, the do not emit their own light but can reflect starlight. Basically, the planets could provide conditions for the emergence of extraterrestrial life until a few years ago was known only to the 8 planets of our solar system (plus dwarf planet Pluto). [3]

In astronomy, moons are defined as celestial bodies running side rotation around a planet or star. The best known is the Earth's satellite, namely Luna even if the two are quite similar in size to be considered a system. Movement of most satellites is direct, from west to east and in the same direction as the planets around which orbits. Only a few satellites of the major planets rotate in reverse. [4]

2. APPLICATION DESIGN AND IMPLEMANTATION

The program was designed with the idea to cover the needs of schools or high schools in terms of imagination and intellectual development. Starting from the idea that the universe is infinite in time and space, infinite variation in the forms that we take the matter in the process of its development, the application is designed to bring in a visual context that is not possible to be processed by human's eyes, because the universe is too big, too vast and too distant.

As a first step run-time depth study of the solar system and all the planets that compose the system was underlying implementation steps. Using accurate information taken from the NASA website were created all the elements in the application. The interface was created using the Microsoft Visual Studio 2015 and C # programming language.

C # is an object-oriented programming language, simple and modern. Programming style is closer to Java and C ++, which is why the transition from one language to another is intuitive.

The language itself, like the other programming languages in the same category, contains a set of predefined types, it is not necessary to include any namespace. The common types are: string, object types integer signed and unsigned types floating point numbers, Boolean types and decimal. [5]

At the time of implementation, it was necessary a detailed study of the Open GL library, which helped me create the objects.

OpenGL is strictly defined as a software interface to graphics hardware. In essence, it is a 3D graphics and modeling library that is extremely portable and very fast. Using OpenGL, you can create elegant and beautiful 3D graphics with nearly the visual quality of a ray-tracer. The greatest advantage to using OpenGL is that it is orders of magnitude faster than a ray-tracer. It uses algorithms carefully developed and optimized by Silicon Graphics, Inc. (SGI), an acknowledged world leader in computer graphics and animation. [5]

OpenGL is intended for use with computer hardware that is designed and optimized for the display and manipulation of 3D graphics. Software-only, generic implementations of OpenGL are also possible, and the Microsoft Windows NT and Windows 95 implementations fall into this category. Soon this may not strictly be the case, because more and more PC graphics hardware vendors are adding 3D acceleration to their products. Although this is mostly driven by the market for 3D games, it closely parallels the evolution of 2D Windows-based graphics accelerators that optimize operations such as line drawing and bitmap filling and manipulation. Just as today no one would consider using an ordinary VGA card to run Windows on a new machine, soon 3D accelerated graphics cards will become commonplace. [6]

The application contains a single form called Main, form in which all elements and all application classes are merged to create our solar system scene.

Solar System and Camera classes are the classes whose structure defines the entire system and how we view existing elements. Besides these two classes, there will be the

InitialiseOpenGL and Texture classes, which are designed to initialize graphics and load textures list, these classes are mostly based on OpenGL library.

