電腦圖學 期中考作業 (108 上)

參數化線性軸曲面設計

資工三乙 406262216 劉品萱

程式架構

依照程式畫面而言可以區分成右側的貝茲曲線（Bézier curve）及 7 個控制點，以及左側透過貝茲曲線所繪出來的 3D 物件。

1. void printString(const char* str)：用來在視窗上繪製字樣，傳入的參數即為要繪製的字。
2. void display() - 控制整個畫面的呈現

- 貝茲曲線可透過 void display()中的給定的 u 值上下界利用函式 glMap1f()、glMapGrid1f()、glEvalMesh1()繪出（code Line 62-78）
- 控制點可利用程式一開始的設定好的 cpts array，透過函式 glBegin(GL_POINTS)繪出（code Line 80-91），及 glBegin(GL_LINE_STRIP)將每個控制點連線起來（code Line93-100）
- 為使 3D 物件具備滑鼠拖拉時有旋轉效果，要加上 Rotatef 搭配視角旋轉（code Line 113-115）
- 兩個 switch 是利用改變 type 的數值來判斷 MainMenu 選單中點選到的選項為何決定繪出的模式，並利用函式 glMap2f()、glMapGrid2f()、glEvalMesh2()繪出 3D 物件
  - void vmult(float m[3][3], float v[3], float r[3]) : 用來搭配 switch 計算旋轉角度的函式

3. 選單處理函式

- void MainMenu(int index) – 用來處理當點選選單的選項時，將 type 改成相對應的數值，讓在 display 中的 switch 能選擇繪出模式。
- void ColorMenu(int index) – 透過直接更改 light0_v array 來決定繪出的顏色
- void RainbowShowMenu(int index) – 利用傳遞進來的參數 index，來改變變數 RainBowFuntion 的數值，達到要開啟或關閉 RainbowShow 功能

4. void RainbowShow(int value) – 此函式搭配主程式中 glutTimerFunc(33,RainbowShow,1);，類似計時器，當秒數到達某值時改變 light0_v array 的值，使得可以呈現出七彩變化的效果。
5. 處來 glut 一些監聽設定的函式
   - void myMotion(int x, int y) 設置滑鼠按鍵回調函式，取得滑鼠拖動時的位置
   - void myMouse(int btn, int state, int x, int y)：用來換算滑鼠位置
     - float dis2p(float a[3], float b[3]):用來搭配 myMouse 的位置計算

- void myReshape(int w, int h)：當應用程式視窗大小調動時調用的函式

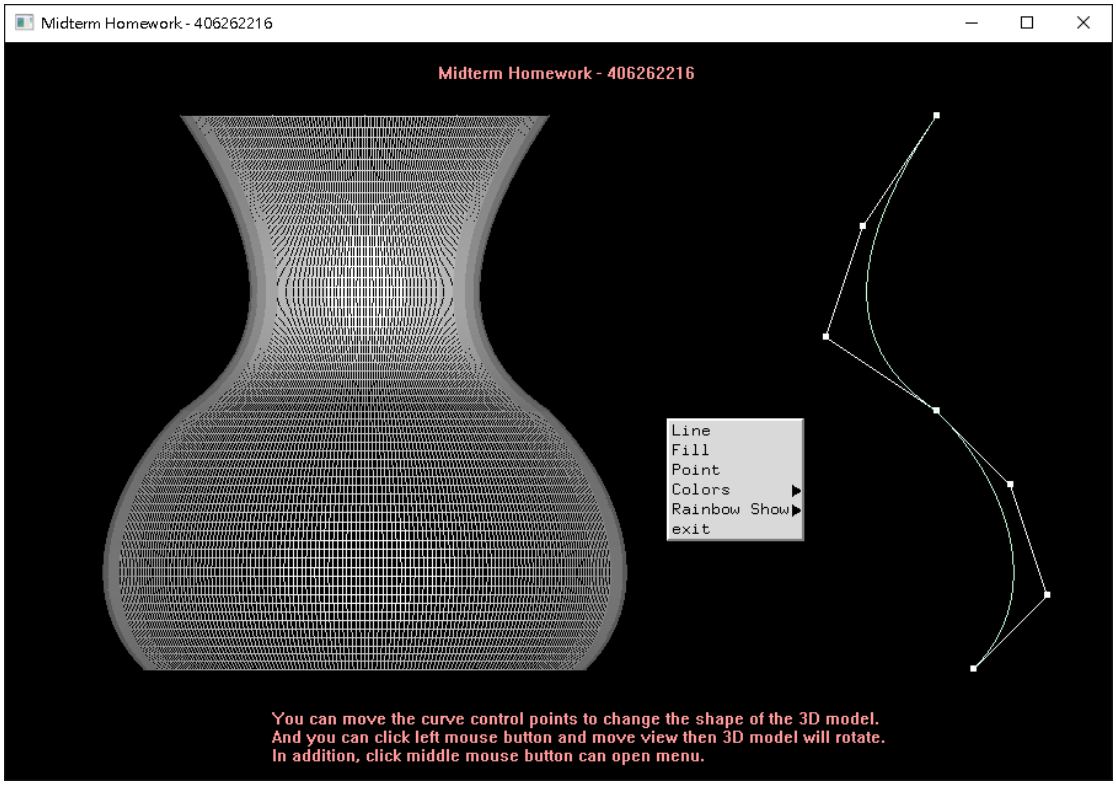
6. 初始化函式 void myinit()：光源、材質、背景顏色等的初始化函式
7. 主程式 int main(int argc, char **argv)

