**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

1. 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

1. 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。
2. 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

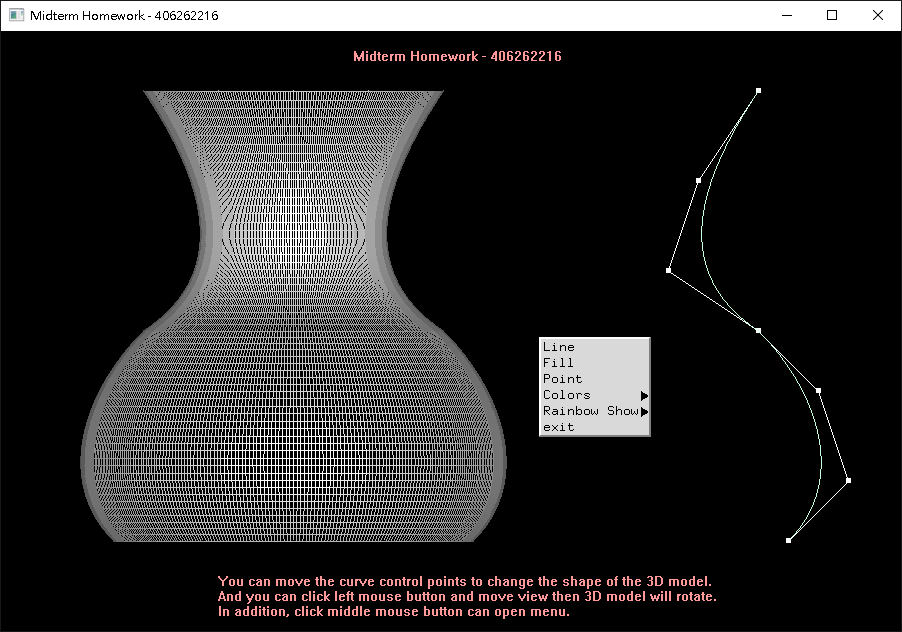
1. 初始化函式 void myinit()：光源、材質、背景顏色等的初始化函式
2. 主程式 int main(int argc, char \*\*argv)

**討論**

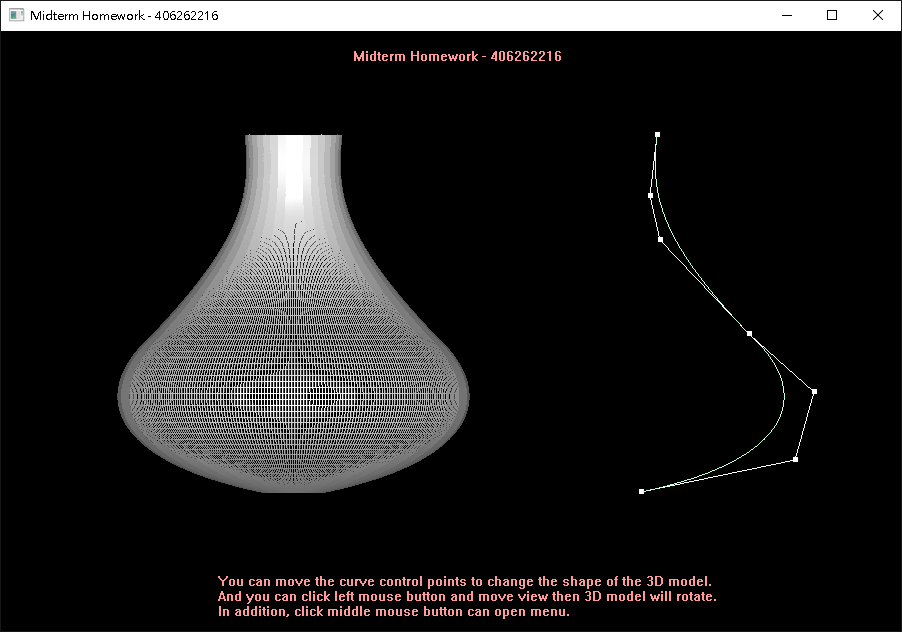
一開始我是使用Mac寫作業，MacOS的環境只要使用framework就可以直接執行，但後來覺得還是繳交exe檔案比較好，所以移植到Windows這個環境來寫，花了點時間在VSCode上架環境，但較可惜的是原本Mac上可以裝起來以OpenGL為基底的UI介面GLUI，在Window上找不到對應的dll檔所以裝不起來。這個作業有很多觀念混在一起，在理解上撞牆了好一陣子，最後在與其他修課的朋友們討論和不斷地爬文、觀看網路上的教學影片才慢慢地抓到要從哪裡下手，在了解到哪些code有什麼用處之後就比較上手，但最後想要新增貼圖的功能又撞牆了，查了網路上load bmp的方式，圖片都無法順利貼上，甚至電腦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**

