Introduction:-

To show a car in 3d model.

Functions:-

All the functions can be access by keyword.

'e' :- bonate close

'E' :- bonate open
'r' :- red sopt-light
'b' :- blue sopt-light
'g' :- green sopt-light
'c' :- clear light
'p' :- open roof
'P' :- close roof
'l' :- head lights on
'L' :- head lights off
'o' :- wheel rotation on
'O' ;- wheel rotation off
ESCAPE :- turn off the window.
'x':- rotate x-angle in +ve
'X':- rotate x-angle in -ve
'y':- rotate y-angle in +ve
'Y':- rotate y-angle in +ve
'z':- rotate z-angle in +ve
'Z':- rotate z-angle in +ve
'u':- Move UP
'U':- Move Down
'f':- Move forward
'F':- Move away
's':- Scale(+ve) w.r.t z
'S':- Scale(1/+ve) w.r.t z
'a':- Scale(+ve) w.r.t y
'A':- Scale(1/+ve) w.r.t y
'q':- Scale(+ve) w.r.t x
'Q':- Scale(1/+ve) w.r.t x

Function in code:-

```
void NormalKey(GLubyte key, GLint x, GLint y)
{
        switch ( key )  {

   case 'e'       :  eflag = 0;                          //bonate close
                                        glutDestroyWindow(window);
                                        break;

   case 'E'       :  eflag = 1;                          //bonate open
                                        glutDestroyWindow(window);
                                        break;


  case 'r'     : red = 1.0;                             //red light
                                        blue = 0.0;
                                        green = 0.0;
                                        glutDestroyWindow(window);
                                        break;

  case 'b'     : red = 0.0;                             //green light
                                        blue = 1.0;
                                        green = 0.0;
                                        glutDestroyWindow(window);
                                        break;

  case 'g'     : red = 0.0;                        // green light
                                        blue = 0.0;
                                        green = 1.0;
                                        glutDestroyWindow(window);
                                        break;

    case 'c'     : red = 0.0;                        //clear light
                                        blue = 0.0;
                                        green = 0.0;
                                        glutDestroyWindow(window);
                                        break;
```

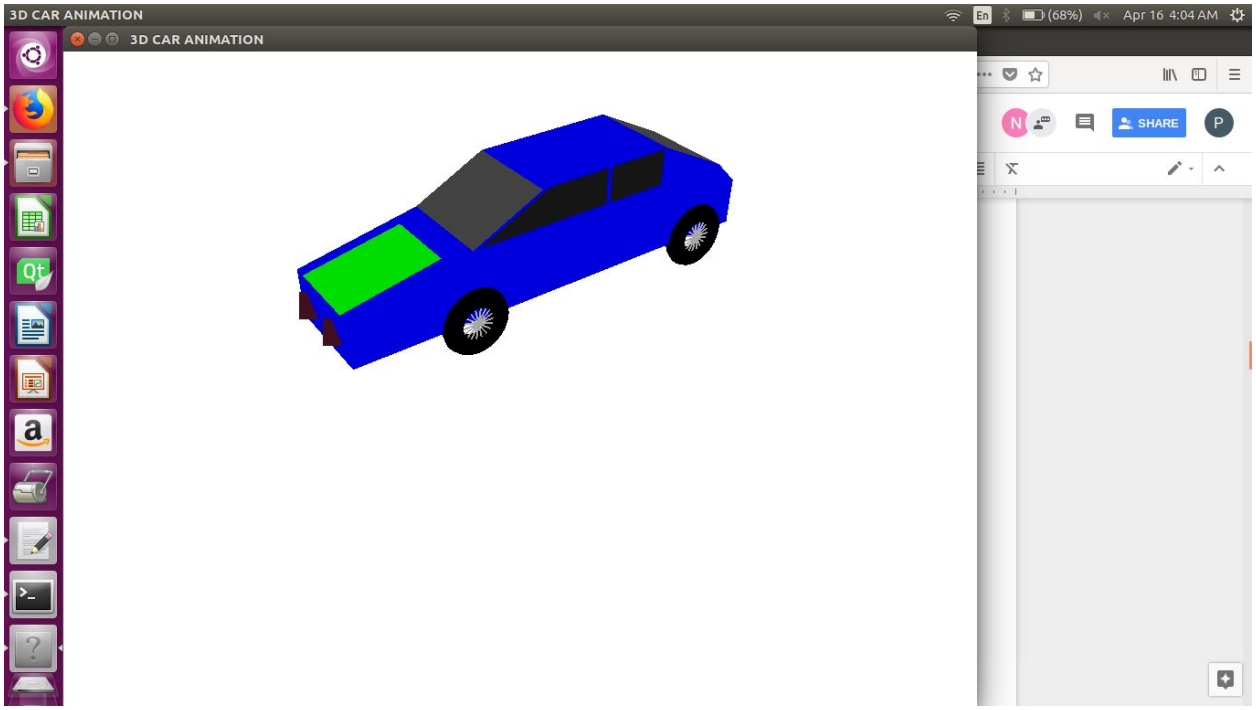
```
case 'p'   :
                flagg1=1;
                glutPostRedisplay();
                break;
 case  'P'  :
                flagg1=0;
                glutPostRedisplay();

   case 'l'       :  flagt=1;                          // head lights on
                              glutDestroyWindow(window);
                                      break;

   case 'L'       :  flagt=0;                           // head lights off
                              glutDestroyWindow(window);
                                      Break;

        case 'O'          :  wheelflag = 1;                  // wheel rotation on
                                    glutDestroyWindow(window);
                                      break;

        case 'o'          :  wheelflag = 0;                  // wheel rotation off
                                    glutDestroyWindow(window);
                                      break;



case ESCAPE : printf("escape pressed. exit.\n");
                glutDestroyWindow(window);    /* Kill our window */
                exit(0);
                break;

 case 'x':      xangle += 5.0;                                  // rotate x-angle in +ve
                glutPostRedisplay();
                break;

  case 'X':     xangle -= 5.0;                                  // rotate x-angle in -ve
                glutPostRedisplay();
                break;

 case 'y':       yangle += 5.0;                                  // rotate y-angle in +ve
                              glutPostRedisplay();
```

