CSC4140 Assignment 2

Computer Graphics

February 25, 2022

Transformation

This assignment is 10% of the total mark.

Strict Due Date: 11:59PM, Feb 25^{th}, 2022

Student ID: 119010265

Student Name: Shi Wenlan

This assignment represents my own work in accordance with University regulations.

Signature: Shi Wenlan

1 The Idea of Implementation

1.1 Implement Model Matrix (40 points)

Firstly, build the Translation Matrix M_trans and the Scale Matrix S_trans using following formulas.

➢ Scale

$$\mathbf{S}(s_x, s_y, s_z) = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

➢ Translation

$$\mathbf{T}(t_x, t_y, t_z) = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Figure 1: Formulas of Translation Matrix and Scale Matrix

Secondly, implement Rodrigues' Rotation Formular using following formula. Note that **u** is normalized (**P1 - P0**), which is the axis of rotation. It is validated that the calculated results are in agreement with those obtained by $AngleAxisf$ (I print them in the terminal).

$$\mathbf{R}(\mathbf{u}, \theta) = \cos\theta\, \mathbf{I} + (1 - \cos\theta)\mathbf{u}\mathbf{u}^{\mathrm{T}} + \sin\theta \begin{bmatrix} 0 & -n_z & n_y \\ n_z & 0 & n_x \\ -n_y & -n_x & 0 \end{bmatrix}$$

Figure 2: Rodrigues' Rotation Formular

Thirdly, the output matrix $model = S\_trans * rotation * M\_trans$.

1.2 Implement perspective projection Matrix (40 Points)

Firstly, convert prospective projection to orthogonal projection using following formula. (n = zNear, f = zFar)

$$\begin{bmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n+f & -fn \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Figure 3: Formula of Prospective to Orthogonal

Secondly, scale to a 2 * 2 * 2 cube, and transfer the center to the origin as following formulas. (t = top, b = bottom, l = left, r = right)

$$M_{ortho} = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & 0 \\ 0 & \frac{2}{t-b} & 0 & 0 \\ 0 & 0 & \frac{2}{n-f} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -\frac{r+l}{2} \\ 0 & 1 & 0 & -\frac{t+b}{2} \\ 0 & 0 & 1 & -\frac{n+f}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figure 4: Formulas of Scale and Translate

Thirdly, output matrix $projection = ortho * persp2ortho$.

1.3 Implement main() function (20 Points)

I use $fstream$ to read parameters from $parameters.txt$.

Note 1: eye_fov is the field of view (unit: degree), and $aspect\_ratio$ is width/hight ratio.

Note 2: Parameters are limited to one triangle.

Attention: do not add Spaces when modifying $parameters.txt$.

2 The Crops

My custom set of parameters is as following:

```
eye_pos={0,0,10}
pos1={2,0,-1}
pos2={0,2,-1}
pos3={-2,0,-1}
ind={0,1,2}
eye_fov={45}
aspect_ratio={1}
zNear={-0.1}
zFar={-50}
T={0,0,0}
S={1,1,1}
P0={0,0,0}
P1={1,2,0}
```

Figure 5: Parameters

The crops of the corresponding results on the screen are as following (I repeatedly pressed the D key to rotate the triangle and cut some crops. Watching these crops continuously would be more like a frame-by-frame animation):

