Computer Graphics Designs – F29GR

Authors:

Tejas Dwarkaram	**H00182776**
Nelio Lucas	**H00182777**
Tyron Mc Donald	**H00182774**

Campus:

CTI Education Group
Nelspruit Campus
Mpumalanga
South Africa
1320

Contents

Executive Summary

The main objective of this project is to create a modelled animal, designed and structured in openGL. The animal chosen is a Unicorn, and is composed of 9 Alphabets which are;

- M
- R
- I
- T
- O
- D
- J
- N
- L

The model consists of 4 Legs, a set of Wings, a Tail, a Neck, a Head, and a Horn.

The Group Members of this project are;

- Tejas Dwarkaram	H00182776
- Nelio Lucas	H00182777
- Tyron Mc Donald	H00182774

Project Diary: Computer Graphics Information

Students: Tejas Dwarkaram, H00182776
Nelio Lucas, H00182777
Tyron Mc Donald, H00182774

Supervisor: Leoni Mullett

13/03/14: The group discussed and agreed upon a division and selection of letters that will be used in the outcome of the project. A Dragon is chosen as the desired model to be created.

To Do: For next week, each of us should have preliminary designs for each of our letters along with how the final modal should look.

21/03/14: Each of us brought forward our designs for how our letters should be structured and also how the dragon should look finally. We decided to base our letters structured on how Tejas designed his letter "M".

To Do: We should try to code our letters in Visual Studio, and report back next week.

31/03/14: Our letters were designed. Tyron had difficulty on designing his "N", so he was assisted by Nelio and I. A division of the parts of the dragon was decided upon. Wings – Tejas; Body, Tail, Head – Nelio; Legs – Tyron.

To Do: Our letters should be designed completely, and each of our parts designs should be structured.

14/04/14 The designs for our parts were designed, but there is a disagreement of our initially proposed outcome model. Our model is now changed to a Gryphon.

To Do: Since the parts are basically the same, but now with 4 legs, we need to get our parts coded at least.

18/04/14: Our parts are now coded, and Nelio decides that we should try to perfect them before we can put them together into a model and start animating.

24/04/14: Another disagreement, we now deciding to base our model on a Unicorn instead of a Gryphon (Gosh!). We all agreed that this would now be our final design.

To Do: All of us need to go restructure our parts based on our now finalised model.

28/04/14: Our parts were all compiled, and Nelio offered to put it all together. Tejas was given the task of animating and articulating the wings, and tail. Tyron was given the task of perfecting all of the scene graphs for the project.

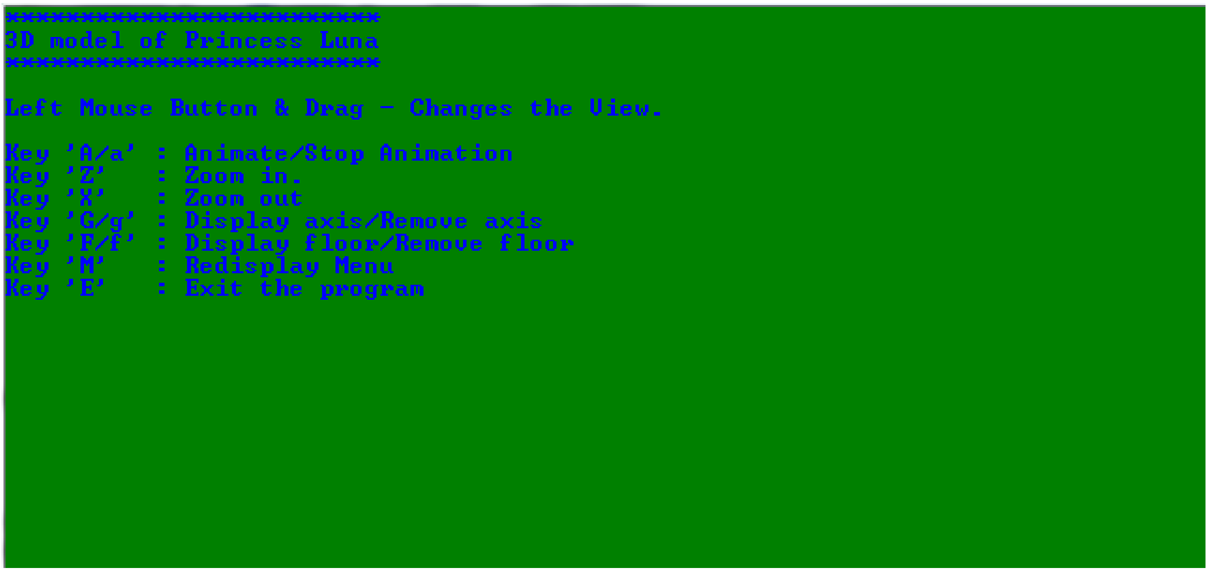
To Do: We all need to have our parts done for next week, a

submission is coming soon (09 May 2014).

01/05/14: All of our parts are animated(Yay!). We are all now going to do the finishing touches on the project, and complete the full documentation on the project

Graphical user manual

Menu Options



```
****************************
3D model of Princess Luna
****************************

Left Mouse Button & Drag - Changes the View.

Key 'A/a' : Animate/Stop Animation
Key 'Z'   : Zoom in.
Key 'X'   : Zoom out
Key 'G/g' : Display axis/Remove axis
Key 'F/f' : Display floor/Remove floor
Key 'M'   : Redisplay Menu
Key 'E'   : Exit the program
```

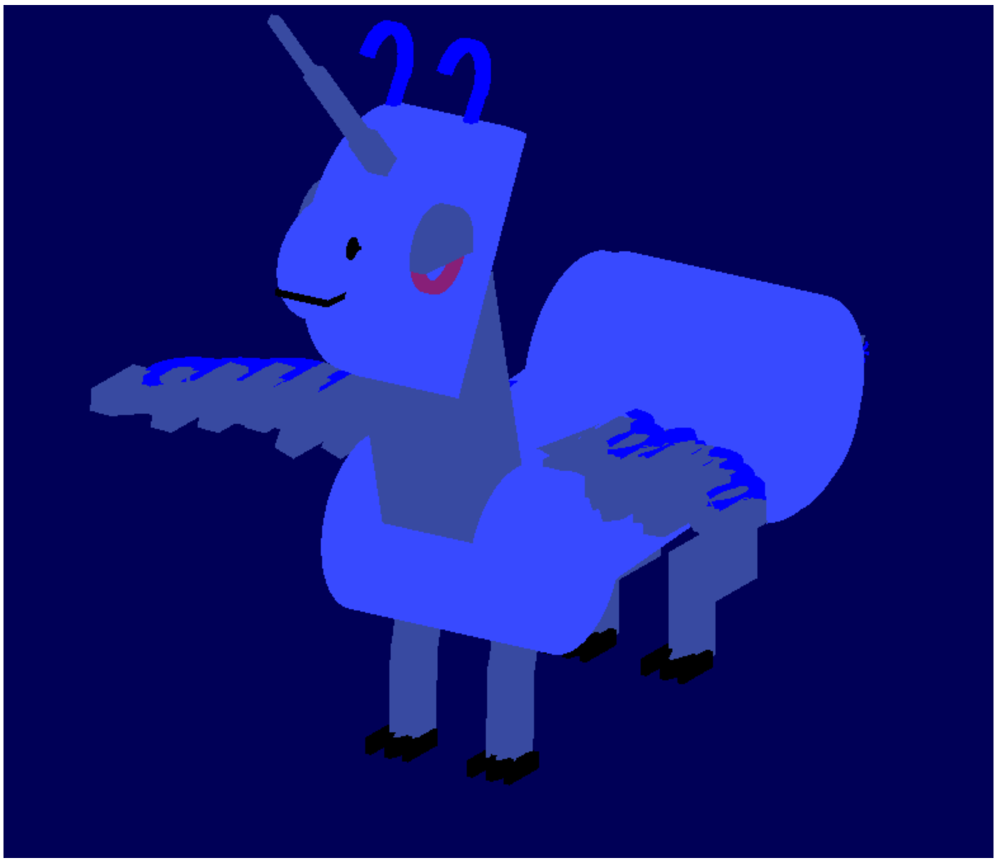
Graphical User Manual 1Layout of Menu Options and Functions

Upon initiating the program, a menu as the one displayed above, will be shown.

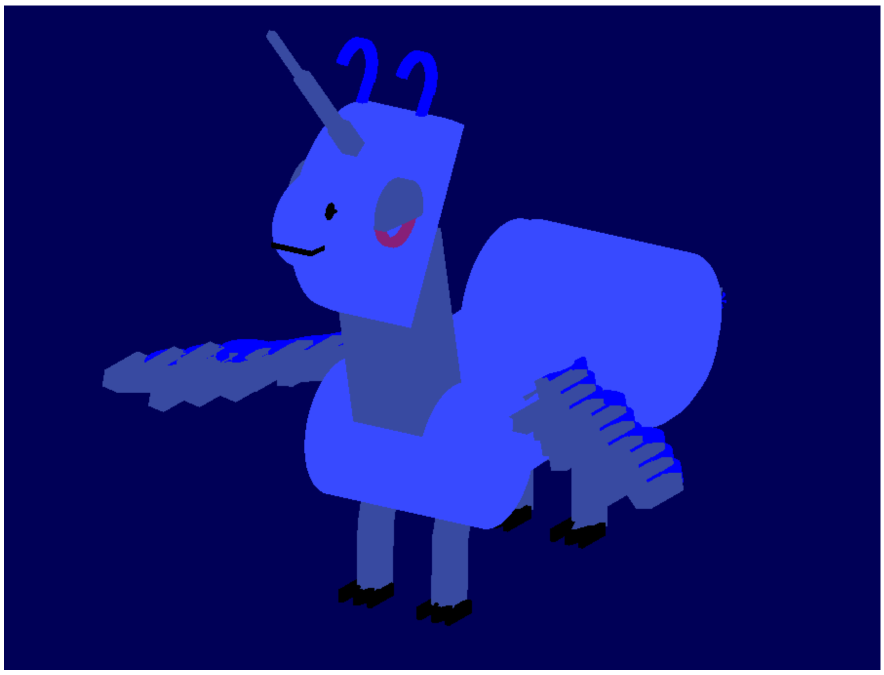
Menu Options:

- **If a capital A is pressed on the keyboard, the model will begin to animate itself**
- **A small A is pressed on the keyboard, the model will cease to animate itself**
- **If a capital Z is pressed on the keyboard, the camera zooms in closer to the model**
- **If a capital X is pressed on the keyboard, the camera zooms further away from the model**
- **If a capital G is pressed an axis on which the model the model is structured will be shown**
- **If a small G is pressed on the keyboard the axis will disappear**
- **If a capital F is pressed on the keyboard, the floor on which the model is displayed will be shown**
- **If a small G is pressed on the keyboard, the floor will disappear**
- **If a capital E is pressed on the keyboard, the program will exit**
- **If a capital M is pressed on the keyboard, the console will redisplay the menu**

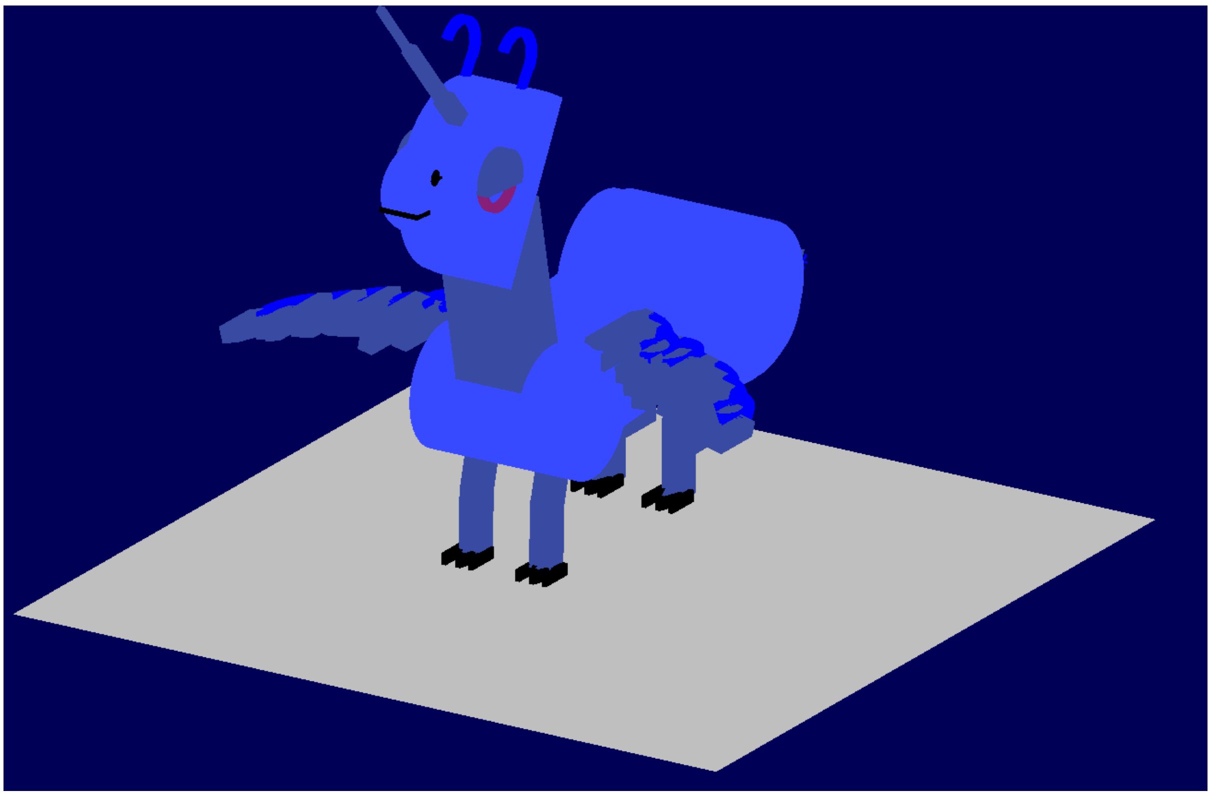
Animation Control


Graphical User Manual 2 Princess Luna before Animation is initiated

When a Capital A is pressed on the keyboard, the model of Princess Luna will begin to animate periodically.

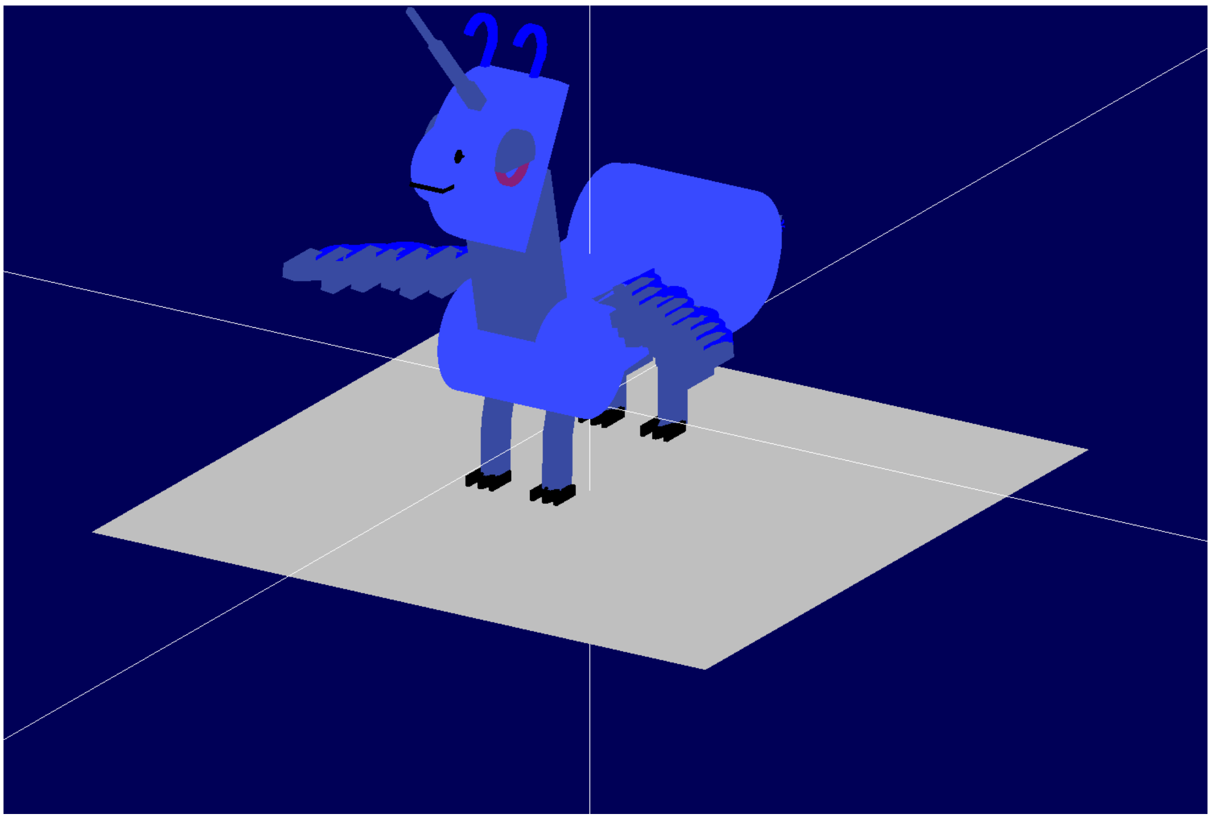

Graphical User Manual 3 Princess Luna after Animation is initiated

Floor View control


Graphical User Manual 4 Floor being drawn

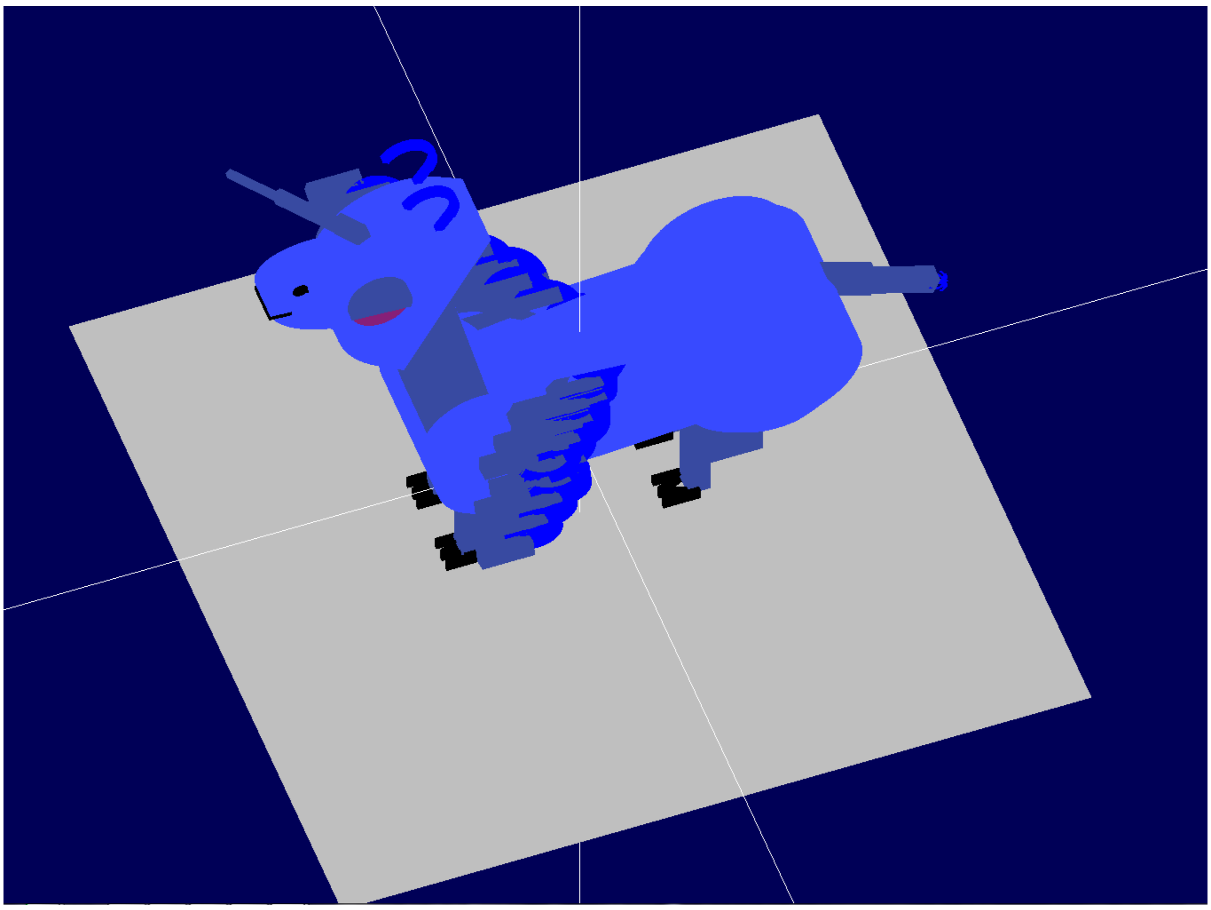
When a Capital F is pressed on the keyboard, the Floor will be displayed under the model.

