3D SOLAR SYSTEM VISUALIZATION

Submitted in partial fulfillment of the requirements of degree of

Bachelor of Engineering

by

Kajal Bagwe (01)

Apurva Khot (14)

Astha Sawant (22)

Guide:
Mrs. Rane Prajakta

DEPARTMENT OF COMPUTER ENGINEERING

Sindhudurga Shikshan Prasarak Mandals's College of Engineering.(Kankavali)

(2021-22)

CERTIFICATE

This is to certify that the project synopsis **for Subject Computer Graphics Lab** entitled " " is a bona fide work of **Kajal Bagwe (01), Apurva Khot (14), Astha Sawant(22)** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **Bachelor of Engineering** in **Computer Engineering**.

(Mrs. Rane Prajakta)
Supervisor/Guide

(Prof. D. P. Mhapasekar) (Dr. A C. Gangal)

Head of Department Principal

Mini Project Report Approval for S.E.

This project synopsis entitled **" "** by **Kajal Bagwe (01), Apurva Khot (14), Astha Sawant(22)** is approved for the degree of **Bachelor of Computer Engineering**.

Examiners 1.————————

————-

2.————————————-

Date:

Place: Kankavali

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, We have adequately cited and referenced the original sources. We also declare that We have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Name and signature of students :

1. Kajal Bagwe
2. Apurva Khot
3. Astha Sawant

Date :

Abstract

Computer graphics is the process of making the design, 2D, 3D and animation of an object. Computer graphics can do many things, including modeling, simulation and visualization of an object or a problem. Modeling is a representation of how people describe or explain an object, system, or a concept, which is usually manifested by simplification or idealization. This can be represented by physical models (mockups, prototypes), the model image (design drawings, computer images), or mathematical formulas. Visualization can also be done to facilitate the delivery of a material in a formal classroom or school. Solar system is a set of celestial bodies bound by gravitational forces. The movement of celestial bodies like the sun, stars, planets and the other will be more easily understood if taught through visualization movement through computer animation. This visualization shows the solar system planetary motion, or we can call it a revolution, that is, when the planets move around the sun, and remain in orbit each using OpenGL API to represent the solar system as a visual. OpenGL support this modeling capability as OpenGL has additional features to better produce something more realistic. OpenGL allows us to create a graph that can be run on any operating system only minor adjustments.
Keywords: Modeling, Solar System, Revolution, Planet.

Contents

Introduction

The development of technology has increased very rapidly and can be found in almost all areas of human life, author can find one of them in the field of education. Nowadays, education learning is using technology in order to perform the delivery of material will become more interesting and easy memorable. The use of technology in education can be implemented with a visualization on a subject of study. The study of the solar system for example, by using the visualization of objects in the solar system would facilitate teachers for the delivery of content. Visualization of the solar system is modified in graphic or animation to display a collection of celestial objects that consist a large star called the sun, and all objects that are bound by the force of gravity. The objects are eight planets that make a revolution or rotation of the sun and remain in orbit respectively. Based on the things that have been described in the background of the problem, the author identify some of the problems that exist, there are:

1. How does the process of making the visualization of the solar system?

2. How to implement a visual method of learning about the solar system?

Many problems that can occur, then the author will limit these problems by:

1. This solar system visualizer application using Turbo c++.

2. This application only displays real comparison of the diameter, distance, lane orbital trajectory, and speed of the planets around the sun.

The purpose of this application is as follows:

1. To create a visualization of the solar system.

2. To apply the visual method of learning about the solar system.

The benefits of the research are as follows:

1. In order to be used as a reference in the process of visualization of the solar system.

2. In order to be used as a reference to learn about the solar system.

3. For an application that is created can be used for the learning process of the solar system

➢ Problem statements :

The development of technology in the 21st century is extremely fast. The urgency of the introduction technology, especially in the educational process, is that the use of such a new system will undoubtedly increase students' motivation, as well as increase the level of assimilation of information due to the variety and interactivity of its visual presentation. It is the process of augmenting reality with virtual objects. Virtual reality communication is performed on-line, and only the camera is required to provide the desired effect – images that will be complemented by virtual objects

➢ Methodologies:

Stages of making solar system visualization:

