My City

Roxana\_Ioana\_Roman\_DOC

1. 1.Contents
2. Subject specification

Using OpenGL, I created a photorealistic application of a city.

The objects were created using the GMAX.

1. Scenario
   1. Scene and object description

In the front view there is a car on a street. In front there’s an intersection. Along the street there are

street lamps and trees. There are also two blocks and a mailbox.

* 1. Functionalities

The user can choose between polygonal, solid, and wireframe views.

The user can choose between different light sources and cast projection shadows on the street.

The user can rotate and move the scene to obtain a better view.

1. Implementation details

* Functions and special algorithms

4.1. Animations

The user can move the car on the street.

4.2 Projected Shadows

The user can see the shadow of the car on the street.

Possible Solution

In order to obtain this I created a light source and a plane on which to cast the shadow of an object by computing the planeEquation and shadowMatrix equation.

* + The motivation of the chosen approach:

I was inspired by the computer games with cars so I tried to reproduce something similar.

* + Graphics model

The models were created in GMAX.

* + Data structures

I used structures, enums.

* + Class hierarchy

I did not use classes.

1. Graphical user interface presentation / user manual

I have a panel in which the user can change the texture og the car and rotate the scene.

switch(key)

{

case 't':

//process

glutPostRedisplay();

break;

case 'w':

deplasare= deplasare+0.30;

glutPostRedisplay();

break;

case 's':

deplasare= deplasare-0.30;

glutPostRedisplay();

break;

case 'd':

directionAngle -= 1.3;

break;

case 'a':

directionAngle += 1.3;

break;

case 27: //esc

exit(1);

break;

//control the global Y rotation angle using 'a' and 'd'

case 'f':

fGlobalAngleY += 1;

if (fGlobalAngleY >= 360) //clamp the rotation angle in the [0,360) interval

fGlobalAngleY = (GLint)fGlobalAngleY % 360;

break;

case 'g':

fGlobalAngleY -= 1;

if (fGlobalAngleY <= -360) //clamp the rotation angle in the [0,360) interval

fGlobalAngleY = (GLint)fGlobalAngleY % 360;

break;

//control the global X rotation angle using 'w' and 's'

case 'h':

fGlobalAngleX += 1;

if (fGlobalAngleX >= 360) //clamp the rotation angle in the [0,360) interval

fGlobalAngleX = (GLint)fGlobalAngleX % 360;

break;

case 'j':

fGlobalAngleX -= 1;

if (fGlobalAngleX <= -360) //clamp the rotation angle in the [0,360) interval

fGlobalAngleX = (GLint)fGlobalAngleX % 360;

break;

//control the global Z rotation angle using 'q' and 'e'

case 'k':

fGlobalAngleZ += 1;

if (fGlobalAngleZ >= 360) //clamp the rotation angle in the [0,360) interval

fGlobalAngleZ = (GLint)fGlobalAngleZ % 360;

break;

//fog and stuff

case 'l':

fGlobalAngleZ -= 1;

if (fGlobalAngleZ <= -360) //clamp the rotation angle in the [0,360) interval

fGlobalAngleZ = (GLint)fGlobalAngleZ % 360;

break;

case '1': //enable fog

fogen=1;

glutPostRedisplay();

break;

case '2': //disable fog

fogen=0;

glutPostRedisplay();

break;

case '3': //disable lights

enlight=0;

glutPostRedisplay();

break;

case '4': //enable lights

enlight=1;

glutPostRedisplay();

break;

case '5': //disable sunlight

light1=0;

glutPostRedisplay();

break;

case '6': //enable sunlight

light1=1;

glutPostRedisplay();

break;

case '7': //disable lamp light

light3=0;

glutPostRedisplay();

break;

case '8': //enable lamp light

light3=1;

glutPostRedisplay();

break;

case '<': //enable projection shadow

shadow1 = 1;

glutPostRedisplay();

break;

case '>': //disable projection shadow

shadow1 = 0;

glutPostRedisplay();

break;

case '?': //draw wireframe

objtype=0;

glutPostRedisplay();

break;

case '!': //draw wireframe

objlook=0;

glutPostRedisplay();

break;

}

1. Conclusions and further developments

I want to include volume shadows and projection shadows for the street lamps.

1. Refrences

www.cgis.utcluj.ro/didactic