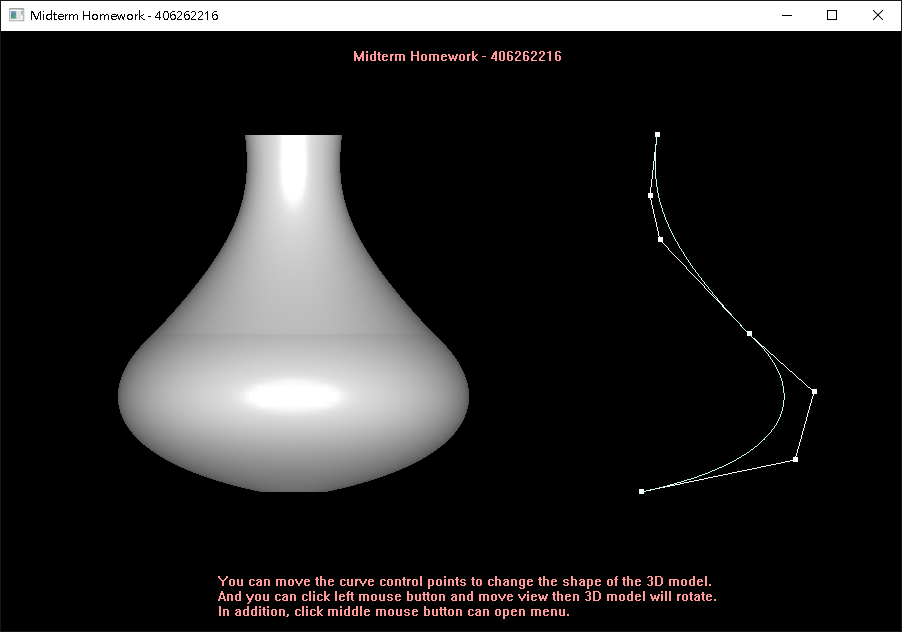
整個執行畫面的最頂和最底分別加上字，分別為標題和使用方法。可以透過滑鼠中鍵來開啟選單，選單主要功能分為繪出3D物件是線框式 （Line）呈現、塗色式（Fill）呈現及點狀式（Point）呈現。而子選單Colors則可以針對3D物件有7種顏色可以選擇，Rainbow Show 則可以呈現七彩變化的特殊效果，另外點選exit變可結束此應用程式。



