**Computer Graphics Project**

**SOLAR GALAXY**

**ABSTRACT:**

This project is developedby using a special main primitive which is solid sphere . The main objective of this project to make a solar galaxy, and give the planets some transition or rotating effects for realistic look.

**INTRODUCTION:**

In this project firstly, We have created the sun and the planets using different radius for each of them in the glutSolidSphere() function.The arguments passed to this function are the different eight radius values for the eight planets and one for the sun which is biggest of all hence has a biggest radius value

Then for showing the stars in the solar galaxy,I have used simple GL\_POINTS which I have taken in a for loop with some number of points and generated them at random location by changing their co-ordinate values in the loop

**THE FUNCTIONS USED IN THE PROJECT ARE:**

1. drawSolarSystem()
2. Initialization()
3. displayFunc()
4. reshapeFunc();
5. idleFunc();

**1. drawSolarSystem()**

This is the main function of this project in which I have generated all planets and the sun at a specific distance with certain radii for each of them by passing their respective radii in the glutSolidSpeher() function too draw them.

Then I have rotated the planets at a arbitrary point with specific rotating year calculated by the year rate in the main body of the project for each of the planets.In the IdleFunc(),These year values are incremented at a year rate which is different for each planet and drawSolarSystem() function is put in a glut processing loop by calling the idleFunc() in the main function hence the planets rotate around the sun.

Then finally I have provided different colors for each planet according to their real look and drew then.In this function I have also used gluLookAt0 function to change the camera angle to make it 3D by giving it a specific position value

Then the planets are created by using the glPushMatrix() and glPopMatrix() functions which helps in simultaneous creation of planets

**2.Initialization()**

In this function,The background color is made black for the solar galaxy and the view is set to MODELVIEW matrix form.

**3.displayFunc()**

In this function,I have cleared the color buffer bit and called the drawSolarSystem() function.

**4.reshapeFunc()**

In this function,In this function I have used the gluPerspective() function to change the perspective and generated a viewport of screen size and called the displayFunc() function.

**5.idleFunc()**

This is again a important function of the project,It is used to put the drawSolarSystem() function is the glut processing loop.In this function I have increased the year of respective planets with their respective year rate for rotation around the sun which is different for each planet depending upon the number of days in a year of that planet.

**CODE OF THE PROJECT:**

/\*The Project made by VARAD PARLIKAR and ABHIJIT PEDHE \*/

#include "windows.h"

#include<GL/glu.h>

#include <GL\glut.h>

#include<math.h>

#include<string.h>

char mercury[]="Mercury";

char venus[]="Venus";

char earth[]="Earth";

char mars[]="Mars";

char jupiter[]="Jupiter";

char saturn[]="Saturn";

char uranus[]="Uranus";

char neptune[]="Neptune";

char sun[]="Sun";

#define sunRaduis 0.5

#define mercuryradius 0.03

#define venusradius 0.05

#define earthRaduis 0.06

#define marsRadius 0.05

#define jupiterRadius 0.08

#define saturnRadius 0.059

#define uranusRadius 0.055

#define neptuneRadius 0.061

#define moonRaduis 0.016

GLfloat rotationSpeed = 0.1;

GLfloat mercurydaysInYear = 88;

GLfloat venusdaysInYear = 225;

GLfloat earthdaysInYear = 365;

GLfloat marsdaysInYear = 687;

GLfloat jupiterdaysInYear = 868;

GLfloat saturndaysInYear = 908;

GLfloat uranusdaysInYear = 967;

GLfloat neptunedaysInYear = 1000;

GLfloat mercuryyear = 1110.0;

GLfloat venusyear = 5530.0;

GLfloat earthyear = 560.0;

GLfloat marsyear = 6760.0;

GLfloat jupiteryear = 780.0;

GLfloat saturnyear = 8880.0;

GLfloat uranusyear = 540.0;

GLfloat neptuneyear = 4440.30;

GLfloat day = 0.0;

GLfloat moonAroundEarth = 0.0;

GLfloat moonItsSelf = 0.0;

GLfloat mercuryOrbitRadius =0.701;

GLfloat venusOrbitRadius =0.72;

GLfloat earthOrbitRadius =0.73;

GLfloat marsOrbitRadius =0.75;

GLfloat jupiterOrbitRadius =1.27;