討論

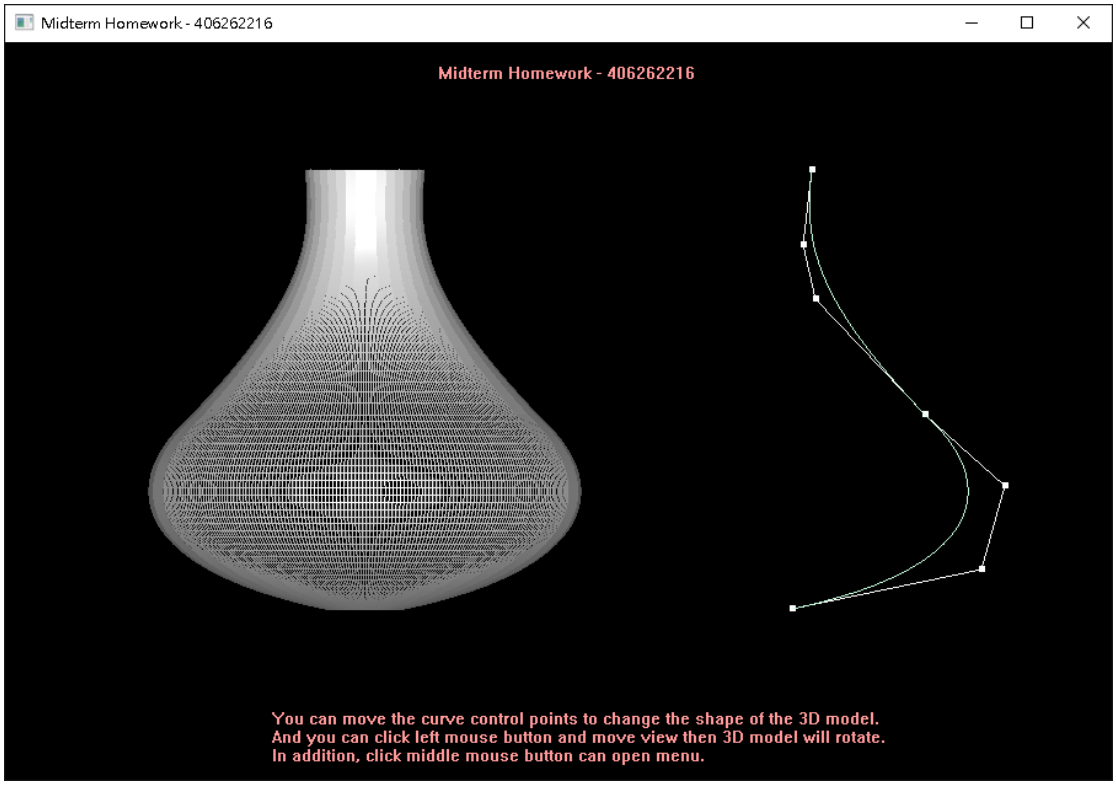
一開始我是使用 Mac 寫作業，MacOS 的環境只要使用 framework 就可以直接執行，但後來覺得還是繳交 exe 檔案比較好，所以移植到 Windows 這個環境來寫，花了點時間在 VSCode 上架環境，但較可惜的是原本 Mac 上可以裝起來以 OpenGL 為基底的 UI 介面 GLUI，在 Window 上找不到對應的 dll 檔所以裝不起來。這個作業有很多觀念混在一起，在理解上撞牆了好一陣子，最後在與其他修課的朋友們討論和不斷地爬文、觀看網路上的教學影片才慢慢地抓到要從哪裡下手，在了解到哪些 code 有什麼用處之後就比較上手，但最後想要新增貼圖的功能又撞牆了，查了網路上 load bmp 的方式，圖片都無法順利貼上，甚至電腦效能太差還一度執行到直接當機。雖然貼圖的部分花了滿多時間試但仍沒做出來稍顯可惜，但透過貝氏曲線在到透過其繪出 3D 物件，慢慢的摸索、做出點成果來，不僅學習到了不少，也讓我因此對圖學這個領域增加了點自信心和興趣。

執行畫面

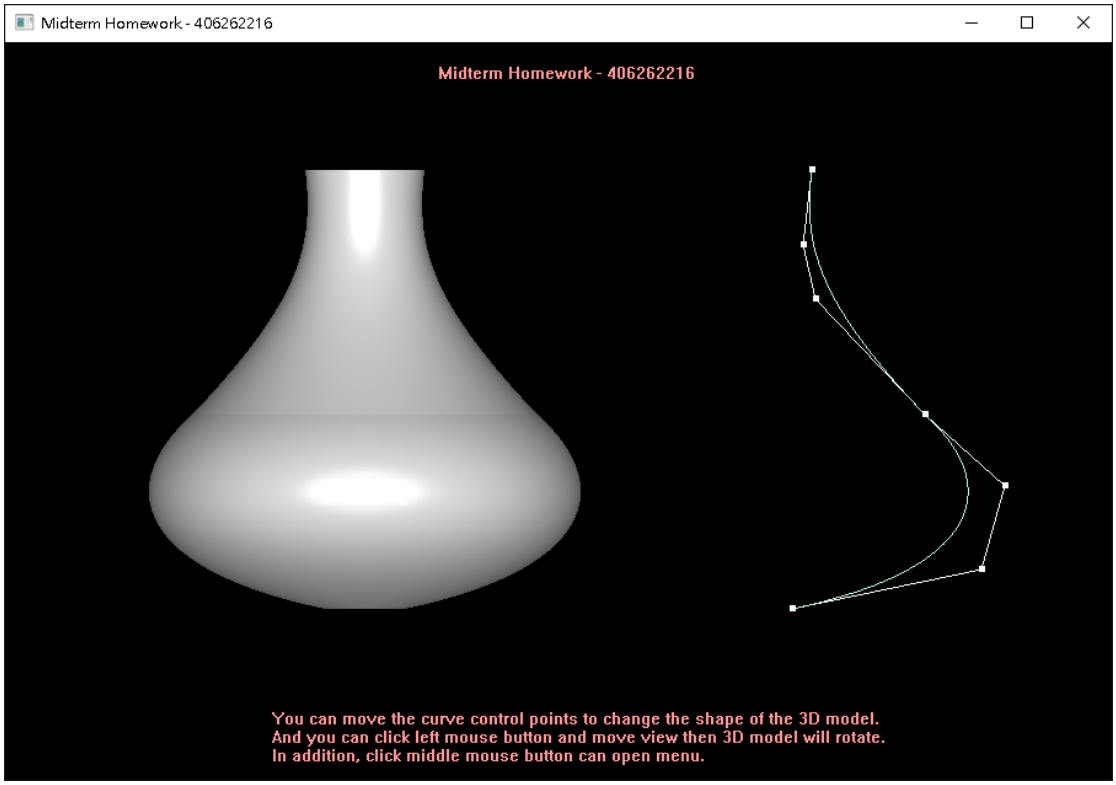
整個執行畫面的最頂和最底分別加上字，分別為標題和使用方法。可以透過滑鼠中鍵來開啟選單，選單主要功能分為繪出 3D 物件是線框式 (Line) 呈現、塗色式 (Fill) 呈現及點狀式 (Point) 呈現。而子選單 Colors 則可以針對 3D 物件有 7 種顏色可以選擇，Rainbow Show 則可以呈現七彩變化的特殊效果，另外點選 exit 變可結束此應用程式。