1. Declare all the attributes that needed for the keyboard function and rotation.
2. Set the speed of rotation of each planet, because the speed of each planet to make one revolution is different.
3. Create the sun with a solid sphere, set the size, and position in the center point (0,0,0) so that the sun became the center of planets that will be surrounding them.
4. Create the planets Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune with a solid sphere, set the size of each planet, and the planet's position in accordance with the ratio of the actual distance.
5. Set the revolution of each planet.
6. Create the orbit of each planet, with torus (ring), set torus in the center, so the planets will move around the sun, and still remain in orbit.
7. Create Keyboard function for regulating the movement of the display.
8. Adjust the lighting to give a 3D effect. In this visualization,user can interacts with the application to perform transformation against planets.

➢ System Requirements:

Material of the solar system based on discussion as needed for design, there are:

1. The composition of the solar system Displays information about the sun and the order of the planets and their orbital trajectory.
2. Various planets Displays the names of the planets in order.
3. The size of the planet Displaying diameter comparison by the actual size of the planet and visualization.
4. Distance planet to the sun Displaying diameter comparison by the actual distance of the planet and visualization.
5. The revolution speed of each planet Revolution speed comparison by the actual speed of the planet and the visualization
6. The orbit of each planet Tracks Display every planetary orbit.

Implementation

Code:

```
#include <conio.h>
#include <dos.h>
#include <graphics.h>
#include <math.h>
#include <stdio.h>

void planetMotion(int xrad, int yrad,

                  int midx, int midy,

                  int x[70], int y[70])
{

   int i, j = 0;

   for (i = 360; i > 0; i = i - 6) {

         x[j] = midx - (xrad * cos((i * 3.14) / 180));

         y[j++] = midy - (yrad * sin((i * 3.14) / 180));

   }


   getch();
}
void main()
{

   int gdriver = DETECT, gmode, err;

   int i = 0, midx, midy;

   int xrad[9], yrad[9], x[9][70], y[9][70];

   int pos[9], planet[9], tmp;

   initgraph(&gdriver, &gmode, "C:\\TURBOC3\\BGI");

   err = graphresult();


   if (err!= grOk)
   {

         printf("Graphics Error: %s",
```

```
            grapherrormsg(err));
        getch();
}
midx = getmaxx() - 220;
midy = getmaxy() - 150;
planet[0] = 8;
for (i = 1; i < 9; i++) {
        planet[i] = planet[i - 1] + 1;
}
for (i = 0; i < 9; i++) {
        pos[i] = i * 6;
}
xrad[0] = 70, yrad[0] = 40;
for (i = 1; i < 9; i++) {
        xrad[i] = xrad[i - 1] + 38;
        yrad[i] = yrad[i - 1] + 20;
}
for (i = 0; i < 9; i++) {
        planetMotion(xrad[i], yrad[i],
                        midx, midy, x[i],
                        y[i]);
}

while (!kbhit())
 {
        setcolor(WHITE);
        for (i = 0; i < 9; i++) {
           setcolor(CYAN);
           ellipse(midx, midy, 0, 360,
                        xrad[i], yrad[i]);
        }
```