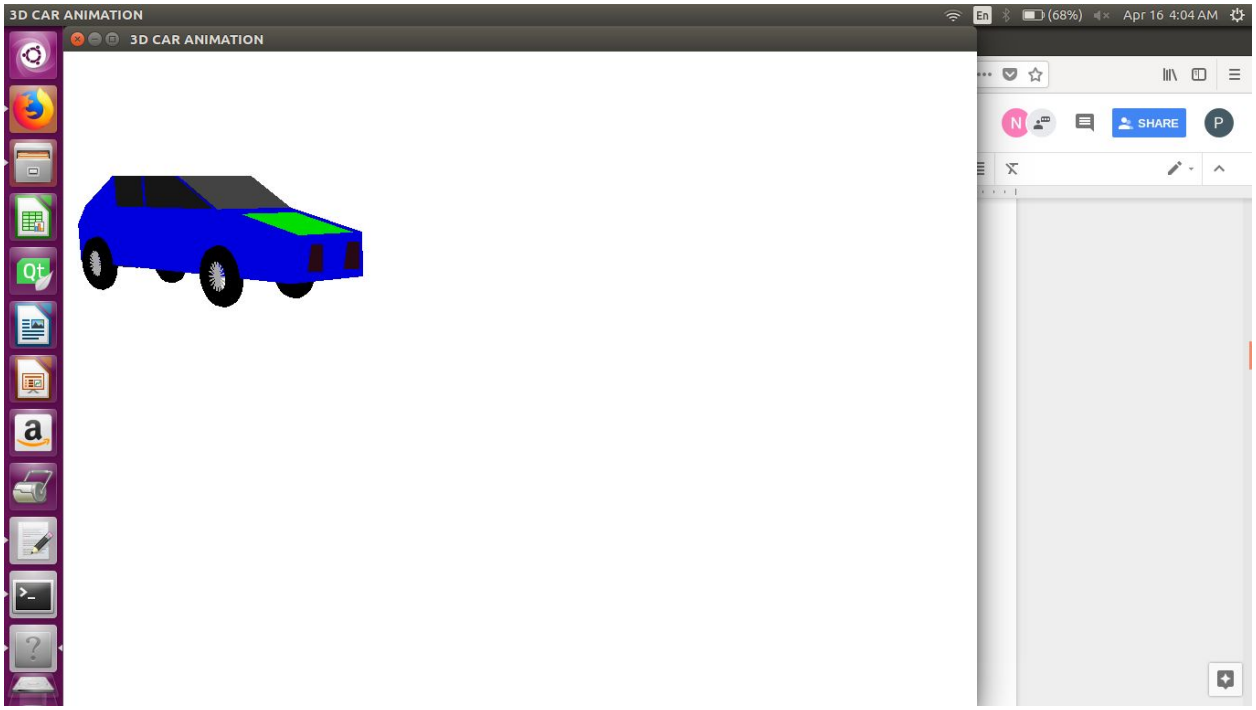
```
                                    break;

case 'Y':       yangle -= 5.0;                          // rotate y-angle in -ve
                                    glutPostRedisplay();
                                    break;

case 'z':       zangle += 5.0;                          // rotate z-angle in +ve
                                    glutPostRedisplay();
                                    break;

case 'Z':       zangle -= 5.0;                          // rotate z-angle in -ve
                                    glutPostRedisplay();
                                    break;


 case 'u':                          /* Move up */
        yt += 0.2;
        glutPostRedisplay();
        break;

 case 'U':
        yt -= 0.2;                  /* Move down */
        glutPostRedisplay();
        break;

 case 'f':                          /* Move forward */
        zt += 0.2;
        glutPostRedisplay();
        break;

 case 'F':
        zt -= 0.2;                  /* Move away */
        glutPostRedisplay();
        break;

 case 's':zs+=.2;                                   //Scale w.r.t z
           glutPostRedisplay();
           break;

 case 'S':zs-=0.2;
           glutPostRedisplay();
           break;
```

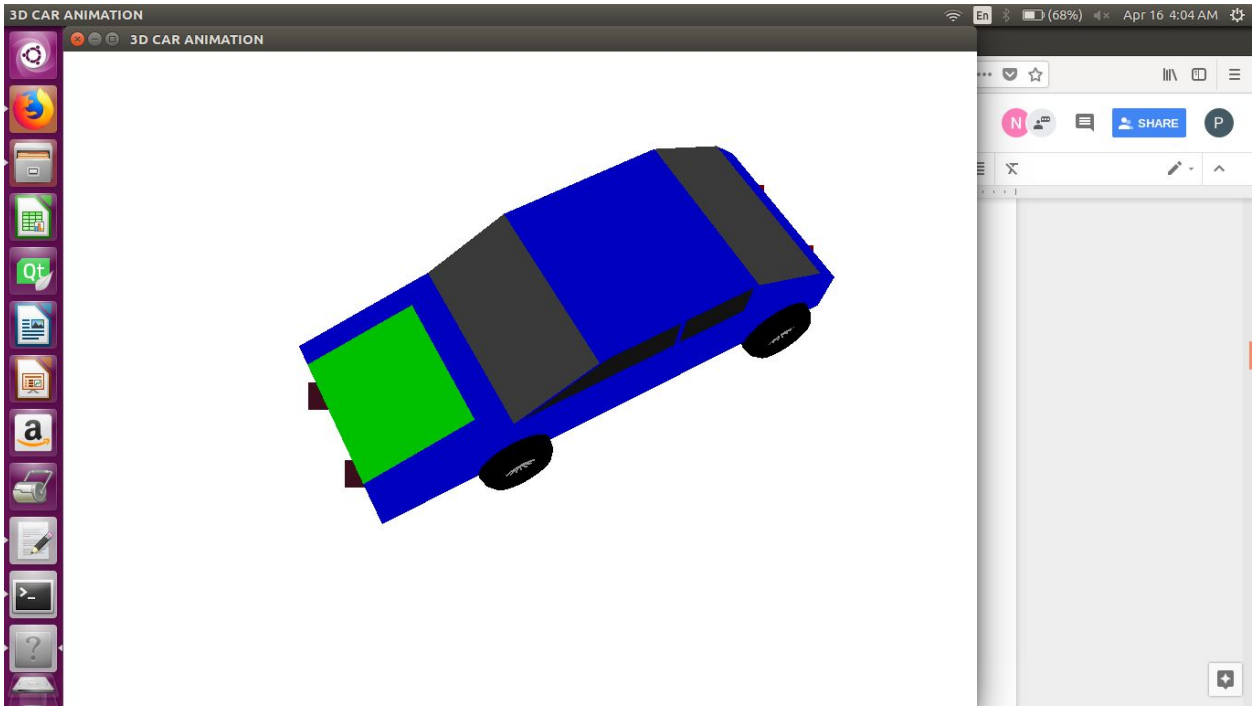
```
 case 'a':ys+=.2;                          //Scale w.r.t y
          glutPostRedisplay();
          break;

 case 'A':ys-=0.2;
          glutPostRedisplay();
          break;

 case 'q':xs+=.2;                          //Scale w.r.t x
          glutPostRedisplay();
          break;

 case 'Q':xs-=0.2;
          glutPostRedisplay();
          break;


 default: DrawGLScene();
          break;
          }

}
```

