TABLE OF CONTENTS

LIST OF FIGURES

List of Tables

CHAPTER 1:

INTRODUCTION.

1.1 Aim

The aim of this project is to develop a 2D traffic simulation,which contains options like control the flow of traffic whit the simple city scenario.And also stopping the simulation when the user wants.The interface should be user friendly and should use mouse and keyboard interface for the interaction with the user. The main goal is to show the users how an user can control the traffic and to understand the start and stop simulation and so that he can can gain knowledge about simulation and 2D shape design

1.2Introduction to Open GL

Distinct commands that you use to specify the objects and operations needed to produce interactive OpenGL is a software interface to graphics hardware. This interface consists of about 150 three-dimensional applications.OpenGL is designed as a streamlined, hardware-independent interface to be implemented on many different hardware platforms. To achieve these qualities, no commands for performing windowing tasks or obtaining user input are included in OpenGL; instead, you must work through whatever windowing system controls the particular hardware you're using. Similarly, OpenGL doesn't provide high-level commands for describing models of three-dimensional objects. Such commands might allow you to specify relatively complicated shapes such as automobiles, parts of the body, airplanes, or molecules. With OpenGL, you must build up your desired model from a small set of *geometric primitives* - points, lines, and polygons.

1.3GLUT

GLUT, short for OpenGL Utility Toolkit, is a set of support libraries available on every major platform.OpenGL does not directly support any form of windowing, menus, or input. That's where GLUT comesin. It provides basic functionality in all of those areas, while remaining platform independent, so thatyou can easily move GLUT-based applications from, for example, Windows to UNIX with few, if any, changes. OpenGL includes the OpenGL Utility Library, GLU, with many useful functions, and most releases of OpenGL also include the OpenGL Utility Toolkit, GLUT. We saw in the first module that GLUT includes window management functions, and both GLU and GLUT include a number of built-in graphical elements that you can use.. GLUT (GL Utilities Toolkit) is a library that provides a simplified way of setting up and using OpenGL. OpenGL is a graphics API. GLUT is a library that makes using OpenGL a little easier.

1.4 Project related concept

The objective is to build an traffic signal controller which can control the flow of the traffic which can convince the about flow of traffic and the start and stop of the traffic simulations . The coding is implemented for the flow control such as stop the vehicles and pass the vehicals and the importance is given on the start and stop simulations

The basic requirements of the traffic signal control are analysed to be:

1.User Interface:

User should be able to using the mode such as day view and the night view.With the night view the traffic signal can looks good and stimulate the traffic.

2.Menu-Bar:

In menu-bar there are different options such as selecting an aeroplane ,comet and quitmode Using quit the user can directly exit from the applications

3.Start/Stop simulation:

User after selecting the view from the interfaces he can start the simulations of the traffic.As soon as they start the traffic the movement of vehicles will be happening.If they select the stop simulation the control will stop this simulation and it can be done to every vehicles

1.5 Interfaces

a.Mouse Interface:

- Opening the menu bar by the right click button to display the options.
- Select the aeroplane options to display the aeroplane.
- Use the comet options todisplay the comet in the night view.
- Use quit option to exit from the applications.

b.Keyboard Interface:

1.Start/stop simulate:

- The user can use the button r and g to start and stop.r or R button helps to stop the traffic g or G button is to start the traffic r and g indicate the “red” and “green” of the traffic signal.

2.Day/Night view:

- Buttons d and n for day and night view . Letter d and n indicated the day and night view of the traffic .

CHAPTER 2:

Requirement Specification

2.1 Hardware Requirements

* Processor-Intel 486/Pentium processor or above

* RAM-2GB Storage capacity or above

* Hard-disk-50 GB

* Monitor-Color Monitor with the minimum resolution of 1000*700

2.2 Software Requirements

* Programming language:C++

* Windows Operating system

Compiler C/C++,Graphics library,OpenGL 2.0

Codeblocks

2.3 Functional Requirements

OpenGL API's

If we want to have a control on the flow of program and if we want to interact with the window system then we use OpenGL API's GL/glut.h

We use a readily available library calles the OpenGL Utility Toolkit (GLUT),which provides the minimum functionality that should be expected in any modern windowing system

CHAPTER 3:

About The Project

3.1 Overveiw of the project

The project “Traffic Signal Controller” is based on the Computer graphics .using OpenGL Functions..It allows the user to stimulate the and control the traffic by making the use of keyboard and mouse.This project also allows to change Day/Night features and add the use of menu bar which contains the aeroplane,comet and quite features.

