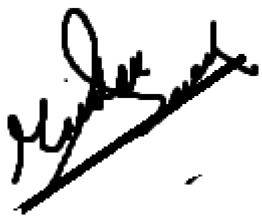
Declaration

I declare that this assignment submission represents my own work (except for allowed material provided in the course), and that ideas or extracts from other sources are properly acknowledged in the report. I have not allowed anyone to copy my work with the intention of passing it off as their own work.

Name : **M. Haider Saeed (msa280)**

Student ID: **68479094**

Date: **28/03/23**



Haider (msa280)

Project Report

Summary

The theme of the project is optical illusion. The scene consists of three separate AAOs which are placed in a gallery with walls. AAO1 and AAO2 represent optical illusion whereas AAO3 is a 3D model of a road with tunnel and traffic lights. Each of these AAO's can be viewed using the control binds and have some animations. The project was created in C++ using OpenGL. We will discuss more on each AAO.

AAO1: Ames Window

The Ames Window is a 2D model created by mapping out the vertices of the provided template and drawing quads. Credits to DESMOS for helping me map out the coordinates more accurately.

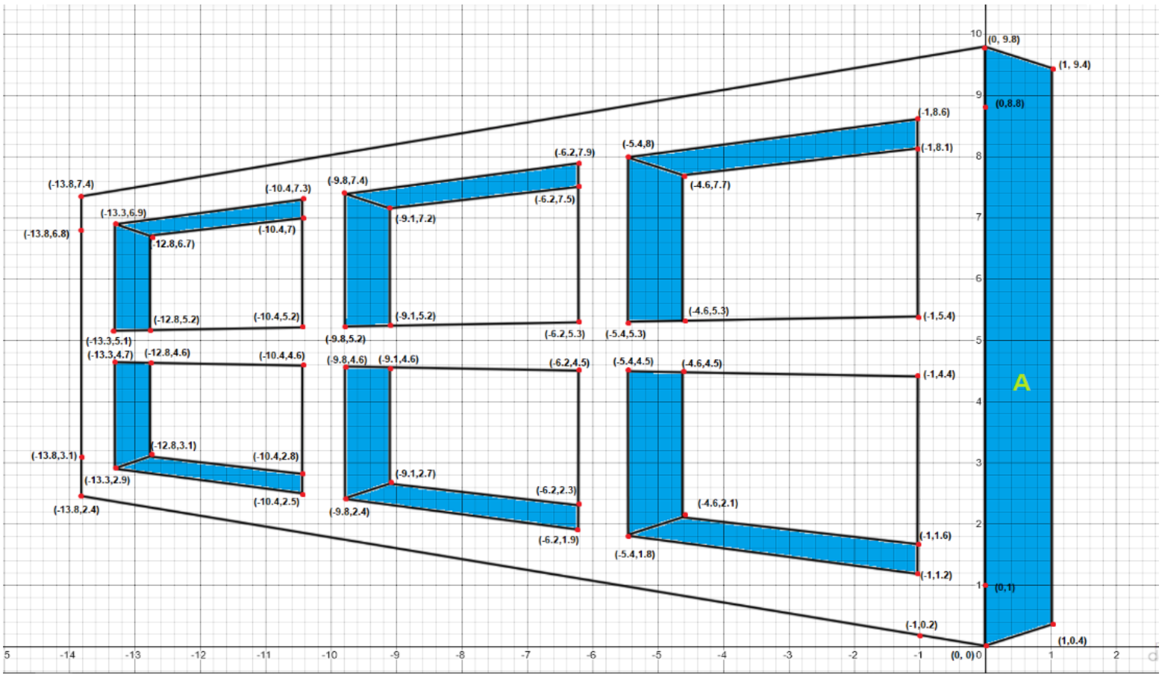


Figure 1: Ames Window Template (DESMOS)

To view the window, press "1". When viewed from a horizontal angle, the Ames Window represents an optical illusion where the window looks like it is swaying from side to side. The truth is that the window is spinning. The window is also under a spotlight source which is rotating at an angle of 45 degrees on each end. This causes the spotlight to move too, which in turn shines the yellow light onto AAO1.