GLfloat saturnOrbitRadius =1.70;

GLfloat uranusOrbitRadius =1.92;

GLfloat neptuneOrbitRadius =2.19;

GLfloat moonOrbitRadius = 0.1;

GLfloat moonAroundEarthRate = 2 \* rotationSpeed;

GLfloat moonRotationItselfRate = 5.0 \* rotationSpeed;

GLfloat dayRate = 5.0 \* rotationSpeed;

GLfloat mercuryyearRate = mercurydaysInYear / 688.0 \* dayRate \* rotationSpeed;

GLfloat venusyearRate = venusdaysInYear / 855.0 \* dayRate \* rotationSpeed;

GLfloat earthyearRate = earthdaysInYear / 960.0 \* dayRate \* rotationSpeed;

GLfloat marsyearRate = marsdaysInYear / 1287.0 \* dayRate \* rotationSpeed;

GLfloat jupiteryearRate = jupiterdaysInYear / 1480.0 \* dayRate \* rotationSpeed;

GLfloat saturnyearRate = saturndaysInYear / 1500.0 \* dayRate \* rotationSpeed;

GLfloat uranusyearRate = uranusdaysInYear / 2190.0 \* dayRate \* rotationSpeed;

GLfloat neptuneyearRate = neptunedaysInYear / 2600.0 \* dayRate \* rotationSpeed;

void drawSolarSystem(void);

void Initialization(void);

void displayFunc(void);

void reshapeFunc(int x, int y);

void idleFunc(void);

int main(int argc, char\* argv[])

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);

glutInitWindowSize(800,800);

glutCreateWindow("Solar system");

Initialization();

glutReshapeFunc(reshapeFunc);

// display call back function reference setting

glutDisplayFunc(displayFunc);

glutIdleFunc(idleFunc);

glutMainLoop();

return 0;

}

void drawSolarSystem(void)

{

glEnable(GL\_TEXTURE\_2D);

glPushMatrix();

gluLookAt( 0.0,10.0,-4.0,

0.0,0.0,1.0,

0.0,-3.0,0.0);

//gluLookAt Function FOr The Camera and setting perspective Don't Change this V

// Color of the sun.

glColor3f(1.0,0.8,0.3);

//Drawing the sun.

glutSolidSphere(sunRaduis,50,50);

glPushMatrix();

glColor3f(1.0,1.0,1.0);

glPointSize(0.9);

glBegin(GL\_POINTS);

for(int j=0;j<10000;j++) // Generate Galaxy Points stars

{

glVertex3f(-1.6+j,-0.6+j,0);

glVertex3f(-3.6+j,-0.9+j,0);

glVertex3f(-4.6+j,-0.9+j,0);

glVertex3f(-5.6+j,-3.9+j,0);

glVertex3f(-0.6+j,-5.9+j,0);

glVertex3f(-0.7-j,-7.4+j,0);

glVertex3f(-0.6+j,-0.5+j,0);

glVertex3f(-0.2+j,-0.7-j,0);

glVertex3f(-0.6+j,-4.2+j,0);

glVertex3f(-0.6+j,-8.1-j,0);

glVertex3f(-0.4+j,-0.6+j,0);

glVertex3f(-0.8+j,-30.9+j,0);

glVertex3f(-0.6+j,-2.9+j,0);

glVertex3f(-0.6-j,-0.9-j,0);

glVertex3f(-0.6-j,-3.9-j,0);

glVertex3f(-0.4+j,-0.6+j,0);

glVertex3f(-0.8-j,-5.9+j,0);

glVertex3f(-0.6+j,-0.9+j,0);

glVertex3f(-0.6+j,-2.9+j,0);

glVertex3f(-0.6+j,-3.9+j,0);

glVertex3f(-0.4-j,-0.6+j,0);

glVertex3f(-0.8-j,-0.9+j,0);

glVertex3f(-0.6+j,-4.9+j,0);

glVertex3f(-0.6+j,-0.9+j,0);

glVertex3f(-6.6+j,-5.9-j,0);

glVertex3f(-0.3-j,-6.6-j,0);

glVertex3f(-0.9-j,-7.9-j,0);

glVertex3f(-5.7-j,-1.9-j,0);

glVertex3f(-4.3-j,-0.9-j,0);