3.2 Set of Instruction

- To stop the traffic we need to use “r” or “R” Alphabet
- To move the traffic we need to use the button “g” or “G” Alphabet
- To change into Night mode ”n” or “N”
- To change into day mode “d” or ”D”
- To display the plane right click to menubar and select aeroplane
- To display the comet reight click to menubar and select comet
- To exit from the project right click and select quite

3.3Objective

The aim is to build the project using the OpenGL and utilize its resroces completely

- It should be able to easily understand and implementable
- To learn the creation of objects such as vehicals
- Implemet the Stop/Start Simulation and to understand it
- Providing the human interaction through mouse and keyboard.

CHAPTER 4:

Design and Implementation

4.1 Window design

Traffic signal consists of the menu bar to select the plane,comet and the quiteoptions in the main window the simulations can be done such as start of the traffic and stop of the traffic by using the mouse and the keyboard.

4.2 Menu-bar

fig 1.1

The Menu-Bar options the selected option will be displayed in the window

4.3Red and Green light

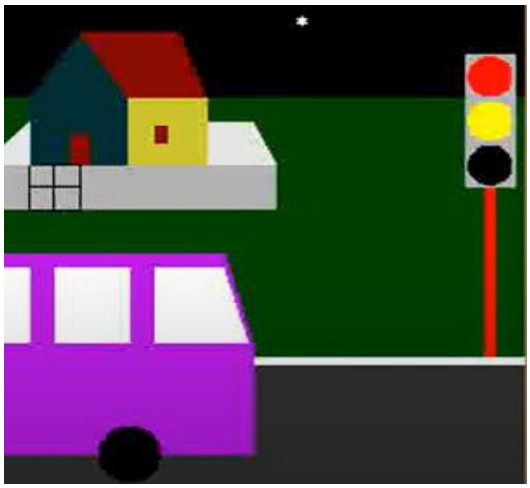


Fig1.2

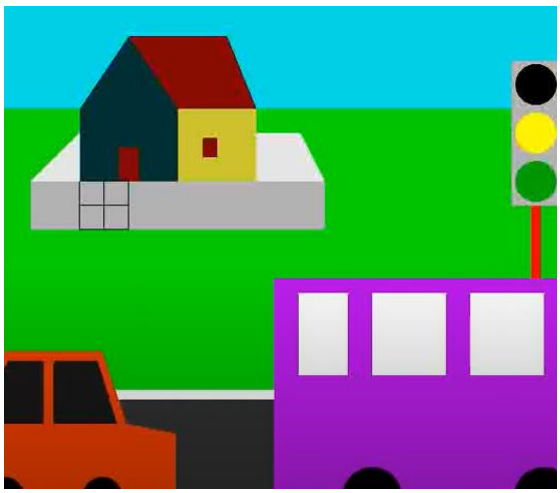


Fig 1.3

As in the figure the red light indicates the vehicles has been stopped and the green light indicates the movement of the traffic can be visualized in this figure

4.4 OpenGL Functions

gl functions for geometric primitives:

- glVertex2f 2D version with z coordinate set to 0 implicitly
- glBegin(GL_POINTS); …; glEnd( );
- glBegin(GL_LINES); …; glEnd( );
- glBegin(GL_LINE_LOOP); …; glEnd( );
- glBegin(GL_LINE_STRIP); …; glEnd( );
- glBegin(GL_POLYGON); …; glEnd( );
- glBegin(GL_TRIANGLES); …; glEnd( );
- glBegin(GL_TRIANGLE_STRIP); …; glEnd( );
- glBegin(GL_TRIANGLE_FAN); …; glEnd( );
- glBegin(GL_QUADS); …; glEnd( );
- glBegin(GL_QUAD_STRIP); …; glEnd( );

gl functions for handling the (color) buffer:

- glClear(GL_COLOR_BUFFER_BIT); //Clear the color buffer
- glFlush //Flush the buffer

gl functions for handling the color settings:

- glColor3f //Set the drawing color
- glClearColor //Set the clear color
- glShadeModel //Set the shading model as GL_FLAT or GL_SMOOTH