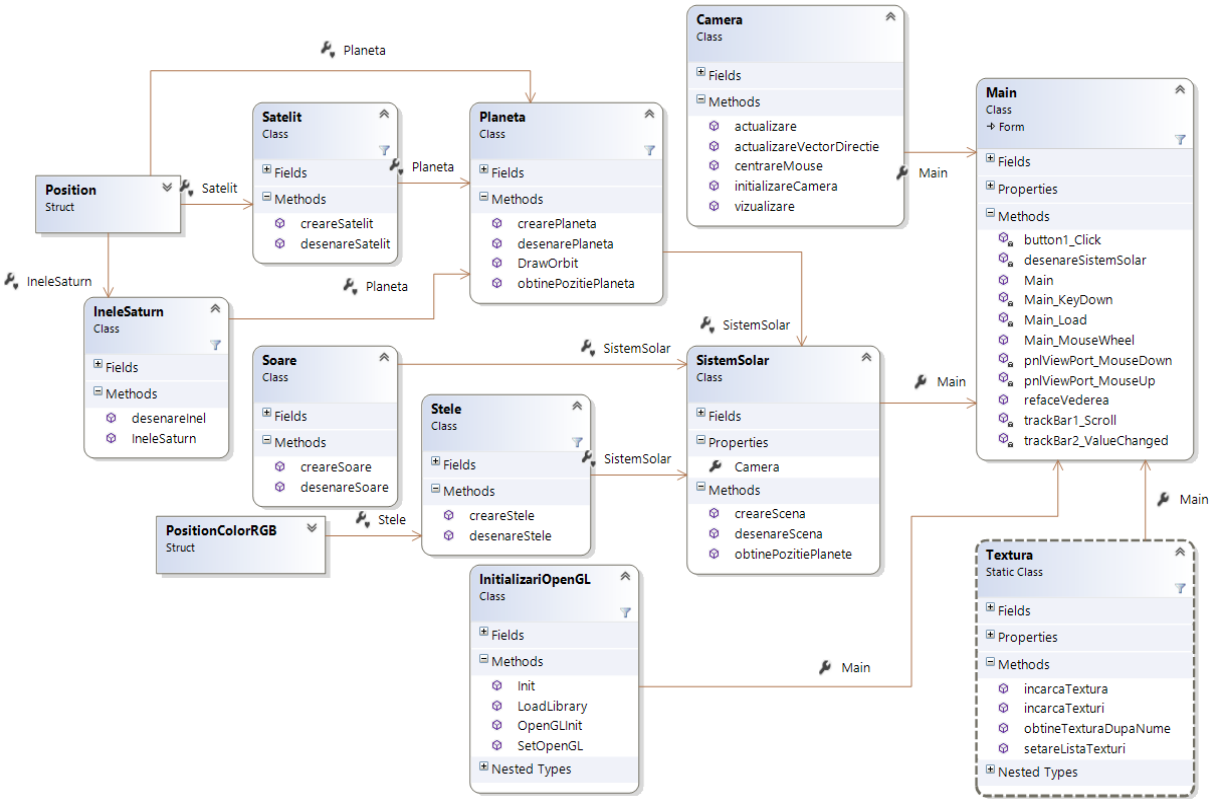

Fig. 2 Class diagram

Solar System class includes three classes (Fig. 2): Planeta, Sun and Stars, in these classes are described the heavenly stars, objects that will be present in our application.

The Planet class will use Position structure to initialize positions of the planets in the solar system.

The next class will be Satellite Class, in which will be described moons of the planets, but also the ring of Saturn class, which will be used to draw to rings of Saturn Planet.

As the scene is infinite and the elements must be drawn continuously, I introduced an element from Windows.Forms called Timer, which automatically calls every 5 milliseconds my described methods for drawing the Solar System.

private void desenareSistemSolar (object sender, EventArgs e)

{

Gl.glClear(Gl.GL_COLOR_BUFFER_BIT | Gl.GL_DEPTH_BUFFER_BIT);

Camera.actualizare (movement);

sistemSolar.desenareScena ();

WindowsApi.SwapBuffers (vpHandle);

Gl.glFlush ();

}

Because this is the class that contains the interface, here will be the actions in my other classes, which will include options that the users use to manipulate the elements of the system. Among the entries we can manipulate the movement of the solar system.

Using the left mouse button will be turning the vision, and by moving forward or backward the mouse wheel will normally move forward and backward through increment or decrement a constant called *forwardSpeed*.

Another entrance will have consequences for the movement of the solar system, it is called trackbar, that element will increment or

decrement the speed of moving objects (planets, satellites, Sun) by the constant *rotationFactor*.

An important component of the input elements is the speed of movement through user system that can be manipulated by a trackbar, it radically changing the speed at which we can advance through the system. But still the most interesting option available to reach users is the ability to jump to a particular planet, which will provide direct vision of the planet on which you selected via a *ComboBox* and the action will be completed by pressing the neighborhood button, besides the vision offered on selected planet, a *RichTextBox* also appears that will be a panel with information about the selected planet, which is the way you can learn all the details of the position and structure of that planet.

Solar System class is the class that combines all the elements of space. Within this class are initialized object class planets, stars, and the sun, which will constitute the main elements of the solar system. This class contains an enumeration type element called Planets, which is how we map the names of the planets from their description. Also in this class can be found the Position structure, a structure that contains the x, y and z coordinates of the elements in the system.

The class contains objects, planets, sun and stars. Object planet is actually a list where all the planets properties were added to the constructor of the class planet, through which were described the elements that help create a planet, for example in this section I have added the Earth:

planete.Add (new Planet (6378, Planets.Earth, new Position (149 600 000, 0, 0), 0.99f, 365, "earth.jpg", true, false));

The eyes you are looking in the Solar System are controlled by the Camera class, which controls the vision and the movement in the space. Because I have added controls similar to planes movement, I used yaw for turning the vision from left to right and pitch to turn the vision up and down.