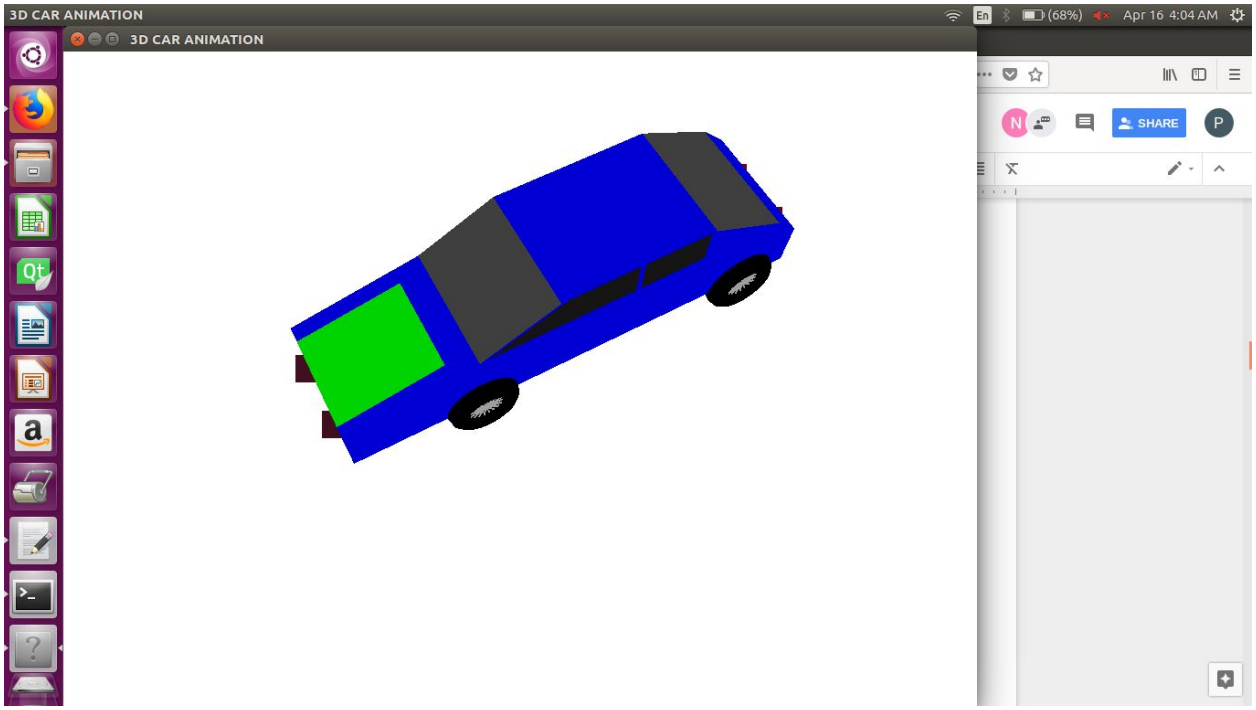
Rotate, Translate and Scaling functions:-

```
glTranslatef(-1.0,0.0,-3.5);
glRotatef(xangle,1.0,0.0,0.0);
glRotatef(yangle,0.0,1.0,0.0);
glRotatef(zangle,0.0,0.0,1.0);
glTranslatef(xt,yt,zt);
glScalef(xs,ys,zs);
```


3D CAR ANIMATION
3D CAR ANIMATION
(68%)
Apr 16 4:04 AM
SHARE


3D CAR ANIMATION
3D CAR ANIMATION
(68%)
Apr 16 4:04 AM
SHARE





Lightining:-

1. Press ‘r’ for red color.
2. Press ‘g’ for green color.
3. Press ‘b’ for blue color.

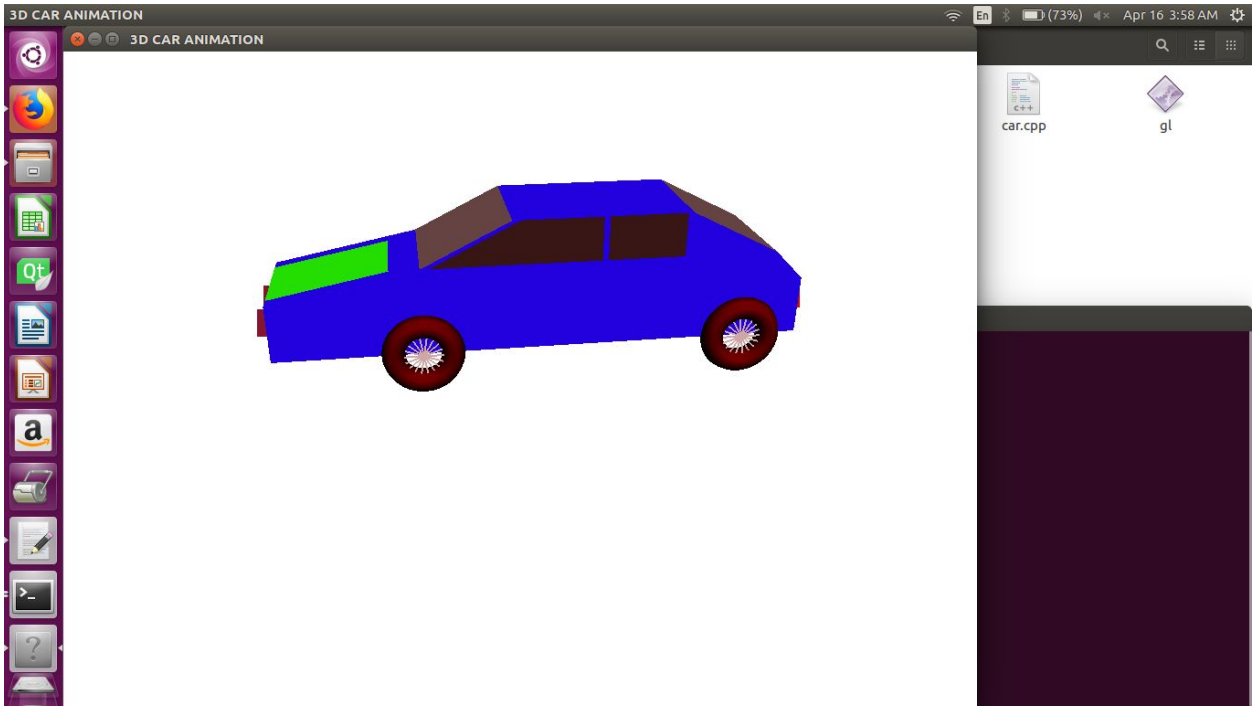
4. Press ‘c’ to above clear color.