Gridline View Control

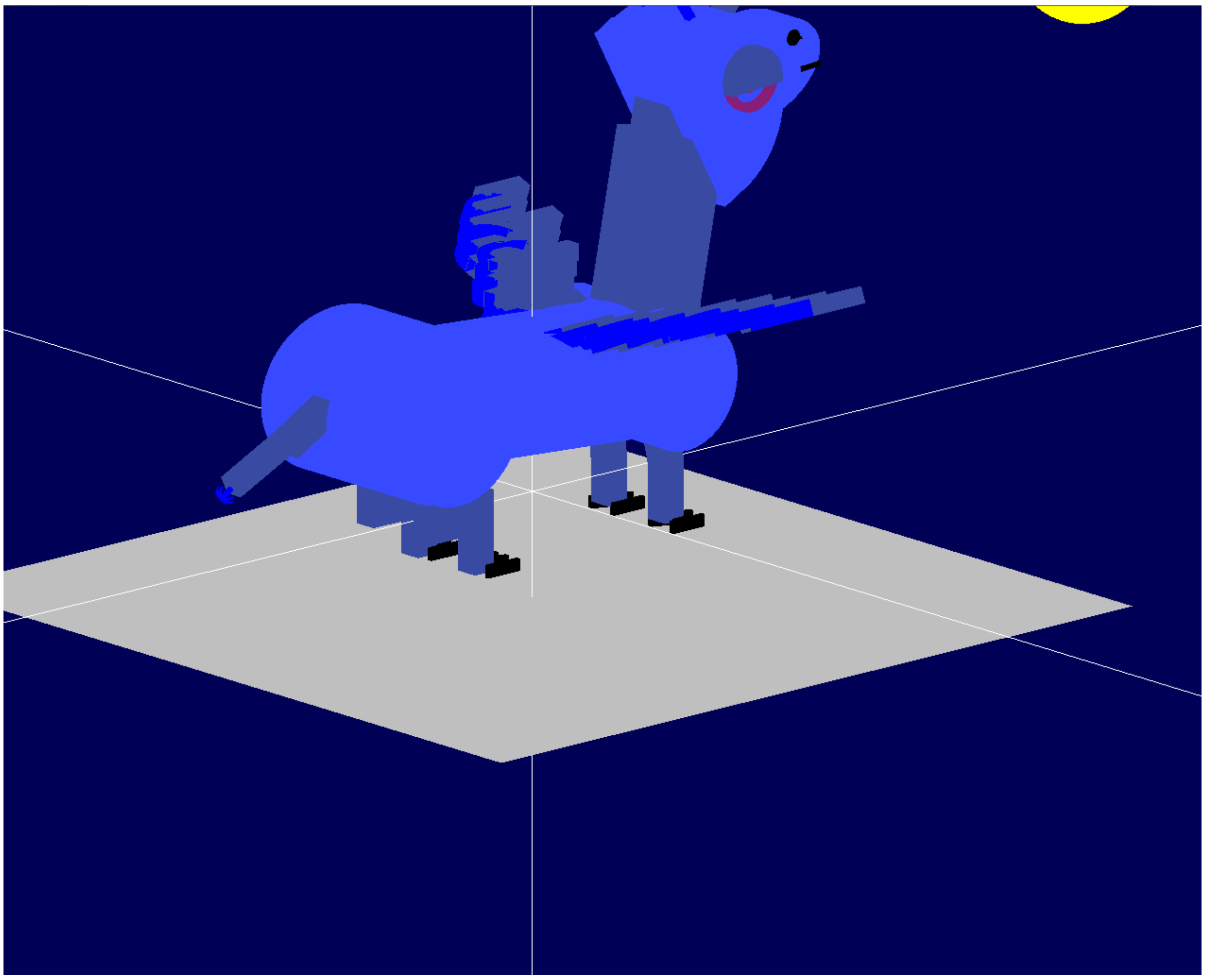

Graphical User Manual 5 Gridlines displayed on model

When a Capital G is pressed on the keyboard, Gridlines will be displayed around the model.

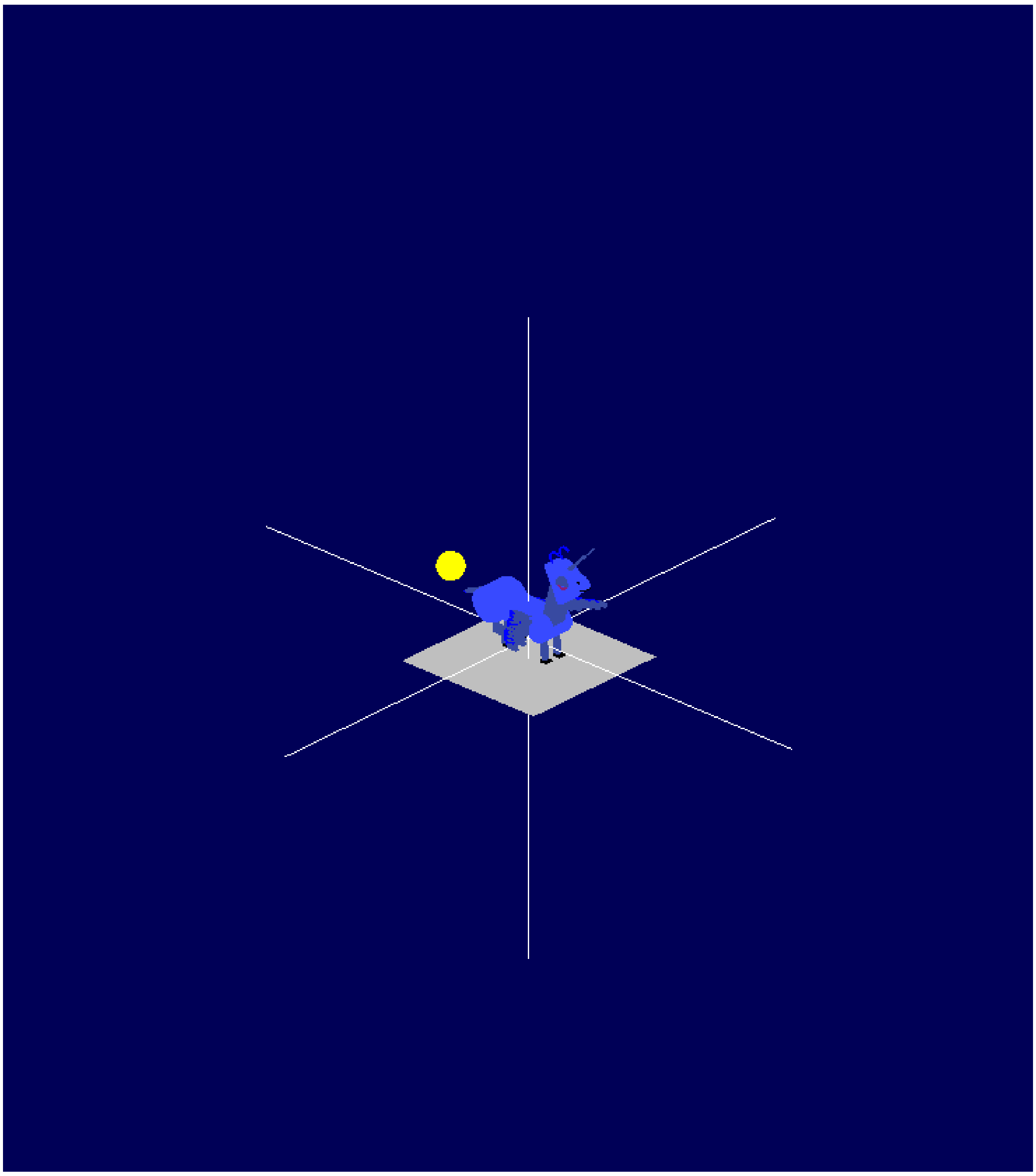
Mouse Rotation Control


Graphical User Manual 6 Mouse Rotation of Model

When the left mouse button clicked and dragged in any manner, the model will be rotated in that manner.

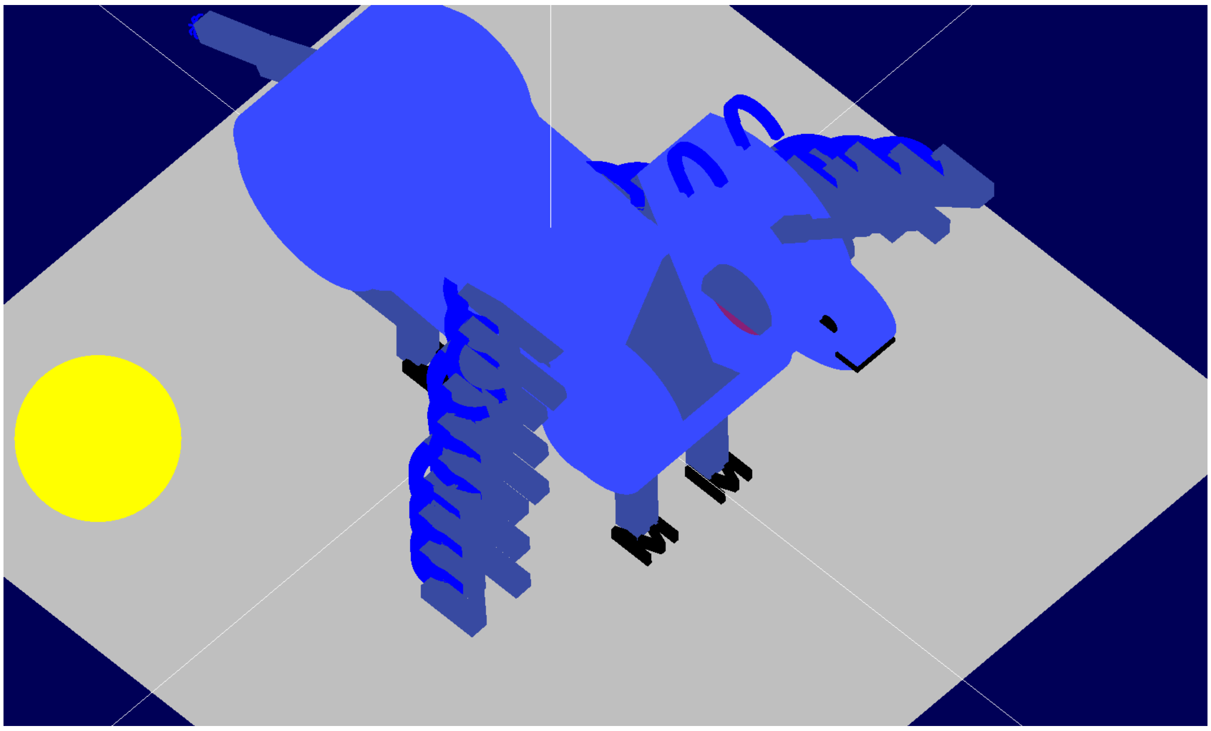

Graphical User Manual 7 Mouse Rotation of Model

Zoom Function Control



Graphical User Manual 8 Zoom function for Zoom Out

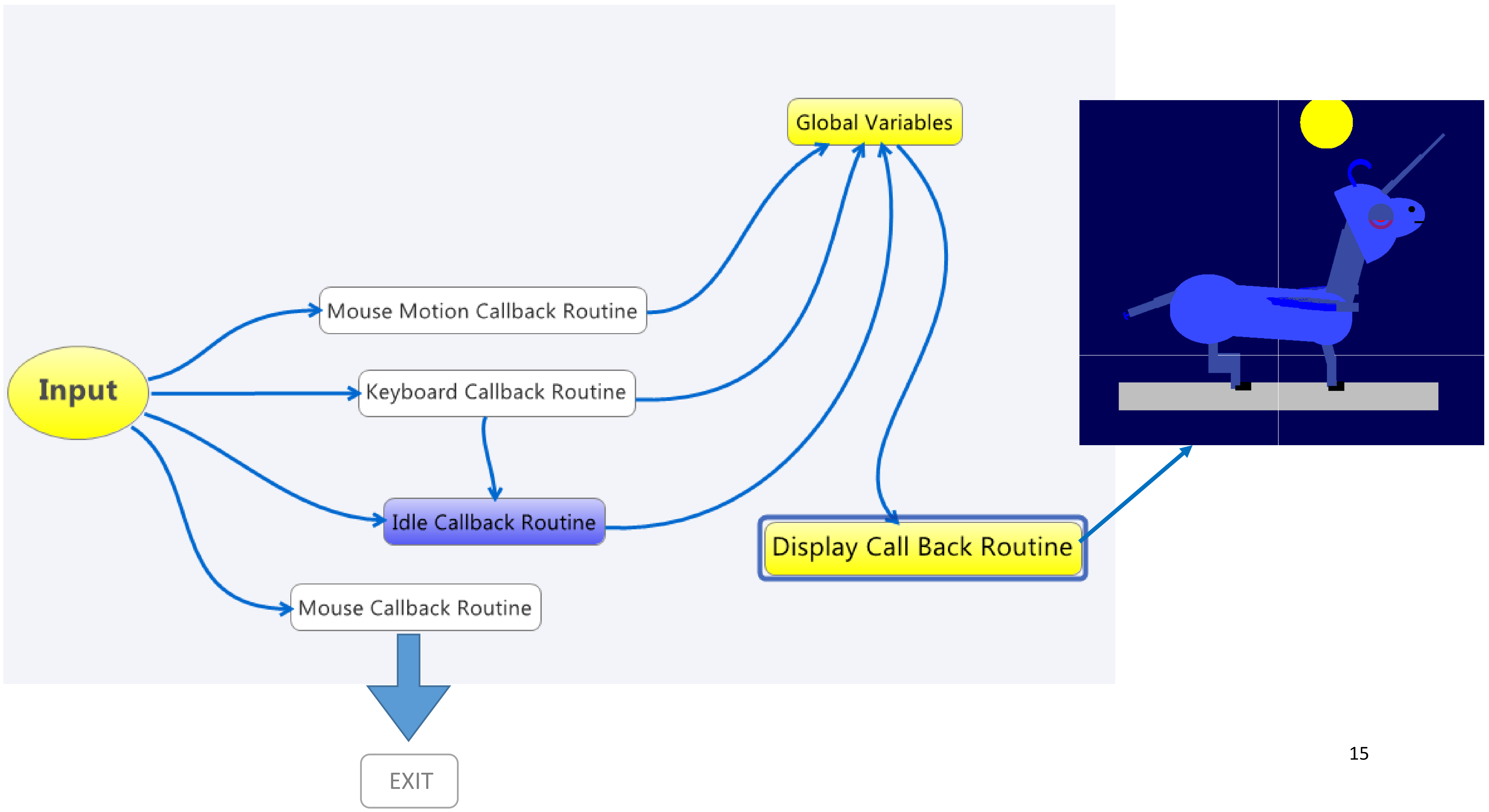
When a Capital X is pressed on the keyboard, the model will be displayed further away from the camera