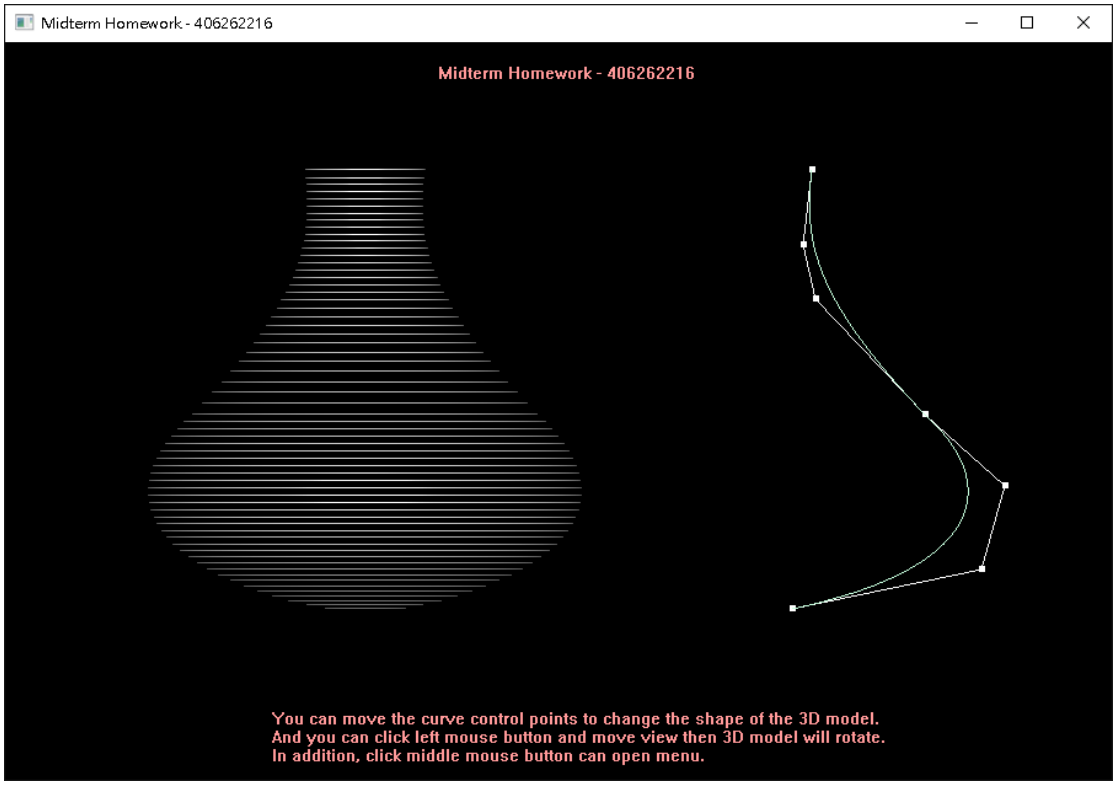
線框式 （Line）呈現



塗色式（Fill）呈現

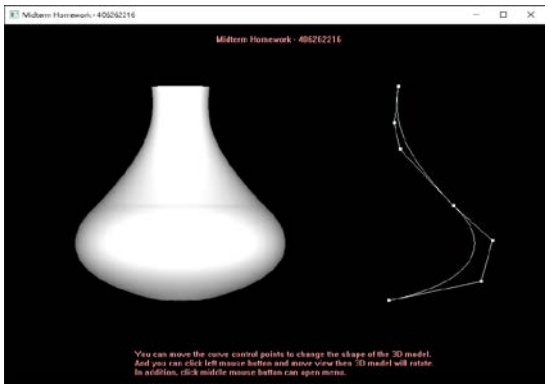

Midterm Homework - 406262216
Midterm Homework - 406262216
You can move the curve control points to change the shape of the 3D model.
And you can click left mouse button and move view then 3D model will rotate.
In addition, click middle mouse button can open menu.

點狀式（Point）呈現

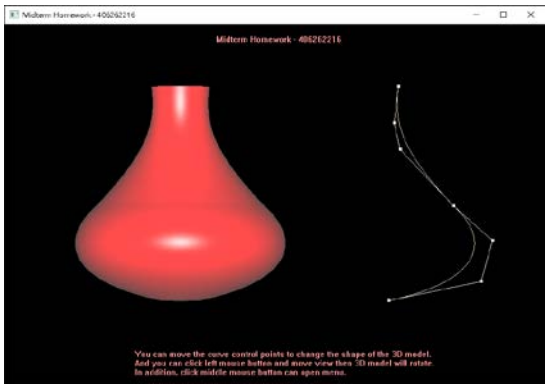

Midterm Homework - 406262216
Midterm Homework - 406262216
You can move the curve control points to change the shape of the 3D model.
And you can click left mouse button and move view then 3D model will rotate.
In addition, click middle mouse button can open menu.

