ECSE−4750 Computer Graphics HW4

Ying Lu

September 24, 2013

1. *(10 pts)* (This is a test of complex a scene your machine can handle.)

   (a) Write a simple OpenGL program to read an integer N, and then draw N random triangles, each of width 1/10 of the width of the window.

   (b) Run the program repeatedly with N=10, 30, 100, 300, ... until something crashes, or it gets so slow that your patience runs out.

   (c) Report your machine HW and SW, what happened, and when it happened.

   Solution :

   (a) The source code is attached at the end of this report. With the random number generator functions attached. Besides, the width of the triangles are 1/10 of the width of the window.

   (b) Done, some of the screen copies are attached. I tested this using the Desktop at my lab, which is a much more powerful machine than my own laptop. So when I increased N to 10^7, it takes about 5.4 seconds to display, (which is measured using my stop watch). When I increased N to 10^8, it takes about 46.7 seconds to display the triangles, which is pretty slow, I think.

   (c) HW:

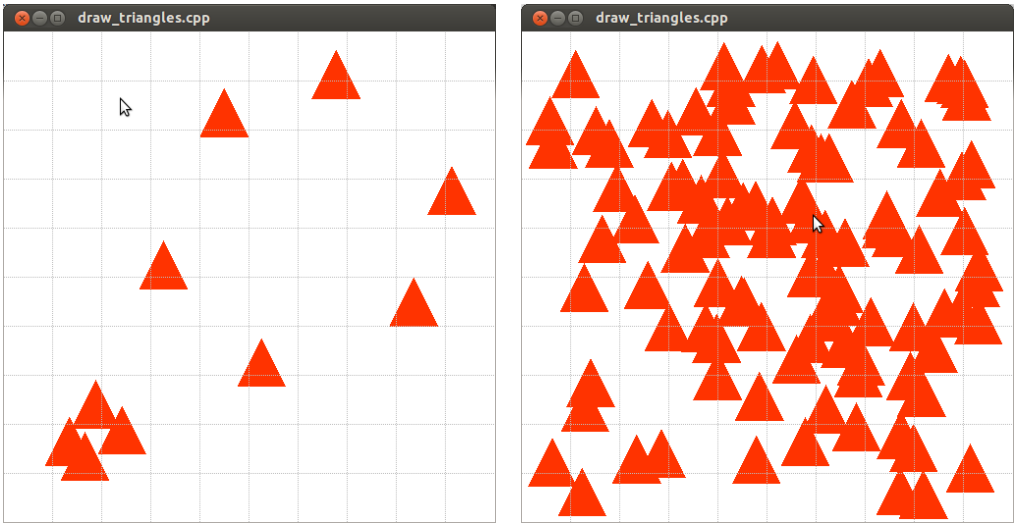
```
$lspci -v | grep VGA

03:00.0 VGA compatible controller: NVIDIA Corporation GF100 [GeForce GTX
       Memory: 16G
```

```
       Disk:   1T + 250G + 256G SSD
```

SW:

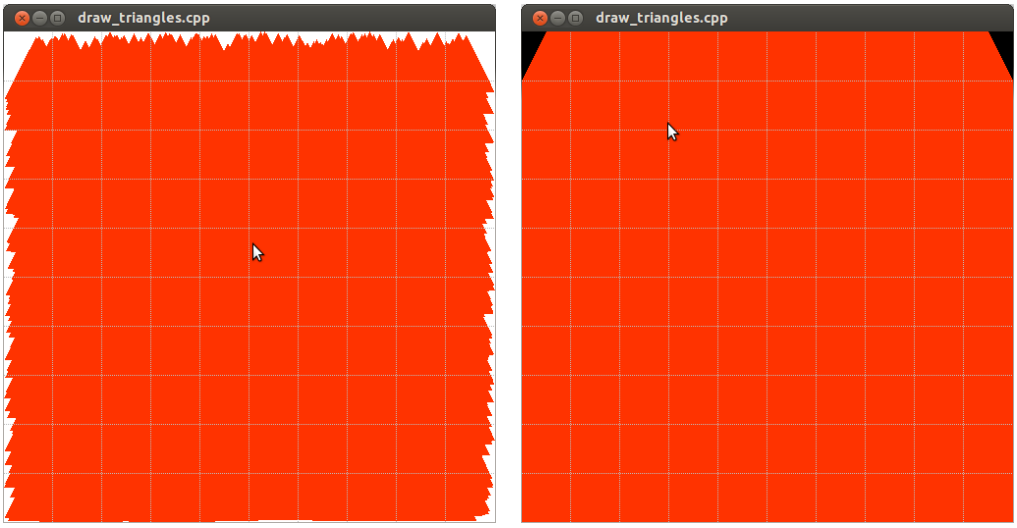
```
       name of display: :0
       display: :0  screen: 0
       direct rendering: Yes
       server glx vendor string: NVIDIA Corporation
       server glx version string: 1.4

OpenGL vendor string: NVIDIA Corporation
OpenGL renderer string: GeForce GTX 480/PCIe/SSE2
OpenGL version string: 4.2.0 NVIDIA 304.88
OpenGL shading language version string: 4.20 NVIDIA via Cg compiler
OpenGL extensions:
    GL_AMD_multi_draw_indirect, GL_ARB_base_instance,
    GL_ARB_blend_func_extended, GL_ARB_color_buffer_float,
    GL_ARB_compatibility, GL_ARB_compressed_texture_pixel_storage,
    GL_ARB_conservative_depth, GL_ARB_copy_buffer, GL_ARB_depth_buffer_fl
    GL_ARB_depth_clamp, GL_ARB_depth_texture, GL_ARB_draw_buffers,
```