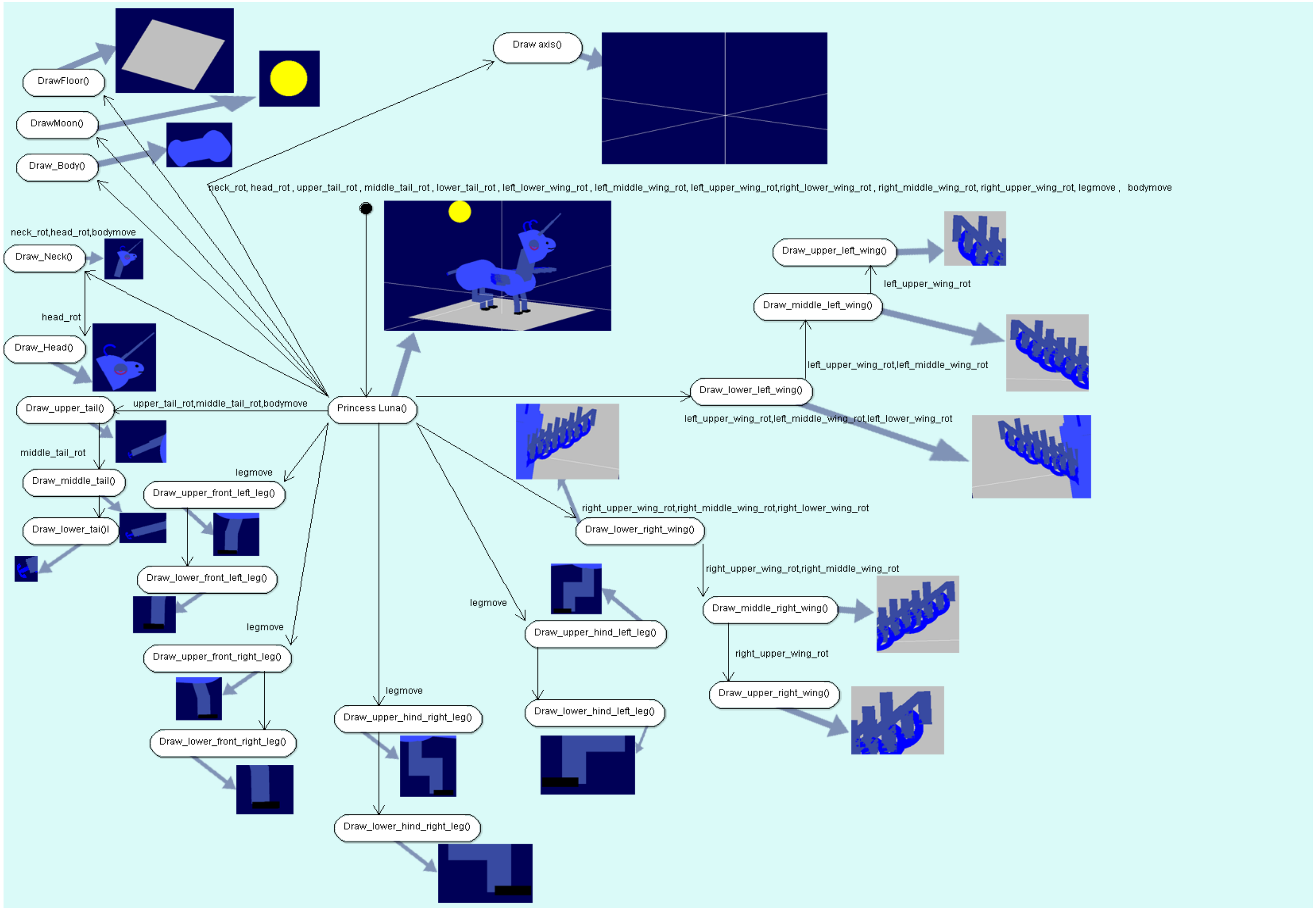
Graphical User Manual 9 Zoom Function for Zoom In

When a Capital Z is pressed on the keyboard, the camera will be displayed closer to the model

Top Level Program Design


Input
Mouse Motion Callback Routine
Keyboard Callback Routine
Idle Callback Routine
Mouse Callback Routine
Global Variables
Display Call Back Routine
EXIT

Top Level Scene Graph


DrawFloor()
DrawMoon()
Draw_Body()
Draw axis()
neck_rot, head_rot , upper_tail_rot , middle_tail_rot , lower_tail_rot , left_lower_wing_rot , left_middle_wing_rot, left_upper_wing_rot,right_lower_wing_rot , right_middle_wing_rot, right_upper_wing_rot, legmove , bodymove
neck_rot,head_rot,bodymove
Draw_Neck()
head_rot
Draw_Head()
Draw_upper_tail()
upper_tail_rot,middle_tail_rot,bodymove
Princess Luna()
middle_tail_rot
Draw_middle_tail()
Draw_lower_tai()l
legmove
Draw_upper_front_left_leg()
Draw_lower_front_left_leg()
legmove
Draw_upper_front_right_leg()
Draw_lower_front_right_leg()
legmove
Draw_upper_hind_right_leg()
Draw_lower_hind_right_leg()
legmove
Draw_upper_hind_left_leg()
Draw_lower_hind_left_leg()
Draw_upper_left_wing()
left_upper_wing_rot
Draw_middle_left_wing()
left_upper_wing_rot,left_middle_wing_rot
Draw_lower_left_wing()
left_upper_wing_rot,left_middle_wing_rot,left_lower_wing_rot
right_upper_wing_rot,right_middle_wing_rot,right_lower_wing_rot
Draw_lower_right_wing()
right_upper_wing_rot,right_middle_wing_rot
Draw_middle_right_wing()
right_upper_wing_rot
Draw_upper_right_wing()

Conclusions by Group Members

Tejas Dwarkaram H00182776

During the duration of this project, a new language was added to my skill list (C++). OpenGL was highly interesting, and extremely fun, with a high sense of satisfaction each time a new component was completed. I learnt that co-operation within a team, on projects like this is very important and when the team communicates well and work is divided equally, the project will be highly successful. As was ours.

If given the chance to do a project of this nature again, I would now offer to complete more tasks, and create more elaborate and detailed, structured letters.

Had I twice the time, I would definitely, have perfected the end model, as well as done a bit more research on texture mapping in order to give our model a definitive and better appearance. I would also have given our model additional features such as perspective changes in terms of camera location, four directional movement in terms of forward, backward, left and right. A timer function to figure out the time of day in order to figure out whether a moon or sun will be displayed for the model, will also be added to a list of possible improvements.

In terms of what can be improved for next years project requirements, I would insist that the basic requirements, should be increased just a little along with the duration, in order to expand the knowledge base of students in terms of more than one double articulation as well as expanding the range of the basis on which the models should be designed.

Nelio Lucas H00182777

During the course of the project, (though it wasn’t very clear at first) I can say that I have mastered drawing 2D and 3D objects, I also learnt how to animate by translation, rotation and scaling to a fair extent, and not to mention how to use GLUT functions to properly interact with the Idle Call back function so that the image is properly drawn onto the screen without any distortions. I can also say that I “self learnt” through all the knowledge I gain in this course to write efficient code (example the parameters set for each letter function we made). If it wasn’t for that, we would have over 2000 lines of code!

Next time I would include “.c” files to the project. This time it was not possible because we used a class to make our letters and all the different body parts.

If I had twice the time to complete the project, I would include texture mapping and make smoother, inline articulations to make the 3D model more life-like. Another thing I thought of was attempting to include sound as an object in order to give the 3D animation more life, though I’m not sure if that’s possible.

For next year I think it would be better to make it a minimum requirement for the 3D model to have triple articulations. Texture mapping should also be a requirement, that way students can get creative and not only create a plain animal but something with more skin and definition, but that means that more time would be required to complete the project, Heriot Watt may have to look into that if considered.

Tyron Mc Donald H00182774

While the project was underway learned how to model letters in 3D and how they could be maneuvered together, and how they could become one animal. If we were given a second chance I would have started working on the project earlier so we could add more to the project.
If we had double the time we could have made a griffin as well so the two animals could \transform into each other.
What I would add to the project next year I would have had a theme with the animal.

Code Listing of Tejas Dwarkaram’s Initials (M, R, I)

Letter M

```
/*****************************************************************************************
********************************************
Author: Tejas Dwarkaram
FileName: Letter M.h
Date: 2014-04-11
Operating system: Windows 7
Description : this file contains the points for creating the letter M
*****************************************************************************************
********************************************/
#ifndef Letter_M_H
#define Letter_M_H

#include "Letter Class.h"

/***********************/
/*    Draw letter M    */
/***********************/

//These parameters change the position , size and angle of the M
void Letters::letter_m(float translateX, float translateY, float translateZ,float
rotateAngle, float rotateX, float rotateY, float rotateZ ,float scaleX, float scaleY,
float scaleZ)
{
       //translation constants
       float x = -0.2;
       float y = 0;


       float a = -0.1;
       float b = -0.1;

       float c = x + a;
       float d = y + b;
       //3D
       float z = -0.1;


       glPushMatrix();
       glTranslatef(translateX, translateY, translateZ);
       glRotatef(rotateAngle, rotateX, rotateY, rotateZ);
       glScalef(scaleX, scaleY, scaleZ);

       //FRONT
       //Column1
       glBegin(GL_POLYGON);
       glColor3f(0,0,0);
       glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
       glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
       glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //B
       glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
       glEnd();

       glBegin(GL_POLYGON);
       glColor3f(0,0,0);
       glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
       glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
       glVertex3f(0.8 + x + a, -0.9 + y + b, z); //B
       glVertex3f(0.5 + x + a, -0.9 + y + b, z); //A
       glEnd();
```

```
glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex3f(0.8 + x + a, -0.9 + y + b, z); //D
glVertex3f(0.5 + x + a, -0.9 + y + b, z); //C
glVertex3f(1.5 + x + a, 0.1 + y + b, z); //B
glVertex3f(1.8 + x + a, 0.1 + y + b, z); //A
glEnd();

//Column2
glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex3f(1.5 + x + a, 0.1 + y + b, z); //D
glVertex3f(1.8 + x + a, 0.1 + y + b, z); //C
glVertex3f(1.8 + x + a, -0.9 + y + b, z); //B
glVertex3f(1.5 + x + a, -0.9 + y + b, z); //A
glEnd();


//3D Sides
glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
glEnd();

glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex3f(1.8 + x + a, 0.1 + y + b, z); //C
glVertex3f(1.8 + x + a, 0.1 + y + b, -0.4); //C
glVertex3f(1.8 + x + a, -0.9 + y + b, -0.4); //B
glVertex3f(1.8 + x + a, -0.9 + y + b, z); //B
glEnd();

glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
glEnd();

glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //B
glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //B
glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
glEnd();


glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex3f(1.5 + x + a, 0.1 + y + b, z); //D
glVertex3f(1.5 + x + a, 0.1 + y + b, -0.4); //D
glVertex3f(1.8 + x + a, 0.1 + y + b, -0.4); //C
glVertex3f(1.8 + x + a, 0.1 + y + b, z); //D
glEnd();

glBegin(GL_POLYGON);
```

```
glColor3f(0,0,0);
glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
glVertex3f(0.5 + x + a, -0.9 + y + b, -0.4); //A
glVertex3f(0.5 + x + a, -0.9 + y + b, z); //A
glEnd();

glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
glVertex3f(0.8 + x + a, -0.9 + y + b, -0.4); //B
glVertex3f(0.8 + x + a, -0.9 + y + b, z); //B
glEnd();

glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex3f(0.8 + x + a, -0.9 + y + b, z); //D
glVertex3f(0.8 + x + a, -0.9 + y + b, -0.4); //D
glVertex3f(1.8 + x + a, 0.1 + y + b, -0.4); //A
glVertex3f(1.8 + x + a, 0.1 + y + b, z); //A
glEnd();

glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex3f(0.5 + x + a, -0.9 + y + b, z); //C
glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
glVertex3f(0.8 + x + a, -0.9 + y + b, -0.4); //B
glVertex3f(1.5 + x + a, 0.1 + y + b, z); //B
glEnd();



//Column2
glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex3f(1.8 + x + a, -0.9 + y + b, z); //B
glVertex3f(1.8 + x + a, -0.9 + y + b, -0.4); //B
glVertex3f(1.5 + x + a, -0.9 + y + b, -0.4); //A
glVertex3f(1.5 + x + a, -0.9 + y + b, z); //A
glEnd();


//REAR
//Column1
glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //B
glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
glEnd();

glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
glVertex3f(0.8 + x + a, -0.9 + y + b, -0.4); //B
glVertex3f(0.5 + x + a, -0.9 + y + b, -0.4); //A
glEnd();

glBegin(GL_POLYGON);
```

```
        glColor3f(0,0,0);
        glVertex3f(0.8 + x + a, -0.9 + y + b, -0.4); //D
        glVertex3f(0.5 + x + a, -0.9 + y + b, -0.4); //C
        glVertex3f(1.5 + x + a, 0.1 + y + b, -0.4); //B
        glVertex3f(1.8 + x + a, 0.1 + y + b, -0.4); //A
        glEnd();

        //Column2
        glBegin(GL_POLYGON);
        glColor3f(0,0,0);
        glVertex3f(1.5 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(1.8 + x + a, 0.1 + y + b, -0.4); //C
        glVertex3f(1.8 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(1.5 + x + a, -0.9 + y + b, -0.4); //A
        glEnd();

        glPopMatrix();

        glFlush();
}


#endif
```

Letter R

```
/*************************************************************************************
********************************************
Author: Tejas Dwarkaram
FileName: Letter R.h
Date: 2014-04-11
Operating system: Windows 7
Description : this file contains the points for creating the letter R
*************************************************************************************
********************************************/
#ifndef Letter_R_H
#define Letter_R_H

#include "Letter Class.h"


/***********************/
/*    Draw letter R    */
/***********************/

//These parameters change the position , size , angle and color of the R
void Letters::letter_r(float translateX, float translateY, float translateZ,float
rotateAngle, float rotateX, float rotateY, float rotateZ ,float scaleX, float scaleY,
float scaleZ ,  float red , float green , float blue , float angle , bool Xaxis , bool
Yaxis ,bool Zaxis)
{



       //translation constants
       float x = -0.2;
       float y = 0;


       float a = -0.1;
       float b = -0.1;

       float c = x + a;
       float d = y + b;
       //3D
       float z = -0.1;

       glPushMatrix();
       glTranslatef(translateX, translateY, translateZ);
       glRotatef(rotateAngle, rotateX, rotateY, rotateZ);
       glScalef(scaleX, scaleY, scaleZ);

       glPushMatrix();
       glTranslatef(0,0,0);
       glRotatef(angle, Xaxis, Yaxis, Zaxis);
       glScalef(1,1,1);

       glBegin(GL_POLYGON);
       glColor3f(0.22 ,0.29 ,0.63);
       glVertex3f(-0.4 + x + a, 0.9 + y + b, z); //D
       glVertex3f(-0.1 + x + a, 0.9 + y + b, z); //C
       glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //B
       glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
       glEnd();
```

```
glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
glVertex3f(0.8 + x + a, -0.9 + y + b, z); //B
glVertex3f(0.5 + x + a, -0.9 + y + b, z); //A
glEnd();

glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(-0.4 + x + a, 0.9 + y + b, z); //D
glVertex3f(-0.4 + x + a, 0.9 + y + b, -0.4); //D
glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
glEnd();


glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(-0.1 + x + a, 0.9 + y + b, z); //C
glVertex3f(-0.1 + x + a, 0.9 + y + b, -0.4); //C
glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //B
glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //B
glEnd();


glPushMatrix();
glScalef(1, 1, 1);
glRotatef(0, 0, 0, 0);
glTranslatef(-0.4, 0.1, -0.25);
//Draw curve using code in 3DCurve.cpp
curve_o(0.3,          //depth
       0.5,                 //inner radius
       0.7,                //outer radius
       -60.0,          //start angle
       93,                        //stop angle
       4.0 , red , green , blue);                //anular increments
glPopMatrix();


glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
glVertex3f(0.8 + x + a, -0.9 + y + b, -0.4); //B
glVertex3f(0.8 + x + a, -0.9 + y + b, z); //B
glEnd();




glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(0.8 + x + a, -0.9 + y + b, z); //B
glVertex3f(0.8 + x + a, -0.9 + y + b, -0.4); //B
glVertex3f(0.5 + x + a, -0.9 + y + b, -0.4); //A
glVertex3f(0.5 + x + a, -0.9 + y + b, z); //A
glEnd();

glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(-0.4 + x + a, 0.9 + y + b, -0.4); //D
```

```
        glVertex3f(-0.1 + x + a, 0.9 + y + b, -0.4); //C
        glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
        glEnd();

        glBegin(GL_POLYGON);
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
        glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
        glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
        glEnd();

        glBegin(GL_POLYGON);
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
        glVertex3f(0.8 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(0.5 + x + a, -0.9 + y + b, -0.4); //A
        glEnd();

        glPopMatrix();
        glPopMatrix();
        glFlush();
}

#endif
```

Letter I

```
/*************************************************************************************
********************************************
Author: Tejas Dwarkaram
FileName: Letter I.h
Date: 2014-04-11
Operating system: Windows 7
Description : this file contains the points for creating the letter I
*************************************************************************************
*******************************************/
#ifndef Letter_I_H
#define Letter_I_H

#include "Letter Class.h"


/***********************/
/*    Draw letter I    */
/***********************/

                                          //These parameters change the position and
size of the  I
void Letters::letter_i(float translateX, float translateY, float translateZ,float
rotateAngle, float rotateX, float rotateY, float rotateZ ,float scaleX, float scaleY,
float scaleZ)
{
//translation constants
       float x = -0.2;
       float y = 0;


       float a = -0.1;
       float b = -0.1;

       float c = x + a;
       float d = y + b;
       //3D
       float z = -0.1;



       glPushMatrix();
       glTranslatef(translateX, translateY, translateZ);
       glRotatef(rotateAngle, rotateX, rotateY, rotateZ);
       glScalef(scaleX, scaleY, scaleZ);

       //FRONT
       //Column1
       glBegin(GL_POLYGON);
       glColor3f(0.22 ,0.29 ,0.63);
       glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
       glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
       glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //B
       glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
       glEnd();


       //Column1
       glBegin(GL_POLYGON);
       glColor3f(0.22 ,0.29 ,0.63);
```

```
        glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
        glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
        glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
        glEnd();

        //Column1
        glBegin(GL_POLYGON);
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
        glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
        glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //B
        glEnd();


        //Column1
        glBegin(GL_POLYGON);
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
        glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
        glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
        glEnd();

        //Column1
        glBegin(GL_POLYGON);
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //B
        glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
        glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
        glEnd();


        //REAR
        //Column1
        glBegin(GL_POLYGON);
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
        glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
        glEnd();


        glPopMatrix();
        glFlush();

}


#endif
```

Code Listing of Nelio Lucas’ Initials (T, O, D)