glVertex3f(-3.2-j,-2.9-j,0);

glVertex3f(-3.5+j,-2.4-j,0);

glVertex3f(-5.7+j,-4.5-j,0);

glVertex3f(-4.2+j,-0.7+j,0);

glVertex3f(-2.1+j,-4.2+j,0);

glVertex3f(-0.3+j,-2.1+j,0);

glVertex3f(-0.8+j,-0.63+j,0);

glVertex3f(-0.9+j,-0.93+j,0);

glVertex3f(-0.6+j,-2.9+j,0);

glVertex3f(-0.6-j,-4.92+j,0);

glVertex3f(-0.6-j,-0.92+j,0);

glVertex3f(-4.4+j,-0.3+j,0);

glVertex3f(-4.8+j,-0.94+j,0);

glVertex3f(-0.6+j,-0.55+j,0);

glVertex3f(-2.6+j,-0.97+j,0);

glVertex3f(-0.6+j,-0.8+j,0);

glVertex3f(-3.4+j,-0.63+j,0);

glVertex3f(-0.8+j,-0.92+j,0);

glVertex3f(-4.3+j,-0.9+j,0);

glVertex3f(-3.66+j,-0.92+j,0);

glVertex3f(-5.6+j,-4.4+j,0);

glVertex3f(-7.6-j,-3.6-j,0);

glVertex3f(-0.6-j,-0.9-j,0);

glVertex3f(-0.6-j,-6.9-j,0);

glVertex3f(-0.6-j,-5.9-j,0);

glVertex3f(-0.6-j,-3.9-j,0);

glVertex3f(-0.7+j,-0.4-j,0);

glVertex3f(-0.6+j,-2.5+j,0);

glVertex3f(-0.2+j,-3.7+j,0);

glVertex3f(-4.6+j,-4.2+j,0);

glVertex3f(-4.6+j,-0.1+j,0);

glVertex3f(-0.4+j,-0.6+j,0);

glVertex3f(-5.8+j,-0.9+j,0);

glVertex3f(-0.6+j,-0.9+j,0);

glVertex3f(-4.6-j,-0.9+j,0);

glVertex3f(-2.6-j,-0.9+j,0);

glVertex3f(-4.4+j,-0.6+j,0);

glVertex3f(-0.8+j,-0.9+j,0);

glVertex3f(-8.6+j,-0.9+j,0);

glVertex3f(-5.6+j,-0.9+j,0);

glVertex3f(-7.6+j,-0.9-j,0);

glVertex3f(-6.4+j,-0.6-j,0);

glVertex3f(-0.8+j,-0.9+j,0);

glVertex3f(-0.6+j,-0.9+j,0);

glVertex3f(-7.6+j,-5.9+j,0);

glVertex3f(-0.6+j,-6.9+j,0);

}

glEnd();

glPopMatrix();

glColor3f(1.0,0.0,0.0);

glRasterPos2i(10,30);

for(int i=0;i<=2;i++){

glutBitmapCharacter(GLUT\_BITMAP\_HELVETICA\_18, sun[i]);

}

glRotatef(mercuryyear,0.0,1.0,0.0); //rotation for mercury

glTranslatef(mercuryOrbitRadius,0.0,0.0); // translation for mercury

glRotatef(-mercuryyear,0.0,1.0,0.0);

glPushMatrix();

glRotatef(day,0.25,1.0,0.0);

glColor3f(1.0,1.0,1.3);

//Drawing the mercury

glutSolidSphere(mercuryradius,10,10);

glPopMatrix();

glRotatef(venusyear,0.0,1.0,0.0); //rotation for venus

glTranslatef(venusOrbitRadius,0.0,0.0); // translation for venus.

glRotatef(-venusyear,0.0,1.0,0.0);

glPushMatrix();

glRotatef(day,0.25,1.0,0.0);

glColor3f(1.0,0.8,0.0);

//Drawing the venus

glutSolidSphere(venusradius,10,10);

glPopMatrix();

glPushMatrix();

glRotatef(earthyear,0.0,1.0,0.0); //rotation for earth

glTranslatef(earthOrbitRadius,0.0,0.0); // translation for earth.

glRotatef(-earthyear,0.0,1.0,0.0);

glPushMatrix();

glRotatef(day,0.25,1.0,0.0);

glColor3f(0.0,0.0,02.75);