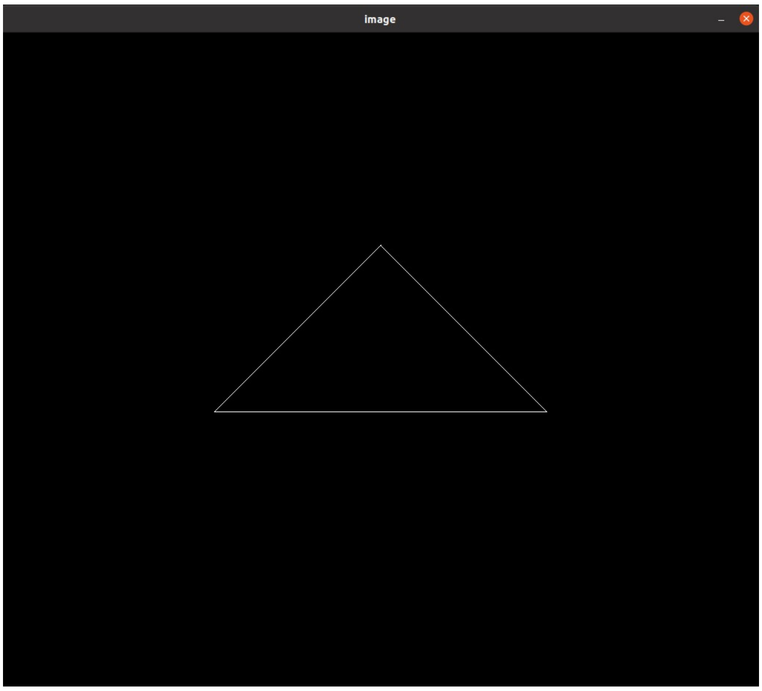

Figure 6: Crop 1

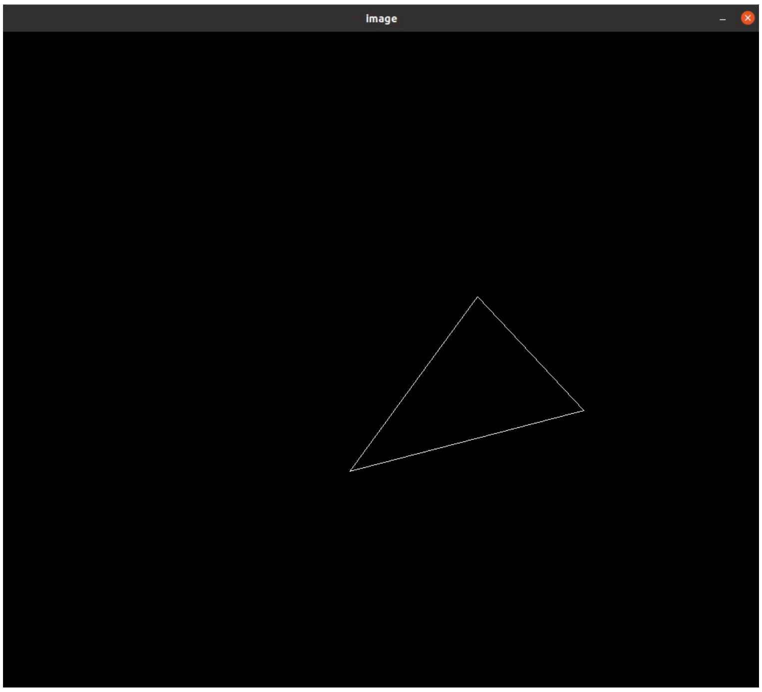

Figure 7: Crop 2

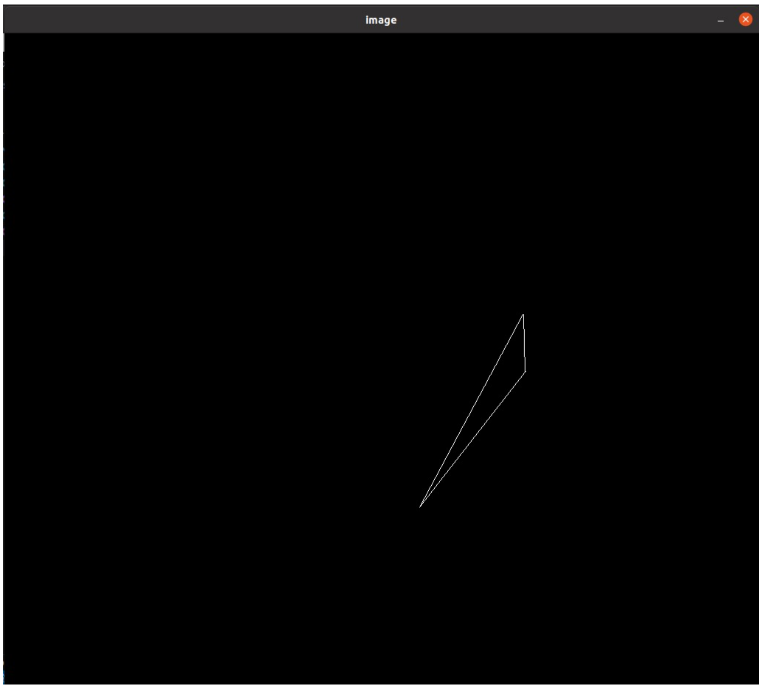