Letter T

```
/*************************************************************************************
********************************************
Author: Nelio Lucas
FileName: Letter T.h
Date: 2014-04-11
Operating system: Windows 7
Description : this file contains the points for creating the letter T
*************************************************************************************
********************************************/
#ifndef Letter_T_H
#define Letter_T_H

#include "Letter Class.h"


/***********************/
/*    Draw letter T    */
/***********************/

//These parameters change the position , size and of the T
void Letters::letter_t(float translateX, float translateY, float translateZ,float 
rotateAngle, float rotateX, float rotateY, float rotateZ ,float scaleX, float scaleY, 
float scaleZ)
{
       //translation constants
       float x = -0.2;
       float y = 0;


       float a = -0.1;
       float b = -0.1;

       float c = x + a;
       float d = y + b;
       //3D


       float z = 0.1;

       glPushMatrix();
       glTranslatef(translateX, translateY, translateZ);
       glRotatef(rotateAngle, rotateX, rotateY, rotateZ);
       glScalef(scaleX, scaleY, scaleZ);

       //REAR
       //glClear(GL_COLOR_BUFFER_BIT);

       glBegin(GL_POLYGON);

       glColor3f(0.22 ,0.29 ,0.63);
       glVertex3f(-0.1 + x, 0.1 + y, 0.0); //D
       glVertex3f(0.9 + x, 0.1 + y, 0.0); //C
       glVertex3f(0.9 + x, -0.1 + y, 0.0); //B
       glVertex3f(-0.1 + x, -0.1 + y, 0.0); //A

       glEnd();

       glBegin(GL_POLYGON);
       glColor3f(0.22 ,0.29 ,0.63);
```

```
glVertex3f(0.3 + x, 0.1 + y, 0.0); //D
glVertex3f(0.5 + x, 0.1 + y, 0.0); //C
glVertex3f(0.5 + x, -0.9 + y, 0.0); //B
glVertex3f(0.3 + x, -0.9 + y, 0.0); //A

glEnd();


//FRONT
glBegin(GL_POLYGON);

glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(-0.1 + x , 0.1 + y , z); //D
glVertex3f(0.9 + x , 0.1 + y , z); //C
glVertex3f(0.9 + x , -0.1 + y , z); //B
glVertex3f(-0.1 + x , -0.1 + y , z); //A

glEnd();

glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(0.3 + x , 0.1 + y , z); //D
glVertex3f(0.5 + x , 0.1 + y , z); //C
glVertex3f(0.5 + x , -0.9 + y , z); //B
glVertex3f(0.3 + x , -0.9 + y , z); //A
glEnd();

//TOP
glBegin(GL_POLYGON);
//glPushMatrix();
//glScalef(0.5, 0.5, 0.5);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(0.7, 0.1, z); //B
glVertex3f(0.7 , 0.1, 0.0); //C
glVertex3f(-0.3 , 0.1, 0.0); //D
glVertex3f(-0.3 , 0.1, z); //A
//glPopMatrix();
glEnd();

//Bottom : Lower Base
glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(0.5 + x , -0.9 + y , z); //B
glVertex3f(0.5 + x , -0.9 + y , 0.0); //C
glVertex3f(0.3 + x , -0.9 + y , 0.0); //D
glVertex3f(0.3 + x , -0.9 + y , z); //A

glEnd();

//Bottom: Medium Base
glBegin(GL_POLYGON);

glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(0.7, -0.1, z); //B
glVertex3f(0.7 , -0.1, 0.0); //C
glVertex3f(-0.3 , -0.1, 0.0); //D
glVertex3f(-0.3 , -0.1, z); //A

glEnd();

//lower side : left
glBegin(GL_POLYGON);
```

```
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(0.3 + x , -0.1 + y , z); //D
        glVertex3f(0.3 + x , -0.1 + y , 0.0);//C
        glVertex3f(0.3 + x , -0.9 + y , 0.0);//B
        glVertex3f(0.3 + x , -0.9 + y , z); //A
        glEnd();

        //lower side : right
        glBegin(GL_POLYGON);
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(0.5 + x , -0.1 + y , z); //D
        glVertex3f(0.5 + x , -0.1 + y , 0.0);//C
        glVertex3f(0.5 + x , -0.9 + y , 0.0);//B
        glVertex3f(0.5 + x , -0.9 + y , z); //A
        glEnd();

        //upper side: right
        glBegin(GL_POLYGON);
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(0.9 + x , 0.1 + y , z); //D
        glVertex3f(0.9 + x , 0.1 + y , 0.0);//C
        glVertex3f(0.9 + x , -0.1 + y , 0.0);//B
        glVertex3f(0.9 + x , -0.1 + y , z); //A
        glEnd();

        //upper side: left
        glBegin(GL_POLYGON);
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(-0.1 + x , 0.1 + y , z); //D
        glVertex3f(-0.1 + x , 0.1 + y , 0.0);//C
        glVertex3f(-0.1 + x , -0.1 + y , 0.0);//B
        glVertex3f(-0.1 + x , -0.1 + y , z); //A
        glEnd();

        glPopMatrix();

        glFlush();

}

#endif
```

Letter O

```
/*************************************************************************************
********************************************
Author: Nelio Lucas
FileName: Letter O.h
Date: 2014-04-11
Operating system: Windows 7
Description : this file contains the points for creating the letter O
*************************************************************************************
********************************************/
#ifndef Letter_O_H
#define Letter_O_H

#include "Letter Class.h"

/************************************************/
/*    Draw letter O and the 360 curve function */
/*************************************************/

//These parameters change the position , size , angle and color of the O
void Letters::letter_o(float translateX, float translateY, float translateZ,float
rotateAngle, float rotateX, float rotateY, float rotateZ ,float scaleX, float scaleY,
float scaleZ ,  float red , float green , float blue)
{
       glPushMatrix();
       glTranslatef(translateX, translateY, translateZ);
       glRotatef(rotateAngle, rotateX, rotateY, rotateZ);
       glScalef(scaleX, scaleY, scaleZ);
       curve_o(1.0 , 1.5 , 2.0 , 0.0 , 360 , 5.0 , red , green , blue);
       glPopMatrix();
}


void Letters::curve_o(double depth, double r1, double r2, double theta_start, double
theta_stop, double theta_inc , float red , float green , float blue)
{
       // Function to draw 3D curve
       // depth = depth centred round z=0
       // r1 = inner radius
       // r2 = outer radius
       // theta_start = start angle in degrees measured from x-axis
       // theta_stop = similar

       double x, y, x1, x2, y1, y2, z, thet, z_front, z_back;
       int i=0;
       double radius=1.5, c=3.14159/180.0;
       z_front=depth/2; z_back=-depth/2;

       // draw rear face (away from viewer)
       glColor3f(red, green, blue);
       z=z_back;
       glBegin(GL_QUAD_STRIP);
       for(thet=theta_start; thet<=theta_stop;thet+=theta_inc) {
              x=cos(c*thet)*r2; y=sin(c*thet)*r2; glVertex3d(x,y,z);
              x=cos(c*thet)*r1; y=sin(c*thet)*r1; glVertex3d(x,y,z);
       }
       glEnd();

       // draw front face (closer to viewer)
       glColor3f(red, green, blue);
```

```
        z=z_front;
        glBegin(GL_QUAD_STRIP);
        for(thet=theta_start; thet<=theta_stop;thet+=theta_inc){
                x=cos(c*thet)*r1; y=sin(c*thet)*r1; glVertex3d(x,y,z);
                x=cos(c*thet)*r2; y=sin(c*thet)*r2; glVertex3d(x,y,z);
        }
        glEnd();

        // draw upper face
        glColor3f(red, green, blue);
        glBegin(GL_QUAD_STRIP);
        for(thet=theta_start; thet<=theta_stop;thet+=theta_inc) {
                x=cos(c*thet)*r2; y=sin(c*thet)*r2;
                z=z_front; glVertex3d(x,y,z);
                z=z_back;  glVertex3d(x,y,z);
        }
        glEnd();

        // draw lower face
        glBegin(GL_QUAD_STRIP);
        for(thet=theta_start; thet<=theta_stop;thet+=theta_inc){
                x=cos(c*thet)*r1; y=sin(c*thet)*r1;
                z=z_back; glVertex3d(x,y,z);
                z=z_front; glVertex3d(x,y,z);
        }
        glEnd();

        // draw bottom end - Fixed by Tom Methven (31/08/2012)
        glColor3f(red, green, blue);
        glBegin(GL_POLYGON);
                x1=cos(c*theta_start)*r1; y1=sin(c*theta_start)*r1;
                x2=cos(c*theta_start)*r2; y2=sin(c*theta_start)*r2;

                glVertex3d(x1,y1,z_front);
                glVertex3d(x2,y2,z_front);
                glVertex3d(x2,y2,z_back);
                glVertex3d(x1,y1,z_back);
        glEnd();

        // draw top end
        glBegin(GL_POLYGON);
                x1=cos(c*theta_stop)*r1; y1=sin(c*theta_stop)*r1;
                x2=cos(c*theta_stop)*r2; y2=sin(c*theta_stop)*r2;

                glVertex3d(x1,y1,z_front);
                glVertex3d(x2,y2,z_front);
                glVertex3d(x2,y2,z_back);
                glVertex3d(x1,y1,z_back);
        glEnd();
}



#endif
```

Letter D

```
/*************************************************************************************
*******************************************
Author: Nelio Lucas
FileName: Letter D.h
Date: 2014-04-11
Operating system: Windows 7
Description : this file contains the points for creating the letter D
*************************************************************************************
*******************************************/
#ifndef Letter_D_H
#define Letter_D_H

#include "Letter Class.h"

/***********************/
/*    Draw letter D    */
/***********************/

//These parameters change the position , size , angle and color of the D
void Letters::letter_d(float translateX, float translateY, float translateZ,float
rotateAngle, float rotateX, float rotateY, float rotateZ ,float scaleX, float scaleY,
float scaleZ, float red , float green , float blue)
{
       glPushMatrix();
       glTranslatef(translateX, translateY, translateZ);
       glRotatef(rotateAngle, rotateX, rotateY, rotateZ);
       glScalef(scaleX, scaleY, scaleZ);
       curveofd(1.7 , 0 , 2.0 , 0 , 180 , 5.0 , red , green , blue);

       glPopMatrix();

}

void Letters::curveofd(double depth, double r1, double r2, double theta_start, double
theta_stop, double theta_inc , float red, float green , float blue)
{
       // Function to draw 3D curve
       // depth = depth centred round z=0
       // r1 = inner radius
       // r2 = outer radius
       // theta_start = start angle in degrees measured from x-axis
       // theta_stop = similar

       double x, y, x1, x2, y1, y2, z, thet, z_front, z_back;
       int i=0;
       double radius=1.5, c=3.14159/180.0;
       z_front=depth/2; z_back=-depth/2;

       // draw rear face (away from viewer)
       glColor3f(red,green, blue);
       z=z_back;
       glBegin(GL_QUAD_STRIP);
       for(thet=theta_start; thet<=theta_stop;thet+=theta_inc) {
              x=cos(c*thet)*r2; y=sin(c*thet)*r2; glVertex3d(x,y,z);
              x=cos(c*thet)*r1; y=sin(c*thet)*r1; glVertex3d(x,y,z);
       }
       glEnd();

       // draw front face (closer to viewer)
```

```
        glColor3f(red,green, blue);
        z=z_front;
        glBegin(GL_QUAD_STRIP);
        for(thet=theta_start; thet<=theta_stop;thet+=theta_inc){
                x=cos(c*thet)*r1; y=sin(c*thet)*r1; glVertex3d(x,y,z);
                x=cos(c*thet)*r2; y=sin(c*thet)*r2; glVertex3d(x,y,z);
        }
        glEnd();

        // draw upper face
        glColor3f(red,green, blue);
        glBegin(GL_QUAD_STRIP);
        for(thet=theta_start; thet<=theta_stop;thet+=theta_inc) {
                x=cos(c*thet)*r2; y=sin(c*thet)*r2;
                z=z_front; glVertex3d(x,y,z);
                z=z_back;  glVertex3d(x,y,z);
        }
        glEnd();

        // draw lower face
        glBegin(GL_QUAD_STRIP);
        for(thet=theta_start; thet<=theta_stop;thet+=theta_inc){
                x=cos(c*thet)*r1; y=sin(c*thet)*r1;
                z=z_back; glVertex3d(x,y,z);
                z=z_front; glVertex3d(x,y,z);
        }
        glEnd();

        // draw bottom end - Fixed by Tom Methven (31/08/2012)
        glColor3f(red,green, blue);
        glBegin(GL_POLYGON);
                x1=cos(c*theta_start)*r1; y1=sin(c*theta_start)*r1;
                x2=cos(c*theta_start)*r2; y2=sin(c*theta_start)*r2;

                glVertex3d(x1,y1,z_front);
                glVertex3d(x2,y2,z_front);
                glVertex3d(x2,y2,z_back);
                glVertex3d(x1,y1,z_back);
        glEnd();

        // draw top end
        glBegin(GL_POLYGON);
                x1=cos(c*theta_stop)*r1; y1=sin(c*theta_stop)*r1;
                x2=cos(c*theta_stop)*r2; y2=sin(c*theta_stop)*r2;

                glVertex3d(x1,y1,z_front);
                glVertex3d(x2,y2,z_front);
                glVertex3d(x2,y2,z_back);
                glVertex3d(x1,y1,z_back);
        glEnd();

        //vertical section of the D
        glPushMatrix();
        glScalef(4,1.8,5.8);
        glRotated(90,0,0,1);
        glTranslated(0.4,0.5,0.256);
        vertical(red,green,blue);
        glPopMatrix();

}
```

```
//******************************************
/*    Draw vertical segment of letter L and D*/
//******************************************

void Letters::vertical(float red,float green,float blue)
{

       //translation constants
       float x = -0.2;
       float y = 0;


       float a = -0.1;
       float b = -0.1;

       float c = x + a;
       float d = y + b;
       //3D
       float z = -0.1;



       //glClear(GL_COLOR_BUFFER_BIT);

       //FRONT
       //Column1
       glBegin(GL_POLYGON);
       glColor3f(red,green,blue);
       //Red
       glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
       glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
       glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //B
       glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
       glEnd();

       //3D Sides

       //Left Long Side
       glBegin(GL_POLYGON);
       glColor3f(red,green,blue);
       //Blue
       glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
       glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
       glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
       glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
       glEnd();

       //Right Long Side
       glBegin(GL_POLYGON);
       glColor3f(red,green,blue);
       //White
       glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //D
       glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //D
       glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //A
       glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //A
       glEnd();

       //Top Cap
       glBegin(GL_POLYGON);
       glColor3f(red,green,blue);
       //White
```

```
        glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
        glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
        glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
        glEnd();

        //Bottom Cap
        glBegin(GL_POLYGON);
        glColor3f(red,green,blue);
        //Green
        glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //B
        glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
        glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
        glEnd();

        //REAR
        //Column1
        glBegin(GL_POLYGON);
        glColor3f(red,green,blue);
        //Red
        glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
        glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
        glEnd();

        //glFlush();




}
//************************************
//Draw horizontal segment of letter L
//************************************
void Letters::horizontal(float red , float green , float blue)
{
        //translation constants
        float x = -0.2;
        float y = 0;


        float a = -0.1;
        float b = -0.1;

        float c = x + a;
        float d = y + b;
        //3D
        float z = -0.1;



        //glClear(GL_COLOR_BUFFER_BIT);

        //FRONT
        //Column1
        glBegin(GL_POLYGON);
        glColor3f(red,green,blue);
        //Red
        glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
        glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
        glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //B
        glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
        glEnd();

        //3D Sides

        //Left Long Side
        glBegin(GL_POLYGON);
        glColor3f(red,green,blue);
        //Blue
        glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
        glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
        glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
        glEnd();

        //Right Long Side
        glBegin(GL_POLYGON);
        glColor3f(red,green,blue);
        //White
        glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //D
        glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //A
        glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //A
        glEnd();

        //Top Cap
        glBegin(GL_POLYGON);
        glColor3f(red,green,blue);
        //White
        glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
        glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
        glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
        glEnd();

        //Bottom Cap
        glBegin(GL_POLYGON);
        glColor3f(red,green,blue);
        //Green
        glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //B
        glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
        glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
        glEnd();

        //REAR
        //Column1
        glBegin(GL_POLYGON);
        glColor3f(red,green,blue);
        //Red
        glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
        glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
        glEnd();

        glFlush();
}
#endif
```

Code Listing of Tyron Mc Donalds’ Initials (N, J, L)

