NATIONAL INSTITUTE OF TECHNOLOGY ,SILCHAR

A MINI PROJECT REPORT

On

URBANIZATION USING OPENGL

Submitted in partial fulfillment of the requirements
For the completion of 4th SEM.

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

By

Purushottam Gaudel (15-1-5-055)
Subham Kamalapuri (15-1-5-059)
Apurbajyoti Das(15-1-5-065)
Atul Singh Rana (15-1-5-073)

Under the guidance of

Mr. Badol Soni

Assistant Professor ,CSE Department

DEPARTMENT OF COMPUTER SCEINCE & ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY, SILCHAR

Acknowledgements

We are pleased to acknowledge Asst. Prof. Mr. Badol Soni, Department of computer science of engineering, NIT Silchar. For his invaluable guidance during the course of this project work.

We are also grateful to our M.Tech seniors Manash jyoti Gogoi, Vishwajeet singh and Rohem Singh for their continuous support and helping to solve some issue.

Last but not least, we would like to extend our thanks to all who are directly or indirectly co-operated with us for the smooth development of this project.

April’17.

Contents

INTRODUCTION

Overview

This report discuss the result of the work done in designing of “A MODEL ON URBANIZATION using OPENGL “

Opengl is a library useful for designing and animation purposes and many more. It can be included either of c or c++ programming and it is very user friendly.

Background and motivation

Opengl can be found in a large variety of application today like to design or develop animation films , cartoons and mostly games. Today’s era is a era where children even the adults are likely to play games. Moreover cartoons are widely seen by the children to adult.Also animation are used in daily newspaper, news channel, advertisements and many more to describe products, social issues, political clashes etc.

Objective

Our objective to development of this project to design a model which could describe the scenario and relationship between the past few decades and the upcoming years in order to make a better or supreme planning for the future. Here we design a model that describe how urbanization is going on.

Methodology

To implement the above goal, the methodology we followed are –

- At the very beginning we design an abstract of the project.
- For designing purpose we use opengl libraries (glut) in c programming.
- Various algorithm are used to draw the various shape like circle using bresenhem midpoint circle algorithm. Etc.
- We use simple translation, rotation, scaling for the movement of object like sun, car etc.

TOOLS DESRIPTION

The tool use for this project is opengl in c programming .

Platform

The model can be run in any platform (os) Linux, windows provided that the glut should be installed and properly working.

Size of source code:

15.4 KB (15,785 bytes)

WORK THAT HAVE DONE

Source code:

```
#include<stdio.h>
#include<GL/glut.h>
#include <GL/gl.h>
#include <stdlib.h>

float i=0.0;
float j=0.0;
float r=0.0;
int d=1;
int k=0;
int gaon=1;
int flag=0;

void draw_pixel(GLint cx, GLint cy)
{

	glBegin(GL_POINTS);
		glVertex2i(cx,cy);
	glEnd();
}

void plotpixels(GLint h,GLint k, GLint x,GLint
y)
{
	draw_pixel(x+h,y+k);
	draw_pixel(-x+h,y+k);
	draw_pixel(x+h,-y+k);
	draw_pixel(-x+h,-y+k);
	draw_pixel(y+h,x+k);
	draw_pixel(-y+h,x+k);
	draw_pixel(y+h,-x+k);
	draw_pixel(-y+h,-x+k);
}

void draw_circle(GLint h, GLint k, GLint r)
{
	GLint d=1-r, x=0, y=r;
	while(y>x)
	{
		plotpixels(h,k,x,y);
		if(d<0) d+=2*x+3;
		else
		{
			d+=2*(x-y)+5;
			--y;
		}
		++x;
	}
	plotpixels(h,k,x,y);
}


void draw_object()
{
//sky
if(d)
 glColor3f(0.22,0.69,0.87);
else glColor3f(0,0,0);
glBegin(GL_POLYGON);
glVertex2f(0,450);
glVertex2f(0,700);
glVertex2f(1100,700);
glVertex2f(1100,450);
glEnd();
```

```
//moon
int l;
       if(d)
         glColor3f(1,.5,0);
            else glColor3f(1,1,1);
            for(l=0;l<=35;l++)
            {
                        draw_circle(100,750-j,l);
            }

//star1
if(!d){
glColor3f(0+r,0+r,0+r);
glBegin(GL_TRIANGLES);
glVertex2f(575,653);
glVertex2f(570,645);
glVertex2f(580,645);
glVertex2f(575,642);
glVertex2f(570,650);
glVertex2f(580,650);
glEnd();

//star2
glColor3f(0+r,0+r,0+r);
glBegin(GL_TRIANGLES);
glVertex2f(975,643);
glVertex2f(970,635);
glVertex2f(980,635);
glVertex2f(975,632);
glVertex2f(970,640);
glVertex2f(980,640);
glEnd();
// star3
glColor3f(0+r,0+r,0+r);
glBegin(GL_TRIANGLES);
glVertex2f(875,543);
glVertex2f(870,535);
glVertex2f(880,535);
glVertex2f(875,532);
glVertex2f(870,540);
glVertex2f(880,540);
glEnd();

//star4
glColor3f(0+r,0+r,0+r);
glBegin(GL_TRIANGLES);
glVertex2f(375,598);
glVertex2f(370,590);
glVertex2f(380,590);
glVertex2f(375,587);
glVertex2f(370,595);
glVertex2f(380,595);
glEnd();
}
//back grass
for(l=0;l<=30;l++){
if(gaon)
  glColor3f(0.0,0.2,0.0);
else
  glColor3f(0.46,0.44,0.09);
for(k=0;k<=1000;k+=55){
draw_circle(k,460,l);
}

}

//grass
if(gaon)
 glColor3f(0.0,0.3,0.0);
else
 glColor3f(.334,.245, .245);
glBegin(GL_POLYGON);
glVertex2f(0,160);
glVertex2f(0,450);
glVertex2f(1100,450);
glVertex2f(1100,160);
glEnd();

glColor3f(0,0,0);
glBegin(GL_POLYGON);
glVertex2f(0,150);
glVertex2f(0,160);
```

```
glVertex2f(1100,160);
glVertex2f(1100,150);
glEnd();

//paddy field
if(gaon){
glColor3f(0.6263,.50,.04151650);
glBegin(GL_POLYGON);
glVertex2f(0,190);
glVertex2f(70,300);
glVertex2f(330,300);
glVertex2f(260,190);
glEnd();

glColor3f(0.0,0.4,0.0);
glLineWidth(3.0);
glBegin(GL_LINES);
glVertex2f(0,190);
glVertex2f(70,300);
glVertex2f(260,190);
glVertex2f(330,300);

for(k=0;k<=10;k+=2){
glVertex2f(0+7*k,190+11*k);
glVertex2f(260+7*k,190+11*k);
}
glEnd();
for(k=1;k<=10;k+=2){
glColor3f( .30,.41,0);
glBegin(GL_TRIANGLES);
for(l=0;l<250;l+=50){
glVertex2f(0+7*k+l,190+11*k);
glVertex2f(30+7*k+l,220+11*k);
glVertex2f(50+7*k+l,195+11*k);
}
glEnd();
}

glEnd();

}
```

```
if(!gaon)
{
glColor3f(.6,.35,.12);
glBegin(GL_POLYGON);
glVertex2f(0,350);
glVertex2f(0,530);
glVertex2f(65,530);
glVertex2f(65,350);
glEnd();
glColor3f(1,.35,.12);
glBegin(GL_POLYGON);
glVertex2f(20,350);
glVertex2f(20,390);
glVertex2f(40,390);
glVertex2f(40,350);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(10,410);
glVertex2f(10,420);
glVertex2f(20,420);
glVertex2f(20,410);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(10,440);
glVertex2f(10,450);
glVertex2f(20,450);
glVertex2f(20,440);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(10,470);
glVertex2f(10,480);
glVertex2f(20,480);
glVertex2f(20,470);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(10,500);
glVertex2f(10,510);
glVertex2f(20,510);
glVertex2f(20,500);
glEnd();
glBegin(GL_POLYGON);
```

```
glVertex2f(40,410);
glVertex2f(40,420);
glVertex2f(50,420);
glVertex2f(50,410);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(40,440);
glVertex2f(40,450);
glVertex2f(50,450);
glVertex2f(50,440);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(40,470);
glVertex2f(40,480);
glVertex2f(50,480);
glVertex2f(50,470);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(40,500);
glVertex2f(40,510);
glVertex2f(50,510);
glVertex2f(50,500);
glEnd();

//
glColor3f(0.7,0.8,1);
glBegin(GL_POLYGON);
glVertex2f(70,350);
glVertex2f(70,570);
glVertex2f(170,570);
glVertex2f(170,350);
glEnd();
glColor3f(0,.5,.2);
glBegin(GL_POLYGON);
glVertex2f(105,350);
glVertex2f(105,390);
glVertex2f(135,390);