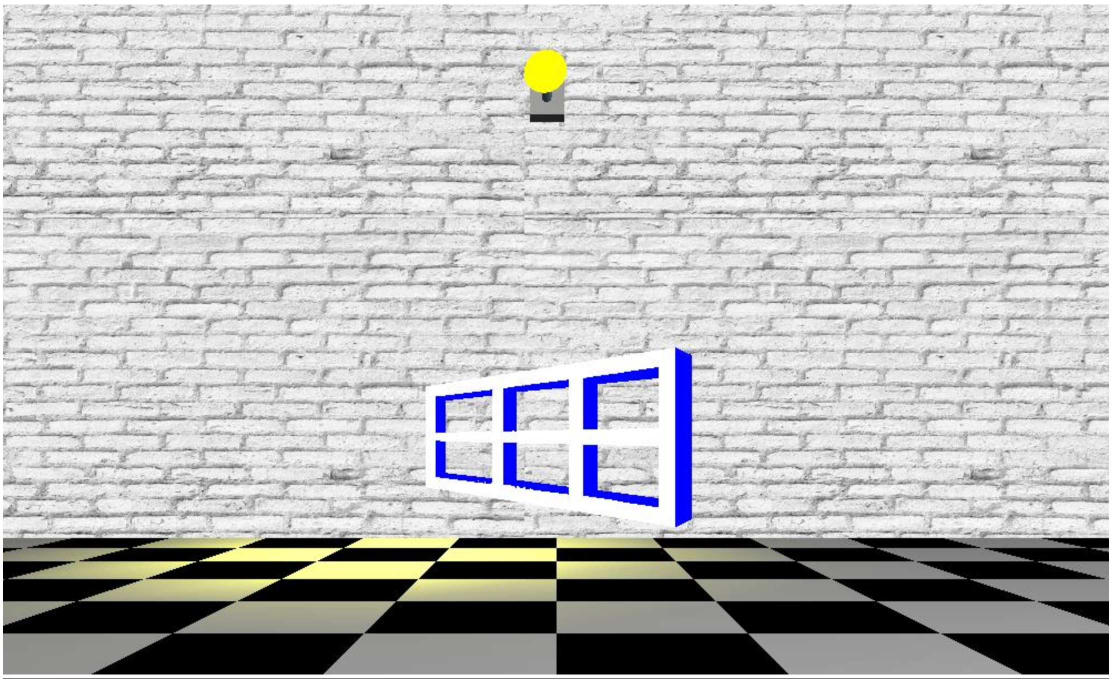


Figure 2: AAO1 - Ames Window

AAO2: Moiré Patterns

This AAO was designed by first creating a circle. The circles were then replicated and multiplied depending on the radius to form a certain pattern. Two of these patterns were then made to overlap each other. This resulted in beautiful moire patterns being seen when the circular figures overlap.