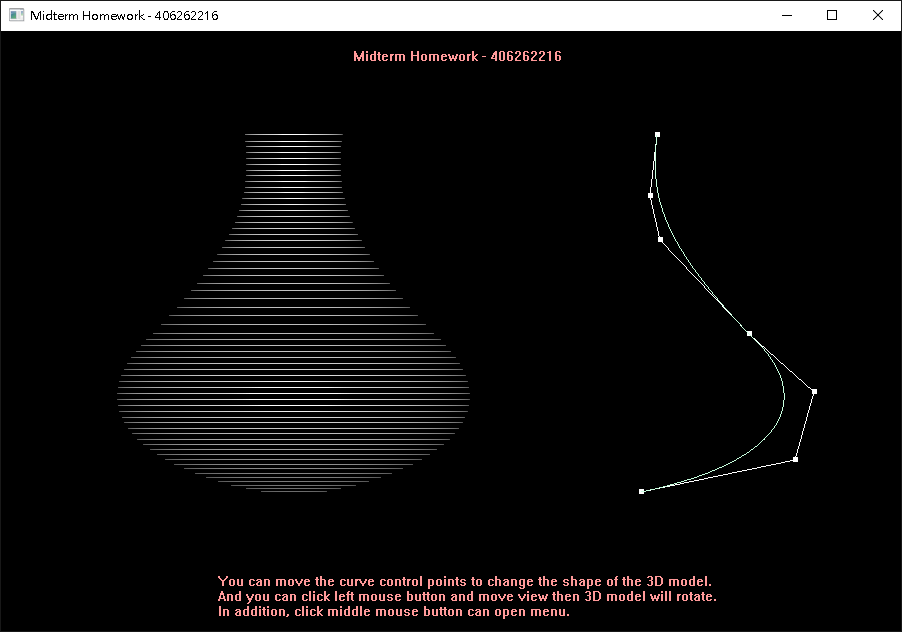
**線框式 （Line）呈現**



**塗色式（Fill）呈現**

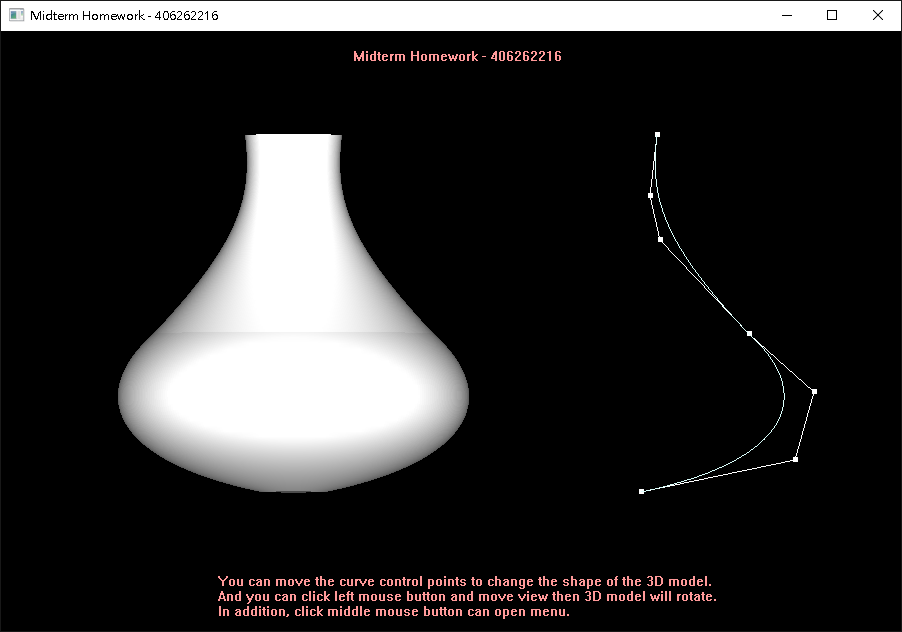
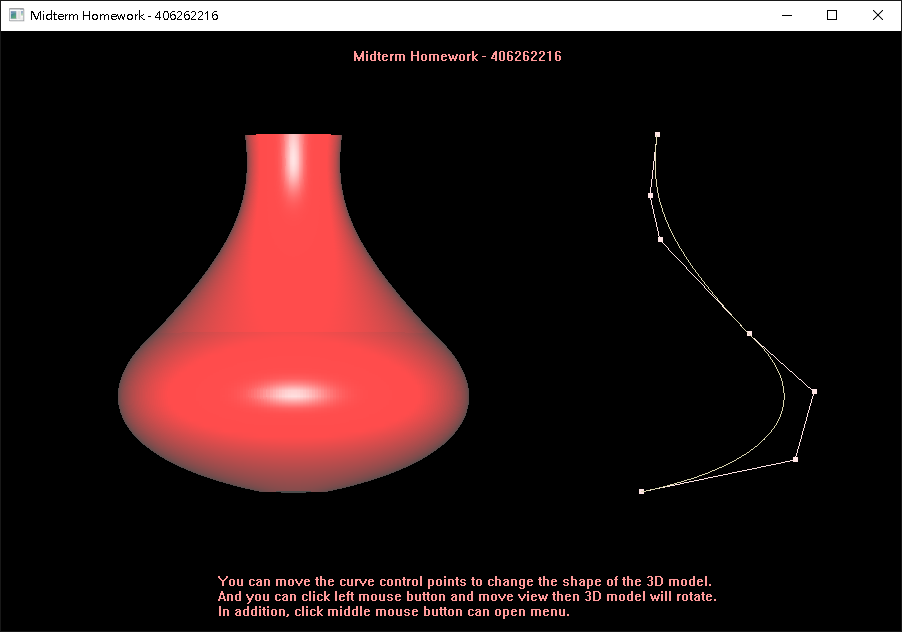