glVertex2f(135,350);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(90,410);
glVertex2f(90,430);
glVertex2f(110,430);
glVertex2f(110,410);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(90,450);
glVertex2f(90,470);
glVertex2f(110,470);
glVertex2f(110,450);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(90,490);
glVertex2f(90,510);
glVertex2f(110,510);
glVertex2f(110,490);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(90,530);
glVertex2f(90,550);
glVertex2f(110,550);
glVertex2f(110,530);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(130,410);
glVertex2f(130,430);
glVertex2f(150,430);
glVertex2f(150,410);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(130,450);
glVertex2f(130,470);
glVertex2f(150,470);
glVertex2f(150,450);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(130,490);
glVertex2f(130,510);
glVertex2f(150,510);
glVertex2f(150,490);
glEnd();
glBegin(GL_POLYGON);
```

```
glVertex2f(130,530);
glVertex2f(130,550);
glVertex2f(150,550);
glVertex2f(150,530);
glEnd();
//LAST 2ND
glColor3f(0.71,.65,.26);
glBegin(GL_POLYGON);
glVertex2f(900,350);
glVertex2f(900,530);
glVertex2f(965,530);
glVertex2f(965,350);
glEnd();
glColor3f(.7,.35,.12);
glBegin(GL_POLYGON);
glVertex2f(920,350);
glVertex2f(920,390);
glVertex2f(940,390);
glVertex2f(940,350);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(910,410);
glVertex2f(910,420);
glVertex2f(920,420);
glVertex2f(920,410);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(910,440);
glVertex2f(910,450);
glVertex2f(920,450);
glVertex2f(920,440);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(910,470);
glVertex2f(910,480);
glVertex2f(920,480);
glVertex2f(920,470);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(910,500);
glVertex2f(910,510);
glVertex2f(920,510);
glVertex2f(920,500);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(940,410);
glVertex2f(940,420);
glVertex2f(950,420);
glVertex2f(950,410);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(940,440);
glVertex2f(940,450);
glVertex2f(950,450);
glVertex2f(950,440);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(940,470);
glVertex2f(940,480);
glVertex2f(950,480);
glVertex2f(950,470);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(940,500);
glVertex2f(940,510);
glVertex2f(950,510);
glVertex2f(950,500);
glEnd();
//a
glColor3f(0.7,0.8,1);
glBegin(GL_POLYGON);
glVertex2f(970,350);
glVertex2f(970,570);
glVertex2f(1070,570);
glVertex2f(1070,350);
glEnd();
glColor3f(0,.5,.2);
glBegin(GL_POLYGON);
glVertex2f(1005,350);
glVertex2f(1005,390);
glVertex2f(1035,390);
glVertex2f(1035,350);
```

```
glEnd();
glBegin(GL_POLYGON);
glVertex2f(990,410);
glVertex2f(990,430);
glVertex2f(1010,430);
glVertex2f(1010,410);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(990,450);
glVertex2f(990,470);
glVertex2f(1010,470);
glVertex2f(1010,450);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(990,490);
glVertex2f(990,510);
glVertex2f(1010,510);
glVertex2f(1010,490);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(990,530);
glVertex2f(990,550);
glVertex2f(1010,550);
glVertex2f(1010,530);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(1030,410);
glVertex2f(1030,430);
glVertex2f(1050,430);
glVertex2f(1050,410);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(1030,450);
glVertex2f(1030,470);
glVertex2f(1050,470);
glVertex2f(1050,450);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(1030,490);
glVertex2f(1030,510);
glVertex2f(1050,510);
glVertex2f(1050,490);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(1030,530);
glVertex2f(1030,550);
glVertex2f(1050,550);
glVertex2f(1050,530);
glEnd();
//powerpuff
glColor3f(.9,.9,.9);
glBegin(GL_POLYGON);
glVertex2f(400,350);
glVertex2f(390,600);
glVertex2f(610,600);
glVertex2f(600,350);
glEnd();
glColor3f(1,1,0);
glBegin(GL_POLYGON);
glVertex2f(430,350);
glVertex2f(430,420);
glVertex2f(470,420);
glVertex2f(470,350);
glEnd();
glColor3f(1,0,0);
glBegin(GL_POLYGON);
glVertex2f(445,398);
glVertex2f(445,408);
glVertex2f(455,408);
glVertex2f(455,398);
glEnd();

glColor3f(.8,.8,.8);
for(l=0;l<=25;l++){
draw_circle(445,530,l);
draw_circle(505,530,l);
draw_circle(565,530,l);
}

//GODOWN
glColor3f(.6,0,.10);
glBegin(GL_POLYGON);
```

```
glVertex2f(180,350);
glVertex2f(180,500);
glVertex2f(380,500);
glVertex2f(380,350);
glEnd();

glColor3f(.32,.32,.32);
glBegin(GL_POLYGON);
glVertex2f(180,350);
glVertex2f(180,380);
glVertex2f(320,380);
glVertex2f(380,350);
glEnd();


glColor3f(.8,.8,.8);
glBegin(GL_POLYGON);
glVertex2f(176,500);
glVertex2f(176,490);
glVertex2f(385,490);
glVertex2f(385,500);
glEnd();
glColor3f(.75,.751,.750);