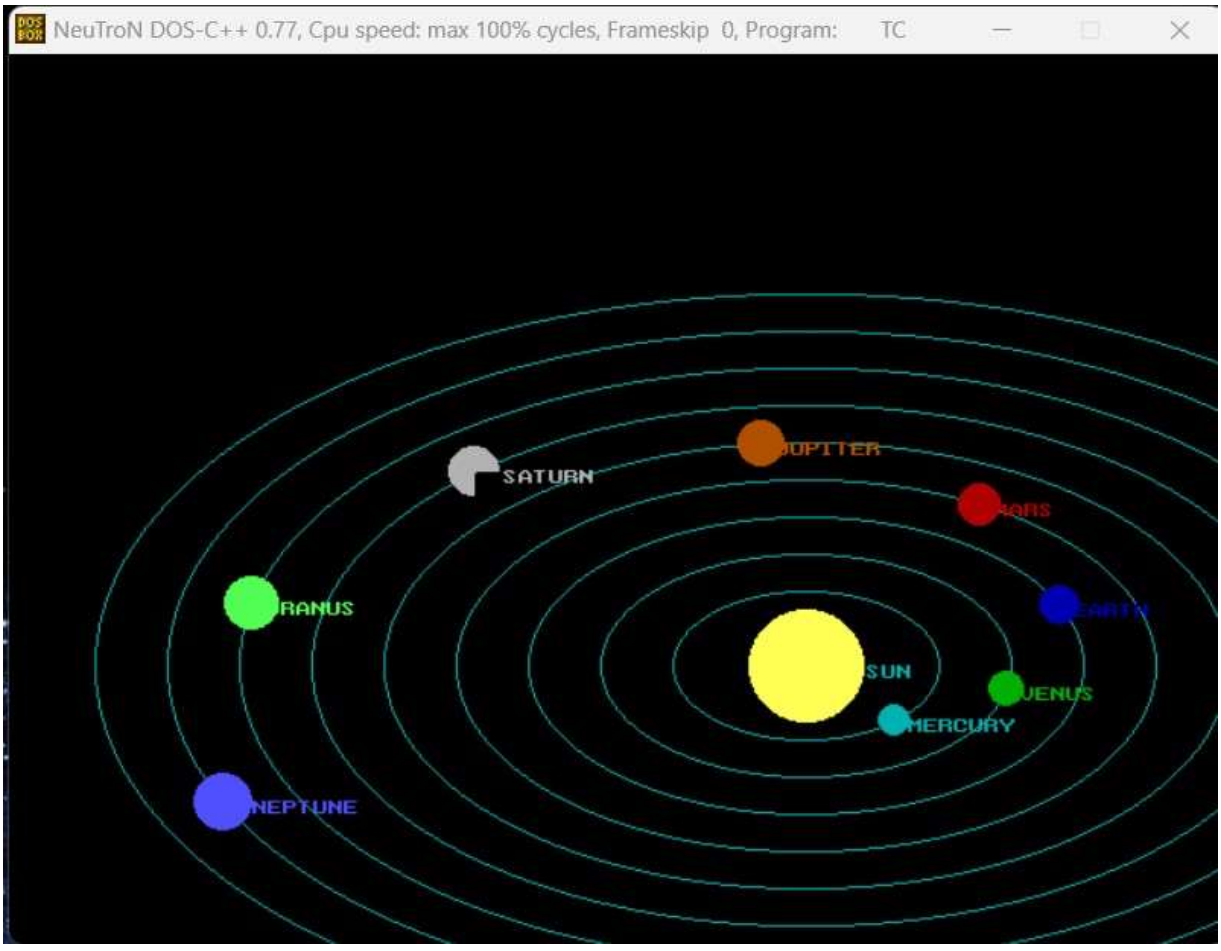
```
outtextxy(midx, midy, "    SUN");
setcolor(YELLOW);
setfillstyle(SOLID_FILL, YELLOW);
circle(midx, midy, 30);
floodfill(midx, midy, YELLOW);
setcolor(CYAN);
setfillstyle(SOLID_FILL, CYAN);
outtextxy(x[0][pos[0]],
          y[0][pos[0]],
          " MERCURY");

pieslice(x[0][pos[0]],
         y[0][pos[0]],
         0, 360, planet[0]);

setcolor(GREEN);
setfillstyle(SOLID_FILL, GREEN);
outtextxy(x[1][pos[1]],
          y[1][pos[1]],
          " VENUS");
pieslice(x[1][pos[1]],
         y[1][pos[1]],
         0, 360, planet[1]);

setcolor(BLUE);
setfillstyle(SOLID_FILL, BLUE);
outtextxy(x[2][pos[2]],
          y[2][pos[2]],
          " EARTH");
pieslice(x[2][pos[2]],
```