Figure 8: Crop 3

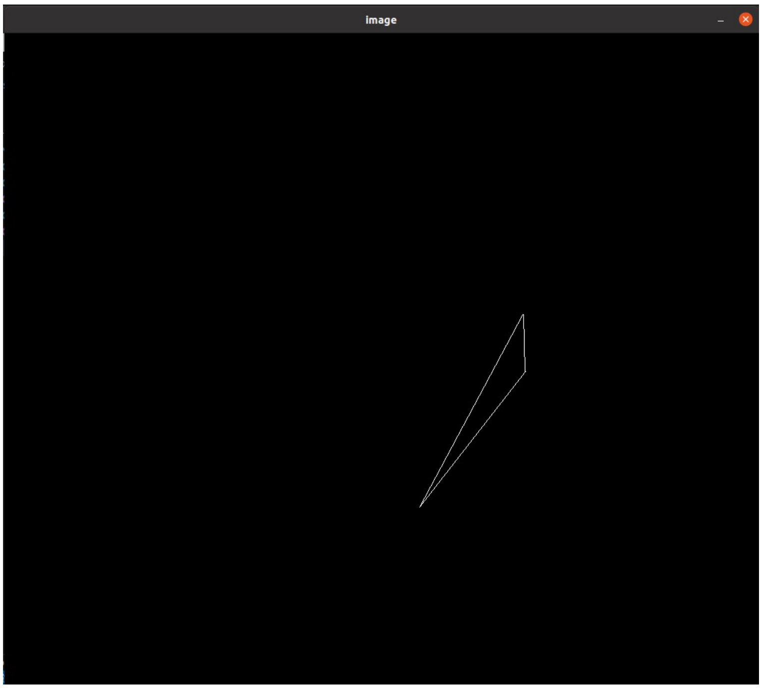

Figure 9: Crop 3

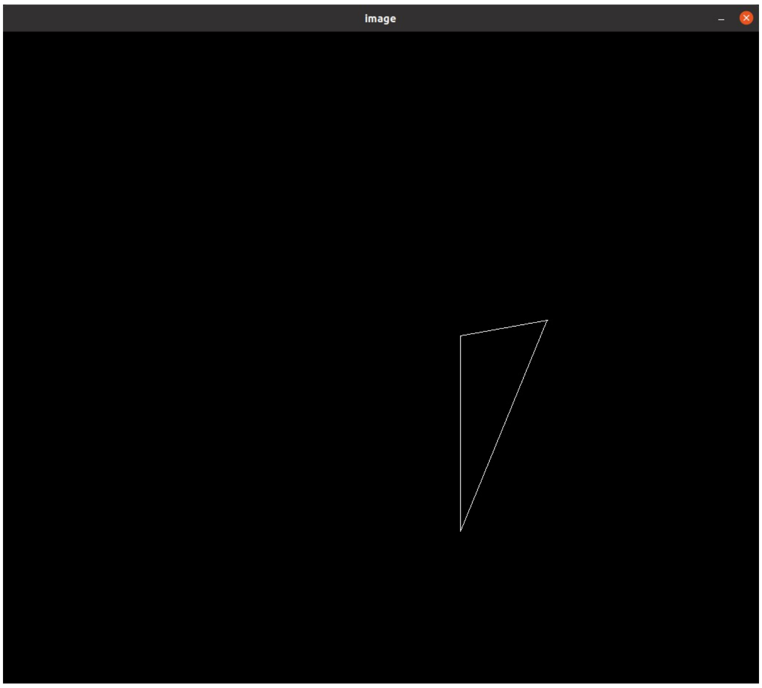

Figure 10: Crop 4