glBegin(GL_POLYGON);
glVertex2f(200,380);
glVertex2f(200,420);
glVertex2f(235,420);
glVertex2f(235,380);
glEnd();
//li
glColor3f(.64,.30,.25);
glBegin(GL_POLYGON);
glVertex2f(860,354);
glVertex2f(860,660);
glVertex2f(867,660);
glVertex2f(867,354);
glEnd();
//curr
glLineWidth(1.0);
glColor3f(0,0,0);
glBegin(GL_LINES);
glVertex2f(0,640);
glVertex2f(1100,640);
glBegin(GL_LINES);
glVertex2f(0,635);
glVertex2f(1100,635);
glEnd();
}

// gaon wall
if(gaon){

glColor3f(0,0,0);
glLineWidth(4.0);
glBegin(GL_LINES);
for(k=0;k<780;k+=30){
glVertex2f(k,150); glVertex2f(k,220);
}

glVertex2f(0,180); glVertex2f(760,180);
glVertex2f(0,200); glVertex2f(760,200);
glEnd();

glBegin(GL_LINES);
for(k=850;k<1100;k+=30){
glVertex2f(k,150); glVertex2f(k,220);
}
glVertex2f(830,180); glVertex2f(1100,180);
glVertex2f(830,200); glVertex2f(1100,200);
glEnd();

}
else
{
  //wall
glColor3f(.91,.76,.65);
glBegin(GL_POLYGON);
glVertex2f(0,150);
glVertex2f(0,220);
glVertex2f(700,220);
glVertex2f(700,150);
glEnd();
glBegin(GL_POLYGON);
```

```
glVertex2f(800,150);
glVertex2f(800,220);
glVertex2f(1100,220);
glVertex2f(1100,150);
glEnd();

glBegin(GL_POLYGON);
glVertex2f(0,221);
glVertex2f(0,230);
glVertex2f(700,230);
glVertex2f(700,221);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(800,221);
glVertex2f(800,230);
glVertex2f(1100,230);
glVertex2f(1100,221);
glEnd();

}

//road
if(!gaon)
 glColor3f(0.2,0.2,0.2);
else
 glColor3f(0.62,0.62,0.37);
glBegin(GL_POLYGON);
glVertex2f(0,0);
glVertex2f(0,150);
glVertex2f(1100,150);
glVertex2f(1100,0);
glEnd();
if(!gaon){
glColor3f(1,1,1);
glBegin(GL_POLYGON);
glVertex2f(0,80);
glVertex2f(0,85);
glVertex2f(1100,85);
glVertex2f(1100,80);
glEnd();
}
```

```
if(gaon){
//road grass
glColor3f(.20,.35,0);
glBegin(GL_POLYGON);
glVertex2f(0,0);glVertex2f(10,80);glVertex2f
(20,60);glVertex2f(30,100);glVertex2f(40,0);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(50+0,0);glVertex2f(50+10,60);glV
ertex2f(50+20,40);glVertex2f(50+30,80);glV
ertex2f(50+40,0);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(90+0,0);glVertex2f(90+10,40);glV
ertex2f(90+20,20);glVertex2f(90+30,60);glV
ertex2f(90+40,0);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(980+0,0);glVertex2f(980+10,60);
glVertex2f(980+20,40);glVertex2f(980+30,8
0);glVertex2f(980+40,0);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(1030+0,0);glVertex2f(1030+10,8
0);glVertex2f(1030+20,60);glVertex2f(1030+
30,100);glVertex2f(1030+40,0);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(1080+0,0);glVertex2f(1080+10,4
0);glVertex2f(1080+20,20);glVertex2f(1080+
30,60);glVertex2f(1080+40,0);
glEnd();
// small tree
glColor3f( 0.647059,0.164706, 0.164706);
glBegin(GL_POLYGON);
glVertex2f(240,400);
glVertex2f(240,445);
glVertex2f(255,445);
glVertex2f(255,400);
glEnd();
```

```
        glColor3f(0.0,0.4,0.0);
          for(l=0;l<=30;l++)
          {

               draw_circle(225,450,l);
               draw_circle(265,450,l);
          }

          for(l=0;l<=20;l++)
          {
               draw_circle(235,490,l);
               draw_circle(255,490,l);
          }

          for(l=0;l<=15;l++)
          {
               draw_circle(245,515,l);
          }
//tree
glColor3f( 0.647059,0.164706, 0.164706);
glBegin(GL_POLYGON);
glVertex2f(145,320);
glVertex2f(145,395);
glVertex2f(170,395);
glVertex2f(170,320);
glEnd();
          for(l=0;l<=45;l++)
          {
               glColor3f(0.0,0.5,0.0);
               draw_circle(140,400,l);
               draw_circle(180,400,l);
          }

          for(l=0;l<=35;l++)
          {
               glColor3f(0.0,0.5,0.0);
               draw_circle(150,450,l);
               draw_circle(170,450,l);
          }

          for(l=0;l<=25;l++)
          {
               glColor3f(0.0,0.5,0.0);
               draw_circle(160,490,l);
          }

// big tree
glColor3f( 0.647059,0.164706, 0.164706);
glBegin(GL_POLYGON);
glVertex2f(1045,320);
glVertex2f(1045,365);
glVertex2f(1070,365);
glVertex2f(1070,320);
glEnd();

     glColor3f(0.0,0.45,0.0);
       glBegin(GL_POLYGON);
     glVertex2f(945,365);
     glVertex2f(1175,365);
     glVertex2f(1060,450);
     glEnd();
     glBegin(GL_POLYGON);
     glVertex2f(960,415);
     glVertex2f(1160,415);
     glVertex2f(1060,500);
     glEnd();
     glBegin(GL_POLYGON);
     glVertex2f(975,465);
     glVertex2f(1150,465);
     glVertex2f(1060,550);
     glEnd();


//house
glColor3f(0.0,0.2,0.2);
glBegin(GL_POLYGON);
glVertex2f(600,375);
glVertex2f(600,450);
glVertex2f(650,525);
glVertex2f(700,450);
glVertex2f(700,375);
glEnd();
```