The class responsible for graphics initializations is called *OpenGLInit(),* which is called in the main class to describe workspace which we will use to draw all other objects. In this method we will call OpenGL features to initialize the model matrix, to set the viewport, to create the effect of light and shadow and to select the model projection. Projection mode is very important because this is the feature that enables you to see 3D objects and not 2D. The projection mode I used is perspective model, which contains zNear and zFar constants that specify the visibility of your eye.

3. RESULTS

As seen in the pictures below, the interface is easy to use; you just have some user controls in the bottom of the screen for changing speeds around the System (Fig. 3 and Fig. 4).

At the time of the implementation, it was necessary a detailed study of the Open GL library with which the objects were created. Of course, software engineering is a complex area and there were problems in implementation. Among the most difficult problems of implementation may be mentioned adding textures of objects. This could be achieved by studying specific books about Open GL, the book there is in the online support (OPEN GL BIBLE - Waite Group Press).

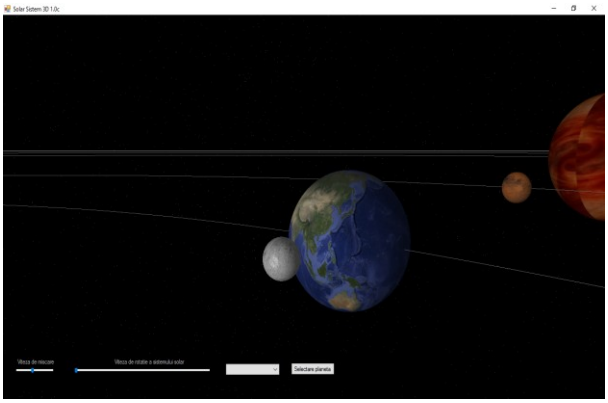

Fig. 3 Application screen capture 1

Planets have texture mapped with the help of the OpenGL library, in order for the System described to look the same like the real System just outside our planet.

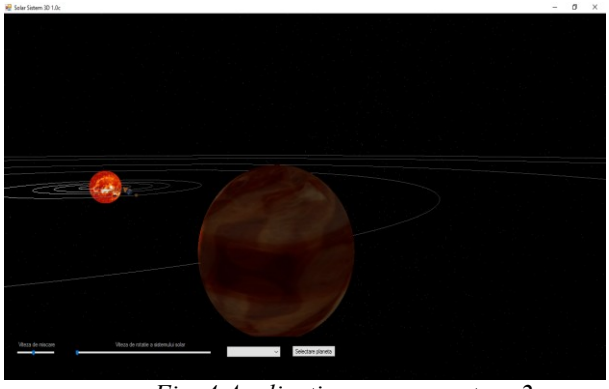

Fig. 4 Application screen capture 2

4. CONCLUSIONS

Nowadays, most people do not have an education well defined on what is happening outside our planet, they do not have a clear vision about how our planet spines around the sun by using gravitational forces acting the revolution movement of the planets on the orbit around the sun. For anyone should be obvious that we need a general knowledge as widely as possible on the things present in our life, and this application comes to their aid in order to discover how the elements of the solar system are positioned and how it moves.

Most of the software's which are developed nowadays are either too expensive or open source but lack graphics and functions for describing the perfect environment.

In future we can make improvements to this application both from a scientific viewpoint (e.g. elliptical orbits for the planets, more accurate equations) and the ergonomically (appearance, controls etc.).

5. REFERENCES

[1]. Wikipedia, the free encyclopedia, "Solar System", accessed 02.07.2016 https://en.wikipedia.org/wiki/Solar_SystemR

[2]. Wikipedia, the free encyclopedia, "Sun", accessed 02.07.2016 https://en.wikipedia.org/wiki/Sun

[3]. Wikipedia, the free encyclopedia, "Planet", accessed 02.07.2016 https://en.wikipedia.org/wiki/Planet

[4]. Wikipedia, the free encyclopedia, "Natural satellite", accessed 03.07.2016 https://en.wikipedia.org/wiki/Natural_satelliteaw

[5]. Herbert Schildt, "C#", published by ALL, Bucharest, 2002

[6]. Waite group Press, Macmillan Computer Publishing, "Open GL Super Bible", 1996

[7]. Presented at local Scientific Communication Session for Students, 14 May 2016. Awarded with 4th place.