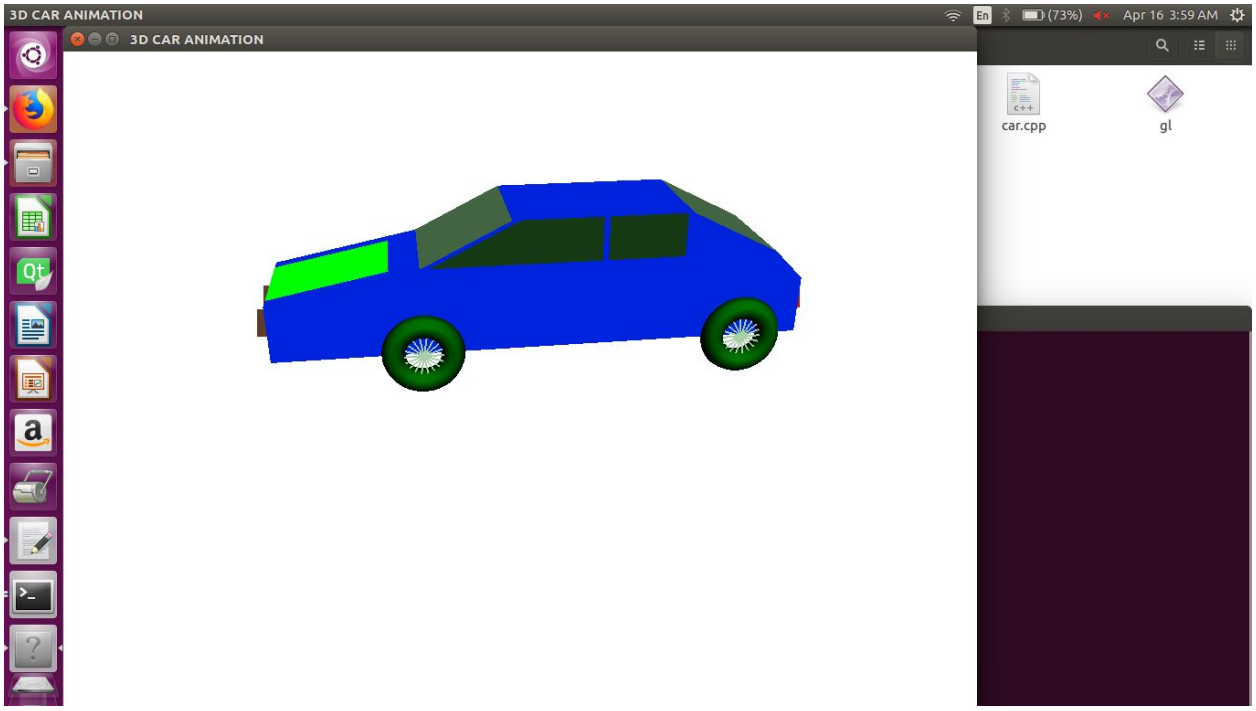
```
GLvoid InitGL(GLfloat Width, GLfloat Height)
{

  glClearColor(1.0, 1.0, 1.0, 1.0);
  glLineWidth(2.0);                /* Add line width,   ditto */
  Transform( Width, Height ); /* Perform the transformation */

  t=gluNewQuadric();
          gluQuadricDrawStyle(t, GLU_FILL);

glEnable(GL_LIGHTING);

glEnable(GL_LIGHT0);

// Create light components
GLfloat ambientLight[] = { 0.2f, 0.2f, 0.2f, 1.0f };
GLfloat diffuseLight[] = { 0.8f, 0.8f, 0.8, 1.0f };
GLfloat specularLight[] = { 0.5f, 0.5f, 0.5f, 1.0f };
GLfloat position[] = { 1.5f, 1.0f, 4.0f, 1.0f };

// Assign created components to GL_LIGHT0
glLightfv(GL_LIGHT0, GL_AMBIENT, ambientLight);
glLightfv(GL_LIGHT0, GL_DIFFUSE, diffuseLight);
glLightfv(GL_LIGHT0, GL_SPECULAR, specularLight);
glLightfv(GL_LIGHT0, GL_POSITION, position);

}
```

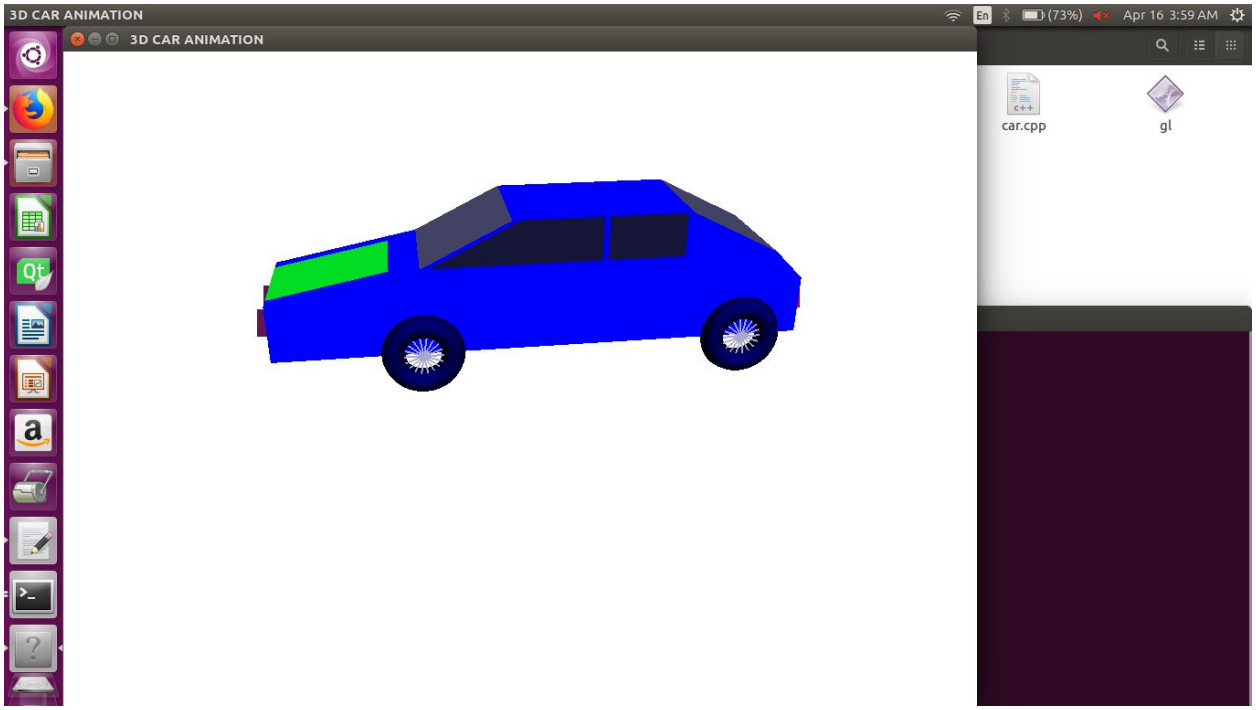
To call the function and assign the values:-