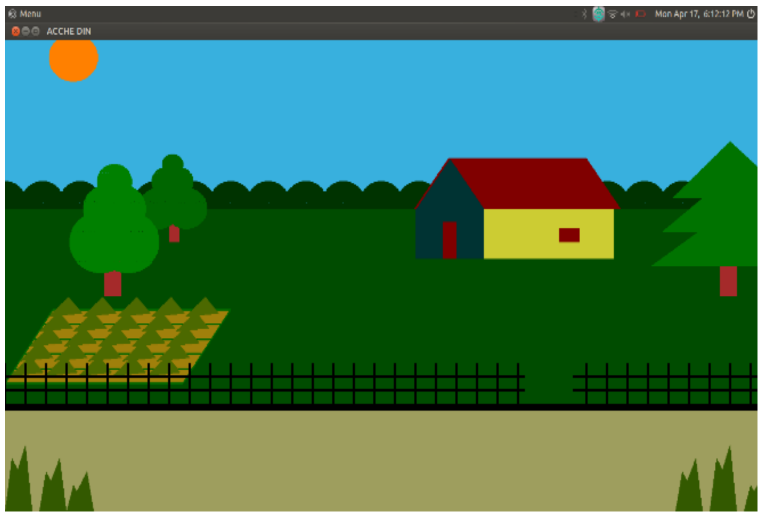
```
//door
glColor3f(0.5,0.0,0.0);
glBegin(GL_POLYGON);
glVertex2f(640,375);
glVertex2f(640,430);
glVertex2f(660,430);
glVertex2f(660,375);
glEnd();

//roof
glColor3f(0.5,0.0,0.0);
glBegin(GL_POLYGON);
glVertex2f(700,450);
glVertex2f(650,525);
glVertex2f(850,525);
glVertex2f(900,450);
glEnd();

glLineWidth(4.0);
glBegin(GL_LINES);
glVertex2f(650,525);
glVertex2f(600,450);
glEnd();

//door wall
glColor3f(0.8,0.8,0.2);
glBegin(GL_POLYGON);
glVertex2f(700,375);
glVertex2f(700,450);
glVertex2f(890,450);
glVertex2f(890,375);
glEnd();

//window
glColor3f(0.5,0.0,0.0);
glBegin(GL_POLYGON);
glVertex2f(810,400);
glVertex2f(810,420);
glVertex2f(840,420);
glVertex2f(840,400);
glEnd();

}

if(!gaon){

//car1
glColor3f(0.0,.5,1.0);
glBegin(GL_POLYGON);
glVertex2f(-470+i,50+70);
glVertex2f(-470+i,112+70);
glVertex2f(-400+i,125+70);
glVertex2f(-372+i,210+70);
glVertex2f(-210+i,210+70);
glVertex2f(-180+i,125+70);
glVertex2f(-135+i,125+70);
glVertex2f(-110+i,50+70);
glEnd();


//windows
glColor3f(0.1,0.1,0.1);
glBegin(GL_POLYGON);
glVertex2f(-410+i,125+70);
glVertex2f(-364+i,200+70);
glVertex2f(-300+i,200+70);
glVertex2f(-300+i,125+70);
glEnd();

glColor3f(0.1,0.1,0.1);
glBegin(GL_POLYGON);
glVertex2f(-290+i,125+70);
glVertex2f(-290+i,200+70);
glVertex2f(-217+i,200+70);
glVertex2f(-192+i,125+70);
glEnd();


for(l=0;l<30;l++)
  {
        glColor3f(0.0,0.0,0.0);
```