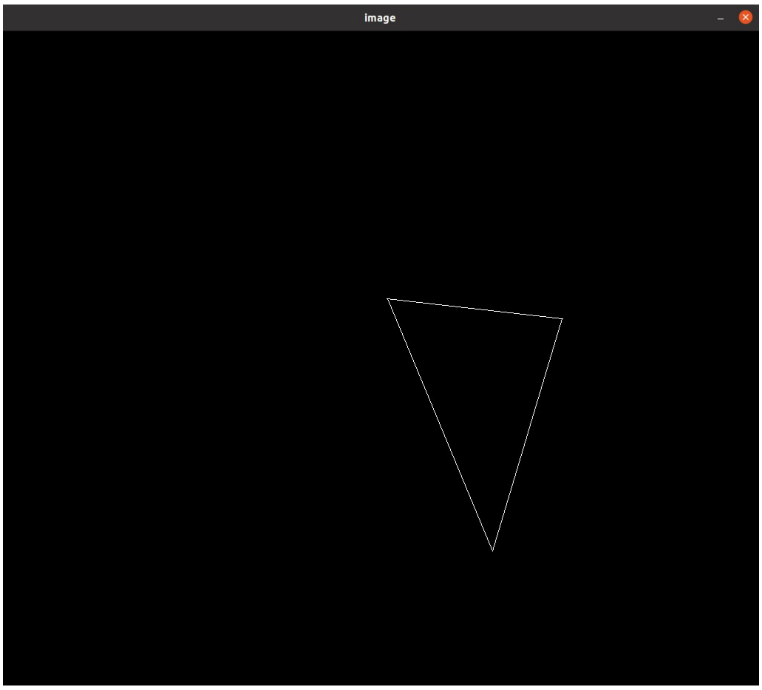

Figure 11: Crop 5

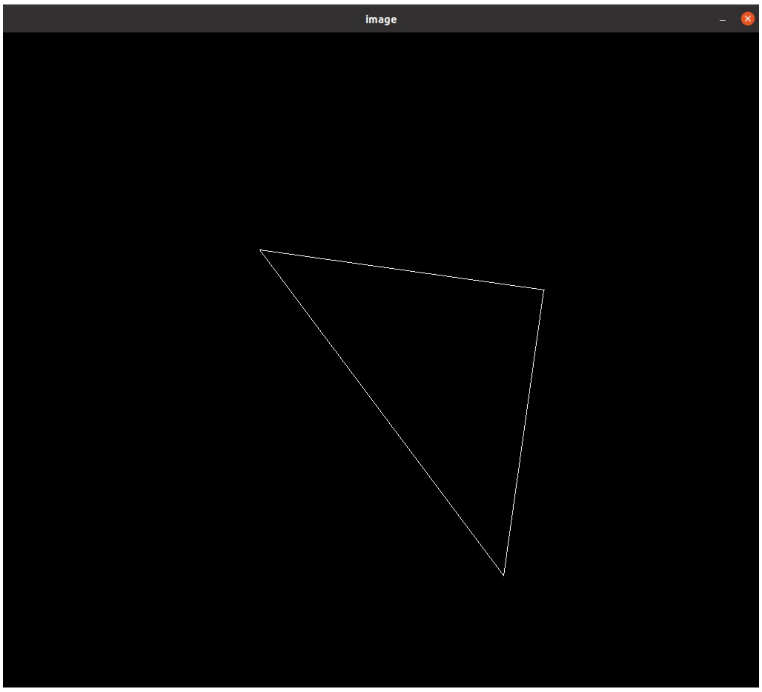

Figure 12: Crop 6

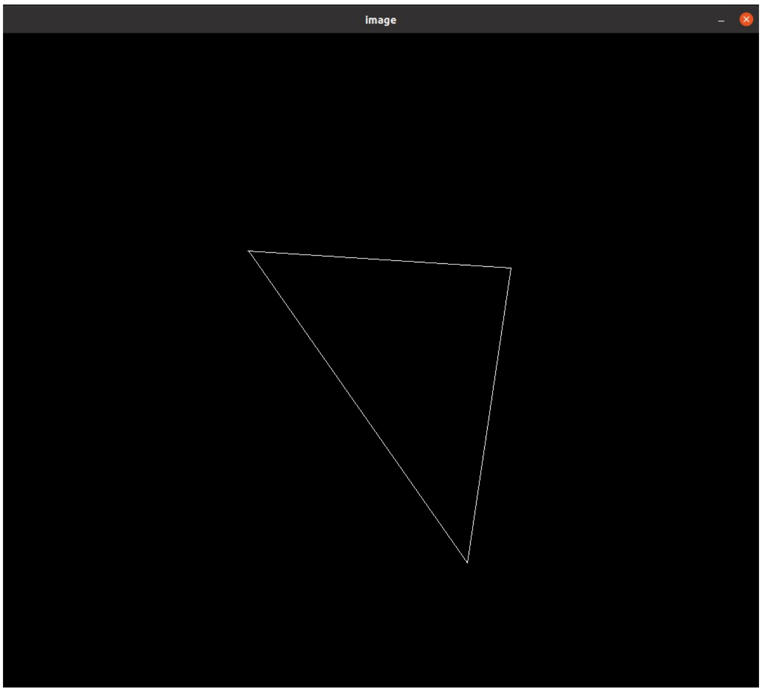