**點狀式（Point）呈現**

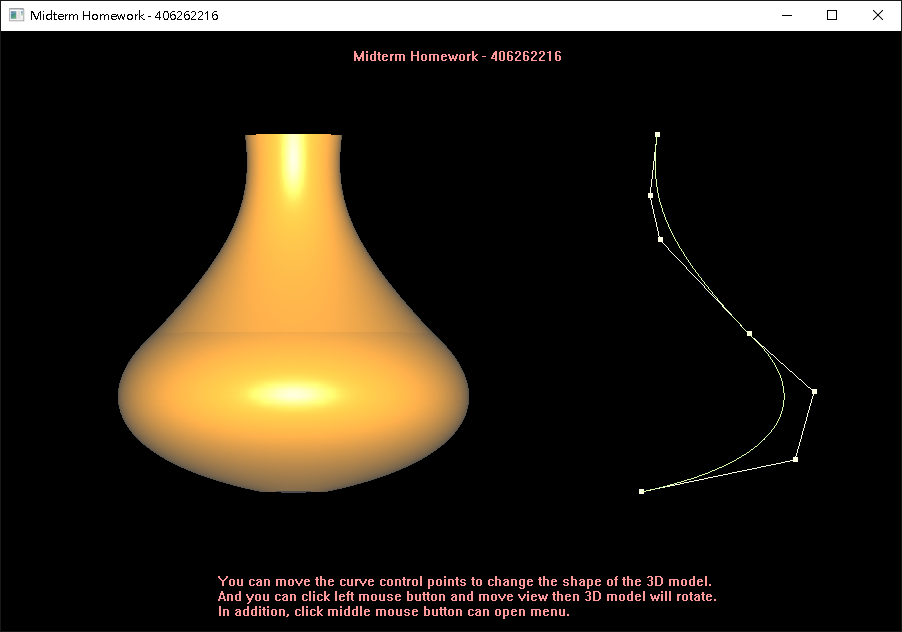
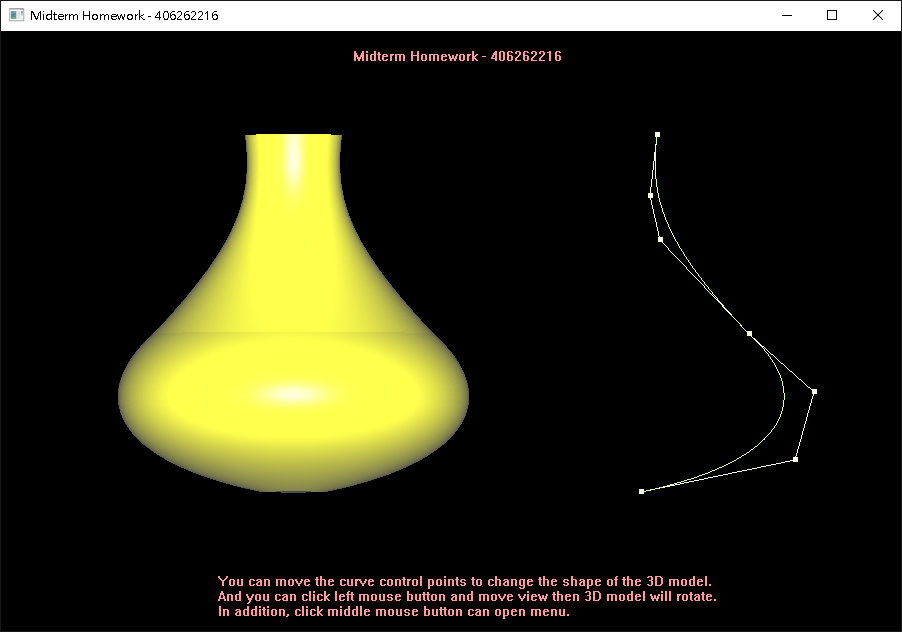