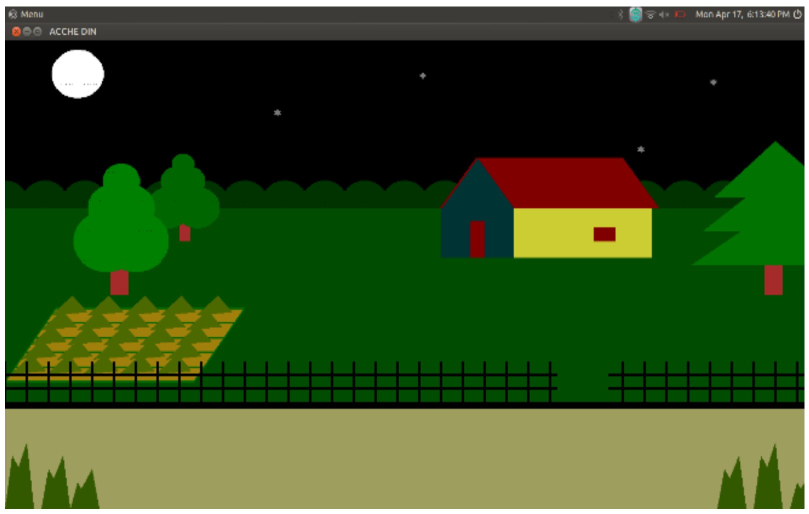
```
            draw_circle(-350+i,50+70,l);
            draw_circle(-200+i,50+70,l);
   }

//car 2
glColor3f(0.5,0.7,0.5);
glBegin(GL_POLYGON);
glVertex2f(-25-i+1600,40);
glVertex2f(-25-i+1600,115);
glVertex2f(-75-i+1600,190);
glVertex2f(-175-i+1600,190);
glVertex2f(-200-i+1600,115);
glVertex2f(-260-i+1600,105);
glVertex2f(-260-i+1600,40);
glEnd();


//windows
glColor3f(0.1,0.1,0.1);
glBegin(GL_POLYGON);
glVertex2f(-35-i+1600,115);
glVertex2f(-80-i+1600,180);
glVertex2f(-115-i+1600,180);
glVertex2f(-115-i+1600,115);
glEnd();

glColor3f(0.1,0.1,0.1);
glBegin(GL_POLYGON);
glVertex2f(-125-i+1600,115);
glVertex2f(-125-i+1600,180);
glVertex2f(-170-i+1600,180);
glVertex2f(-190-i+1600,115);
glEnd();

for(l=0;l<25;l++)
   {
            glColor3f(0.0,0.0,0.0);
            draw_circle(-75-i+1600,40,l);
            draw_circle(-175-i+1600,40,l);
   }
}
glFlush();

}

void idle()
{
            glClearColor(1.0,1.0,1.0,1.0);
            i+=2;j+=.05;
            if(i>1630)
             i=0.0;

            if(j>302){
              j=0.0;
              if(d==0)d=1; else d=0;
              flag++;
              if(flag==2)
          {
           if(gaon)gaon=0;
                else gaon=1;
                flag=0;
          }
             }
            r+=.03;
            if(r>1)r=0;
            glutPostRedisplay();
}


void myinit()
{
glClearColor(1.0,1.0,1.0,1.0);
glColor3f(0.0,0.0,1.0);
glPointSize(2.0);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0.0,1100.0,0.0,700.0);
}

void display()
{
```