7 種 Color 呈現

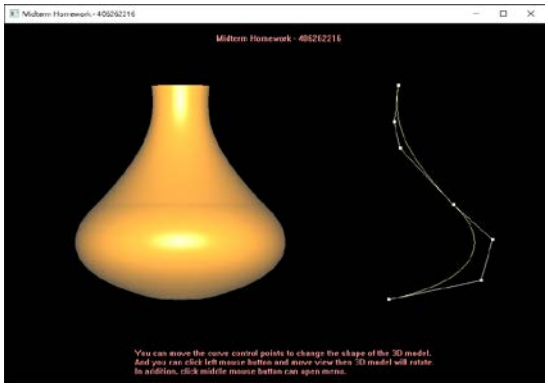
白色（White）



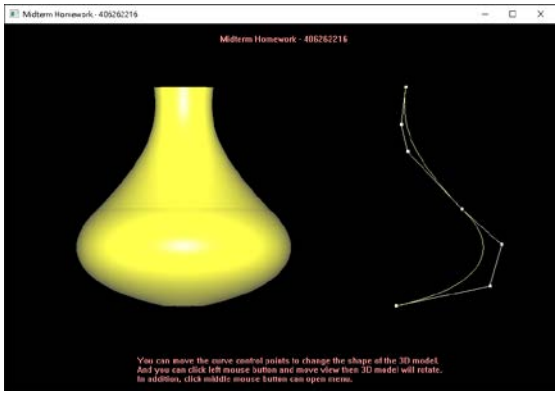
紅色（Red）



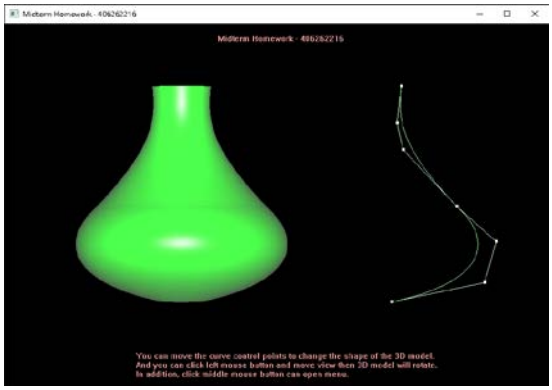
橘色（Orange）



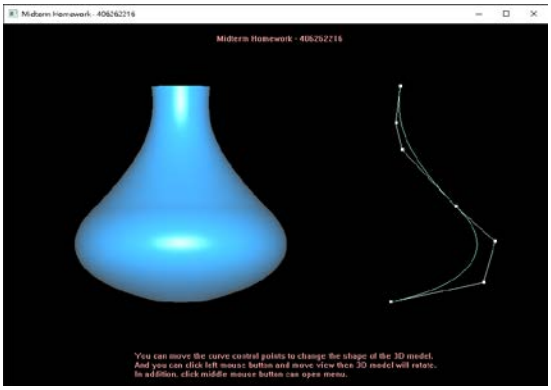
黃色（Yellow）



綠色（Green）

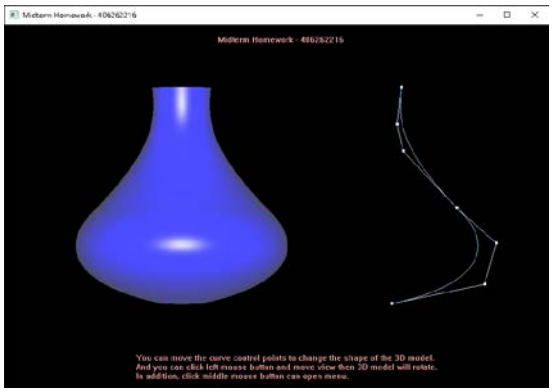


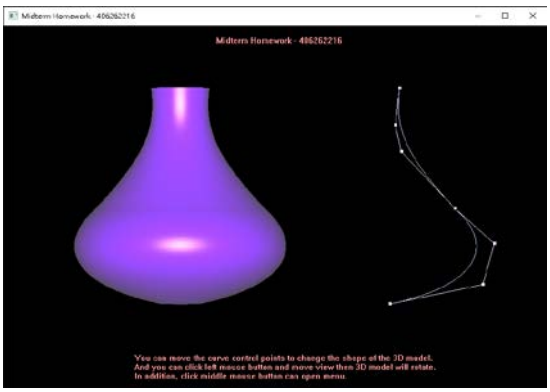
藍色（Blue）



靛色（Indigo）

紫色（Purple）





程式碼

```
1.  //
2.  //  main.cpp
3.  //  openGLMidterm
4.  //
5.  //  Created by 劉品萱 on 2019/11/15.
6.  //  Copyright © 2019 劉品萱. All rights reserved.
7.  //
8.
9.  #include <cstdlib>
10. #include <cmath>
11. #include <cstdio>
12. #include <GL/glut.h>
13. #include <GL/glui.h>
14. #include <windows.h>
15.
16. #define PI 3.14159265
17. #define MS 10.0
18. #define TN 3.5
19. #define FRAME 10
20. #define MAX_CHAR 128
21.
22. int type = 1;
23. int timer = 0;
24. int RainBowFuntion = 0;
25. int WindowId;
26.
27. // 光源參數
28. GLfloat light0_v[4] =  {0.6, 0.6, 0.6, 1};
```

```
29.
30.  // 設置控制點座標
31.  GLfloat cpts[2][7][3] = {{{5.0, 8.0, 0.0}, {3.0, 5.0, 0.0}, {2.0, 2, 0.0}, {5.0, 0.0, 0.0}, {7.0, -2.0, 0.0},
     {8.0, -5.0, 0.0}, {6.0, -7.0, 0.0}}};
32.  GLubyte image[64][64][3];
33.  GLfloat r1[3][3] = {{static_cast<GLfloat>(cos(PI / 180.0)), 0.0, static_cast<GLfloat>(sin(PI /
     180.0))}, {0.0, 1.0, 0.0}, {static_cast<GLfloat>(-sin(PI / 180.0)), 0.0,
     static_cast<GLfloat>(cos(PI / 180.0))}};
34.  GLfloat r10[3][3] = {{static_cast<GLfloat>(cos(PI / 18.0)), 0.0, static_cast<GLfloat>(sin(PI /
     18.0))}, {0.0, 1.0, 0.0}, {static_cast<GLfloat>(-sin(PI / 18.0)), 0.0, static_cast<GLfloat>(cos(PI /
     18.0))}};
35.  GLfloat r100[3][3] = {{static_cast<GLfloat>(cos(PI / 180.0)), 0.0, static_cast<GLfloat>(sin(PI /
     180.0))}, {0.0, 1.0, 0.0}, {static_cast<GLfloat>(-sin(PI / 180.0)), 0.0,
     static_cast<GLfloat>(cos(PI / 180.0))}};
36.  GLfloat thta[2] = {0.0, 0.0};
37.  float p[3], mp[3], interw, interh, mrgb[3] = {0.0, 0.4, 0.8};
38.  int pn = 7, ww, hh;
39.
40.  // 繪製字樣
41.  void printString(const char* str)
42.  {
43.      static int isFirstCall = 1;
44.      static GLuint lists;
45.      // 如果是第一次調用，則執行初始化。並且為每一個 ASCII 字符產生一個顯示列表
46.      // 为每一个 ASCII 字符产生一个显示列表
47.      if( isFirstCall ) {
48.          isFirstCall = 0;
49.          // 申請 MAX_CHAR 個連續的顯示列表編號
50.          lists = glGenLists(MAX_CHAR);
51.          // 把每個字符的繪製命令都裝到對應的顯示列表中
52.          wglUseFontBitmaps(wglGetCurrentDC(), 0, MAX_CHAR, lists);
53.       }
54.      // 調用每個字符對應的顯示列表，繪製每個字符
55.      for(; *str!='\0'; ++str)
56.          glCallList(lists + *str);
57.  }
58.
59.  // 計算旋轉角度使用，搭配 display
```

```
60.  void vmult(float m[3][3], float v[3], float r[3])
61.  {
62.      int i, j;
63.      for(i = 0; i < 3; ++i){
64.          for(j = 0, r[i] = 0.0; j < 3; ++j){
65.              r[i] += m[i][j] * v[j];
66.          }
67.      }
68.  }
69.
70.  void display(void)
71.  {
72.      glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
73.      // 初始化 Martix
74.      glLoadIdentity();
75.
76.      glPushMatrix();
77.      glTranslated(TN * (float)ww / (float)hh, 0.0, 0.0);
78.
79.      /****************產生貝式曲線 ****************/
80.      glColor3f(0.2, 0.8, 0.3);
81.
82.      // 為曲線建立對應, u 的數值由 0 到 1
83.      glMap1f(GL_MAP1_VERTEX_3, 0.0, 1.0, 3, 4, &cpts[0][0][0]);
84.      /*  glMap1f(控制點代表頂點座標, u 值起頭, u 值結尾, 每個儲存塊之間單精度或雙精度浮點
    數的數量, 控制點數量, 指向第一個控制點的第一個座標)*/
85.      glMapGrid1f(30, 0.0, 1.0);
86.      /*  glMapGrid1f(網格中間的階段, u 值起頭, u 值結尾)*/
87.      glEvalMesh1(GL_LINE, 0, 30);
88.      /*  glEvalMesh1(mode,i1,i2)
89.       對所有已經啟用的求值器應用定義一維網格。
90.       mode 參數可以是 GL_POINT、GL_LINE。
91.       这个函数相當於從 i1 到 i2 的每一步都調用 glMapGrid1d 函数，其中 0<i1,i2<n*/
92.
93.      glMap1f(GL_MAP1_VERTEX_3, 0.0, 1.0, 3, 4, &cpts[0][3][0]);
94.      glMapGrid1f(30, 0.0, 1.0);
95.      glEvalMesh1(GL_LINE, 0, 30);
96.
```

```
97.         // 設置控制點大小
98.         glPointSize(5.0);
99.         glColor3f(1.0, 1.0, 1.0);
100.        glBegin(GL_POINTS);
101.        glVertex3fv(cpts[0][0]);
102.        glVertex3fv(cpts[0][1]);