//Drawing the earth

glutSolidSphere(earthRaduis,10,10);

glPopMatrix();

glPushMatrix();

glRotatef(marsyear,0.0,1.0,0.0); //rotation for mars

glTranslatef(marsOrbitRadius,0.0,0.0); // translation for mars

glRotatef(-marsyear,0.0,1.0,0.0);

glPushMatrix();

glRotatef(day,0.25,1.0,0.0);

glColor3f(1.0,0.0,0.0);

//Drawing the mars

glutSolidSphere(marsRadius,10,10);

glPopMatrix();

glPushMatrix();

glRotatef(jupiteryear,0.0,1.0,0.0); //rotation for jupiter

glTranslatef(jupiterOrbitRadius,0.0,0.0); // translation for jupiter

glRotatef(-jupiteryear,0.0,1.0,0.0);

glPushMatrix();

glRotatef(day,0.25,1.0,0.0);

glColor3f(0.5,0.60,0.0);

//Drawing the jupiter

glutSolidSphere(jupiterRadius,10,10);

glPopMatrix();

glPushMatrix();

glRotatef(saturnyear,0.0,1.0,0.0); //rotation for saturn

glTranslatef(saturnOrbitRadius,0.0,0.0); // translation for saturn

glRotatef(-saturnyear,0.0,1.0,0.0);

glPushMatrix();

glRotatef(day,0.25,1.0,0.0);

glColor3f(0.1,0.0,0.0);

//Drawing the saturn

glutSolidSphere(saturnRadius,10,10);

glPopMatrix();

glPushMatrix();

glRotatef(uranusyear,0.0,1.0,0.0); //rotation for uranus

glTranslatef(uranusOrbitRadius,0.0,0.0); // translation for uranus

glRotatef(-uranusyear,0.0,1.0,0.0);

glPushMatrix();

glRotatef(day,0.25,1.0,0.0);

glColor3f(0.5,0.60,0.0);

//Drawing the uranus

glutSolidSphere(uranusRadius,10,10);

glPopMatrix();

glPushMatrix();

glRotatef(neptuneyear,0.0,1.0,0.0); //rotation for neptune

glTranslatef(neptuneOrbitRadius,0.0,0.0); // translation for neptune

glRotatef(-neptuneyear,0.0,1.0,0.0);

glPushMatrix();

glRotatef(day,0.25,1.0,0.0);

glColor3f(0.5,0.60,0.0);

//Drawing the neptune

glutSolidSphere(neptuneRadius,10,10);

glPopMatrix();

// rotation for moon.

glRotatef(moonAroundEarth,0.0,1.0,0.0);

// translation for moon.

glTranslatef(moonOrbitRadius,0.0,0.0);

// around earth rotation.

glRotatef(-moonAroundEarth,0.0,1.0,0.0);

// moon rotation about it self.

glColor3f(0.3,0.3,0.5);

// draw the moon

glutSolidSphere(moonRaduis,8,8);

glPopMatrix();

glPopMatrix();

}

void Initialization(void)

{

glClearColor(0.0,0.0,0.0,0.0);

glClearDepth(10.0);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

}

void displayFunc(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT);

drawSolarSystem();

// flush to screen.

glFlush();

// swap buffers , yes we need this ( double buffering used!)

glutSwapBuffers();

}

void reshapeFunc(int x, int y)

{

if (y == 0 || x==0) return;

glLoadIdentity();

gluPerspective(40.0,(GLdouble)x/(GLdouble)y,0.5,20.0);

glMatrixMode(GL\_MODELVIEW);

glViewport(0,0,x,y);

displayFunc();

}

void idleFunc(void)

{

// idle event call back in animation , here we increase the values and redisply .

day += dayRate;

mercuryyear += mercuryyearRate;

venusyear += venusyearRate;

earthyear += earthyearRate;

marsyear += marsyearRate;

jupiteryear += jupiteryearRate;

saturnyear += saturnyearRate;

uranusyear += uranusyearRate;

neptuneyear += neptuneyearRate;

moonItsSelf += moonRotationItselfRate;

moonAroundEarth += moonAroundEarthRate;

displayFunc();

}

**OUTPUT OF THE PROJECT**