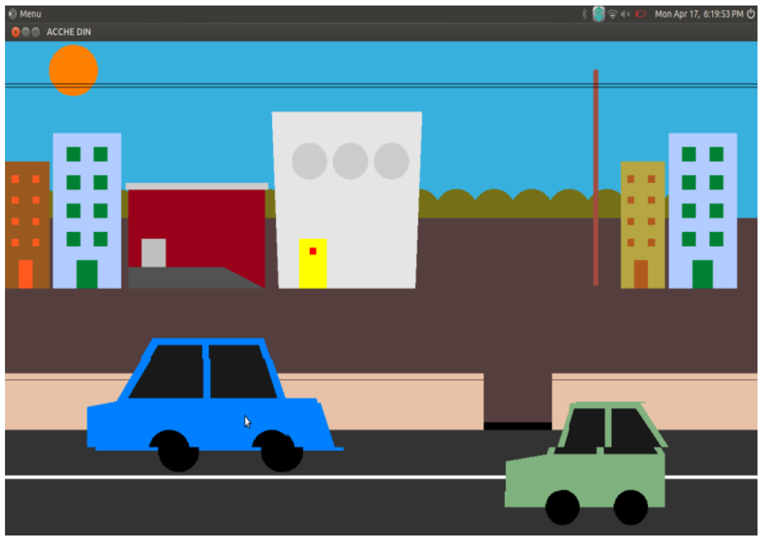
```
glClear(GL_COLOR_BUFFER_BIT);
draw_object();
glFlush();
}


int main(int argc,char** argv)
{

        glutInit(&argc,argv);
        glutInitDisplayMode(GLUT_SINGLE|
GLUT_RGB);
        glutInitWindowSize(1100.0,700.0);
        glutInitWindowPosition(0,0);
        glutCreateWindow("ACCHE DIN");
        glutDisplayFunc(display);
        glutIdleFunc(idle);
        myinit();
        glutMainLoop();
        return 0;
}
```