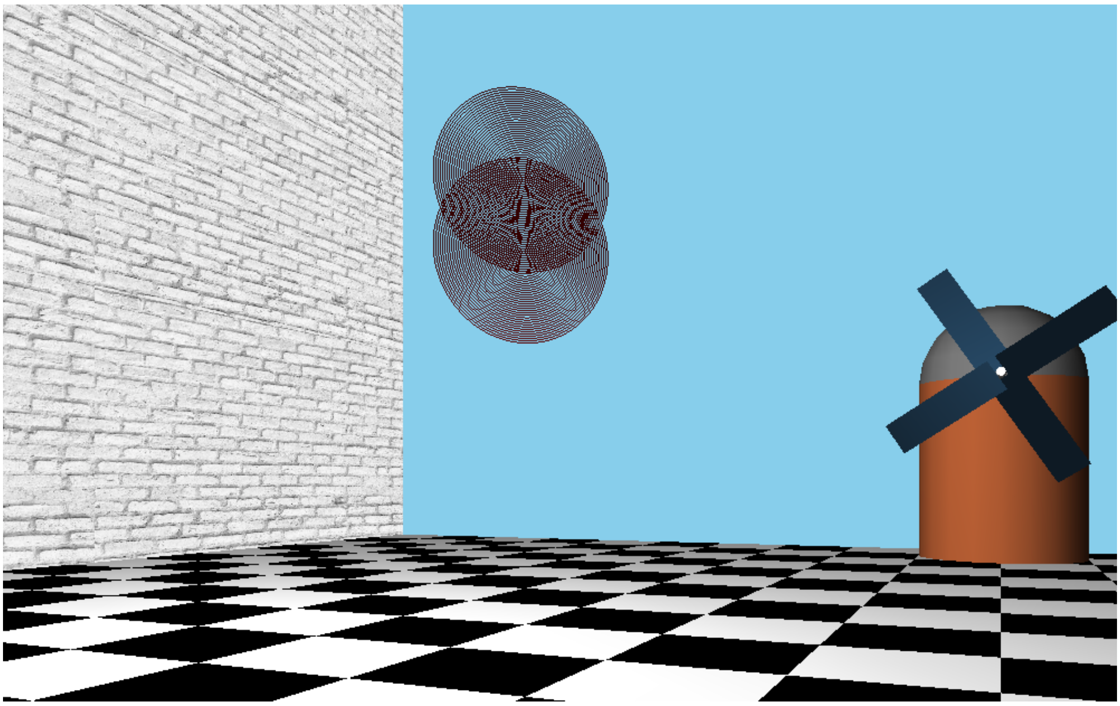


Figure 3: AAO2 - Moire Patterns

<u>AAO3: 3D Road Scene</u>

This scene consists of a car and traffic lights made using basic shapes and primitives. The user can control the traffic lights using the controls provided. The lights can have three states: red, yellow, and green. The car is placed on a road. This road is a textured quad strip. There are also two static illusions displayed on the side of the wall. On both ends of the road is a dark tunnel from which the car enters and leaves. When the light turns green, the car starts accelerating slowly until it reaches its maximum speed. On yellow or red lights, the car starts slowing down depending on its speed. If the car was going very fast, it is going to take a while for the car to stop. However, if the car is going slowly, it is able to stop relatively quickly. This was a physics based approach to the scene. Model for the car, traffic lights and windmill were drawn in MS Paint.