103.        glVertex3fv(cpts[0][2]);
104.        glVertex3fv(cpts[0][3]);
105.        glVertex3fv(cpts[0][4]);
106.        glVertex3fv(cpts[0][5]);
107.        glVertex3fv(cpts[0][6]);
108.        glEnd();
109.
110.        // 將每個控制點連線
111.        glColor3f(1.0, 1.0, 1.0);
112.        glBegin(GL_LINE_STRIP);
113.        {
114.            for(int i = 0 ; i < 7 ; i++)
115.                glVertex3fv(cpts[0][i]);
116.        }
117.        glEnd();
118.
119.        // 繪製字
120.        glColor3f(1.0f, 0.0f, 0.0f);
121.        glRasterPos2f(-8.5f, 9.0f);
122.        printString("Midterm Homework - 406262216");
123.        glRasterPos2f(-13.0f, -8.5f);
124.        printString("You can move the curve control points to change the shape of the 3D
    model.");
125.        glRasterPos2f(-13.0f, -9.0f);
126.        printString("And you can click left mouse button and move view then 3D model will
    rotate.");
127.        glRasterPos2f(-13.0f, -9.5f);
128.        printString("In addition, click middle mouse button can open menu.");
129.
130.        // 模型顏色
131.        glColor3f(1, 1, 1);
132.        glBegin(GL_LINE);
```

```
133.    glVertex3fv(cpts[0][0]);
134.    glVertex3fv(cpts[0][1]);
135.    glEnd();
136.
137.    glPopMatrix();
138.    glTranslated(-TN * (float)ww / (float)hh, 0.0, 0.0);
139.
140.    // 讓模型能夠旋轉
141.    glRotatef(thta[1], 1.0, 0.0, 0.0);
142.    glRotatef(thta[2], 0.0, 1.0, 0.0);
143.    glRotatef(thta[0], 0.0, 0.0, 1.0);
144.
145.    int i;
146.
147.    // type 對應主選單點選的選項，並依照設定好的角度讓運算控制點後將數值存起來
148.    switch(type){
149.        case 1:
150.            for(i = 0; i < 7; ++i)
151.                vmult(r10, cpts[0][i], cpts[1][i]);
152.            break;
153.        case 2:
154.            for(i = 0; i < 7; ++i)
155.                vmult(r1, cpts[0][i], cpts[1][i]);
156.            break;
157.        case 3:
158.            for(i = 0; i < 7; ++i)
159.                vmult(r100, cpts[0][i], cpts[1][i]);
160.            break;
161.    }
162.
163.    // type 對應主選單點選的選項，決定呈現的方式：Line、Fill、Point
164.    switch(type){
165.        case 1:
166.            // Line
167.            for(i = 0; i < 36; ++i){
168.                glRotatef(10.0, 0.0, 1.0, 0.0);
169.
170.                glMap2f(GL_MAP2_VERTEX_3, 0.0, 1.0, 3, 4, 0.0, 1.0, 21, 2, &cpts[0][0][0]);
```

```
171.                    //glMapGrid2f() 從 a 到 b 的範圍對映一個包含 n 個點的網格
172.                    glMapGrid2f(30,       // 產生網格數
173.                                0.0,      // u 值下界
174.                                1.0,      // u 值上界
175.                                10,
176.                                0.0,
177.                                1.0);
178.                    // 計算網格
179.                    glEvalMesh2(GL_LINE, 0, 30, 0, 10);
180.
181.                    glMap2f(GL_MAP2_VERTEX_3, 0.0, 1.0, 3, 4, 0.0, 1.0, 21, 2, &cpts[0][3][0]);
182.                    glMapGrid2f(30, 0.0, 1.0, 10, 0.0, 1.0);
183.                    glEvalMesh2(GL_LINE, 0, 30, 0, 10);
184.                }
185.                break;
186.            case 2:
187.                // Fill
188.                for(i = 0; i < 360; ++i){
189.                    glRotatef(1.0, 0.0, 1.0, 0.0);
190.
191.                    glMap2f(GL_MAP2_VERTEX_3, 0.0, 1.0, 3, 4, 0.0, 1.0, 21, 2, &cpts[0][0][0]);
192.                    glMapGrid2f(30, 0.0, 1.0, 10, 0.0, 1.0);
193.                    glEvalMesh2(GL_FILL, 0, 30, 0, 10);
194.
195.                    glMap2f(GL_MAP2_VERTEX_3, 0.0, 1.0, 3, 4, 0.0, 1.0, 21, 2, &cpts[0][3][0]);
196.                    glMapGrid2f(30, 0.0, 1.0, 10, 0.0, 1.0);
197.                    glEvalMesh2(GL_FILL, 0, 30, 0, 10);
198.                }
199.                break;
200.            case 3:
201.                // Point
202.                for(i = 0; i < 360; ++i){
203.                    glEnable(GL_POINT_SIZE);
204.                    glEnable(GL_TEXTURE_2D);
205.                    glPointSize(0.5f);
206.                    glRotatef(1.0, 0.0, 1.0, 0.0);
207.