Letter_N

```
/*******************************************************************************
********************************************
Author: Tyron Mc Donald
FileName: Letter N.h
Date: 2014-04-11
Operating system: Windows 7
Description : this file contains the points for creating the letter N
*******************************************************************************
********************************************/
#ifndef Letter_N_H
#define Letter_N_H

#include "Letter Class.h"

/**********************/
/*    Draw letter N    */
/**********************/

//These parameters change the position and size of the N
void Letters::letter_n(float translateX, float translateY, float translateZ,float
rotateAngle, float rotateX, float rotateY, float rotateZ ,float scaleX, float scaleY,
float scaleZ, float angle,bool Xaxis , bool Yaxis ,bool Zaxis)
{
       //translation constants
       float x = -0.2;
       float y = 0;


       float a = -0.1;
       float b = -0.1;

       float c = x + a;
       float d = y + b;
       //3D
       float z = -0.1;


       glPushMatrix();
       glTranslatef(translateX, translateY, translateZ);
       glRotatef(rotateAngle, rotateX, rotateY, rotateZ);
       glScalef(scaleX, scaleY, scaleZ);

       glPushMatrix();
       glTranslatef(0,0,0);
       glRotatef(angle, Xaxis, Yaxis, Zaxis);
       glScalef(1,1,1);
       //FRONT
       //Column1
       glBegin(GL_POLYGON);
       glColor3f(0.22 ,0.29 ,0.63);
       glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
       glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
       glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //B
       glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
       glEnd();

       glBegin(GL_POLYGON);
       glColor3f(0.22 ,0.29 ,0.63);
       glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
```

```
glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
glVertex3f(0.8 + x + a, -0.9 + y + b, z); //B
glVertex3f(0.5 + x + a, -0.9 + y + b, z); //A
glEnd();

//Column2
glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(0.5 + x + a, 0.1 + y + b, z); //D
glVertex3f(0.8 + x + a, 0.1 + y + b, z); //C
glVertex3f(0.8 + x + a, -0.9 + y + b, z); //B
glVertex3f(0.5 + x + a, -0.9 + y + b, z); //A
glEnd();


//3D Sides
glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
glEnd();


glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(0.8 + x + a, 0.1 + y + b, z); //C
glVertex3f(0.8 + x + a, 0.1 + y + b, -0.4); //C
glVertex3f(0.8 + x + a, -0.9 + y + b, -0.4); //B
glVertex3f(0.8 + x + a, -0.9 + y + b, z); //B
glEnd();

glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
glEnd();

glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(-0.1 + x + a, -0.9 + y + b, z); //B
glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //B
glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
glVertex3f(-0.4 + x + a, -0.9 + y + b, z); //A
glEnd();


glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(0.5 + x + a, 0.1 + y + b, z); //D
glVertex3f(0.5 + x + a, 0.1 + y + b, -0.4); //D
glVertex3f(0.8 + x + a, 0.1 + y + b, -0.4); //C
glVertex3f(0.8 + x + a, 0.1 + y + b, z); //C
glEnd();


glBegin(GL_POLYGON);
glColor3f(0.22 ,0.29 ,0.63);
glVertex3f(0.8 + x + a, -0.9 + y + b, z); //B
```

```
        glVertex3f(0.8 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(0.5 + x + a, -0.9 + y + b, -0.4); //A
        glVertex3f(0.5 + x + a, -0.9 + y + b, z); //A
        glEnd();


        glBegin(GL_POLYGON);
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(-0.4 + x + a, 0.1 + y + b, z); //D
        glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(0.5 + x + a, -0.9 + y + b, -0.4); //A
        glVertex3f(0.5 + x + a, -0.9 + y + b, z); //A
        glEnd();


        glBegin(GL_POLYGON);
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(-0.1 + x + a, 0.1 + y + b, z); //C
        glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
        glVertex3f(0.8 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(0.8 + x + a, -0.9 + y + b, z); //B
        glEnd();

        //REAR
        //Column1
        glBegin(GL_POLYGON);
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
        glVertex3f(-0.1 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(-0.4 + x + a, -0.9 + y + b, -0.4); //A
        glEnd();

        glBegin(GL_POLYGON);
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(-0.4 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(-0.1 + x + a, 0.1 + y + b, -0.4); //C
        glVertex3f(0.8 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(0.5 + x + a, -0.9 + y + b, -0.4); //A
        glEnd();

        //Column2
        glBegin(GL_POLYGON);
        glColor3f(0.22 ,0.29 ,0.63);
        glVertex3f(0.5 + x + a, 0.1 + y + b, -0.4); //D
        glVertex3f(0.8 + x + a, 0.1 + y + b, -0.4); //C
        glVertex3f(0.8 + x + a, -0.9 + y + b, -0.4); //B
        glVertex3f(0.5 + x + a, -0.9 + y + b, -0.4); //A
        glEnd();

        glPopMatrix();
        glPopMatrix();

        glFlush();
}


#endif
```

Letter_J

```
/*************************************************************************************
********************************************
Author: Tyron Mc Donald
FileName: Letter J.h
Date: 2014-04-11
Operating system: Windows 7
Description : this file contains the points for creating the letter J
*************************************************************************************
********************************************/
#ifndef Letter_J_H
#define Letter_J_H

#include "Letter Class.h"


/**********************/
/*    Draw letter J    */
/**********************/

//These parameters change the position and size of the J
void Letters::letter_j(float translateX, float translateY, float translateZ,float
rotateAngle, float rotateX, float rotateY, float rotateZ ,float scaleX, float scaleY,
float scaleZ)
{
       //translation constants
       float x = -0.2;
       float y = 0;


       float a = -0.1;
       float b = -0.1;

       float c = x + a;
       float d = y + b;
       //3D
       float zf = 0.1;
       float zr = 0.0;
       float zt = 0.1;



       //horizontal
       glPushMatrix();
       glTranslatef(translateX, translateY, translateZ);
       glRotatef(rotateAngle, rotateX, rotateY, rotateZ);
       glScalef(scaleX, scaleY, scaleZ);

       glPushMatrix();
       glScalef(2,2,1.9);
    glRotatef(0,0,0,0);
    glTranslatef(-0.1,0.5,0.0);

       glBegin(GL_POLYGON);
       glColor3f(0, 0, 125);
       glVertex3f(-0.1 + x, 0.1 + y, zr); //D
       glVertex3f(0.9 + x, 0.1 + y, zr); //C
       glVertex3f(0.9 + x, -0.1 + y, zr); //B
       glVertex3f(-0.1 + x, -0.1 + y, zr); //A
```

```
        glEnd();


    glBegin(GL_POLYGON);

        glColor3f(0, 0, 125);
        glVertex3f(-0.1 + x , 0.1 + y , zf); //D
        glVertex3f(0.9 + x , 0.1 + y , zf); //C
        glVertex3f(0.9 + x , -0.1 + y , zf); //B
        glVertex3f(-0.1 + x , -0.1 + y , zf); //A

        glEnd();

//TOP
        glBegin(GL_POLYGON);
        //glPushMatrix();
        //glScalef(0.5, 0.5, 0.5);
        glColor3f(0 , 0, 125);
        glVertex3f(0.7, 0.1, zt); //B
        glVertex3f(0.7 , 0.1, 0.0); //C
        glVertex3f(-0.3 , 0.1, 0.0); //D
        glVertex3f(-0.3 , 0.1, zt); //A
        //glPopMatrix();
        glEnd();

//Bottom: Medium Base
        glBegin(GL_POLYGON);

        glColor3f(0 , 0, 125);
        glVertex3f(0.7, -0.1, zt); //B
        glVertex3f(0.7 , -0.1, 0.0); //C
        glVertex3f(-0.3 , -0.1, 0.0); //D
        glVertex3f(-0.3 , -0.1, zt); //A

        glEnd();

//upper side: right
        glBegin(GL_POLYGON);
        glColor3f(0, 0, 125);
        glVertex3f(0.9 + x , 0.1 + y , zf); //D
        glVertex3f(0.9 + x , 0.1 + y , 0.0);//C
        glVertex3f(0.9 + x , -0.1 + y , 0.0);//B
        glVertex3f(0.9 + x , -0.1 + y , zf); //A
        glEnd();

        //upper side: left
        glBegin(GL_POLYGON);
        glColor3f(0, 0, 125);
        glVertex3f(-0.1 + x , 0.1 + y , zf); //D
        glVertex3f(-0.1 + x , 0.1 + y , 0.0);//C
        glVertex3f(-0.1 + x , -0.1 + y , 0.0);//B
        glVertex3f(-0.1 + x , -0.1 + y , zf); //A
        glEnd();
        glPopMatrix();

        //verticle
        glPushMatrix();
        glScalef(1,2,1.9);
        glRotatef(0,0,0,0);
        glTranslatef(0,0.5,0.0);
        glBegin(GL_POLYGON);
        glColor3f(0, 0, 125);
        glVertex3f(0.3 + x, 0.1 + y, zr); //D
        glVertex3f(0.5 + x, 0.1 + y, zr); //C
        glVertex3f(0.5 + x, -0.9 + y, zr); //B
        glVertex3f(0.3 + x, -0.9 + y, zr); //A


        glBegin(GL_POLYGON);
        glColor3f(0, 0, 125);
        glVertex3f(0.3 + x , 0.1 + y , zf); //D
        glVertex3f(0.5 + x , 0.1 + y , zf); //C
        glVertex3f(0.5 + x , -0.9 + y , zf); //B
        glVertex3f(0.3 + x , -0.9 + y , zf); //A
        glEnd();

//Bottom : Lower Base
        glBegin(GL_POLYGON);
        glColor3f(0 , 0 , 125);
        glVertex3f(0.5 + x , -0.9 + y , zf); //B
        glVertex3f(0.5 + x , -0.9 + y , 0.0); //C
        glVertex3f(0.3 + x , -0.9 + y , 0.0); //D
        glVertex3f(0.3 + x , -0.9 + y , zf); //A

        glEnd();

//lower side : left
        glBegin(GL_POLYGON);
        glColor3f(0, 0, 125);
        glVertex3f(0.3 + x , -0.1 + y , zf); //D
        glVertex3f(0.3 + x , -0.1 + y , 0.0);//C
        glVertex3f(0.3 + x , -0.9 + y , 0.0);//B
        glVertex3f(0.3 + x , -0.9 + y , zf); //A
        glEnd();

//lower side : right
        glBegin(GL_POLYGON);
        glColor3f(0, 0, 125);
        glVertex3f(0.5 + x , -0.1 + y , zf); //D
        glVertex3f(0.5 + x , -0.1 + y , 0.0);//C
        glVertex3f(0.5 + x , -0.9 + y , 0.0);//B
        glVertex3f(0.5 + x , -0.9 + y , zf); //A
        glEnd();
        glPopMatrix();

        //J curve
        glPushMatrix();
        glScalef(0.4,0.4,0.4);
        glRotatef(180,1,0,0);
        glTranslatef(-1.25,2,-0.2);
        curveofj(0.53,1.5, 2.0 , 0.0,180, 5.0);
        glPopMatrix();


        glPopMatrix();

        glFlush();


}
```

```
void Letters::curveofj(double depth, double r1, double r2, double theta_start, double
theta_stop, double theta_inc)
{
       //Function to draw 3D curve
       // depth = depth centred round z=0
       // r1 = inner radius
       // r2 = outer radius
       // theta_start = start angle in degrees measured from x-axis
       // theta_stop = similar

       double x, y, x1, x2, y1, y2, z, thet, z_front, z_back;
       int i=0;
       double radius=1.5, c=3.14159/180.0;
       z_front=depth/2; z_back=-depth/2;

       // draw rear face (away from viewer)
       glColor3f(0, 0, 125);
       z=z_back;
       glBegin(GL_QUAD_STRIP);
       for(thet=theta_start; thet<=theta_stop;thet+=theta_inc) {
              x=cos(c*thet)*r2; y=sin(c*thet)*r2; glVertex3d(x,y,z);
              x=cos(c*thet)*r1; y=sin(c*thet)*r1; glVertex3d(x,y,z);
       }
       glEnd();

       // draw front face (closer to viewer)
       glColor3f(0, 0, 125);
       z=z_front;
       glBegin(GL_QUAD_STRIP);
       for(thet=theta_start; thet<=theta_stop;thet+=theta_inc){
              x=cos(c*thet)*r1; y=sin(c*thet)*r1; glVertex3d(x,y,z);
              x=cos(c*thet)*r2; y=sin(c*thet)*r2; glVertex3d(x,y,z);
       }
       glEnd();

       // draw upper face
       glColor3f(0, 0, 125);
       glBegin(GL_QUAD_STRIP);
       for(thet=theta_start; thet<=theta_stop;thet+=theta_inc) {
              x=cos(c*thet)*r2; y=sin(c*thet)*r2;
              z=z_front; glVertex3d(x,y,z);
              z=z_back;  glVertex3d(x,y,z);
       }
       glEnd();

       // draw lower face
       glBegin(GL_QUAD_STRIP);
       for(thet=theta_start; thet<=theta_stop;thet+=theta_inc){
              x=cos(c*thet)*r1; y=sin(c*thet)*r1;
              z=z_back; glVertex3d(x,y,z);
              z=z_front; glVertex3d(x,y,z);
       }
       glEnd();

       // draw bottom end - Fixed by Tom Methven (31/08/2012)
       glColor3f(0, 0, 125);
       glBegin(GL_POLYGON);
              x1=cos(c*theta_start)*r1; y1=sin(c*theta_start)*r1;
              x2=cos(c*theta_start)*r2; y2=sin(c*theta_start)*r2;

              glVertex3d(x1,y1,z_front);
              glVertex3d(x2,y2,z_front);
```

```
		glVertex3d(x2,y2,z_back);
		glVertex3d(x1,y1,z_back);
	glEnd();

	// draw top end
	glBegin(GL_POLYGON);
		x1=cos(c*theta_stop)*r1; y1=sin(c*theta_stop)*r1;
		x2=cos(c*theta_stop)*r2; y2=sin(c*theta_stop)*r2;

		glVertex3d(x1,y1,z_front);
		glVertex3d(x2,y2,z_front);
		glVertex3d(x2,y2,z_back);
		glVertex3d(x1,y1,z_back);
	glEnd();
}

#endif
```

Letter_L

```
/*************************************************************************************
********************************************
Author: Tyron Mc Donald
FileName: Letter L.h
Date: 2014-04-11
Operating system: Windows 7
Description : this file contains the points for creating the letter L
*************************************************************************************
********************************************/
#ifndef Letter_L_H
#define Letter_L_H

#include "Letter Class.h"

//These parameters change the position , size , angle and color of the L
void Letters::letter_l(float translateX, float translateY, float translateZ,float
rotateAngle, float rotateX, float rotateY, float rotateZ ,float scaleX, float scaleY,
float scaleZ , float red , float green ,float blue)
{
       //VERTICAL SEGMENT OF THE L
       glPushMatrix();
       glTranslatef(translateX, translateY, translateZ);
       glRotatef(rotateAngle, rotateX, rotateY, rotateZ);
       glScalef(scaleX, scaleY, scaleZ);
       vertical(red,green,blue);

       //HORIZONTAL SEGMENT OF THE L
       glPushMatrix();
              glScalef(1,1,1);
              glRotatef(90,0,0,1);
              glTranslatef(-0.3,0.7,0);
              horizontal(red,green,blue);
              glPopMatrix();
       glPopMatrix();
}

#endif
```

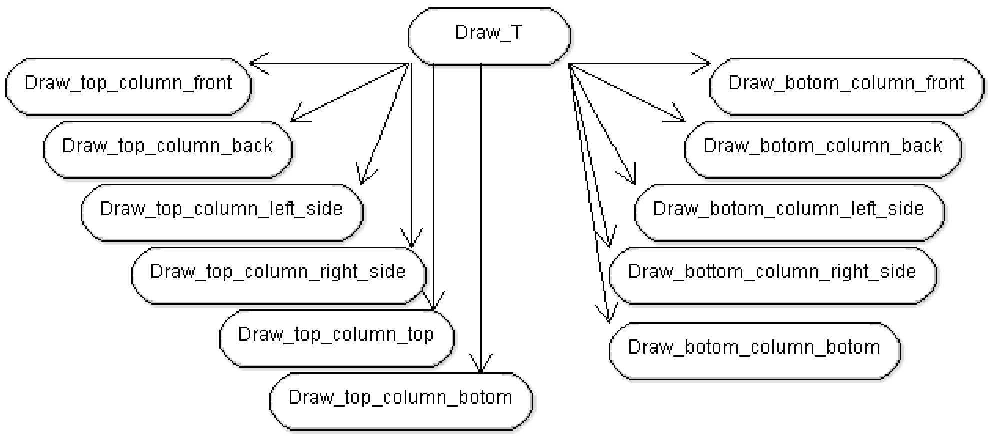
Scene Graphs

Letters designed by Nelio Lucas

Letter T

Parameters required by function

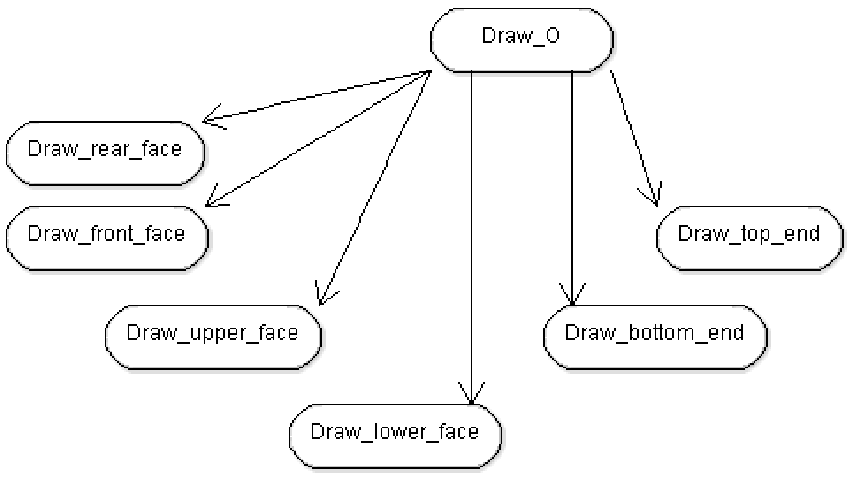
- `float translateX`
- `float translateY`
- `float translateZ`
- `float rotateAngle`
- `float rotateX`
- `float rotateY`
- `float rotateZ`
- `float scaleX`
- `float scaleY`
- `float scaleZ`



Letter O

Parameters required by function

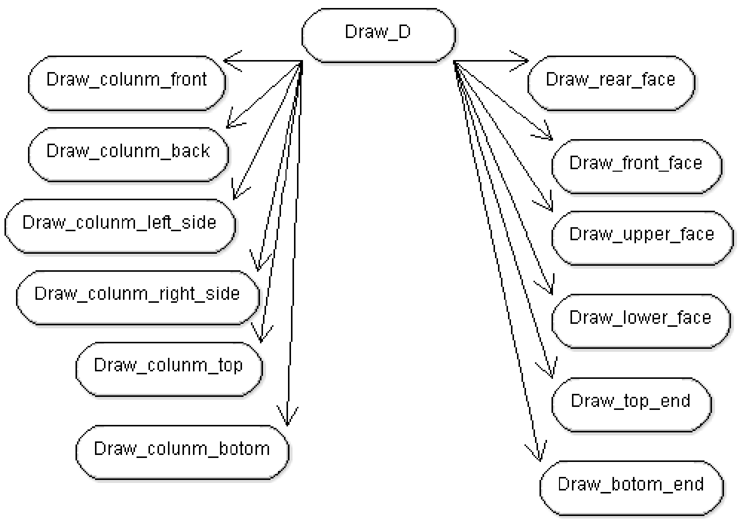
- `float translateX`
- `float translateY`
- `float translateZ`
- `float rotateAngle`
- `float rotateX`
- `float rotateY`
- `float rotateZ`
- `float scaleX`
- `float scaleY`
- `float scaleZ`
- `float red`
- `float green`
- `float blue`
- `float angle`



Letter D

Parameters required by function

```
-   float translateX
-   float translateY
-   float translate
-   float rotateAngle
-   float rotateX
-   float rotateY
-   float rotateZ
-   float scaleX
-   float scaleY
-   float scaleZ
-   float red
-   float green
-   float blue
-   float angle
```