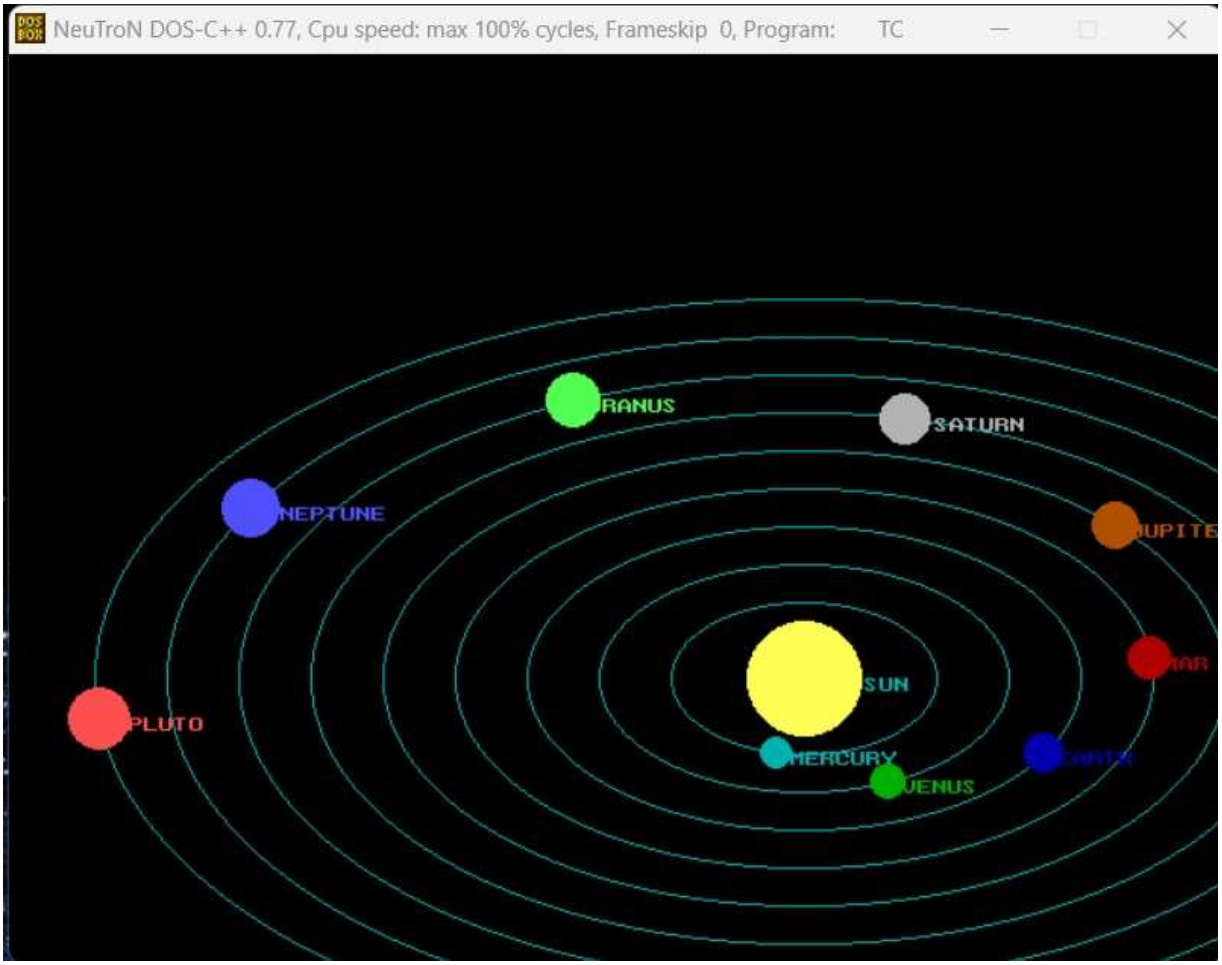
```
                y[2][pos[2]],
                0, 360, planet[2]);

setcolor(RED);
setfillstyle(SOLID_FILL, RED);
outtextxy(x[3][pos[3]],
                 y[3][pos[3]],
                 " MARS");
pieslice(x[3][pos[3]],
                y[3][pos[3]],
                0, 360, planet[3]);

setcolor(BROWN);
setfillstyle(SOLID_FILL, BROWN);
outtextxy(x[4][pos[4]],
                 y[4][pos[4]],
                 " JUPITER");
pieslice(x[4][pos[4]],
                y[4][pos[4]],
                0, 360, planet[4]);

setcolor(LIGHTGRAY);
setfillstyle(SOLID_FILL, LIGHTGRAY);
outtextxy(x[5][pos[5]],
                 y[5][pos[5]],
                 "  SATURN");
pieslice(x[5][pos[5]],
                y[5][pos[5]],
                0, 360, planet[5]);
setcolor(LIGHTGREEN);
```

```
setfillstyle(SOLID_FILL, LIGHTGREEN);
        outtextxy (x [6] [pos [6]],
                   y [6] [pos [6]],

                   (" URANUS"));
        pieslice (x [6] [pos [6]],
                  y [6] [pos [6]],
                  0, 360, planet [6]),
setcolor (LIGHTBLUE);
setfillstyle (SOLID_FILL, LIGHTBLUE);
outtextxy (x [7] [pos [7]],
           y [7] [pos [7]],
           "  NEPTUNE");
pieslice (x [7] [pos [7]],
          y [7] [pos [7]],
          0, 360, planet [7]);

setcolor (LIGHTRED);
setfillstyle (SOLID_FILL, LIGHTRED);
outtextxy (x [8] [pos [8]],
           y [8] [pos [8]],
           "  PLUTO");
pieslice (x [8] [pos [8]],
          y [8] [pos [8]],
          0, 360, planet [8]);

for (i = 0; i < 9; i++) {
   if (pos[i] <= 0) {
        pos[i] = 59;
   }
```

```
            else {
                    pos[i] = pos[i] - 1;
               }
          }
          delay (100);

          cleardevice ();
     }
   closegraph();
   getch();
}
```