Some additional gl functions to fine tune the attributes of geometric primitives

glPointSize //default point size is 1 pixel

- glLineWidth //default width is 1 pixel
- glLineStipple //to set the stipple patterns of line segment, default is a solid segment
- glEnable(GL_LINE_STIPPLE) //Enable the use of stipple pattern, by default this feature is disabled
- glDisable(GL_LINE_STIPPLE) //Disable the use of stipple pattern

gl functions for the setting of image projection:

- glMatricxMode(GL_PROJECTION); //Choose the projection matrix
- glLoadIdentity(); //Reset the matrix into the identity matrix
- glOrtho //Use orthogonal projection and set up the parameters
- gluOrtho2D //A specialized version of glOrtho with the far and near set to -1 and 1 respectively

glut functions:

- glutInit
- glutCreateWindow
- glutDisplayFunc
- glutMainLoop

glutIntitDisplayMode //Default is GLU_RGB | GLUT_SINGLE

- glutInitWindowSize
- glutInitWindowPosition

4.5 Basic Pesudo code

ROAD:

```
glColor3f(1.0,1.0,1.0);
glBegin(GL_POLYGON);
glVertex2f(0,150);
glVertex2f(0,160);
glVertex2f(1100,160);
glVertex2f(1100,150);
glEnd();
glColor3f(0.2,0.2,0.2);
glBegin(GL_POLYGON);
glVertex2f(0,0);
glVertex2f(0,150);
glVertex2f(1100,150);
glVertex2f(1100,0);
glEnd();
```

VEHICLES:

```
glColor3f(0.9,0.2,0.0);
glBegin(GL_POLYGON);
glVertex2f(25+i,50);
glVertex2f(25+i,125);
glVertex2f(75+i,200);
glVertex2f(175+i,200);
glVertex2f(200+i,125);
glVertex2f(250+i,115);
glVertex2f(250+i,50);
glEnd();
glColor3f(0.1,0.1,0.1);
glBegin(GL_POLYGON);
glVertex2f(35+i,125);
glVertex2f(80+i,190);
glVertex2f(115+i,190);
glVertex2f(115+i,125);
glEnd();
glColor3f(0.1,0.1,0.1);
glBegin(GL_POLYGON);
glVertex2f(125+i,125);
```

```
glVertex2f(125+i,190);
glVertex2f(170+i,190);
glVertex2f(190+i,125);
glEnd();
```

TRAFFIC SIGNAL:

```
for(a=0;a<2;a++)
for(b=0;b<2;b++)
 {
glColor3f(0.0,0.0,0.0);
        glBegin(GL_LINE_LOOP);
         glVertex2f(x[a],y[b]);
    glVertex2f(x[a],y[b+1]);
         glVertex2f(x[a+1],y[b+1]);
         glVertex2f(x[a+1],y[b]);
        glEnd();
 }
        glColor3f(1.0,0.0,0.0);
        glBegin(GL_POLYGON);
                glVertex2f(1060,160);
                glVertex2f(1060,350);
                glVertex2f(1070,350);
                glVertex2f(1070,160);
        glEnd();
        glColor3f(0.7,0.7,0.7);
        glBegin(GL_POLYGON);
                glVertex2f(1040,350);
                glVertex2f(1040,500);
                glVertex2f(1090,500);
                glVertex2f(1090,350);
        glEnd();
```

```
        for(l=0;l<=20;l++)
        {
                glColor3f(0.0,0.0,0.0);
                draw_circle(1065,475,l);
                glColor3f(1.0,1.0,0.0);
                draw_circle(1065,425,l);
                glColor3f(0.0,0.0,0.0);
                draw_circle(1065,375,l);
        }
```