208.                    glMap2f(GL_MAP2_VERTEX_3, 0.0, 1.0, 3, 4, 0.0, 1.0, 21, 2, &cpts[0][0][0]);
```

```
                    glMapGrid2f(30, 0.0, 1.0, 10, 0.0, 1.0);
                    glEvalMesh2(GL_POINT, 0, 30, 0, 10);

                    glMap2f(GL_MAP2_VERTEX_3, 0.0, 1.0, 3, 4, 0.0, 1.0, 21, 2, &cpts[0][3][0]);
                    glMapGrid2f(30, 0.0, 1.0, 10, 0.0, 1.0);
                    glEvalMesh2(GL_POINT, 0, 30, 0, 10);
                }
                break;
        }
        glFlush();
        glutSwapBuffers();
}

// 設置滑鼠按鍵回調函式，取得滑鼠拖動時的位置
void myMotion(int x, int y){
        y = hh - y;
        float tmp[3] = {static_cast<float>(2.0 * interw * (float)x / (float)ww - interw),
    static_cast<float>(2.0 * interh * (float)y / (float)hh - interh), 0.0};
        int i;
        if(pn == 7){
            for(i = 0; i < 2; ++i){
                if(mp[i] < tmp[i])
                    --thta[i];
                else if(mp[i] > tmp[i])
                    ++thta[i];
                if(thta[i] >= 360.0 || thta[i] <= -360.0)
                    thta[i] = 0.0;
            }
        }
        else{
            for(i = 0; i < 3; ++i)
                cpts[0][pn][i] = tmp[i];
            cpts[0][pn][0] -= TN * (float)ww / (float)hh;
        }
        mp[0] = tmp[0];
        mp[1] = tmp[1];
        glutPostRedisplay();
}
```

```
246.
247. // myMouse 使用
248. float dis2p(float a[3], float b[3]){
249.     return sqrt((a[0] - b[0]) * (a[0] - b[0]) + (a[1] - b[1]) * (a[1] - b[1]));
250. }
251.
252. // 用來換算滑鼠位置，搭配 dis2p 使用
253. void myMouse(int btn, int state, int x, int y){
254.     if(btn == GLUT_LEFT_BUTTON && state == GLUT_DOWN){
255.         y = hh - y;
256.         p[0] = mp[0] = 2.0 * interw * (float)x / (float)ww - interw - TN * (float)ww / (float)hh;
257.         p[1] = mp[1] = 2.0 * interh * (float)y / (float)hh - interh;
258.         int i;
259.
260.         for(i = 0; i < 7; ++i){
261.             if(dis2p(cpts[0][i], p) <= 1.0){
262.                 pn = i;
263.                 break;
264.             }
265.         }
266.     }
267.     else
268.         pn = 7;
269. }
270.
271. // 當應用程式視窗大小調動時調用的函式
272. void myReshape(int w, int h)
273. {
274.     // w 代表 window 寬度，h 代表 window 高度
275.     // 檢視設定，即告訴 openGL 渲染之後的圖形會繪製在哪
276.     glViewport(0,    // 視見區域的左下角在視窗中的 x 位置
277.                0,    // 視見區域的左下角在視窗中的 y 位置
278.                w,    // 視見區域的寬度
279.                h);   // 視見區域的高度
280.
281.     // 聲明接下來要做什麼
282.     glMatrixMode(GL_PROJECTION);
283.     glLoadIdentity();
```

```
284.
285.        if (w <= h){
286.            // glOrtho(left, right, bottom, top, near, far)
287.            glOrtho(-FRAME,        // 視景體左面座標（left）
288.                    FRAME,         // 視景體右面座標（right）
289.                    -FRAME * ((GLfloat) h / (GLfloat) w),     //  （buttom）
290.                    FRAME * ((GLfloat) h / (GLfloat) w),      //  （top）
291.                    -FRAME,        // (near）
292.                    FRAME);        //（far）
293.            interw = FRAME;
294.            interh = FRAME * (GLfloat) h / (GLfloat) w;
295.        }
296.        else{
297.            glOrtho(-FRAME * (GLfloat) w / (GLfloat) h, FRAME * (GLfloat) w / (GLfloat) h,
    -FRAME, FRAME, -FRAME, FRAME);
298.            interw = FRAME * (GLfloat) w / (GLfloat) h;
299.            interh = FRAME;
300.        }
301.
302.        // 聲明接下來要做什麼
303.        glMatrixMode(GL_MODELVIEW);
304.        ww = w;
305.        hh = h;
306. }
307.
308. // 七彩變換選單，改變 RainBowFunction 數值來開關效果
309. void RainbowShowMenu(int index)
310. {
311.        if(index == 1)
312.            RainBowFuntion = 1;
313.        if(index == 2)
314.            RainBowFuntion = 0;
315. }
316.
317. // 顏色選單，直接改變 light0_v array 的數值
318. void ColorMenu(int index){
319.        RainBowFuntion = 0;
320.        switch(index){
```

```
321.            case 1:     //White
322.                light0_v[0] = 1.000000;
323.                light0_v[1] = 1.000000;
324.                light0_v[2] = 1.000000;