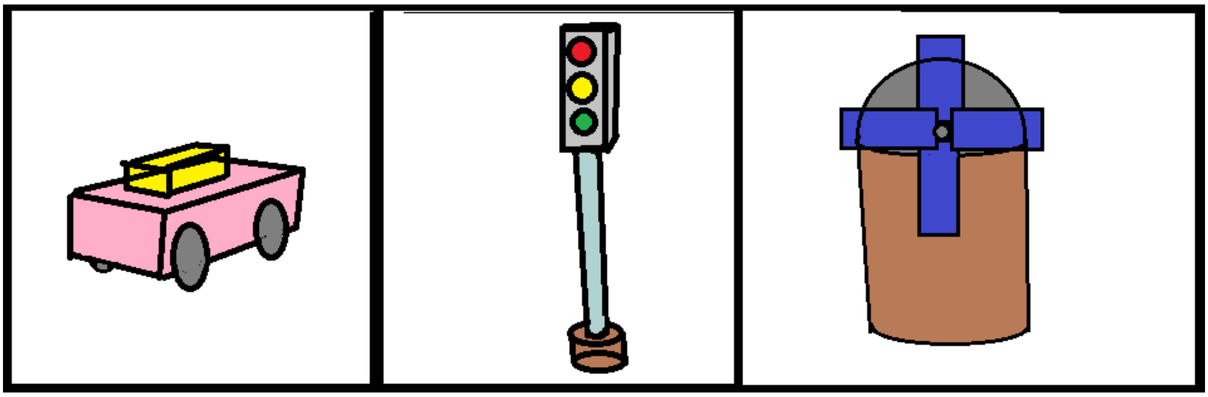


Figure 4: Car and Traffic Light Template

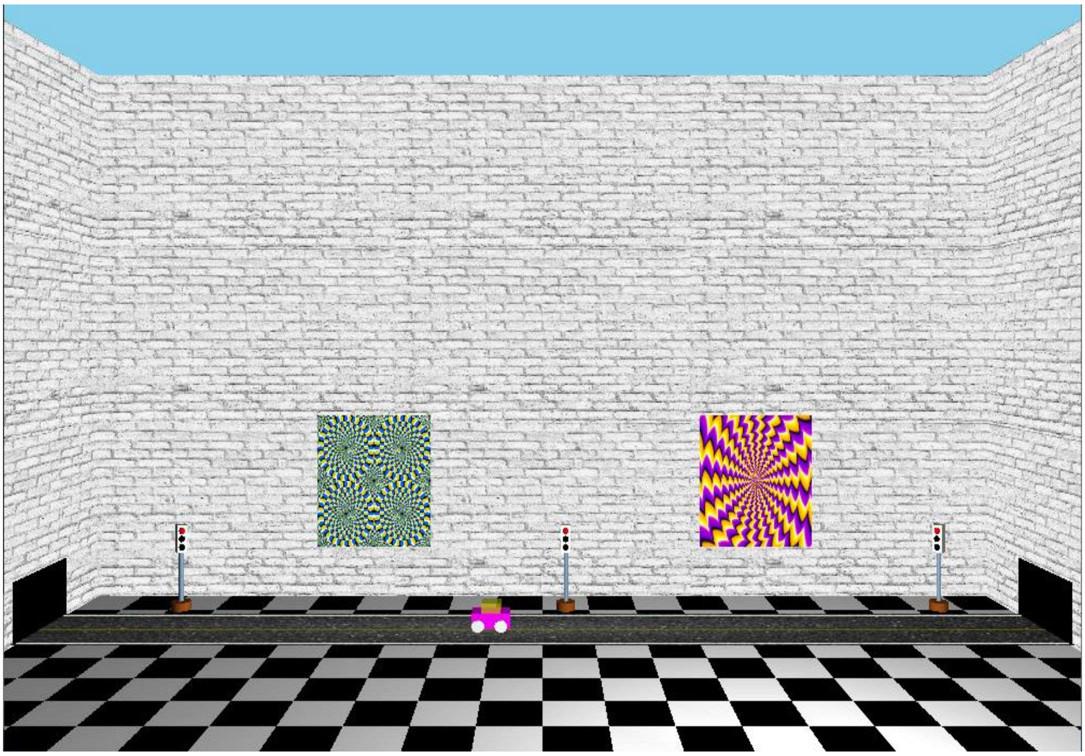


Figure 5: AAO3 - 3D Road Scene

The following code was used to predict the way the car behaves:

```
//-- This is the timer for car behavior which is dependent on
//-- traffic light and car speed.
void aao3_timer(int value)
{
    if (car_pos_z < 200 and green == 1) {
        car_pos_z += 0.5 * speed_rate;
        if (speed_rate < 15) {            // Max car speed
            speed_rate += 0.1;            // Acceleration rate increasing by 0.1 if car has not reached the max acceleration.
        }
    }
    else if (red == 1 or yellow == 1 and car_pos_z < 200) {
        if (speed_rate > 0) {
            car_pos_z += 0.5 * speed_rate;
            if (car_pos_z > 200) {
                car_pos_z = -200;
            }
            speed_rate -= 0.25;           // Car decelerating on red and yellow light. Deceleration is faster than acceleration.
        }
        else if (car_pos_z > 200) {
            car_pos_z = -200;
        }
    }
    else {
        car_pos_z = -200;
    }

    glutTimerFunc(50, aao3_timer, value);
    glutPostRedisplay();
}
```

Figure 6: Car Movement (Physics Code)