Figure 13: Crop 7

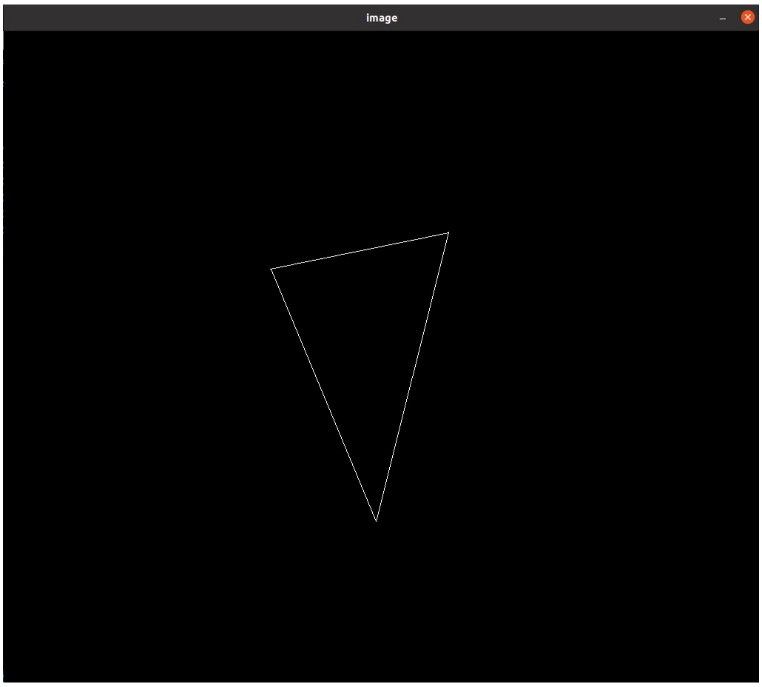

Figure 14: Crop 8

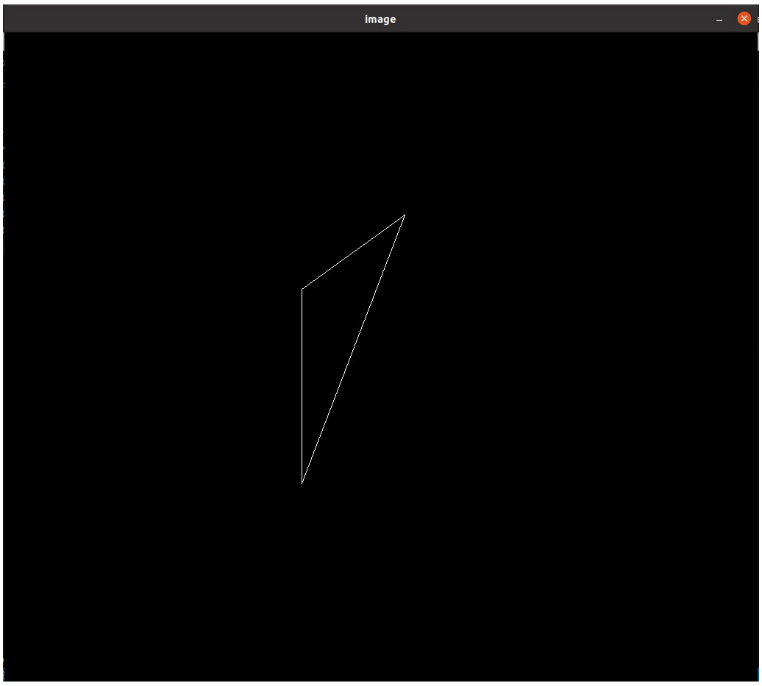

Figure 15: Crop 9