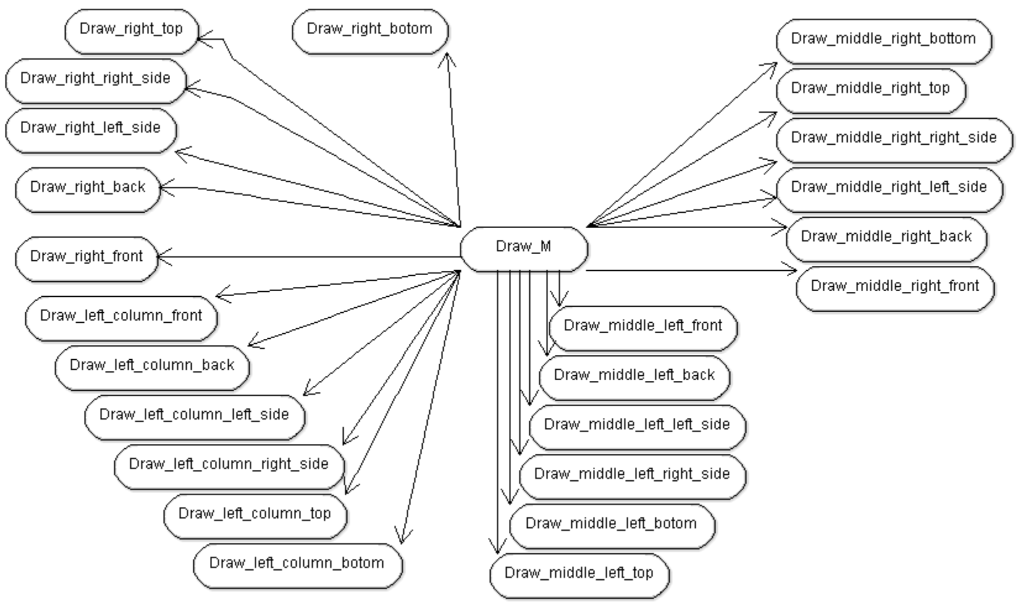


Letters designed by Tejas Dwarkaram

Letter M

Parameters required by function

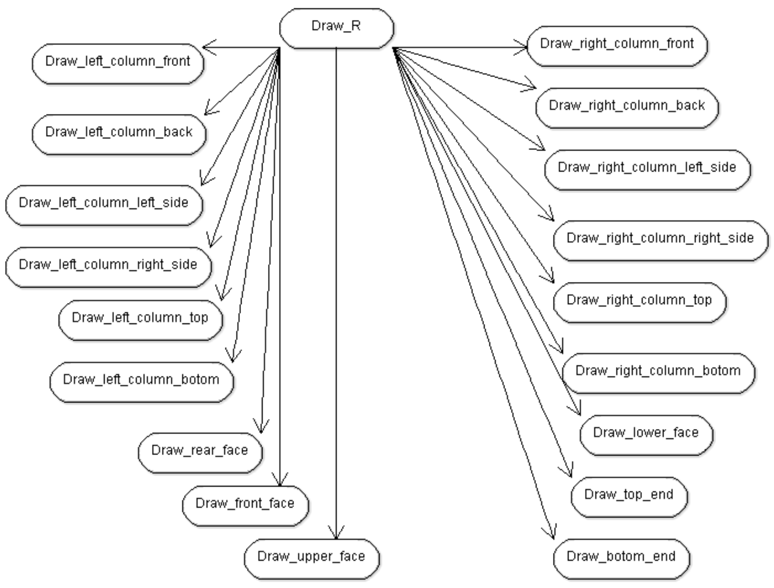
- `float translateX`
- `float translateY`
- `float translateZ`
- `float rotateAngle`
- `float rotateX`
- `float rotateY`
- `float rotateZ`
- `float scaleX`
- `float scaleY`
- `float scaleZ`



Letter R

Parameters required by function

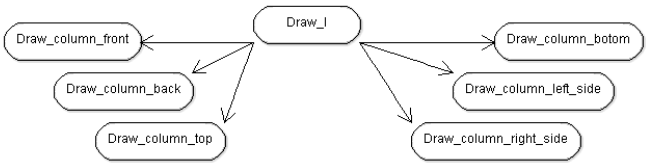
- `float translateX`
- `float translateY`
- `float translateZ`
- `float rotateAngle`
- `float rotateX`
- `float rotateY`
- `float rotateZ`
- `float scaleX`
- `float scaleY`
- `float scaleZ`
- `float red`
- `float green`
- `float blue`
- `float angle`
- `bool Xaxis`
- `bool Yaxis`
- `bool Zaxis`



Letter I

Parameters required by function

- `float translateX`
- `float translateY`
- `float translate`
- `float rotateAngle`
- `float rotateX`
- `float rotateY`
- `float rotateZ`
- `float scaleX`
- `float scaleY`
- `float scaleZ`

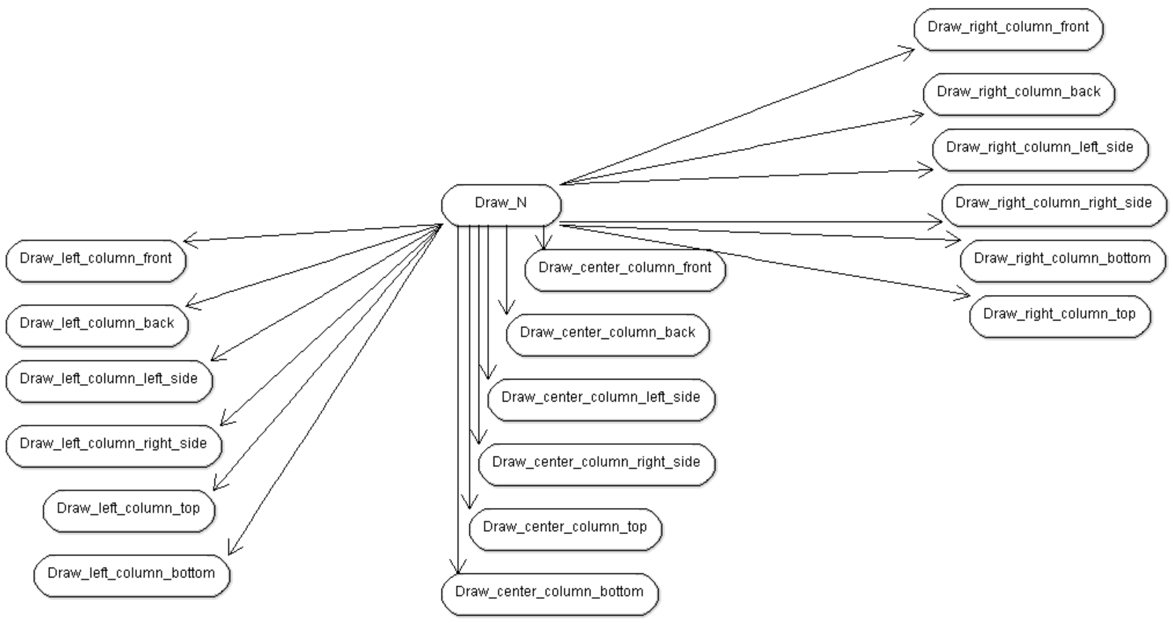


Letters designed by Tyron Mc Donald

Letter N

Parameters required by function

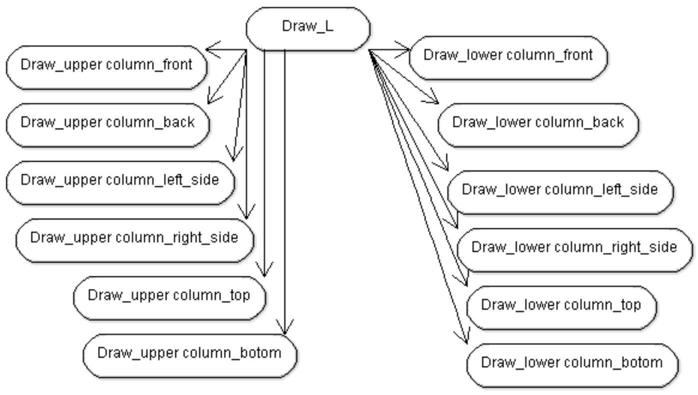
- `float translateX`
- `float translateY`
- `float translateZ`
- `float rotateAngle`
- `float rotateX`
- `float rotateY`
- `float rotateZ`
- `float scaleX`
- `float scaleY`
- `float scaleZ`
- `float angle`
- `bool Xaxis`
- `bool Yaxis`
- `bool Zaxis`



Letter L

Parameters required by function

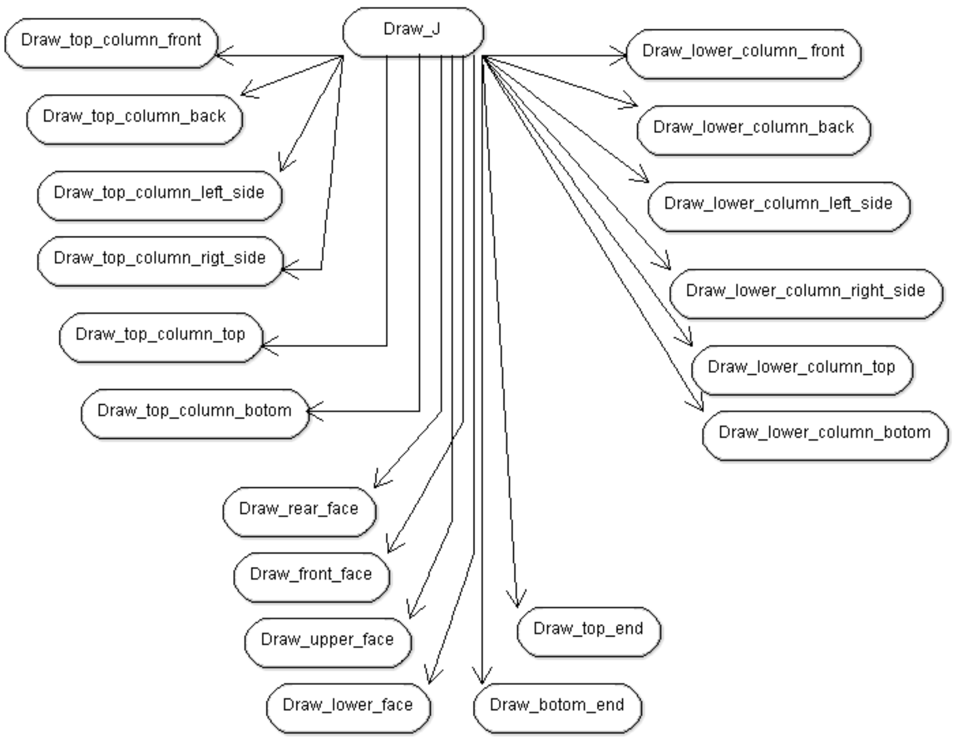
- `float translateX`
- `float translateY`
- `float translateZ`
- `float rotateAngle`
- `float rotateX`
- `float rotateY`
- `float rotateZ`
- `float scaleX`
- `float scaleY`
- `float scaleZ`
- `float red`
- `float green`
- `float blue`



Letter J

Parameters required by function

- `float translateX`
- `float translateY`
- `float translate`
- `float rotateAngle`
- `float rotateX`
- `float rotateY`
- `float rotateZ`
- `float scaleX`
- `float scaleY`
- `float scaleZ`



Code Listing for Princess Luna.cpp

```
/*****************************************************************************************
********************************************
Author: Nelio Lucas , Tyron Mc Donald , Tejas Dwarkaram
FileName: Dragon_Source.cpp
Date: 2014-04-11
Operating system: Windows 7
Description : this file contains the main function that displays the 3D dragon onto 
the screen
*****************************************************************************************
********************************************/
#include "Drawing class.h"



//======================================================
// MAIN PROGRAM
//======================================================
int main(int argc, char **argv)
{
       Drawing o;
       // Create and Name window
       // Add Display , Mouse CallBacks and Keyboard Callback
    glutInit(&argc, argv);
       glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
       glutInitWindowSize(800, 600);
       glutCreateWindow("Princess Luna");
       glutReshapeFunc(o.reshapeCallBack);
       glutDisplayFunc(o.displayCallback);
       glutMouseFunc(o.mouseClickCallback);
       glutMotionFunc(o.mouseMotionCallback);
       glutKeyboardFunc(o.keyboardCallBack);
    o.menu();

       glClearColor(0.0, 0.0, 0.34, 1.0);
       glColor3f(0.0, 0.0, 12.0);
       glEnable(GL_DEPTH_TEST);
       glutMainLoop();
       return 0;
}
```