Gallery View

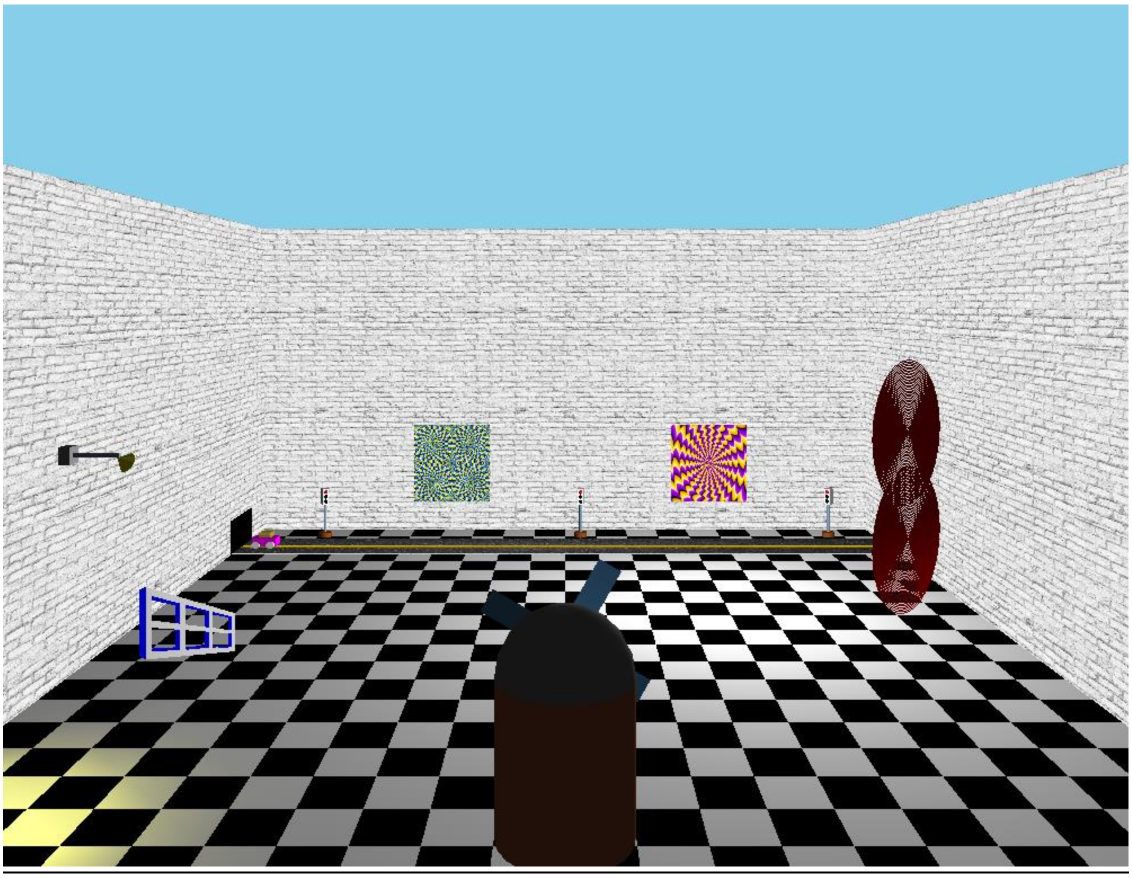


Figure 7: Gallery View

Haider (msa280)

Equations for designing animations:

```
//-- AAO1 Timer ----------------------------
void aao1_timer(int value)
{
    amesWinAngle += 5;
    if (amesWinAngle > 360)
    {
        amesWinAngle = 0;
    }
    glutTimerFunc(50, aao1_timer, value);
    glutPostRedisplay();
}


//-- AAO2 Timer 1 ---------------------------
void aao2_timer(int value)
{
    aao2_height += 0.1 * aao2_height_dir;
    if (aao2_height <= 19) {
        aao2_height_dir = 1;            // Going Up
    }
    else if (aao2_height >= 28) {
        aao2_height_dir = -1;           // Going Down
    }
    glutTimerFunc(50, aao2_timer, value);
    glutPostRedisplay();
}


//-- AAO2 Timer 2 ---------------------------
void aao2_timer_2(int value)
{
    aao2_height_2 += 0.1 * aao2_height_dir_2;
    if (aao2_height_2 >= 35) {
        aao2_height_dir_2 = -1;            // Going Down
    }
    else if (aao2_height_2 <= 26) {
        aao2_height_dir_2 = 1;             // Going Up
    }
    glutTimerFunc(50, aao2_timer_2, value);
    glutPostRedisplay();
}


//-- This is the timer for car behavior which is dependent on
//-- traffic light and car speed.
void aao3_timer(int value)
{
    if (car_pos_z < 200 and green == 1) {
        car_pos_z += 0.5 * speed_rate;
        if (speed_rate < 15) {            // Max car speed
            speed_rate += 0.1;            // Acceleration rate increasing by 0.1 if car has not reached the max acceleration.
        }
    }
    else if ((red == 1 or yellow == 1) and car_pos_z < 200) {
        if (speed_rate > 0) {
            car_pos_z += 0.5 * speed_rate;
            if (car_pos_z > 200) {
                car_pos_z = -200;
            }
            speed_rate -= 0.25;           // Car decelerating on red and yellow light. Deceleration is faster than acceleration.
        }
        else if (car_pos_z > 200) {
            car_pos_z = -200;
        }
    }
    else {
        car_pos_z = -200;
    }

    glutTimerFunc(50, aao3_timer, value);
    glutPostRedisplay();
}
```

```
//-- Windmill Timer ----------------------------
void windmill_timer(int value)
{
    fan_rot += 3;
    if (fan_rot > 360)
    {
        fan_rot = 0;
    }
    glutTimerFunc(50, windmill_timer, value);
    glutPostRedisplay();
}


//-- This is the timer for the spotlight and its source to change angle. --------
void spotlight_timer(int value)
{
    sl_dir += 0.5 * light_rate;
    sl_source_angle += 2 * light_rate;
    if (sl_dir <= -5 and sl_source_angle <= -45) {
        light_rate = 1;
    }
    else if (sl_dir >= 5 and sl_source_angle >= 45)
    {

        light_rate = -1;
    }
    glutTimerFunc(50, spotlight_timer, value);
    glutPostRedisplay();
}
```