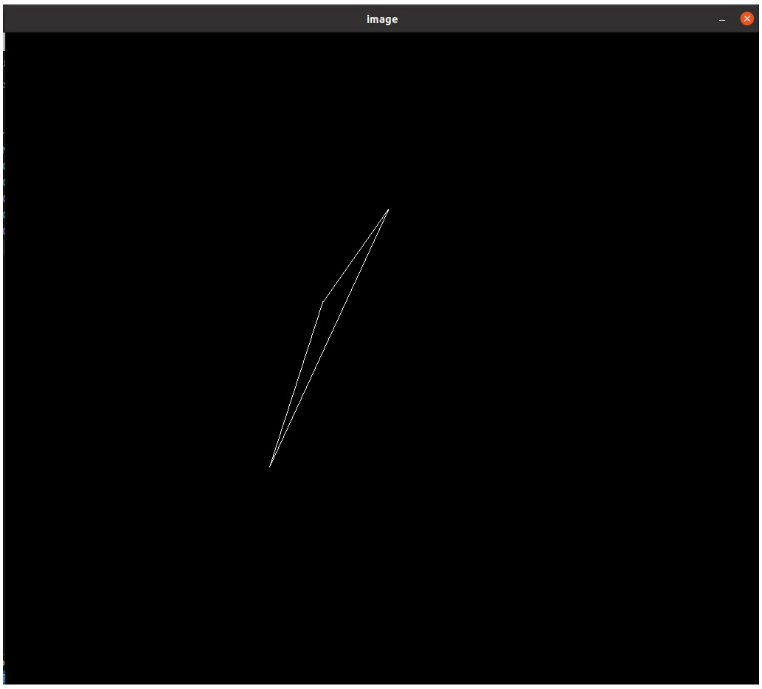

Figure 16: Crop 10

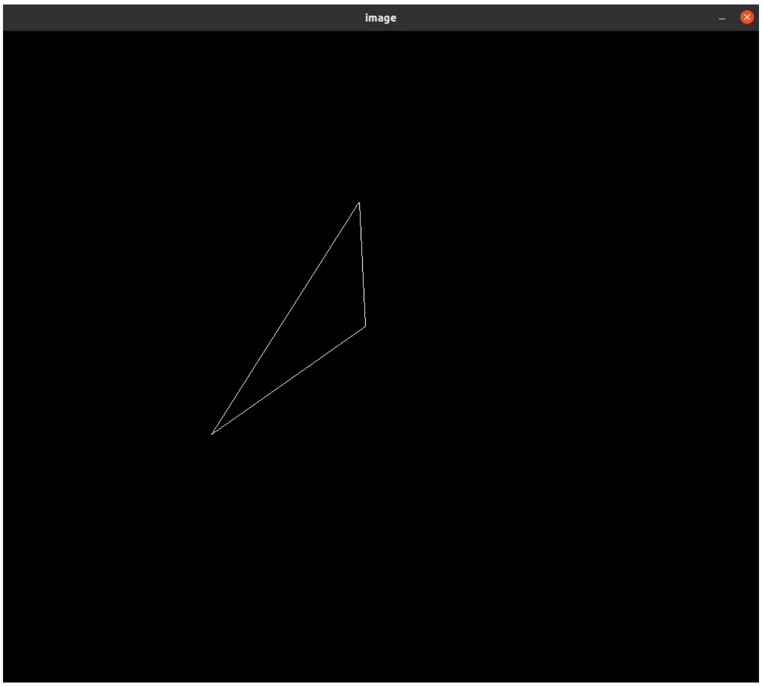

Figure 17: Crop 11

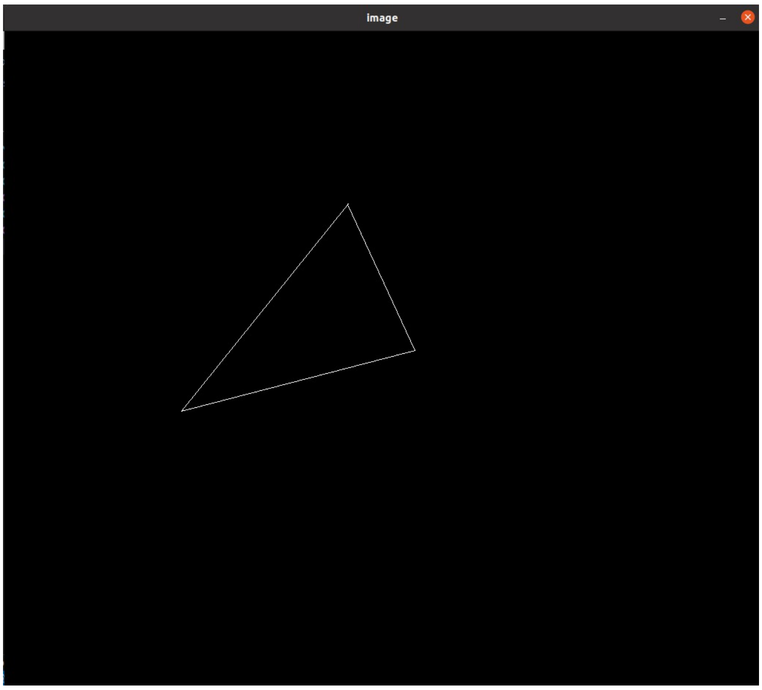