Code Listing for Top Level Scene Graph

```
/*******************************************************************************
Author: Nelio Lucas
FileName: TitleFrame.h
Date:2013-01-22
Operating System: Windows 7
Description:this file contains the framing function
*******************************************************************************/
#ifndef GUARD_frame_h
#define GUARD_frame_h
#include <string>
#include <iostream>


using std::string;
using std::cout;
using std::cin;
using std::endl;

/*******************************************************************************
                                 * frame *
                     this function frames  any string value
******************************************************************************/
string frame(string& v)
{
     int maxlen = 0;
     maxlen = v.size();
     system("cls");
     string star(maxlen,'*' );
     //top frame
     cout << star << endl;

     for(int i = 0; i != maxlen; i++)
     {

        //display each character
         cout << v[i];
     }

     cout << endl;
     //bottom frame
     cout << star;
     return v;
}

#endif
```

```
/*****************************************************************************************
********************************************
Author: Nelio Lucas , Tyron Mc Donald , Tejas Dwarkaram
FileName: Drawing class.h
Date: 2014-04-11
Operating system: Windows 7
Description : this file contains all the class and all the function declarations used
for creating the body parts that builds
the 3D model of princess luna
*****************************************************************************************
********************************************/

#include "Body.h"
#include "Environment.h"
#include "Left front leg.h"
#include "Left hind leg.h"
#include "Right front leg.h"
#include "Right Hind leg.h"
#include "Lower left wing.h"
#include "Lower right wing.h"
#include "Luna.h"
#include "Tail.h"
#include "NeckHead.h"

#ifndef Drawing_class_H
#define Drawing_class_H

#include "Letter Class.h"
#include "TitleFrame.h"



//======================================================
// GLOBAL VARIABLES
//======================================================

//floor and grid display
int dont_display_floor=0;
int x_y_display=0, y_z_display=0, x_z_display=0;

//for view control
static float G_theta[3]; // View X,Y,Z
static float zoom=0.25;

bool MousePressed; // Used in Mouse Drag to change the Viewpoint
float pitch0, yaw0;
int mouseX0, mouseY0;

//rotation and translation of body parts
static float neck_rot=0.0;
static float head_rot= 0.0;
static float upper_tail_rot=0.0;
static float middle_tail_rot=0.0;
static float lower_tail_rot =0.0;
static float left_lower_wing_rot = 0.0;
static float left_middle_wing_rot = 0.0;
static float left_upper_wing_rot =0.0;
static float right_lower_wing_rot = 0.0;
static float right_middle_wing_rot= 0.0;
static float right_upper_wing_rot = 0.0;
static float legmove = 0.0;
static float bodymove = 0.0;
```

```
bool allrot = false;
bool legs = false;
bool body = false;
bool neck_u= false;
bool head = false;
bool left_low_wing = false;
bool left_mid_wing = false;
bool left_up_wing = false;
bool tail = false;
bool right_low_wing = false;
bool right_mid_wing = false;
bool right_up_wing = false;
//displaying and removing floor and axis
bool ground = true;
bool axis = true;

class Drawing:Letters
{
   public:
            //drawing the scene(luna) , body , neck and head
            static void
luna(float,float,float,float,float,float,float,float,float,float, float, float, float
);
            static void draw_body(float);
            static void draw_neck(float,float,float);
            static void draw_head(float);


          //LEFT HIND LEG
           static void draw_upper_hind_left_leg(float);
           static void draw_lower_hind_left_leg();

          //RIGHT HIND LEG
              static void draw_upper_hind_right_leg(float);
              static void draw_lower_hind_right_leg();

          //LEFT FRONT LEG
               static void draw_upper_front_left_leg(float);
               static void  draw_lower_front_left_leg();

          //RIGHT FRONT LEG
               static void draw_upper_front_right_leg(float);
               static void draw_lower_front_right_leg();

          //TAIL
                     static void draw_upper_tail(float,float,float);
                     static void draw_middle_tail(float);
                     static void draw_lower_tail();

          //LEFT WING
                     static void draw_lower_left_wing(float, float,float);
               static void draw_middle_left_wing(float,float);
                     static void draw_upper_left_wing(float);

          //RIGHT WING
                     static void draw_lower_right_wing(float,float,float);
                     static void draw_middle_right_wing(float,float);
                     static void draw_upper_right_wing(float);

       static void drawFloor();
          static void drawMoon();
          static void viewControl();
          static void displayCallback();
          static void reshapeCallBack(int w , int h);
          static void mouseMotionCallback(int x , int y);
          static void mouseClickCallback(int button , int state , int x , int y);
          static void gridLines(void);
          static void idleCallBack();
          static void keyboardCallBack(unsigned char, int , int );
          static void menu();



};

/**********************/
/*    Draw Menu        */
/**********************/

void Drawing::menu()
{
       string title = "3D model of Princess Luna";
       frame(title);
       cout << endl << endl;
       cout << "Left Mouse Button & Drag - Changes the View.\n" << endl;
       cout <<"Key 'A/a' : Animate/Stop Animation "<< endl;
       cout <<"Key 'Z'   : Zoom in." << endl;
       cout <<"Key 'X'   : Zoom out" << endl;
       cout <<"Key 'G/g' : Display axis/Remove axis" << endl;
       cout <<"Key 'F/f' : Display floor/Remove floor " << endl;
       cout <<"Key 'M'   : Redisplay Menu " << endl;
       cout <<"Key 'E'   : Exit the program " << endl;
       //console color
       system("COLOR 29");

}


//=======================================================
// KEYBOARD CALLBACK ROUTINE
//=======================================================
void Drawing::keyboardCallBack(unsigned char key, int x , int y)
{

       switch(key)
       {
       case 'A':
              glutIdleFunc(idleCallBack);printf("Keyboard call back: key=%c, x=%d,
y=%d \n", key, x, y);
              cout << "\n Animation ON" << endl;
       break;
       case 'a':
              glutIdleFunc(NULL);printf("Keyboard call back: key=%c, x=%d, y=%d \n",
key, x, y);
              cout << "\n Animation OFF" << endl;
       break;
       case 'Z':
              zoom += 0.1;printf("Keyboard call back: key=%c, x=%d, y=%d \n", key, x,
y);
              cout << "zoomed in by " << zoom << " "<< "units" << endl;
       break;
       case 'X':
              zoom -= 0.1;printf("Keyboard call back: key=%c, x=%d, y=%d \n", key, x,
y);
```

```
			cout << "zoomed out by "<< zoom <<" units" << endl;
		break;
		case 'G':
			axis = true;
			cout << "\n Grid ON" << endl;
		break;
		case 'g':
			axis = false;
			cout << "\n Grid OFF" << endl;
		break;
		case 'F':
			ground = true;
			cout << "\n Floor ON" << endl;
		break;
		case 'f' :
			ground = false;
			cout << "\n Floor OFF" << endl;
		break;
		case 'E':
			exit(0);
		break;
		case 'M':
			system("CLS");
			menu();
		break;

		default:
			cout << "\n Wrong Keyboard entry , please enter A, a, Z, X, G, g, F, f,
E or M" << endl;
		}

		glutPostRedisplay();
}

//=======================================================
// VIEW CONTROL ROUTINE
//=======================================================
void Drawing::viewControl()
{
		//Resets the matrix
		glLoadIdentity();

		//Rotate entire model
		glRotatef(G_theta[0], 1.0, 0.0, 0.0);
		glRotatef(G_theta[1], 0.0, 1.0, 0.0);
		glRotatef(G_theta[2], 0.0, 0.0, 1.0);

		//zoom (NB glOrtho projection)
		glScalef(zoom,zoom,zoom);
}

//=======================================================
// DISPLAY CALLBACK ROUTINE
//=======================================================
void Drawing::displayCallback(void)
{
		// display callback,
		glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
		//movement of scene
		viewControl();

		//draws scene
```

```
       glEnable(GL_DEPTH_TEST);
       luna(neck_rot,head_rot ,  upper_tail_rot ,  middle_tail_rot ,  lower_tail_rot ,
left_lower_wing_rot ,  left_middle_wing_rot,  left_upper_wing_rot,
right_lower_wing_rot ,  right_middle_wing_rot, right_upper_wing_rot, legmove ,
bodymove );
       glFlush();
       glutSwapBuffers();
}

//=======================================================
// DISPLAY RESHAPE CALLBACK ROUTINE
//=======================================================
void Drawing::reshapeCallBack(int w, int h)
{
    glViewport(0, 0, w, h);
       gluPerspective(90,60,1,100);
    glMatrixMode(GL_PROJECTION);
              glLoadIdentity();
              if (w <= h)
                     glOrtho(-2.0, 2.0, -2.0 * (float) h / (float) w,
                            2.0 * (float) h / (float) w, -10.0, 10.0);
              else
                     glOrtho(-2.0 * (float) w / (float) h,
                            2.0 * (float) w / (float) h, -2.0, 2.0, -10.0, 10.0);
    glMatrixMode(GL_MODELVIEW);
}
//=======================================================
// IDLE CALLBACK ROUTINE
//=======================================================
void Drawing::idleCallBack()
{
       //LEG movement
       if (legs)
       {
              legmove = legmove + 0.005;
       }
       else
       {
              legmove = legmove - 0.005;
       }
       if (legmove<= 0.2)
       {
              legs = true;
       }
       if (legmove>= 0.4)
       {
              legs= false;
       }


       //UPPER TAIL movement
       if(allrot)
       {
              upper_tail_rot += 0.1;
    }
       else
       {
              upper_tail_rot -= 0.1;
    }

       if (upper_tail_rot <= 0)
       {
              allrot = true;
       }
       if (upper_tail_rot >= 5)
       {
              allrot= false;
       }

       //MIDDLE TAIL movement
       if(!tail)
       {
              middle_tail_rot += 0.08;
    }
       else
       {
              middle_tail_rot -= 0.08;
    }

       if (middle_tail_rot > -0.1)
       {
              tail = true;
       }
       if (middle_tail_rot < 0.25)
       {
              tail= false;
       }

       //BODY movement
       if(body)
       {
              bodymove += 0.005;
    }
       else
       {
              bodymove -= 0.005;
    }

       if (bodymove <= 0)
       {
              body = true;
       }
       if (bodymove >= 0.2)
       {
              body= false;
       }

       //NECK movement
       if (neck_u)
       {
              neck_rot = neck_rot + 0.03;
       }
       else
       {
              neck_rot = neck_rot - 0.03;
       }
       if (neck_rot<= 0.0)
       {
              neck_u = true;
       }
       if (neck_rot>= 3.5)
       {
              neck_u= false;
       }
```

```
//HEAD movement
if (head)
{
       head_rot = head_rot + 0.03;
}
else
{
       head_rot = head_rot - 0.03;
}
if (head_rot<= -0.4)
{
       head = true;
}
if (head_rot>= 2)
{
       head= false;
}

//LEFT WING LOWER movement
if (left_low_wing)
{
       left_lower_wing_rot = left_lower_wing_rot + 0.5;
}
else
{
       left_lower_wing_rot = left_lower_wing_rot - 0.5;
}
if (left_lower_wing_rot<= -13)
{
       left_low_wing = true;
}
if (left_lower_wing_rot>= 15)
{
       left_low_wing= false;
}

//LEFT WING MIDDLE movement
if (left_mid_wing )
{
       left_middle_wing_rot = left_middle_wing_rot + 0.4;
}
else
{
       left_middle_wing_rot = left_middle_wing_rot- 0.4;
}
if (left_middle_wing_rot <= -10)
{
       left_mid_wing = true;
}
if (left_middle_wing_rot >= 15)
{
       left_mid_wing= false;
}

//LEFT WING UPPER movement
if (left_up_wing)
{
       left_upper_wing_rot = left_upper_wing_rot + 0.45;
}
else
{
        left_upper_wing_rot = left_upper_wing_rot- 0.45;
}
if (left_upper_wing_rot<= -13)
{
        left_up_wing = true;
}
if (left_upper_wing_rot>= 15)
{
        left_up_wing= false;
}

//RIGHT WING LOWER movement
if (!left_low_wing)
{
        right_lower_wing_rot = right_lower_wing_rot + 0.5;
}
else
{
        right_lower_wing_rot = right_lower_wing_rot - 0.5;
}
if (right_lower_wing_rot>= 13)
{
        right_low_wing = true;
}
if (left_lower_wing_rot<= -15)
{
        right_low_wing= false;
}

//RIGHT WING MIDDLE movement
if (!right_mid_wing )
{
        right_middle_wing_rot = right_middle_wing_rot + 0.4;
}
else
{
        right_middle_wing_rot = right_middle_wing_rot- 0.4;
}
if (right_middle_wing_rot >= 13)
{
        right_mid_wing = true;
}
if (right_middle_wing_rot <= -15)
{
        right_mid_wing= false;
}

//RIGHT WING  UPPER movement
if (!right_up_wing)
{
        right_upper_wing_rot = right_upper_wing_rot + 0.45;
}
else
{
        right_upper_wing_rot = right_upper_wing_rot- 0.45;
}
if (right_upper_wing_rot>= 13)
{
        right_up_wing = true;
}
if (right_upper_wing_rot<= -15)
{
```

```
            right_up_wing= false;
        }

        glutPostRedisplay();
}

//=========================================================
// MOUSE CALLBACK ROUTINES
//=========================================================

void Drawing::mouseMotionCallback(int x , int y)
{
        // Called when the Mouse is moved with left button down
        G_theta[0] = pitch0 + (y - mouseY0);
    G_theta[1] = yaw0 + (x - mouseX0);
        glutPostRedisplay();
}

void Drawing::mouseClickCallback(int button, int state, int x, int y)
{
        // Called on button press or release
    switch (state)
    {
            case GLUT_DOWN:
                MousePressed = true;
                pitch0 = G_theta[0];
                yaw0 = G_theta[1];
                mouseX0 = x; mouseY0 = y;
                break;
            default:
            case GLUT_UP:
                MousePressed = false;
                break;
    }
}
#endif
```

```
/*************************************************************************************
********************************************
Author: Nelio Lucas
Date: 2014-04-11
Operating system: Windows 7
IDE : Microsoft Visual Studio 2012 for Windows Desktop
Description : this file contains the drawMoon , DrawFloor and gridLines function
definitions
*************************************************************************************
********************************************/

#ifndef ENVIRONMENT_H
#define ENVIRONMENT_H

#include "Drawing Class.h"

/***********************/
/*    Draw Moon*/
/***********************/
void Drawing::drawMoon()
{
       for(int x =0; x < 360 ; x++)
              letter_o(8,8,-1.5,x,0,1,0,0.4,0.4,0.4,255.0 ,255.0 ,0);

}

/***********************/
/*    Draw Floor   */
/***********************/

void Drawing::drawFloor()
{

       glPushMatrix();
              glTranslatef(0, -1.1, 0); //draw slightly below y=0 so we can see grid
              glScalef(ground,ground,ground); //variables for adding and removing
              glBegin(GL_POLYGON);
                     glColor3f(.75,.75,.75);
                     glVertex3f(-5,0,5);
                     glVertex3f(-5,0,-5);
                     glVertex3f(5,0,-5);
                     glVertex3f(5,0,5);
              glEnd();
       glPopMatrix();
}

/***********************/
/*    Draw Axis        */
/***********************/
void Drawing::gridLines(void)
{

       glPushMatrix();
       glScalef(axis,axis,axis);
       float offset; //int gd;
       glBegin(GL_LINES);
              glColor3f(255, 255, 255);
              glVertex3f(-20, 0, 0);
              glVertex3f(+20, 0, 0);
              glVertex3f( 0 ,-20, 0);
              glVertex3f(   0, +20, 0);
              glVertex3f( 0, 0,-20);
```

```
            glVertex3f(   0, 0, +20);

        glEnd();

        glLineStipple(1, 0xAAAA); //line style = fine dots
        glEnable(GL_LINE_STIPPLE);

        glBegin(GL_LINES);

            if (x_y_display) {glColor3f(0.0,0.7,0.7);
            for (offset=-10.0;offset<10.1;offset++){
                //draw lines in x-y plane
                glVertex3f(-10.0, offset, 0.0);                              //
Top Left
                glVertex3f(+10.0, offset, 0.0);                              //
Top Right
                glVertex3f( offset,-10, 0.0);                                //
Bottom Right
                glVertex3f(   offset,+10.0, 0.0);                            //
Bottom Left
            }}

            if (y_z_display) {glColor3f(0.7,0.0,0.7);
            for (offset=-10.0;offset<10.1;offset++){
                //draw lines in y-z plane
                glVertex3f( 0, offset, -10);
                glVertex3f(   0, offset, 10.0);
                glVertex3f( 0, -10, offset);
                glVertex3f(   0, 10, offset);
            }}

            if (x_z_display) {glColor3f(0.7,0.7,0.0);
            for (offset=-10.0;offset<10.1;offset++){
                //draw lines in x-z plane
                glVertex3f( offset, 0, -10);
                glVertex3f(   offset, 0, 10.0);
                glVertex3f( -10, 0, offset);
                glVertex3f(   10, 0, offset);
            }}

        glEnd();
        glDisable(GL_LINE_STIPPLE);
        glPopMatrix();

}

#endif
```

```
/*************************************************************************************
********************************************
Author: Tyron Mc Donald
Date: 2014-04-11
Operating system: Windows 7
IDE : Microsoft Visual Studio 2012 for Windows Desktop
Description : this file contains the draw_upper_front_left_leg and
draw_lower_front_left_leg function definition
*************************************************************************************
********************************************/

#ifndef LEFT_FRONT_LEG_H
#define LEFT_FRONT_LEG_H

#include "Drawing Class.h"

/***********************/
/*    Left front leg   */
/***********************/
void Drawing::draw_upper_front_left_leg(float legmove)
{
	glPushMatrix();
	glTranslatef(0,legmove,0);
	letter_i(0.05, 0.9, -1.1, 15, 180, 1, 1, 1, 1.2, 1);
	letter_m(-0.6, -1.1, -1.55, 270, -90, 1, 1, 0.2, 0.5, 0.5);
	draw_lower_front_left_leg();
	glPopMatrix();
}
void Drawing::draw_lower_front_left_leg()
{
	letter_i(0.05 ,-0.1, -1.4, 1, 180, 1, 1, 1, 0.85, 1);
}
#endif
```

```
/*****************************************************************************************
********************************************
Author: Tyron Mc Donald
Date: 2014-04-11
Operating system: Windows 7
IDE : Microsoft Visual Studio 2012 for Windows Desktop
Description : this file contains the draw_upper_hind_left_leg and
draw_lower_hind_left_leg function definition
*****************************************************************************************
********************************************/
#ifndef LEFT_HIND_LEG_H
#define LEFT_HIND_LEG_H

#include "Drawing Class.h"


/***********************/
/*    Left hind leg   */
/***********************/
void Drawing::draw_upper_hind_left_leg(float legmove)
{
       glPushMatrix();
       glTranslatef(0,legmove,0);
       letter_l(-0.25,0.5,1.5,90,0,1,0,1,1,1,0.22 ,0.29 ,0.63);
       draw_lower_hind_left_leg();
       glPopMatrix();
}
void Drawing::draw_lower_hind_left_leg()
{
       glPushMatrix();
       glTranslatef(0.0,-1.5,2.25);
       glRotatef(180,1,0,0);
       glScalef(1,1,1);
       letter_l(-0.25,-0.5,0.35,90,0,1,0,1,1,1,0.22 ,0.29 ,0.63);
       glPopMatrix();
       //feet
       letter_m(-0.58,-1.1,1.35,90,1,0,0,0.2,0.5,0.5);
}
#endif
```

```
/*****************************************************************************************
********************************************
Author: Nelio Lucas , Tyron Mc Donald , Tejas Dwarkaram
FileName: Letter class.h
Date: 2014-04-11