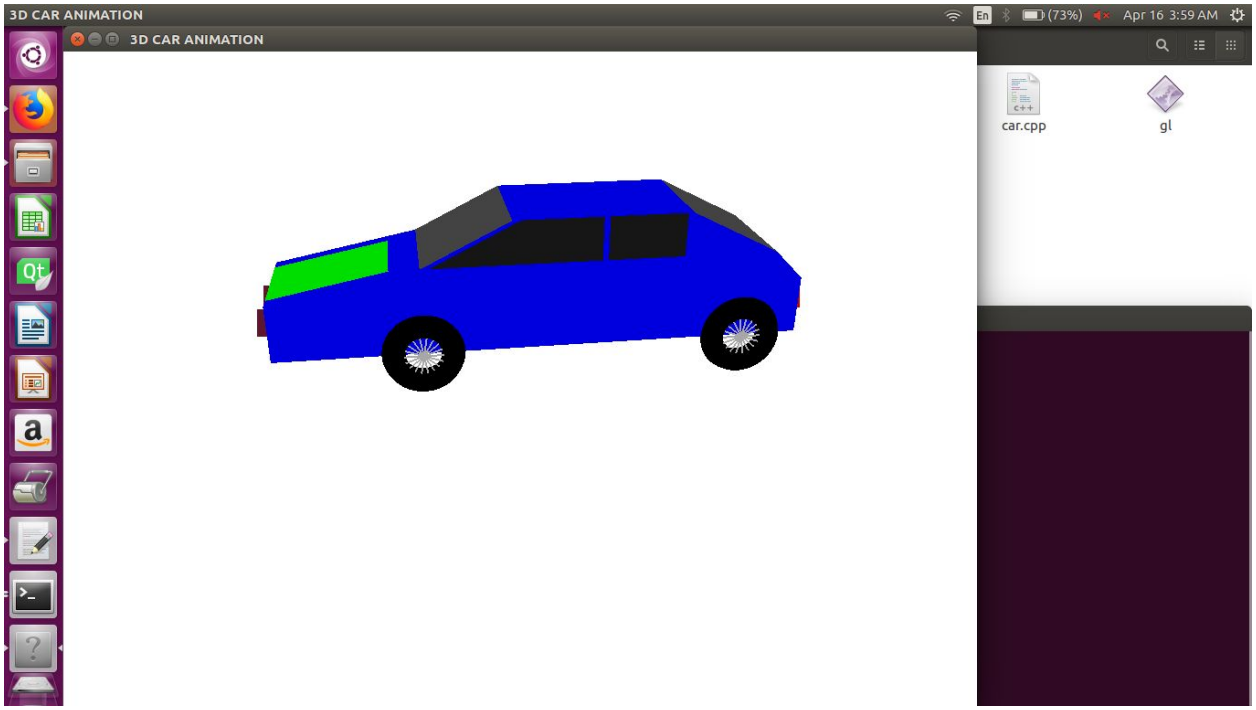
```
  GLfloat mat_specular1[] = { red, green, blue, 1.0 };                    // Lightning
  GLfloat mat_shininess1[] = { 2.0 };
   GLfloat light_position1[] = { 0.1, 0.3, 0.3, 0. };
   glClearColor (1.0, 1.0, 1.0, 0.0);
   glShadeModel (GL_SMOOTH);

   glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular1);
   glMaterialfv(GL_FRONT, GL_SHININESS, mat_shininess1);
   glLightfv(GL_LIGHT0, GL_POSITION, light_position1);
```

```
glEnable(GL_LIGHTING);
glEnable(GL_LIGHT0);
glEnable(GL_DEPTH_TEST);
```



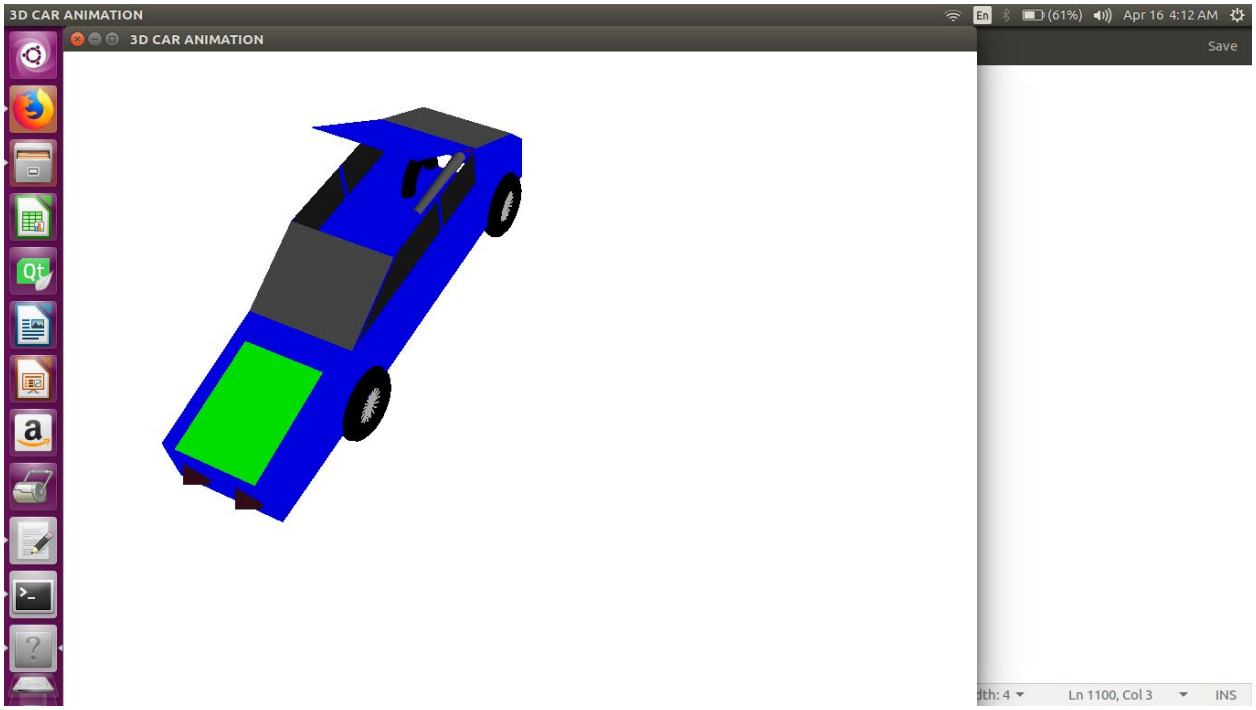






Open Roof:-

1. Press ‘p’ to open roof of car.
2. Press ‘P’ to close roof of car.



Wheel Rotation:-

1.Press “O” to check the wheel rotation and then press the left or right arrow to check the rotation.