The following are several snapshots with different Ns.



(a) N = 10 (b) N = 100

Figure 1: N triangles



(a) N = 5000 (b) N = 10^8

Figure 2: N triangles

2. Write the 4x4 matrix for a 3D translation by (2,3,4).

   Solution:

$$\begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3. Apply it to the 3D point (1, 0, 1).
   Solution:

$$\boxed{(3, 3, 5)}$$

4. Write the 4×4 matrix for a multiplication about the Y axis by 90 degrees.
   Solution:
   First of all, we could find the 3×3 matrix $\mathbf{M}$ on the left upper part of the 4×4 matrix, using the definition of:

$$\mathbf{M} = \begin{bmatrix} \cos\theta + (1-\cos\theta)a_1^2 & (1-\cos\theta)a_1a_2 - \sin\theta a_3 & (1-\cos\theta)a_1a_3 + \sin\theta a_2 \\ (1-\cos\theta)a_1a_2 + \sin\theta a_3 & \cos\theta + (1-\cos\theta)a_2^2 & (1-\cos\theta)a_2a_3 + \sin\theta a_1 \\ (1-\cos\theta)a_1a_3 - \sin\theta a_2 & (1-\cos\theta)a_2a_3 + \sin\theta a_1 & \cos\theta + (1-\cos\theta)a_3^2 \end{bmatrix}$$

   Here we are going to rotate about the Y axis by 10 degrees, then we have:

$$\begin{aligned} a_1 &= 0 \\ a_2 &= 1 \\ a_3 &= 0 \\ \theta &= 90^\circ \end{aligned}$$

   Substitute into the matrix, we would get:

$$\mathbf{M} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \tag{1}$$

Then the 4×4 matrix for a multiplication about the Y axis by 90 degrees is:

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

5. Apply it to the answer from question 3.
Solution:

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 3 \\ 5 \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 3 \\ -3 \\ 1 \end{bmatrix}$$

$$\boxed{(5, 3, -3, 1)} \tag{2}$$

6. Multiply the matrices that are the answers to questions 2 and 4, with 2 being on the right.
Solution:

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 4 \\ 0 & 1 & 0 & 3 \\ -1 & 0 & 0 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

7. Apply that matrix to the point $(1, 0, 1)$. The answer should be the same as question 5.
Solution:

$$\begin{bmatrix} 0 & 0 & 1 & 4 \\ 0 & 1 & 0 & 3 \\ -1 & 0 & 0 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 3 \\ -3 \\ 1 \end{bmatrix} \tag{4}$$

The answer is the same as question 5.

8. Why is it useful that translation be a matrix multiplication?
   Solution:
   If applying several translations, it is faster to first multiply the matrices, then just multiply all the points by that one matrix.

9. Suppose that you specify two colors before drawing a point thus:

```
glColor3f(1.,0.,1.);
glColor3f(0.,1.,0.);
glVertex3f(10.,20.,30.);
```

   What color is the point? E.g, do the magenta and green mix to make the point white?
   Solution:
   The most recent color will work, which is green here. So the point is green

10. Suppose that you draw a second point w/o specifying the color again. What color is this point? E.g. does it revert to black?
    Solution:
    The color of this point is still the most recent defined one, which is still green here, since we draw a second point w/o specifying the color again.

11. (Reverse engineering rotations) In 2D, if the point $(1, 2)$ rotates about the origin to $(2, -1)$, what's the angle?
    Solution:
    It is 270 degrees (or -90 degrees) if we take the counter-clock direction as positive. We could easily verify this, since $\theta = 270°(-90°)$, we have $\cos\theta = 0, \sin\theta = -1$, then:

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

    Then we will have:

$$\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ -1 \end{bmatrix} \tag{5}$$