Operating system: Windows 7
Description : this file contains all the main functions used for creating the letters
that builds
the 3D model of princess luna
*****************************************************************************************
********************************************/
#include "Letter T.h"
#include "Letter O.h"
#include "Letter D.h"
#include "Letter J.h"
#include "Letter R.h"
#include "Letter M.h"
#include "Letter N.h"
#include "Letter I.h"
#include "Letter L.h"

#ifndef Letter_class_H
#define Letter_class_H

#include <iostream>
#include <GL/glut.h>


using std::cout;
using std::endl;




class Letters
{

public:

       //the following fuctions draw letters
       static void
letter_t(float,float,float,float,float,float,float,float,float,float);
       static void
letter_d(float,float,float,float,float,float,float,float,float,float, float,float
,float);
       static void
letter_n(float,float,float,float,float,float,float,float,float,float,float,bool,bool,b
ool);
       static void
letter_j(float,float,float,float,float,float,float,float,float,float);
       static void
letter_o(float,float,float,float,float,float,float,float,float,float,float,float,float
);
       static void
letter_l(float,float,float,float,float,float,float,float,float,float,float,float,float
);
       static void
letter_m(float,float,float,float,float,float,float,float,float,float);
       static void
letter_r(float,float,float,float,float,float,float,float,float,float,float,float,float
,float,bool,bool,bool);
```

```
       static void
letter_i(float,float,float,float,float,float,float,float,float,float);

       //SEGMENTS OF L
    static void horizontal(float,float,float);
    static void  vertical(float,float,float);

       //CURVE OF J , D and O
       static void curveofj(double, double, double, double, double, double);
    static void curveofd(double, double, double, double, double, double, float , float
, float);
    static void  curve_o(double , double , double , double , double , double , float ,
float , float);


};




#endif
```

```
/*********************************************************************************
*********************************************
Author: Nelio Lucas
Date: 2014-04-11
Operating system: Windows 7
IDE : Microsoft Visual Studio 2012 for Windows Desktop
Description : this file contains the draw body function definition
*********************************************************************************
*********************************************/
#ifndef BODY_H
#define BODY_H

#include "Drawing Class.h"



/***********************/
/*    Draw Body     */
/***********************/
void Drawing::draw_body(float bodymove)
{
       //FRONT BODY
       glPushMatrix();
       glTranslatef(0,bodymove,0);
       glRotatef(0,0,0,0);
       glScalef(1,1,1);

       //right
    letter_o(0,1.2,-1.5,90,0,1,0,0.4,0.4,2.1,0.22 ,0.29 ,1.10);
       letter_o(-0.9,1.2,-1.5,90,0,1,0,0.3,0.3,0.3 , 0.22 ,0.29 ,1.10);
       letter_o(-1.0,1.2,-1.5,90,0,1,0,0.25,0.25,0.25,0.22 ,0.29 ,1.10);
       letter_o(-1.1,1.2,-1.5,90,0,1,0,0.15,0.15,0.15 ,0.22 ,0.29 ,1.10);
       letter_o(-1.15,1.2,-1.5,90,0,1,0,0.1,0.1,0.1,0.22 ,0.29 ,0.63);
       letter_o(-1.17,1.2,-1.5,90,0,1,0,0.15,0.15,0.15 ,0.22 ,0.29 ,1.10);

       //left
       letter_o(0.9,1.2,-1.5,90,0,1,0,0.3,0.3,0.3 , 0.22 ,0.29 ,1.10);
       letter_o(1.0,1.2,-1.5,90,0,1,0,0.25,0.25,0.25,0.22 ,0.29 ,1.10);
       letter_o(1.1,1.2,-1.5,90,0,1,0,0.15,0.15,0.15 ,0.22 ,0.29 ,1.10);
       letter_o(1.15,1.2,-1.5,90,0,1,0,0.1,0.1,0.1,0.22 ,0.29 ,1.10);
       letter_o(1.17,1.2,-1.5,90,0,1,0,0.15,0.15,0.15 ,0.22 ,0.29 ,1.10);
       //front caps
       glPushMatrix();
       glTranslatef(-1.17,1.2,-1.5);
       glRotatef(90,0,1,0);
       glScalef(0.2,0.2,0.2);
       curve_o(1.0 , 0.0 , 2.0 , 0.0 , 360 , 5.0, 0.22 ,0.29 ,1.10);
       glPopMatrix();

       //back cap
       glPushMatrix();
       glTranslatef(1.17,1.2,-1.5);
       glRotatef(90,0,1,0);
       glScalef(0.2,0.2,0.2);
       curve_o(1.0 , 0.0 , 2.0 , 0.0 , 360 , 5.0 ,0.22 ,0.29 ,1.10);
       glPopMatrix();
     glPopMatrix();
       //BACK BODY


       glPushMatrix();
       glTranslatef(0,bodymove,0);
```

```
        glRotatef(0,0,0,0);
        glScalef(1,1,1);

     letter_o(0,1.45,2.2,90,0,1,0,0.6,0.5,2.0,0.22 ,0.29 ,1.10);

        letter_o(-0.9,1.45,2.2,90,0,1,0,0.45,0.45,0.45 , 0.22 ,0.29 ,1.10);
        letter_o(-1.0,1.45,2.2,90,0,1,0,0.4,0.4,0.4 ,0.22 ,0.29 ,1.10);
        letter_o(-1.1,1.45,2.2,90,0,1,0,0.35,0.35,0.35 ,0.22 ,0.29 ,1.10);
        letter_o(-1.15,1.45,2.2,90,0,1,0,0.3,0.3,0.3,0.22 ,0.29 ,1.10);
        letter_o(-1.17,1.45,2.2,90,0,1,0,0.25,0.25,0.25 ,0.22 ,0.29 ,1.10);


        letter_o(0.9,1.45,2.2,90,0,1,0,0.5,0.5,0.5 , 0.22 ,0.29 ,1.10);
        letter_o(1.0,1.45,2.2,90,0,1,0,0.4,0.4,0.4 ,0.22 ,0.29 ,1.10);
        letter_o(1.1,1.45,2.2,90,0,1,0,0.35,0.35,0.35 ,0.22 ,0.29 ,1.10);
        letter_o(1.15,1.45,2.2,90,0,1,0,0.3,0.3,0.3,0.22 ,0.29 ,1.10);
        letter_o(1.17,1.45,2.2,90,0,1,0,0.25,0.25,0.25 ,0.22 ,0.29 ,1.10);

        //front caps
        glPushMatrix();
        glTranslatef(-1.17,1.45,2.2);
        glRotatef(90,0,1,0);
        glScalef(0.18,0.18,0.18);
        curve_o(1.0 , 0.0 , 2.0 , 0.0 , 360 , 5.0, 0.22 ,0.29 ,1.10);
        glPopMatrix();

        //back cap
        glPushMatrix();
        glTranslatef(1.17,1.45,2.2);
        glRotatef(90,0,1,0);
        glScalef(0.2,0.2,0.2);
        curve_o(1.0 , 0.0 , 2.0 , 0.0 , 360 , 5.0 ,0.22 ,0.29 ,1.10);
        glPopMatrix();

        //middle
        letter_o(0,1.3,0,-4,1,0,0,0.6,0.4,2.9,0.22 ,0.29 ,1.10);

        glPopMatrix();

}
#endif
```

```
/*****************************************************************************************
********************************************
Author: Tejas Dwarkaram
Date: 2014-04-11
Operating system: Windows 7
IDE : Microsoft Visual Studio 2012 for Windows Desktop
Description : this file contains the draw_lower_left_wing , draw_middle_left_wing and
draw_upper_left_wing function definitions
*****************************************************************************************
********************************************/
#ifndef LOWER_LEFT_WING_H
#define LOWER_LEFT_WING_H

#include "Drawing Class.h"



/***********************/
/*    Left wing        */
/***********************/


void Drawing::draw_lower_left_wing(float left_upper_wing_rot, float
left_middle_wing_rot,float left_lower_wing_rot)
{

       glPushMatrix();
       glTranslatef(0,bodymove,0);
       glRotatef(left_lower_wing_rot,0,0,1);
    glScalef(1,1,1);



       letter_n(0.8, 1.6, 0,left_lower_wing_rot,0,0,1, 0.7, 0.7, 0.7,90,1,0,0);
       letter_r(0.8, 1.6, 0,left_lower_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,90,1,0,0);
       letter_n(1.1, 1.6, -0.3,left_lower_wing_rot,0,0,1, 0.7, 0.7, 0.7,90,1,0,0);
       letter_r(1.1, 1.6, -0.3,left_lower_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,90,1,0,0);
       letter_n(1.4, 1.6, -0.6,left_lower_wing_rot,0,0,1, 0.7, 0.7, 0.7,90,1,0,0);
       letter_r(1.4, 1.6, -0.6,left_lower_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,90,1,0,0);

       draw_middle_left_wing(left_middle_wing_rot,left_upper_wing_rot);
       glPopMatrix();

}

//middle left wing
void Drawing::draw_middle_left_wing(float left_middle_wing_rot, float
left_upper_wing_rot)
{


       letter_n(1.7, 1.6, -0.6,left_middle_wing_rot,0,0,1, 0.7, 0.7, 0.7,90,1,0,0);
       letter_r(1.7, 1.6, -0.6, left_middle_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,90,1,0,0);
       letter_n(2.0, 1.6, -0.9,left_middle_wing_rot,0,0,1, 0.7, 0.7, 0.7,90,1,0,0);
       letter_r(2.0, 1.6, -0.9,left_middle_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,90,1,0,0);
       letter_n(2.3, 1.6, -1.2,left_middle_wing_rot,0,0,1, 0.7, 0.7, 0.7,90,1,0,0);
       letter_r(2.3, 1.6, -1.2,left_middle_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,90,1,0,0);
       draw_upper_left_wing(left_upper_wing_rot);

}

//upper left wing
void Drawing::draw_upper_left_wing(float left_upper_wing_rot)
{

       letter_n(2.6, 1.6, -1.2, left_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7,90,1,0,0);
       letter_r(2.6, 1.6, -1.2,left_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,90,1,0,0);
       letter_n(2.9, 1.6, -1.5,left_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7,90,1,0,0);
       letter_r(2.9, 1.6, -1.5, left_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,90,1,0,0);
       letter_n(3.2, 1.6, -1.8, left_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7,90,1,0,0);
       letter_r(3.2, 1.6, -1.8,left_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,90,1,0,0);/**/



}
#endif
```

```
/*************************************************************************************
********************************************
Author: Tejas Dwarkaram
Date: 2014-04-11
Operating system: Windows 7
IDE : Microsoft Visual Studio 2012 for Windows Desktop
Description : this file contains the draw_lower_right_wing , draw_middle_right_wing
and draw_upper_right_wing function definitions
*************************************************************************************
********************************************/
#ifndef LOWER_RIGHT_WING_H
#define LOWER_RIGHT_WING_H

#include "Drawing Class.h"




/***********************/
/*    Right wing           */
/***********************/

void Drawing::draw_lower_right_wing(float right_upper_wing_rot, float
right_middle_wing_rot,float right_lower_wing_rot)
{
       glPushMatrix();
       glTranslatef(0,bodymove,0);
       glRotatef(right_lower_wing_rot,0,0,1);
    glScalef(1,1,1);

       letter_n(-0.8, 1.95, 0, right_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7,180, 0, -180,
-180);
       letter_r(-0.8, 1.95, 0, right_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,180, 0, -180, -180);
       letter_n(-1.1, 1.95, -0.3, right_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7,180, 0, -
180, -180);
       letter_r(-1.1, 1.95, -0.3, right_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,180, 0, -180, -180);
    letter_n(-1.4, 1.95, -0.6, right_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7,180, 0, -180,
-180);
       letter_r(-1.4, 1.95, -0.6, right_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,180, 0, -180, -180);
       draw_middle_right_wing( right_upper_wing_rot,  right_middle_wing_rot);
       glPopMatrix();
}
//middle right wing
void Drawing::draw_middle_right_wing(float right_upper_wing_rot, float
right_middle_wing_rot)
{

       letter_n(-1.7, 1.95, -0.6, right_middle_wing_rot,0,0,1, 0.7, 0.7, 0.7,180, 0, -
180, -180);
       letter_r(-1.7, 1.95, -0.6, right_middle_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,180, 0, -180, -180);
       letter_n(-2.0, 1.95, -0.9, right_middle_wing_rot,0,0,1, 0.7, 0.7, 0.7,180, 0, -
180, -180);
       letter_r(-2.0, 1.95, -0.9, right_middle_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,180, 0, -180, -180);
       letter_n(-2.3, 1.95, -1.2, right_middle_wing_rot,0,0,1, 0.7, 0.7, 0.7,180, 0, -
180, -180);
       letter_r(-2.3, 1.95, -1.2, right_middle_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,180, 0, -180, -180);
```

```
        draw_upper_right_wing(right_upper_wing_rot);
}

//upper right wing
void Drawing::draw_upper_right_wing(float right_upper_wing_rot)
{

        letter_n(-2.6, 1.95, -1.2,right_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7,180, 0, -
180, -180);
        letter_r(-2.6, 1.95, -1.2,right_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,180, 0, -180, -180);
        letter_n(-2.9, 1.95, -1.5,right_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7,180, 0, -
180, -180);
        letter_r(-2.9, 1.95, -1.5,right_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,180, 0, -180, -180);
        letter_n(-3.2, 1.95, -1.8,right_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7,180, 0, -
180, -180);
        letter_r(-3.2, 1.95, -1.8,right_upper_wing_rot,0,0,1, 0.7, 0.7, 0.7, 0 , 0 ,
120,180, 0, -180, -180);
}
#endif
```

```
/*****************************************************************************************
********************************************
Author: Tyron Mc Donald
Date: 2014-04-11
Operating system: Windows 7
IDE : Microsoft Visual Studio 2012 for Windows Desktop
Description : this file contains the draw_upper_front_right_leg and
draw_lower_front_right_leg function definitions
*****************************************************************************************
********************************************/

#ifndef RIGHT_FRONT_LEG_H
#define RIGHT_FRONT_LEG_H

#include "Drawing Class.h"




/***********************/
/*    Right front leg  */
/***********************/
void Drawing::draw_upper_front_right_leg(float legmove)
{
       glPushMatrix();
       glTranslatef(0,legmove,0);
       letter_i(1.05, 0.9, -1.1, 15, 180, 1, 1, 1, 1.2, 1);
       letter_m(0.45, -1.1, -1.55, 270, -90, 1, 1, 0.2, 0.5, 0.5);
       draw_lower_front_right_leg();
       glPopMatrix();
}
//lower front right leg
void Drawing::draw_lower_front_right_leg()
{
              letter_i(1.05 ,-0.1, -1.4, 1, 180, 1, 1, 1, 0.85, 1);

}
#endif
```

```
/*******************************************************************************************
********************************************
Author: Nelio Lucas
Date: 2014-04-11
Operating system: Windows 7
IDE : Microsoft Visual Studio 2012 for Windows Desktop
Description : this file contains the draw_head and draw_neck function definitions
*******************************************************************************************
********************************************/

#ifndef NECKHEAD_H
#define NECKHEAD_H

#include "Drawing Class.h"







/***********************/
/*    Draw Head    */
/***********************/

void Drawing::draw_head(float head_rot)
{

       glPushMatrix();
       glTranslatef(0,-0.6,-1.2);
       glRotatef(head_rot,1,0,0); //articulation of head
       glScalef(1,1,1);


       //snout
       glPushMatrix();
       glTranslatef(0,4.2,1.555);
       glRotatef(-80,1,0,0);
       glScalef(0.6,1.0,0.6);
    letter_d(0,4,0,-90,0,1,0,0.5,0.5,0.5, 0.22 ,0.29 ,1.10);
       glPopMatrix();

       //head
       glPushMatrix();
       glTranslatef(0,2.7,3);
       glRotatef(-65,1,0,0);
       glScalef(1.2,1.2,1.2);
    letter_d(0,4,0,-90,0,1,0,0.5,0.5,0.5,0.22 ,0.29 ,1.10);
       glPopMatrix();


       //ears
       glPushMatrix();
       glTranslatef(0.4,5.85,-1.35);
       glRotatef(-150,1,0,0);
       glScalef(1,1,1);
       letter_j(0,0,0,90,0,1,0,0.5,0.5,0.5);
       glPopMatrix();

       glPushMatrix();
       glTranslatef(-0.4,5.85,-1.35);
       glRotatef(-150,1,0,0);
       glScalef(1,1,1);
       letter_j(0,0,0,90,0,1,0,0.5,0.5,0.5);
       glPopMatrix();

       //mouth
       glPushMatrix();
       glTranslatef(-0.26,4.9,-2.95);
       glRotatef(90,0,0,1);
       glScalef(0.2,0.54,1);
       horizontal(0,0,0);
       glPopMatrix();

       //nostrils
       glPushMatrix();
       glTranslatef(-0.21,5.2,-2.96);
       glRotatef(90,0,1,0);
       glScalef(0.05,0.05,0.1);
       curveofd(1.0 , 0.0 , 2.0 , 0.0 , 360 , 5.0, 0 ,0 ,0);
       glPopMatrix();

       glPushMatrix();
       glTranslatef(0.21,5.2,-2.96);
       glRotatef(90,0,1,0);
       glScalef(0.05,0.05,0.1);
       curveofd(1.0 , 0.0 , 2.0 , 0.0 , 360 , 5.0, 0 ,0 ,0);
       glPopMatrix();

       //eyes
       letter_o(0.5,5,-2.0,90,0,1,0,0.2,0.2,0.2 ,0.53 ,0.12 ,0.47);
       letter_o(0.5,5.0,-2.0,90,0,1,0,0.09,0.09,0.09 , 125 , 0 , 0);
       letter_d(0.5,5,-2.0,90,0,1,0,0.2001,0.2001,0.2001 ,0.22 ,0.29 ,0.63);


       letter_o(-0.5,5,-2.0,90,0,1,0,0.2,0.2,0.2 , 0.53 ,0.12 ,0.47);
       letter_o(-0.5,5.0,-2.0,90,0,1,0,0.09,0.09,0.09 , 125 , 0 , 0);
       letter_d(-0.5,5,-2.0,90,0,1,0,0.2001,0.2001,0.2001 ,0.22 ,0.29 ,0.63);
    letter_o(-0.5,5,-2.0,90,0,1,0,0.2,0.2,0.2,125 , 0 , 12);

       //horn
       glPushMatrix();
       glTranslatef(0,3.2,3.8);
       glRotatef(-45,1,0,0);
       glScalef(1,1,1);
       letter_t(0.2,5.5,-2.5,180,0,0,1,1,1,2);
       glPopMatrix();
       letter_i(0.27,5.5,-2.1,135,1,0,0,0.5,1.8,0.5);
       letter_i(0.165,5.5,-2.05,135,1,0,0,0.3,2.8,0.3);

       glPopMatrix();

}

/***********************/
/*    Draw Neck    */
/***********************/

void Drawing::draw_neck(float head_rot, float neck_rot, float bodymove)
{
       glPushMatrix();
       glTranslatef(0,bodymove,0);
       glRotatef(neck_rot,1,0,0);
              glScalef(1,1,1);
              letter_i(1.7,4,-1.7,-15,1,0,0,3.1,2.7,3);
              letter_i(1.1,4.7,-2.05,-15,1,0,0,2,2,2);
              draw_head(head_rot);
       glPopMatrix();
}

#endif
```

```
/***************************************************************************************
********************************************
Author: Tyron Mc Donald
Date: 2014-04-11
Operating system: Windows 7
IDE : Microsoft Visual Studio 2012 for Windows Desktop
Description : this file contains the draw_upper_hind_right_leg and
draw_lower_hind_right_leg function definitions
****************************************************************************************
********************************************/
#ifndef RIGHT_HIND_LEG_H
#define RIGHT_HIND_LEG_H

#include "Drawing Class.h"




/***********************/
/*    Right hind leg   */
/***********************/
void Drawing::draw_upper_hind_right_leg(float legmove)
{
       glPushMatrix();
       glTranslatef(0,legmove,0);
       letter_l(0.75,0.5,1.5,90,0,1,0,1,1,1,0.22 ,0.29 ,0.63);
       draw_lower_hind_right_leg();
       glPopMatrix();
}

//lower hind right leg
void Drawing::draw_lower_hind_right_leg()
{

       glPushMatrix();
       glTranslatef(1,-1.5,2.25);
       glRotatef(180,1,0,0);
       glScalef(1,1,1);
       letter_l(-0.25,-0.5,0.35,90,0,1,0,1,1,1,0.22 ,0.29 ,0.63);
       glPopMatrix();
       //feet
       letter_m(0.425,-1.1,1.35,90,1,0,0,0.2,0.5,0.5);
}
#endif
```

```
/*****************************************************************************************
********************************************
Author: Nelio Lucas
Date: 2014-04-11
Operating system: Windows 7
IDE : Microsoft Visual Studio 2012 for Windows Desktop
Description : this file contains the draw_upper_tail , draw_middle_tail and
draw_lower_tail function definitions
*****************************************************************************************
********************************************/
#ifndef TAIL_H
#define TAIL_H

#include "Drawing Class.h"



/***********************/
/*        Tail         */
/***********************/
void Drawing::draw_upper_tail(float upper_tail_rot,float middle_tail_rot,float
bodymove)
{
       glPushMatrix();
       glTranslatef(0,bodymove,0);
       glRotatef(upper_tail_rot,1,0,0);
       glScalef(1,1,1);

       letter_i(0.55,2.2,3,-70+upper_tail_rot,1,0,0,1,1,1);
       draw_middle_tail(middle_tail_rot);
       glPopMatrix();
}

//draw middle tail
void Drawing::draw_middle_tail(float middle_tail_rot)
{
       glPushMatrix();
       glTranslatef(0,0,0);
       glRotatef(middle_tail_rot,1,0,0);
       glScalef(1,1,1);
       letter_i(0.425,1.8,3.9,-70+middle_tail_rot,1,0,0,0.8,0.9,0.8);
       draw_lower_tail();
       glPopMatrix();

}

//draw lower tail
void Drawing::draw_lower_tail()
{
       glPushMatrix();
       glTranslatef(-0.02,-0.09,4.05);
       glRotatef(-65,1,0,0);
       glScalef(0.5,0.5,0.5);

       float aa =0.2;
       float bb =0.2;
       float cc =0.2;
       letter_j(0,0,3,45,0,1,0,aa,bb,cc);
       letter_j(0,0,3,90,0,1,0,aa,bb,cc);
       letter_j(0,0,3,135,0,1,0,aa,bb,cc);
       letter_j(0,0,3,180,0,1,0,aa,bb,cc);
       letter_j(0,0,3,225,0,1,0,aa,bb,cc);
```

```
        letter_j(0,0,3,270,0,1,0,aa,bb,cc);
        letter_j(0,0,3,315,0,1,0,aa,bb,cc);
        letter_j(0,0,3,360,0,1,0,aa,bb,cc);

        glPopMatrix();

}
#endif
```