2.Press “o” to end the wheel rotation.

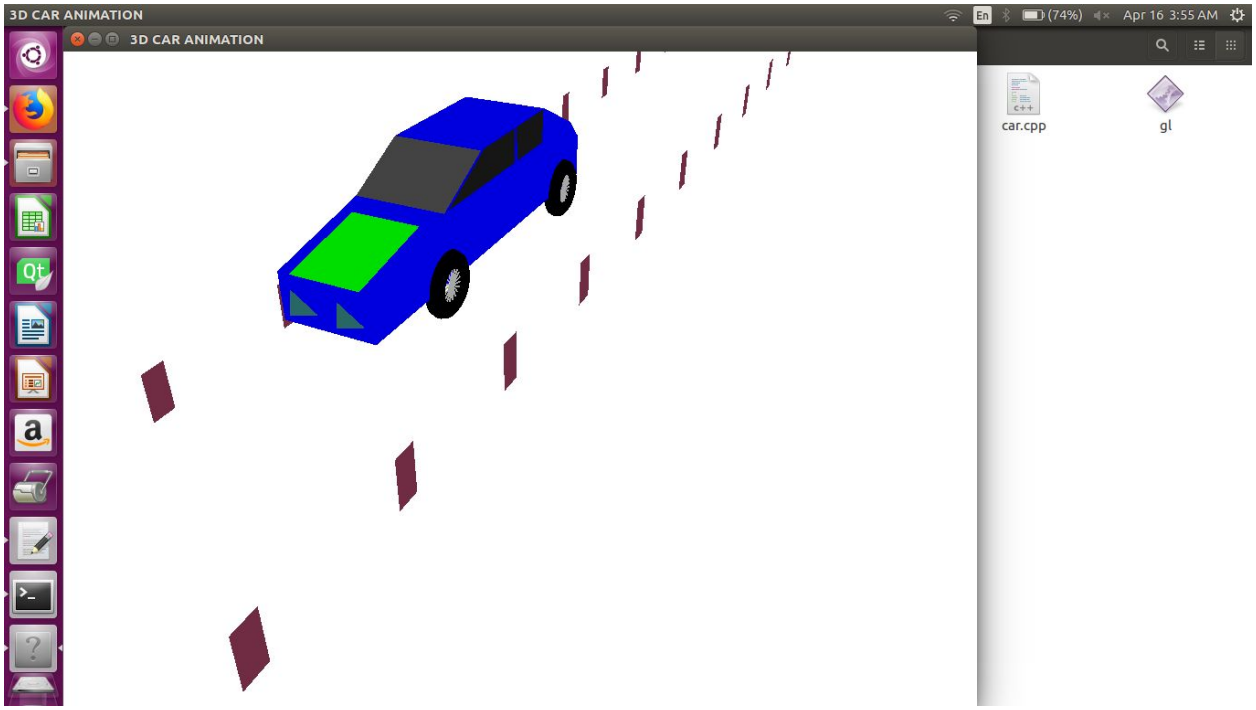
Code:-

```
if(wheelflag)
{
    glPushMatrix();
    glTranslatef(xw,0,0);
    glColor3f(0.5,.2,0.3);
    glBegin(GL_QUADS);
        glEnd();
    glPopMatrix();
}
```

Mouse Rotation:-

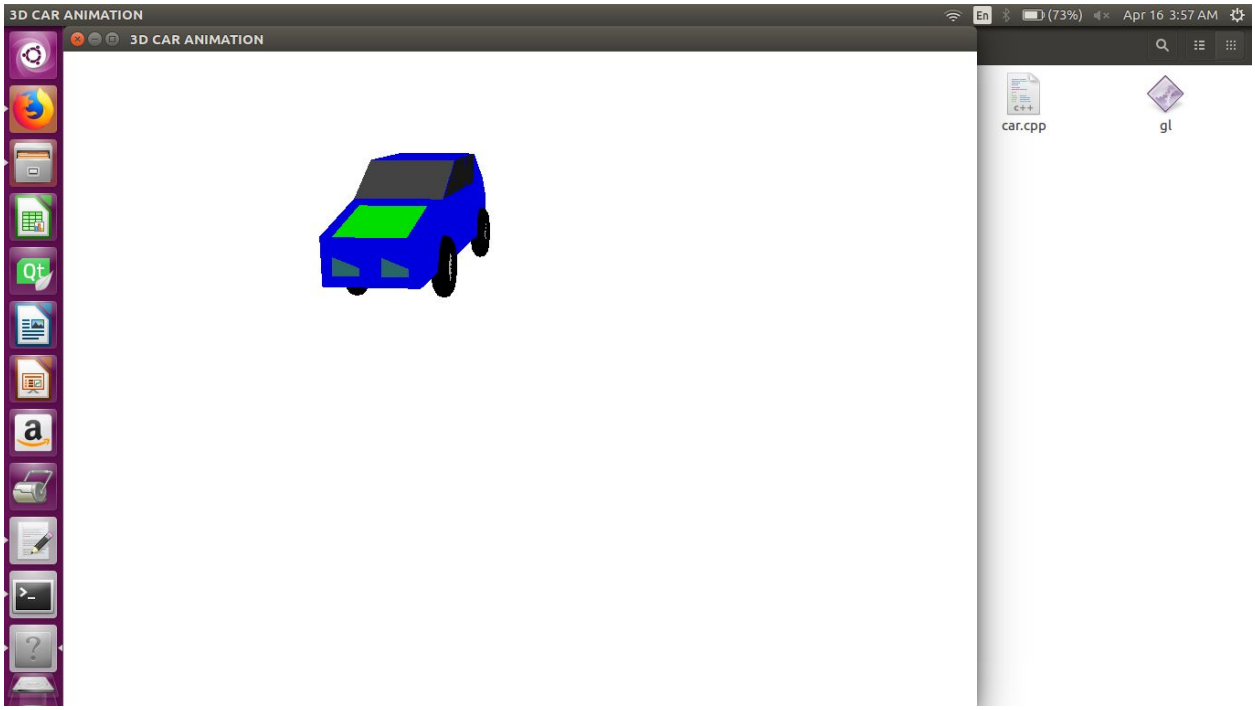
1. Left click and move the mouse to check the rotation.