325.                light0_v[3] = 1.000000;
326.                break;
327.            case 2:     //Red
328.                light0_v[0] = 1.000000;
329.                light0_v[1] = 0.000000;
330.                light0_v[2] = 0.000000;
331.                light0_v[3] = 0.000000;
332.                break;
333.            case 3:     //Orange
334.                light0_v[0] = 1.000000;
335.                light0_v[1] = 0.560976;
336.                light0_v[2] = 0.000000;
337.                light0_v[3] = 1.000000;
338.                break;
339.            case 4:     //Yellow
340.                light0_v[0] = 1.000000;
341.                light0_v[1] = 1.000000;
342.                light0_v[2] = 0.000000;
343.                light0_v[3] = 0.000000;
344.                break;
345.            case 5:     //Green
346.                light0_v[0] = 0.000000;
347.                light0_v[1] = 1.000000;
348.                light0_v[2] = 0.000000;
349.                light0_v[3] = 0.000000;
350.                break;
351.            case 6:     //Blue
352.                light0_v[0] = 0.000000;
353.                light0_v[1] = 0.560976;
354.                light0_v[2] = 1.000000;
355.                light0_v[3] = 1.000000;
356.                break;
357.            case 7:     //DarkBlue
358.                light0_v[0] = 0.000000;
```

```
359.                    light0_v[1] = 0.000000;
360.                    light0_v[2] = 1.000000;
361.                    light0_v[3] = 1.000000;
362.                    break;
363.            case 8:       //Purple
364.                    light0_v[0] = 0.414634;
365.                    light0_v[1] = 0.000000;
366.                    light0_v[2] = 1.000000;
367.                    light0_v[3] = 1.000000;
368.                    break;
369.        }
370.        glLightfv(GL_LIGHT0, GL_DIFFUSE, light0_v);
371.        glutPostRedisplay();
372. }
373.
374. // 主選單
375. void MainMenu(int index){
376.        if(index == 4)
377.            exit(0);
378.        type = index;
379.        glutPostRedisplay();
380. }
381.
382. // 當 RainBowFuntion == 1 時，七彩變化效果開啟，隨時間改變 light0_v array 的值
383. void RainbowShow(int value)
384. {
385.        if(RainBowFuntion == 1)
386.        {
387.            timer = timer + 1;
388.            // 紅
389.            if(timer % 7 == 0)
390.            {
391.                    light0_v[0] = 1.000000;
392.                    light0_v[1] = 0.000000;
393.                    light0_v[2] = 0.000000;
394.                    light0_v[3] = 0.000000;
395.            }
396.            // 橙
```

```
397.        if(timer % 7 == 1)
398.        {
399.            light0_v[0] = 1.000000;
400.            light0_v[1] = 0.560976;
401.            light0_v[2] = 0.000000;
402.            light0_v[3] = 1.000000;
403.        }
404.        if(timer % 7 == 2)
405.        {
406.            light0_v[0] = 1.000000;
407.            light0_v[1] = 1.000000;
408.            light0_v[2] = 0.000000;
409.            light0_v[3] = 0.000000;
410.        }
411.        if(timer % 7 == 3)
412.        {
413.            light0_v[0] = 0.000000;
414.            light0_v[1] = 1.000000;
415.            light0_v[2] = 0.000000;
416.            light0_v[3] = 0.000000;
417.        }
418.        if(timer % 7 == 4)
419.        {
420.            light0_v[0] = 0.000000;
421.            light0_v[1] = 0.560976;
422.            light0_v[2] = 1.000000;
423.            light0_v[3] = 1.000000;
424.        }
425.        if(timer % 7 == 5)
426.        {
427.            light0_v[0] = 0.000000;
428.            light0_v[1] = 0.000000;
429.            light0_v[2] = 1.000000;
430.            light0_v[3] = 1.000000;
431.        }
432.        if(timer % 7 == 6)
433.        {
434.            light0_v[0] = 0.414634;
```

```
435.                    light0_v[1] = 0.000000;
436.                    light0_v[2] = 1.000000;
437.                    light0_v[3] = 1.000000;
438.                }
439.        }
440.        glLightfv(GL_LIGHT0, GL_DIFFUSE, light0_v);