Figure 18: Crop 12

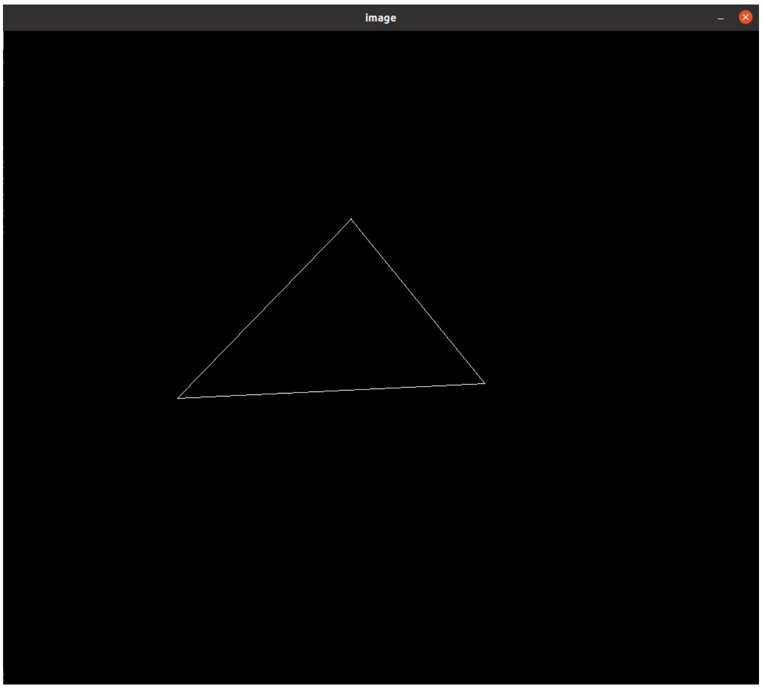

Figure 19: Crop 13

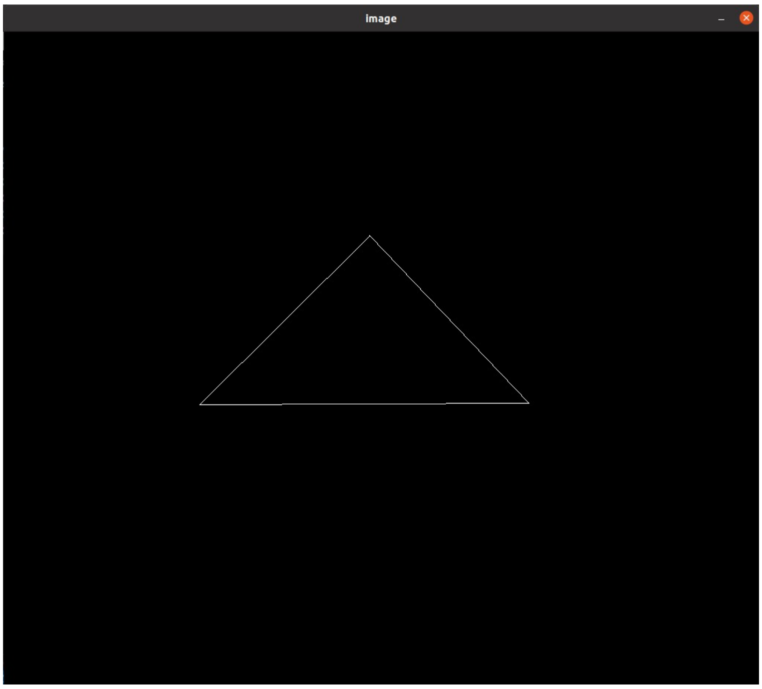

Figure 20: Crop 14