Note: All images, files, libraries, documents and the final cpp file are located in the zip folder. They are labelled for clarity.

Extra Features

- A spotlight on a moving object. The movement of spotlight is visible.
- Physics based animation shown in AAO3. Relevant equation screenshot provided for car movements.
- A texture mapped quad strip (road) which isn't part of a sweep surface.
- Two texture mapped quads showing static optical illusions.
- Texture mapped brick walls around the room.
- Extra controls like moving up and down as well as control of the traffic lights.
- Added a windmill with rotating blades.
- Well documented and readable code.

Controls:

```
↑ Arrow - Move Forwards
↓ Arrow - Move Backwards
← Arrow - Turn Left
→ Arrow - Turn Right
1 - AAO1 View (Ames Window)
2 - AAO2 View (Moire Patterns)
3 - AAO3 View (3D Road Scene)
0 - Gallery View
F1 - Move Up
F2 - Move Down
F5 - Turn Red Traffic Signal On
F6 - Turn Yellow Traffic Signal On
F7 - Turn Green Traffic Signal On
```

Instruction to run project:

1. Download the project zip file and drag it to your desktop.
2. Extract items from the zip. A folder called “msa280” would appear.
3. Go to terminal and change directory by using: **cd Desktop/msa280**
4. Now compile the file using following: **g++ Assignment1.cpp -lGL -lGLU -lglut**
5. A folder called a.out would appear. Run the program using: **./a.out**

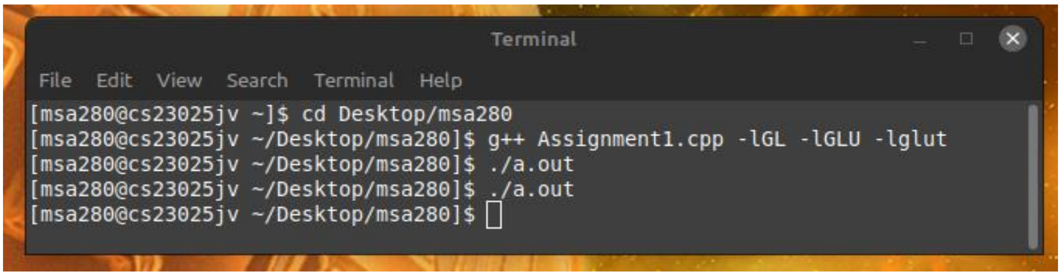


Figure 8: Linux Terminal Instructions

References:

Road Texture: https://www.istockphoto.com/photo/background-texture-of-rough-asphalt-gm463476651-32700288

Static Illusion 1: https://myvision.org/education/optical-illusions/

Static Illusion 2: https://www.talkcocksingsong.net/tricks-your-eyes-with-these-optical-illusions/

Brick Wall: https://unsplash.com/s/photos/white-brick-wall

Windmill Brick: https://architextures.org/textures/472

Colours: https://web.archive.org/web/20180301041827/https://prideout.net/archive/colors.php