Output:




NeuTroN DOS-C++ 0.77, Cpu speed: max 100% cycles, Frameskip 0, Program: TC
RANUS
SATURN
NEPTUNE
SUN
PLUTO
MERCURY
VENUS

Conclusion

Based on the results of discussion that has been done, the authors can conclude that:

1. Animation circulation of planets around the sun interface design is the design of the visual and interpretive models designed.

2. Application of learning visualization of the solar system was designed using TurboC++ that can facilitate computer users to know the objects of the solar system and the solar system information more easily Advice can be given by the author in connection with the limitations of the system is built, is:

Acknowledgement

We sincerely acknowledged with deep sense of gratitude to Project Guide **Mrs. Rane Prajakta** for their valuable guidance, genuine suggestion and constant encouragement during preparation of project synopsis work without which completion of this task would be a difficult task.

We are also thankful to all of our faculty member of **Computer Engineering De- partment** especially our head of department **Prof. D. P. Mhapasekar** and our respected principal **Dr. A C. Gangal** who give us idea of significant cooperation during completion of this work.

We are immensely grateful to all who involved in this project work because without their cooperation,I nspiration, constant promoting and useful suggestion it would be im- possible to complete this task and synopsis report within this allotted time.

Project Members:

1. Kajal Bagwe
2. Apurva Khot
3. Astha Sawant

November 2021

Dr. A C. Gangal

Sindhudurga Shikshan Prasrak Mandal's College of Engineering

Referance

(a) Angel, Edward, 2012. Interactive computer graphics: a Top-Down Approach with Shader-Based TurboC++ 7th ed. Addison Wesley.

(b) F.S.Hill, Jr., 2001. Computer Graphics – TurboC++, Second Edition, Prentice Hall.

(c) Foley, van Dam, Feiner, Hughes and Philips, 2000. Introduction to Computer Graphics, Addison Wesley.

(d) Neider,Jackie. 1997. TurboC++ Programming Guide:

the official guide to learning TurboC++. second edition. Addison Wesley.

(e) Richard S.Wright Jr. 2011. Turboc++ Super Bible:

Comprehensive Tutorial and Reference 5th ed. Pearson Education, Inc.

(f) Shreiner. Dave, 2004. An Interactive Introduction to TurboC++ Programming Courese #29, Siggraph.

(g) PDS: The Planetary Data Systems, https://pds.jpl.nasa.gov, accessed at 29 January 2015

(h) PDS: The Planetary Data Systems, https://pds.jpl.nasa.gov/planets, accessed at 29 January 2015

Bibliography

Riri Safitri is a lecturer in Department of Informatics Engineering, Faculty of Science Technology, University of Al Azhar Indonesia, Jakarta. She received her Master of Electrical Engineering: Digital Media and Game Technology from Institut Technology Bandung in 2009. Her research interests are in the modelling and visualization in Augmented Reality.