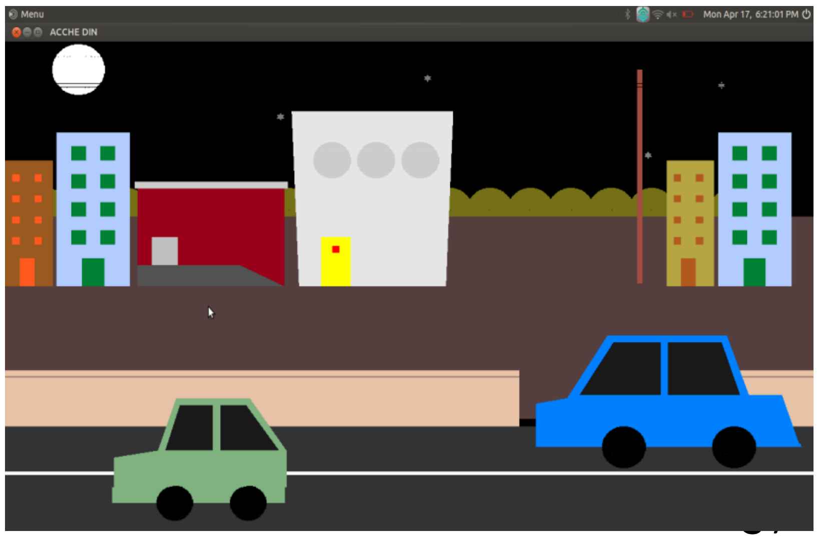
FEATURE WORK

OUTPUT:









REFERENCES/BIBLIOGRAPHY:

Websites:

- Lighthouse3d.com
- https://www.opengl.org/
- https://en.wikipedia.org/wiki/OpenGL
- Stackoverflow.com

Books:

- Computer graphics with openGL HEARN & BAKER.

THANK YOU