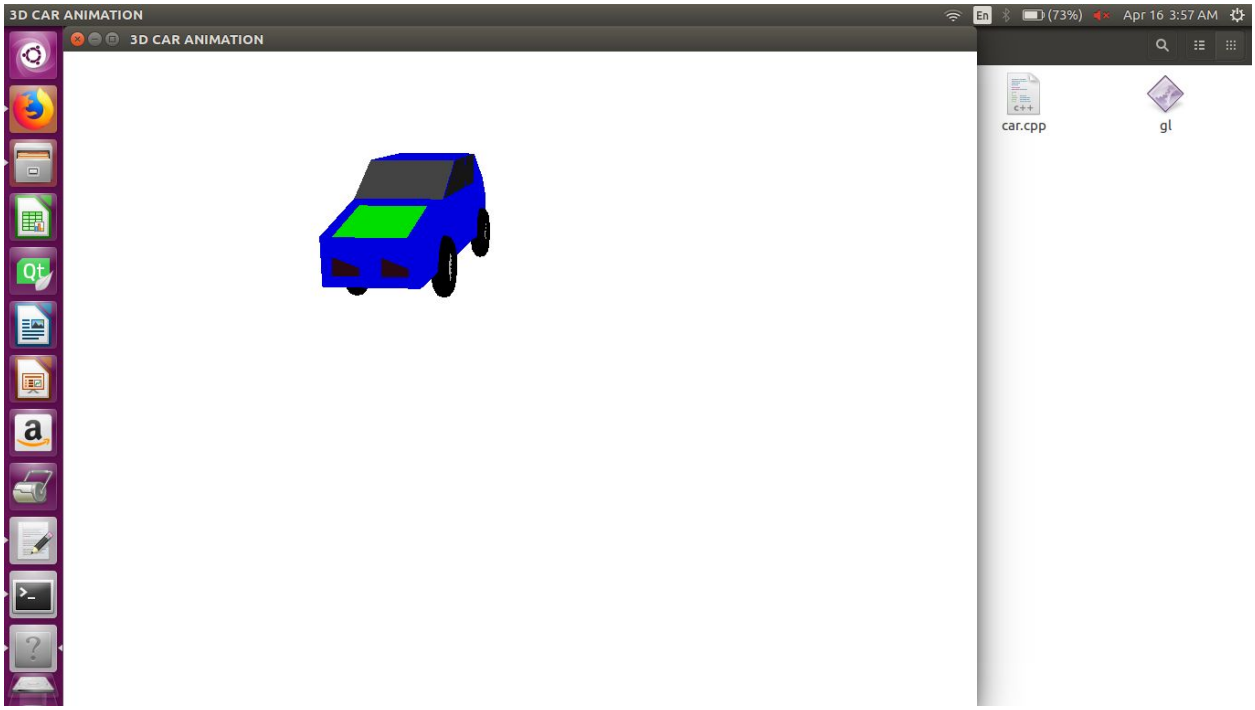
```
void mouse(int button, int state, int x, int y)
{
if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN)
{
mouseDown = true;

xdiff = x - yangle;
ydiff = -y + xangle;
}
else
  mouseDown = false;
}
void mouseMotion(int x, int y)
{
if (mouseDown)
{
yangle = x - xdiff;
xangle = y + ydiff;

glutPostRedisplay();
}
}
```



Light On and Off:-

```
if(flagt==0)                                        //headlights On
{
  glColor3f(.4,.1,0.2);
  glPointSize(30.0);
  glBegin(GL_POINTS);
  glVertex3f(0.1,0.3,0.3);
  glVertex3f(0.1,0.3,0.5);
  glEnd();
  glPointSize(200.0);
}
Else                                                // headlights Off
{
 glColor3f(.4,1,1);
  glPointSize(30.0);
  glBegin(GL_POINTS);
  glVertex3f(0.1,0.3,0.3);
  glVertex3f(0.1,0.3,0.5);
  glEnd();
  glPointSize(200.0);
}
```


3D CAR ANIMATION
3D CAR ANIMATION
(73%)
Apr 16 3:57 AM
car.cpp
gl


3D CAR ANIMATION
3D CAR ANIMATION
(73%)
Apr 16 3:57 AM
car.cpp
gl

Sound:-

```
int fun()
        {
                if (SDL_Init(SDL_INIT_AUDIO) < 0)
                return 1;

          // local variables
          static Uint32 wav_length; //length of our sample
          static Uint8 *wav_buffer; //buffer containing our audio file
          static SDL_AudioSpec wav_spec; //the specs of our piece of music


          /* Load the WAV */
          // the specs, length and buffer of our wav are filled
          if( SDL_LoadWAV(MUS_PATH, &wav_spec, &wav_buffer, &wav_length) ==
        NULL ){
                return 1;
          }
          // set the callback function
          wav_spec.callback = my_audio_callback;
          wav_spec.userdata = NULL;
          // set our global static variables
          audio_pos = wav_buffer; // copy sound buffer
          audio_len = wav_length; // copy file length

          /* Open the audio device */

                if ( SDL_OpenAudio(&wav_spec, NULL) < 0 ){
                fprintf(stderr, "Couldn't open audio: %s\n", SDL_GetError());
                exit(-1);
                }

                /* Start playing */
                SDL_PauseAudio(0);

                // wait until we're don't playing
                while ( audio_len > 0 ) {
                SDL_Delay(100);
                }

                // shut everything down
                SDL_CloseAudio();
                SDL_FreeWAV(wav_buffer);
```