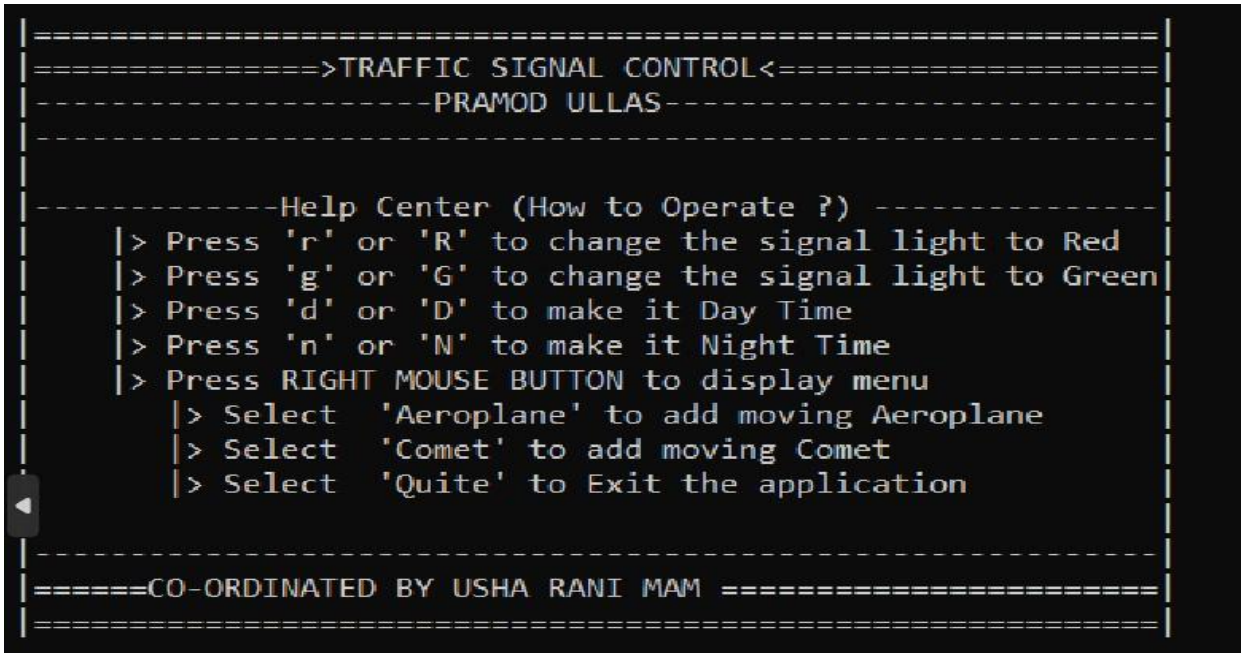
MAIN FUNCTION:

```
int main(int argc,char*argv[])
{  int c_menu;
 printf("\n");
printf("|================================================|\n");
   printf(" |===>ULLAS PRAMOD <=======================|\n");
   printf(" #----Project:-'Simulation of Traffic Control'------#\n");
   printf(" |-------------------------------------------------------|\n");
   printf(" |                                                |\n");
   printf(" #----Help Center (How to Operate ?) ------------------------#\n");
        printf(" |    |> Press 'r' or 'R' to change the signal light to Red  |\n");
        printf(" |    |> Press 'g' or 'G' to change the signal light to Green|\n");
        printf(" |    |> Press 'd' or 'D' to make it Day Time              |\n");
        printf(" |    |> Press 'n' or 'N' to make it Night Time            |\n");
        printf(" |    |> Press RIGHT MOUSE BUTTON to display menu          |\n");
        printf(" |      |> Select  'Aeroplane' to add moving Aeroplane     |\n");
        printf(" |      |> Select  'Comet' to add moving Comet             |\n");
        printf(" |      |> Select  'Quite' to Exit the application         |\n");
        printf(" |                                                |\n");
        printf(" |-------------------------------------------------------|\n");
   printf(" |======CO-ORDINATED BY:USHA RANI MAM======|\n");
```

