ECS 175 Project 2

“A 3D TRANSFORMATION AND PROJECTION SYSTEM”

Author

Name: Tianyang Zhan

SID: 913739374

Environment

- Operating System:Ubuntu 12.04 amd64
- IDE:Using Code::Blocks 10.05 for coding, Make for compiling and running
- G++ Version:Ubuntu/Linaro 4.6.3-1ubuntu5

Prerequisite

- OpenGL
- GLUT

Commands

1.Compile the program using Makefile

```
$   make
```

2. Run the program

```
$   ./graphics
```

3. Clean up the folder AFTER program ends

```
$   make clean
```

Running the program

1. Setup config files (not required) (Refer to *Config Files*)

2. Run the program (Refer to *Commands*)

```
$ make
$ ./graphics
```

3. Use Command Line to set pre-launch settings (Refer to *User Menu*)

4. Use Keyboard to operate on objects (Refer to *Keyboard Interactions*)

5. Right Click On Window for Menu to set parameters (Refer to *User Menu*)

6. Exit the program, outputs save to files automatically

7. Clean up the folder (Refer to *Commands*)

```
$ make clean
```

Config Files

1.Data file used to display shape (default file: data) (World Coordinates)

```
1                   number of polygons
                        definition of 1st polygon:
4                           number of points of 1st polygon
0.0 0.0 0.0                 coordinates of 1st point
1.0 0.0 0.0                 coordinates of 2nd point
0.0 1.0 0.0                 coordinates of 3rd point
0.0 0.0 1.0                 coordinates of 4th point
6                       number of edges of 1st object
1 2                         edge from point 1 to point 2
1 3                         edge from point 1 to point 3
1 4                         edge from point 1 to point 4
2 3                         edge from point 2 to point 3
2 4                         edge from point 2 to point 4
3 4                         edge from point 3 to point 4
```

2. Setting file used to config the window (default file: setting)

300 300	Viewport width and height	(0~) (0~)
0 0 0	Viewport background color R,G,B	(0~255) (0~255) (0~255)
255 255 255	Line and shape color R,G,B	(0~255) (0~255) (0~255)
2	Oblique Projection Type	(2 / 4)
660 660	Window width and height	(0~) (0~)
1 1 1	Window background color R,G,B	(0~1) (0~1) (0~1)

Input/Output Files

Inputs (1 OR 2 Files) : data file (REQUIRED), setting file

- User is able to specify the name of the **data file** when prompted

Outputs (2 Files) : data file, setting file

- Program automatically save all changes **(World Coordinates)** to output files at exit

User Menu

1. Pre-launch Settings (Command Line)

- **Specify input file name:**

  Do you want to specify the input data file? (y/n) **y**

  Changing Input File to (ex: data.txt) **data1.txt**

- **Specify display window size:**

  Do you want to specify the window size? (y/n) **y**

  Setting Window Size (ex: 660 660): **800 800**

2. Runtime Menu (Right Click On the Graphic Window)

** For *(Input in Terminal)* items, check command line prompt for custom input **

Menu Entries:

- **Current Object:** Select object by ID to do operation
  …
- **Transformation Parameters:** Set Transformation parameters
  Scaling factor (Input in Terminal)
  Translation vector (Input in Terminal)
  Rotation axis (Input in Terminal)
  Rotation angle (Input in Terminal)
  Set All to Default
- **Oblique Projection Type:** Set the type of oblique projection
  Cavalier Projection
  Cabinet Projection
- **Resize:**
  - **Resize Window:** Set the Graphics Window size
    ...
    Custom (Input in Terminal)
  - **Resize Viewport:** Set the Viewport size
    ...
    Custom (Input in Terminal)
- **Color:**
  - **Line Color:** Set Line and Shape color
    ...
    Custom (Input in Terminal)
  - **Rotation Axis Color:** Set Rotation Axis color
    ...
    Custom (Input in Terminal)
  - **ViewPort Color:** Set ViewPort background color
    ...
    Custom (Input in Terminal)
  - **Background Color:** Set Graphic Window background color
    ...
- **Exit** Exit the program

Input Units:

Custom Color: **R G B**	**(0-255) (0-255) (0-255)**
Custom Translation Vector: **X Y Z**	**(Real World Coordinates)**
Custom Rotation Axis: **X1 Y1 Z1, X2 Y2 Z2**	**(Real World Coordinates)**
Custom Rotation Angle	**(Unit: Radian)**

DEFAULT PARAMETERS:

Translation Vector: 5.0 5.0 5.0	(Unit: Pixels)
Rotation Axis: Point 1: 0 0 0 Point 2: 0 100 100	(Unit: Pixels)
Rotation Angle: 0.1	(Unit: Radian)
Scaling Factor: 1.1	

Keyboard Interactions

FUNCTIONAL KEYS:

1. **Translation**
   - **W :** translate current object by **+y** defined in translation vector
   - **A :** translate current object by **-x** defined in translation vector
   - **S :** translate current object by **-y** defined in translation vector
   - **D :** translate current object by **+x** defined in translation vector
   - **UP_ARROW_KEY:** translate current object by **+z** defined in translation vector
   - **DOWN_ARROW_KEY:** translate current object by **-z** defined in translation vector
   - **T :** translate current object by **+x & +y & +z** defined in translation vector
2. **Rotation**
   - **Q :** rotate current object **clockwise** by **α** radian around rotation axis
   - **E :** rotate current object **counterclockwise** by **α** radian around rotation axis
3. **Display Rotation Axis**
   - **L :** Display / Hide The Rotation Axis
4. **Scaling**
   - **X :** scale current object by **(1 * scale)** defined in scaling factor
   - **Z :** scale current object by **(1 / scale)** defined in scaling factor
5. **Object Selection**
   - **. :** Switch to the next object
   - **, :** Switch to the previous object
6. **Exit Program**
   - **Esc :** Exit the program

Project Overview

Required:

At least three different polygons (Statue: **Complete**) (Location: **Data File**)
Projections Onto XY, XZ, YZ Planes (Statue: **Complete**) (Location: **[124-218]main.cpp**)
3D Translation Algorithm (Statue: **Complete**) (Location: **[35-41]parser.cpp**)
3D Rotation Algorithm (Statue: **Complete**) (Location: **[55-97]parser.cpp**)
3D Scaling Algorithm (Statue: **Complete**) (Location: **[43-53]parser.cpp**)

User Input Customization (Statue: **Complete**) (Location: **Runtime Menu**)

- **ID** of the polygon to be manipulated
- **scaling factor, translation vector, rotation angle, and rotation axis**

Output written to the input data file (Statue: **Complete**)
Makefile and Manual (Statue: **Complete**)

Extra Credit:

Oblique Projections
(Location: **[66-95, 220-250]main.cpp**)
Separate setting of Window and Viewport size
(Location: **Runtime Menu**)
Separate setting of Window, Viewport, background, rotation axis, and Line color
(Location: **Runtime Menu**)
Rescale All Objects according to the bounding box
(Location: **[30-95]main.cpp**)
Intuitive Keyboard Interactions (Location: **[199-244]main.cpp**)
Save Settings to a setting file at exit (Location: **[247-250]parser.cpp**)