}

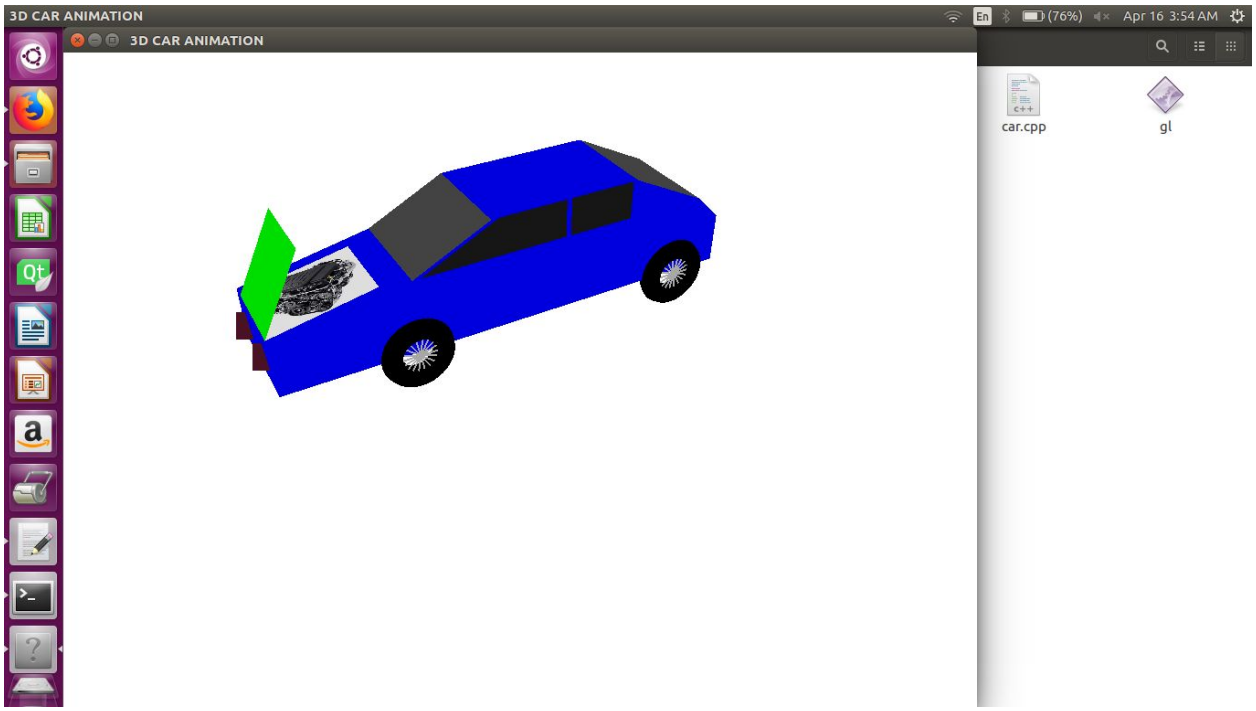
Texture:-

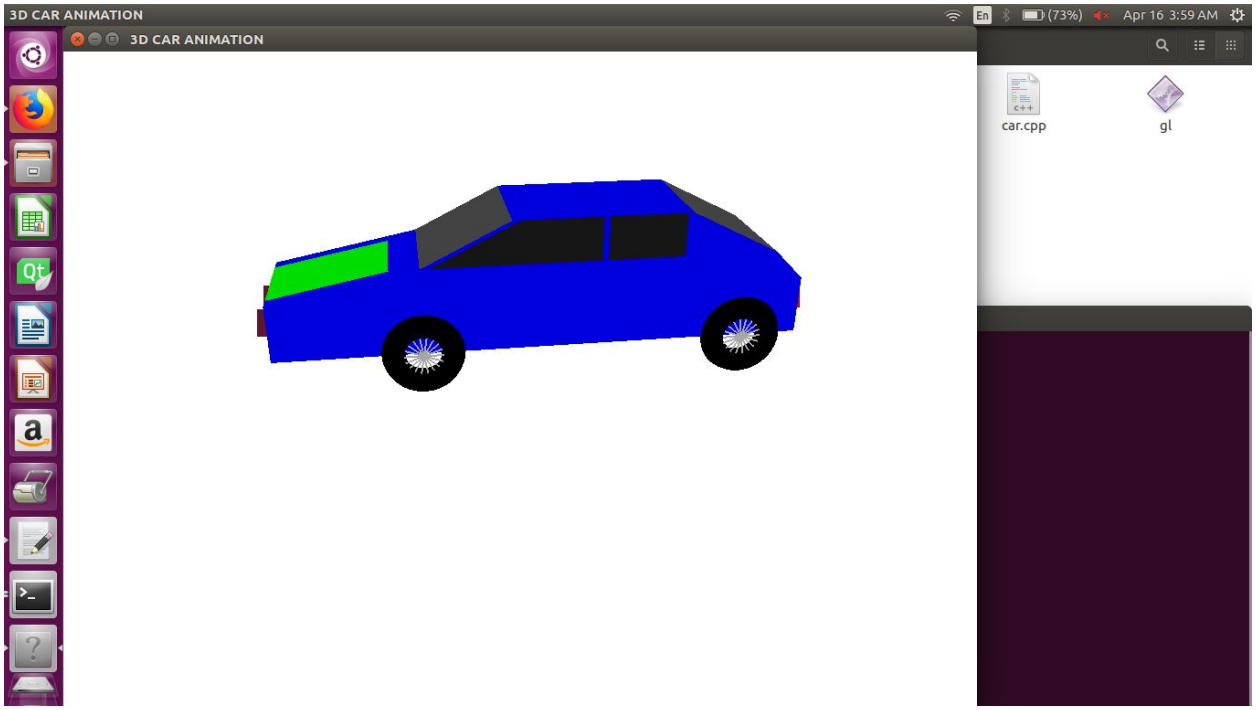
1. To load image give the path in grass = loadimage(“path”), which can be found in init_texture() function.
2. To Load texture call function draw_texture() and call init_texture() in main so that the photo gets uploaded in grass(int). Define grass as global variable.

To check the feature:- 1.Press “E” to check the feature.
2.Press “e” to back to original feature.

```
void draw_texture()                          // Texture
{
   glEnable(GL_TEXTURE_2D);
   glBindTexture(GL_TEXTURE_2D,grass);
   glBegin(GL_QUADS);
   glTexCoord2f(0.0f, 0.0f); glVertex3f(0.1, 0.4,0.55);
   glTexCoord2f(1.0f, 0.0f);   glVertex3f(0.5, 0.48,0.55);
   glTexCoord2f(1.0f, 1.0f);  glVertex3f(0.5, 0.48,0.25);
   glTexCoord2f(0.0f, 1.0f); glVertex3f(0.1,0.4,0.25);
   glEnd();
   glDisable(GL_TEXTURE_2D);
}

GLuint loadimage(const char *fileName)       // load image
{
   FILE *file;
   unsigned char header[54],*data;
   unsigned int dataPos,size,width, height;
   file = fopen(fileName, "rb");
   fread(header, 1, 54, file);
   dataPos       = *(int*)&(header[0x0A]);
   size          = *(int*)&(header[0x22]);
   width         = *(int*)&(header[0x12]);
   height        = *(int*)&(header[0x16]);
   if (size == NULL)
         size = width * height * 3;
   if (dataPos == NULL)
         dataPos = 54;
   data = new unsigned char[size];
   fread(data, 1, size, file);
```

```
    fclose(file);
    GLuint texture;
    glGenTextures(1, &texture);
    glBindTexture(GL_TEXTURE_2D, texture);

    glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
    glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);

    glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, width, height, 0, GL_BGR_EXT,
GL_UNSIGNED_BYTE, data);
    return texture;
}

void init_texture()
{
        grass = loadimage("h.bmp") ;
}
```





Installation Steps for opengl in ubuntu:-

1. sudo apt-get install freeglut3 freeglut3-dev
2. sudo apt-get install binutils-gold
3. g++ -lGL -lglut test.cpp -o test (test - case)

Installation Steps for SDL2 in ubuntu:-

It is used to produce sound when a key is pressed. Instructions to install library are given below:-

1.sudo apt-get install libsdl2-2.0.

2.save the medium.wav and texture (bmp file) file in the folder of code.

3.g++ 3dcar.cpp -I /usr/include/SDL2 -o gl -lGL -lGLU -lglut -lSDL (to run).