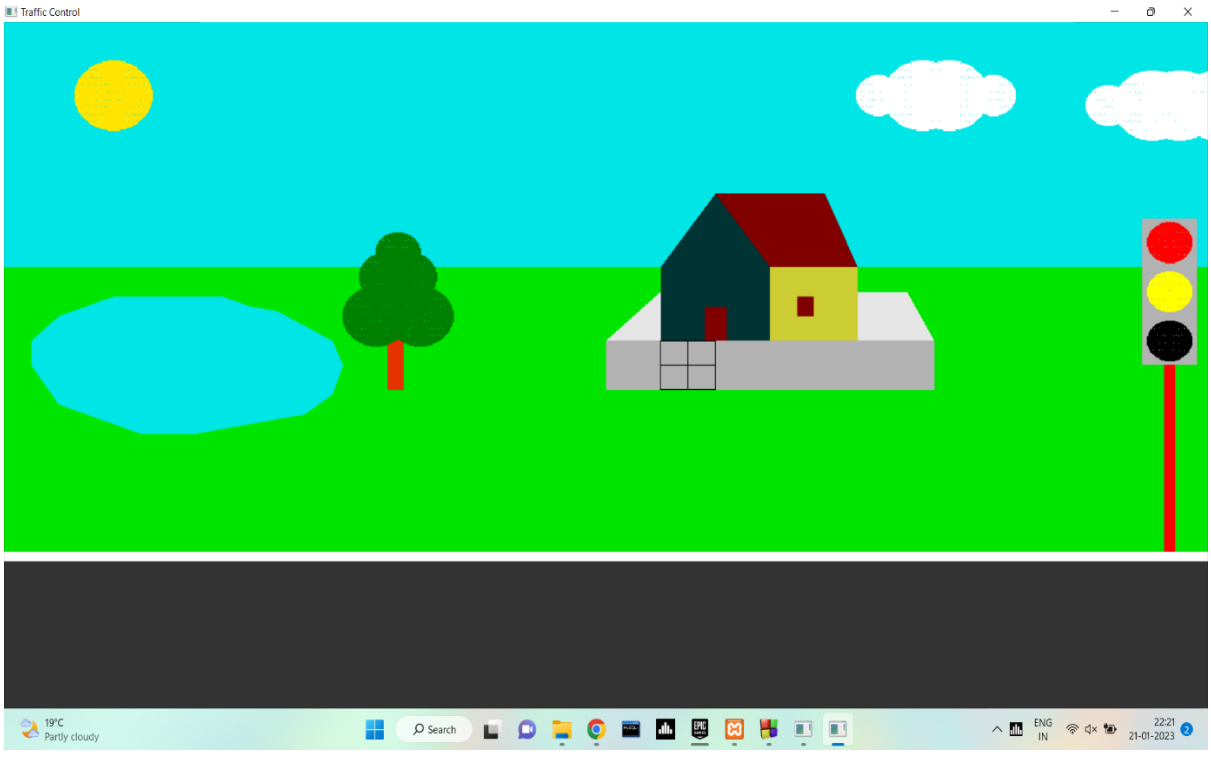
```
 printf("|=============================================|\n");
   glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
        glutInitWindowSize(1100.0,700.0);
        glutInitWindowPosition(250,0);
        glutCreateWindow("Traffic Control");
        myinit();
        glutDisplayFunc(display);
        glutIdleFunc(idle);
        glutKeyboardFunc(keyboardFunc);
        c_menu=glutCreateMenu(main_menu);
        glutAddMenuEntry("Aeroplane",1);
        glutAddMenuEntry("Comet",2);
        glutAddMenuEntry("Quite",3);
        glutAttachMenu(GLUT_RIGHT_BUTTON);
        glutMainLoop();
        return 0;
}
```

CHAPTER 5:

SCREEN SHOTS



a)This is the instruction page fig 2.1



b)This is the scenario image fig2.2

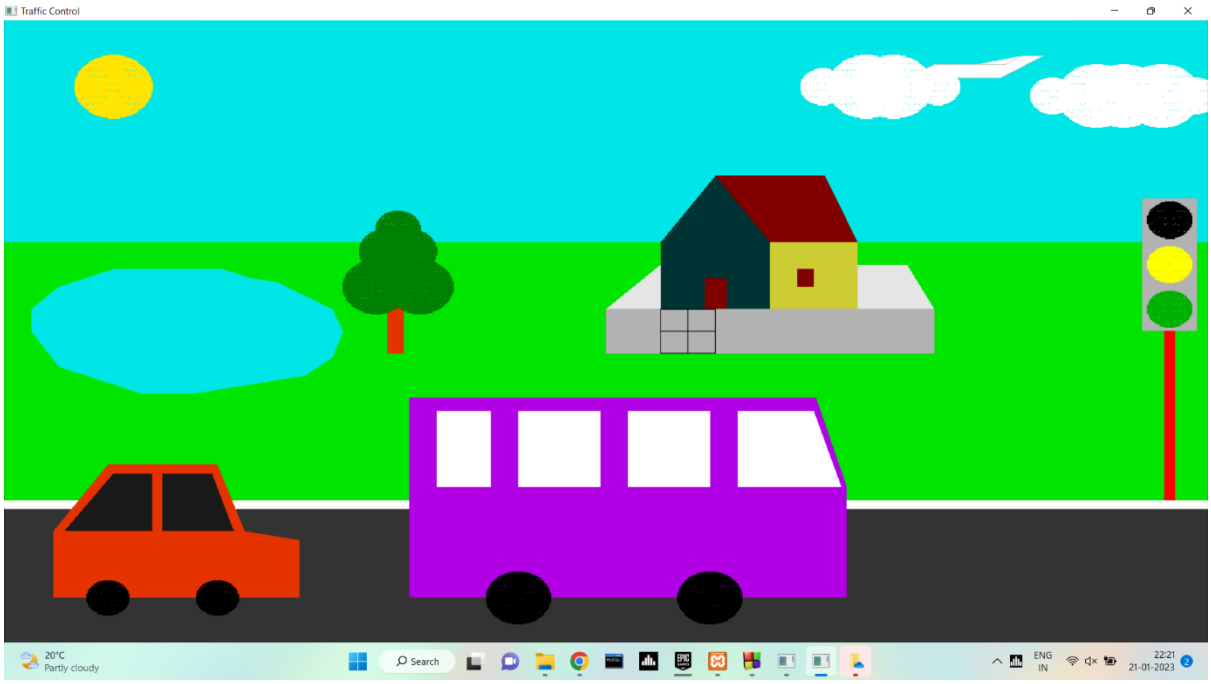


Fig2.3

c) Images of day view traffic and the vehicals

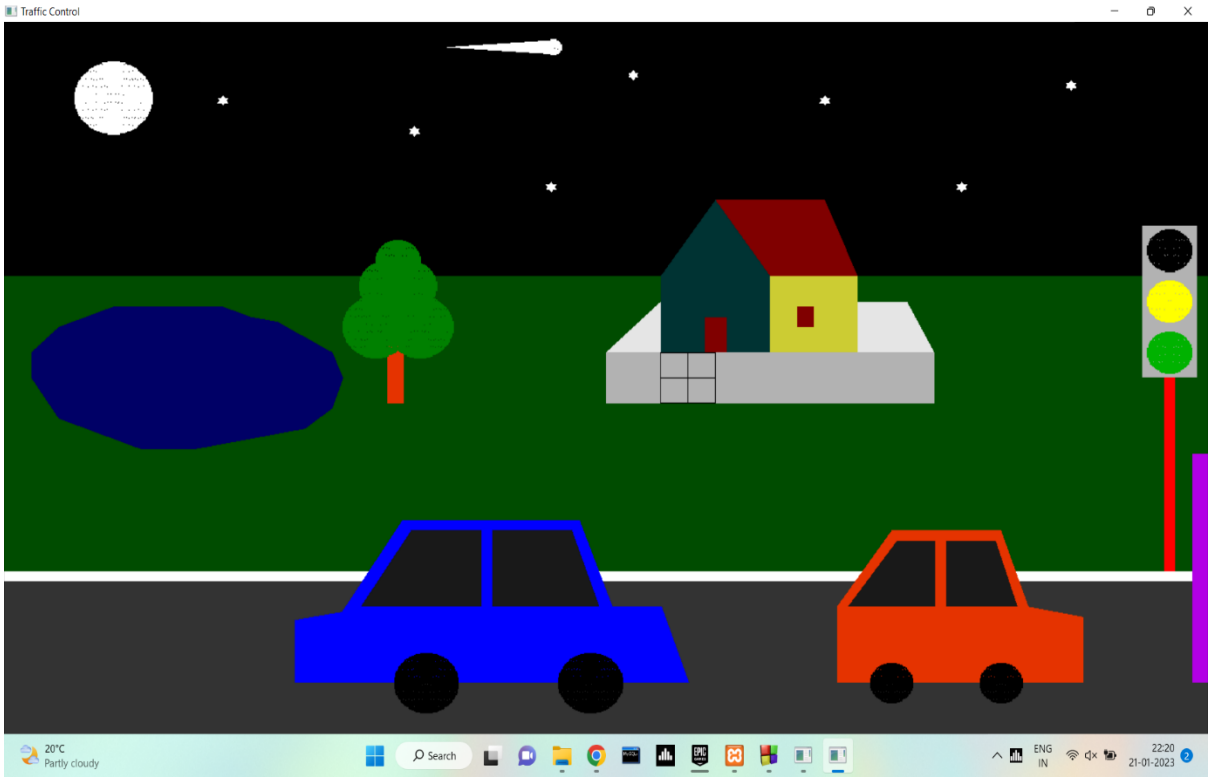


Fig2.4

d)Image of the comet

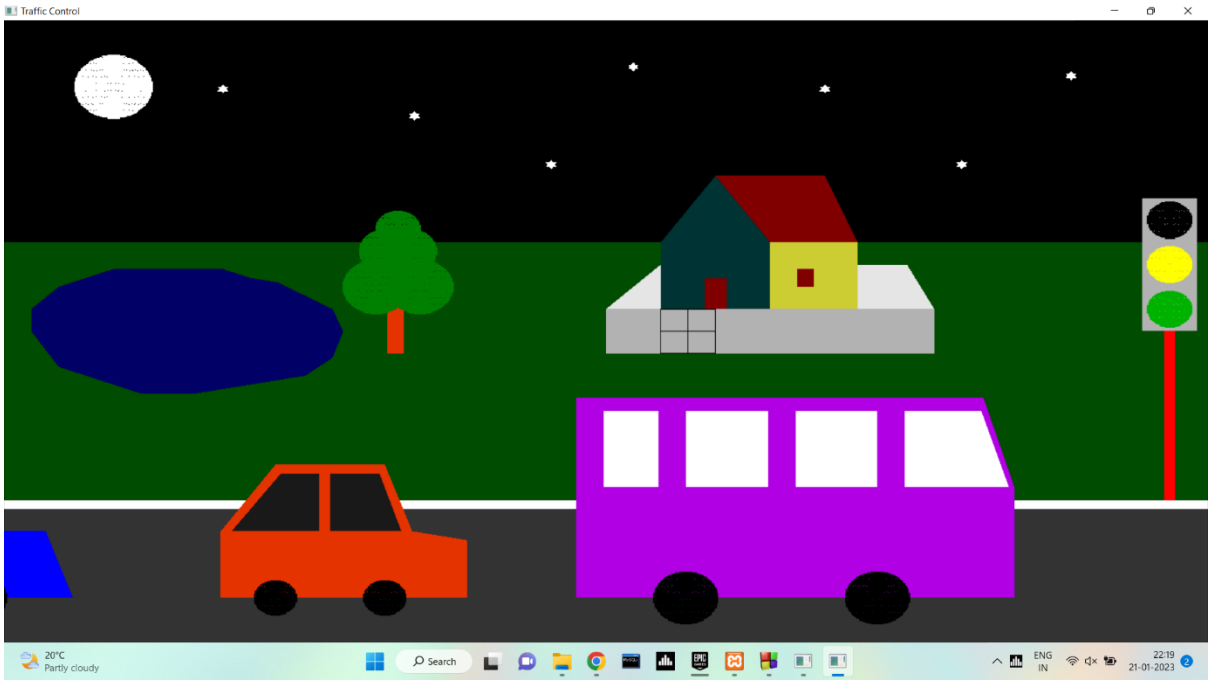


Fig2.5

e)Images of the Stop view

\

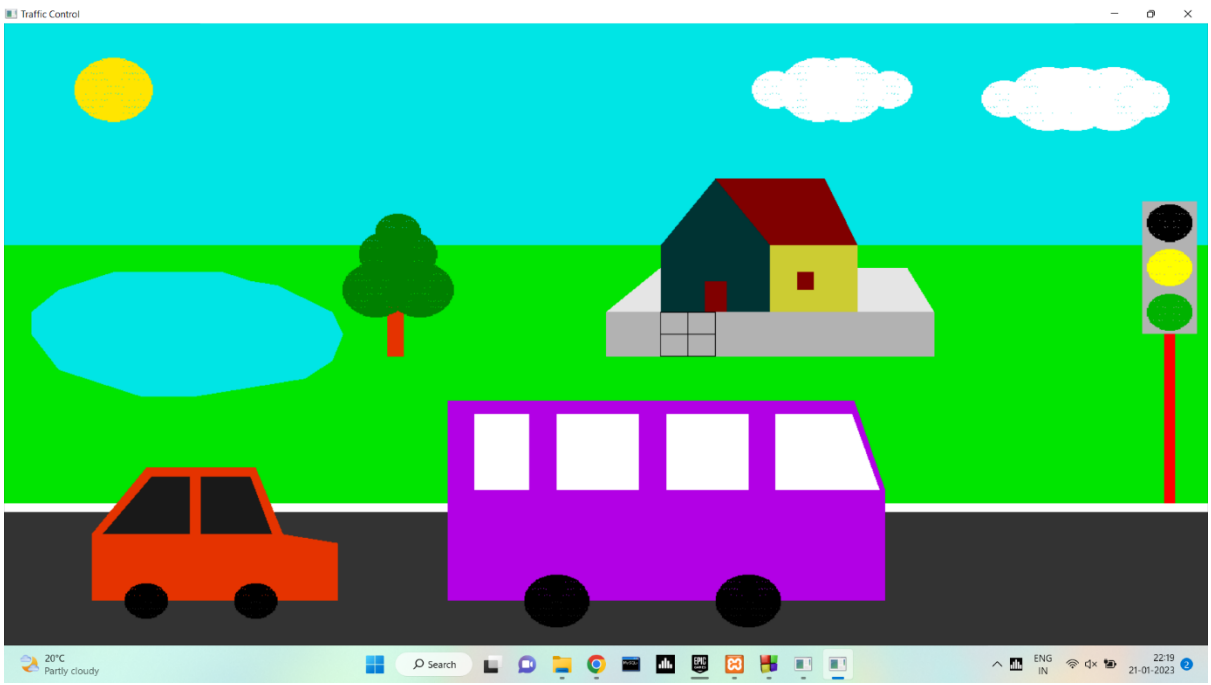


Fig 2.6

f)Image of Start-View

CHAPTER 6

TESTING

6.1 INTRODUCTION

Testing process started with the testing of individual program units such as functions objects.These were then integrated into sub-systems and systems, and interactions of these units were tested.

Testing involves verification and validation.

Validation: “Are we building right product?”

Verification: “Are we building the product right?”

The ultimate goal of the verification and validation process is to establish confidence that the software system is ‘fit for purpose’. The level of required confidence depends on the system’s purpose, the expectations of the system users and the current marketing environment for the system. With the verification and validation process, there are two complementary approaches to the system checking and analysis:

Software inspections or peer reviews analyses and check system representations such as the requirements document, design diagrams, and the program source code. Software testing involves running an implementation of the software with test data.

6.2 TEST CASES:

Mouse Interface Test Cases:

Sl.No	Functionality	Comments	Remarks
1	Mouse Right Click	It shows Menu Bar	Pass
2	Selecting the options	It shows the list of options to user	Pass
3	Select Aeroplane	Plane Displays	Pass
4	Select Comet	Movement of Comet	Pass
5	Select Quite	Exit	Pass

Tb1.1

Keyboard Interface Test Cases :

Sl-No	Functionality	Comments	Remarks
1	Start Simulation	Press “g” or “G”	Pass
2	Stop Simulation	Press “r” or “R”	Pass
3	Day Veiw	Press ”d” or “D”	Pass
4	Night Veiw	Press “n” or “N”	Pass

Tb 1.2

CHAPTER 7

7.1 CONCLUSION

The ‘Traffic Signal Controller’ has been tested under Windows 11 and has been found to provide ease of use and manipulation to the user.This projectis created for the windows operating system used to draw lines,boxes ,circles,ellipses, and polygons . It is very simple and effective user interface We found designing and developing this traffic signal control as a very interesting and learning experience. It helped us to learn about computer graphics,design of Graphical User Interfaces, interface to the user, user interaction handling and screen management,The graphics editor provides all and more than the features that have been detailed in the university syllabus.

7.2 FUTURE ENHANCEMENTS

The following are some of the features that are planned to be supported in the future versions of the ‘Traffic Signal Control’ .

- Adding more Two-Way directions road
- Controlling the traffic using the Time and Density of the traffic.
- Creating the Zebra-crossing.
- If it possible then can improving the project 3D-Development.

CHAPTER 8:

Bibilography

- Azzikha's Text book of 'The OpenGL and its uses' 5th Edition 2008.
- Interactive computer Graphics --A top down approach using open GL--by Edward Angle
- Jackie.L.Neider,Mark Warhol,Tom.R.Davis,"OpenGL Red Book",Second Revised Edition,2005.
- Donald D Hearn and M.Pauline Baker,"Computer Graphics with OpenGL", 3rd Edition.
- Portion of the code for implementing GEAR has been borrowed from Brian Paul's MESA.
- https://www.javatpoint.com/computer-graphics-tutorial
- https://www.opengl.org/