441.        glutPostRedisplay();
442.        glutTimerFunc(60,RainbowShow,1);
443. }
444.
445. // 光源、材質、背景顏色等的初始化函式
446. void myinit()
447. {
448.        GLfloat mat_specular[]={1.0, 1.0, 1.0, 1.0};
449.        GLfloat mat_diffuse[]={1.0, 1.0, 1.0, 1.0};
450.        GLfloat mat_ambient[]={0.8, 0.8, 0.8, 1.0};
451.        GLfloat mat_shininess={50.0};
452.
453.        GLfloat light_specular[]={0.6, 0.6, 0.6, 1.0};
454.        GLfloat light_diffuse[]={0.6, 0.6, 0.6, 1.0};
455.        GLfloat light_ambient[]={0.1, 0.1, 0.1, 1.0};
456.
457.        // 設置光源 glLightfv(光源編號, 光源特性, 參數數據)
458.        glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient);      // GL_AMBIENT 設置環境光屬性
459.        glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);      // GL_DIFFUSE 設置散射光屬性
460.        glLightfv(GL_LIGHT0, GL_SPECULAR, light_specular);   // GL_SPECULAR 設置鏡面反射光
    屬性
461.
462.        // 設置材質 決定打光出來的效果、對應選單的 color
463.        glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
464.        glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient);
465.        glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
466.        glMaterialf(GL_FRONT, GL_SHININESS, mat_shininess);
467.
468.        // 控制繪製指定兩點間其他點顏色的過度模式
469.        glShadeModel(GL_SMOOTH);
470.        glEnable(GL_LIGHTING);
471.        glEnable(GL_LIGHT0);
```

```
472.        glEnable(GL_LIGHT1);
473.
474.        glColorMaterial(GL_FRONT_AND_BACK, GL_AMBIENT_AND_DIFFUSE);
475.        glEnable(GL_COLOR_MATERIAL);
476.
477.        // 將視窗背景顏色設置為黑色
478.        glClearColor(0, 0, 0, 0);
479.        glColor3f(0.0, 0.0, 0.0);
480.
481.        glEnable(GL_TEXTURE_2D);
482.        //glTexImage2D(GL_TEXTURE_2D,0,3,64,64,0,GL_RGB,GL_UNSIGNED_BYTE, image);
483.        glTexParameterf(GL_TEXTURE_2D,GL_TEXTURE_WRAP_S,GL_REPEAT);
484.        glTexParameterf(GL_TEXTURE_2D,GL_TEXTURE_WRAP_T,GL_REPEAT);
485.        glTexParameterf(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_NEAREST);
486.        glTexParameterf(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_NEAREST);
487.  }
488.
489.  int main(int argc, char **argv)
490.  {
491.        glutInit(&argc, argv);
492.        glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
493.        glutInitWindowSize(900, 600);
494.        WindowId = glutCreateWindow("Midterm Homework - 406262216 ");
495.        myinit();
496.
497.        // 創建 Color 子選單
498.        int Color_Menu;
499.        Color_Menu = glutCreateMenu(ColorMenu);
500.
501.        glutAddMenuEntry("White", 1);
502.        glutAddMenuEntry("Red", 2);
503.        glutAddMenuEntry("Orange", 3);
504.        glutAddMenuEntry("Yellow", 4);
505.        glutAddMenuEntry("Green", 5);
506.        glutAddMenuEntry("Blue", 6);
507.        glutAddMenuEntry("Indigo", 7);
508.        glutAddMenuEntry("Purple", 8);
509.
```

```
510.        int RainbowShow_Menu;
511.        RainbowShow_Menu = glutCreateMenu(RainbowShowMenu);
512.        glutAddMenuEntry("Open", 1);
513.        glutAddMenuEntry("Close", 2);
514.
515.        // 創建主選單
516.        glutCreateMenu(MainMenu);
517.        glutAddMenuEntry("Line", 1);
518.        glutAddMenuEntry("Fill", 2);
519.        glutAddMenuEntry("Point", 3);
520.        glutAddSubMenu("Colors", Color_Menu);
521.        glutAddSubMenu("Rainbow Show", RainbowShow_Menu);
522.        glutAddMenuEntry("exit", 4);
523.        glutAttachMenu(GLUT_MIDDLE_BUTTON);
524.        // 設置當 Window 視窗大小改變時，作業系統也察覺到了，作業系統會呼叫 myReshape
    Function
525.        glutReshapeFunc(myReshape);
526.        // OpenGL 的繪圖動作都寫在其所指定的自定函式 display
527.        glutDisplayFunc(display);
528.        glutMouseFunc (myMouse);
529.        glutMotionFunc(myMotion);
530.        glutTimerFunc(33,RainbowShow,1);
531.
532.        glEnable(GL_DEPTH_TEST);
533.        glEnable(GL_MAP1_VERTEX_3);
534.        glEnable(GL_MAP2_VERTEX_3);
535.        glEnable(GL_AUTO_NORMAL);
536.        glEnable(GL_MAP2_TEXTURE_COORD_2);
537.
538.        glutMainLoop();
539.
540.        return 0;
541.  }
```