**7種Color呈現**

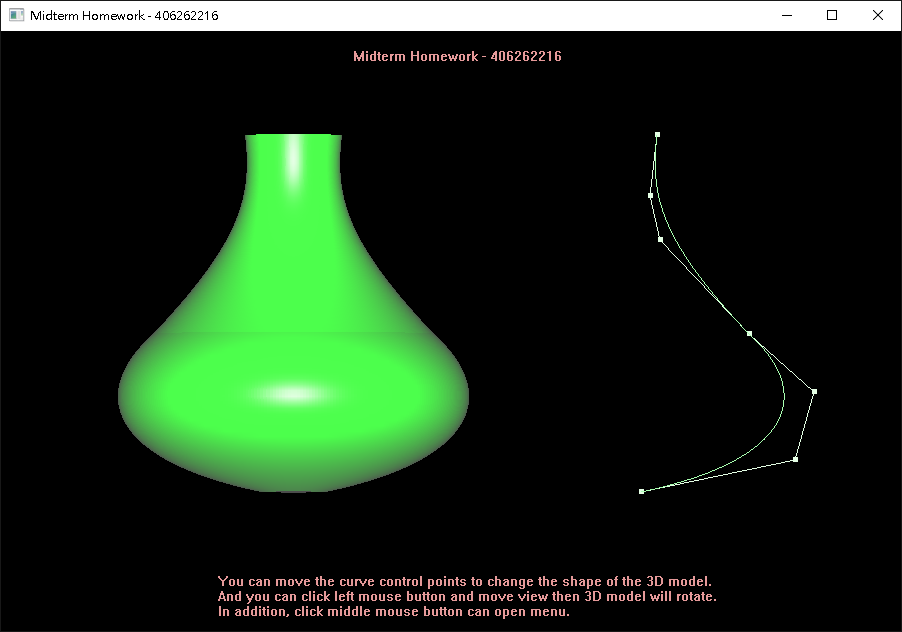
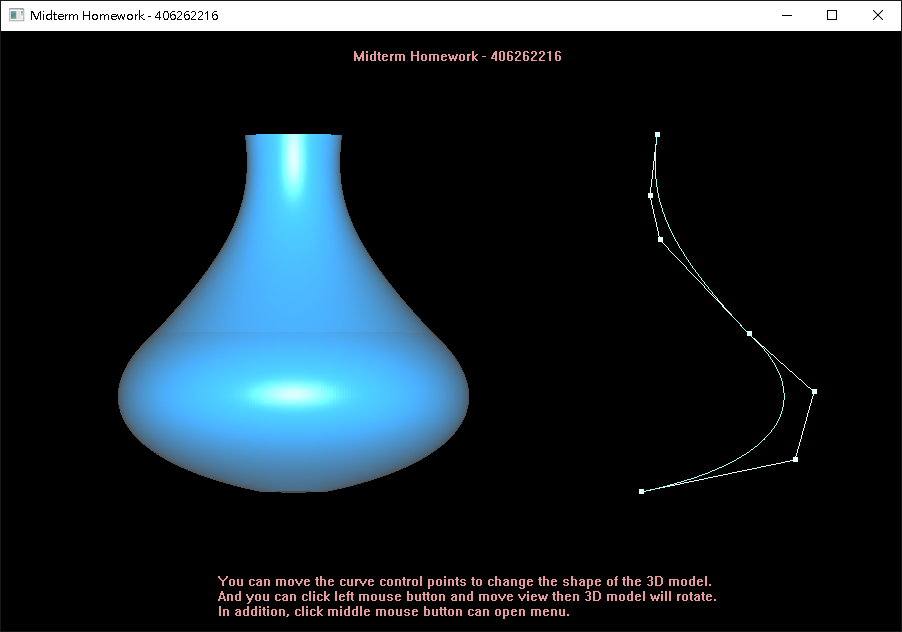
**白色（White） 紅色（Red）**

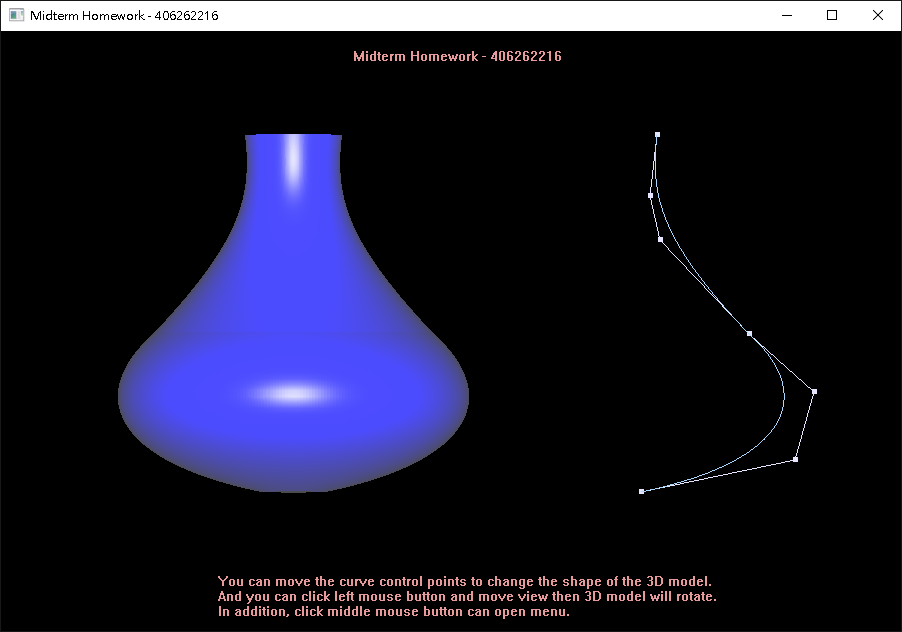
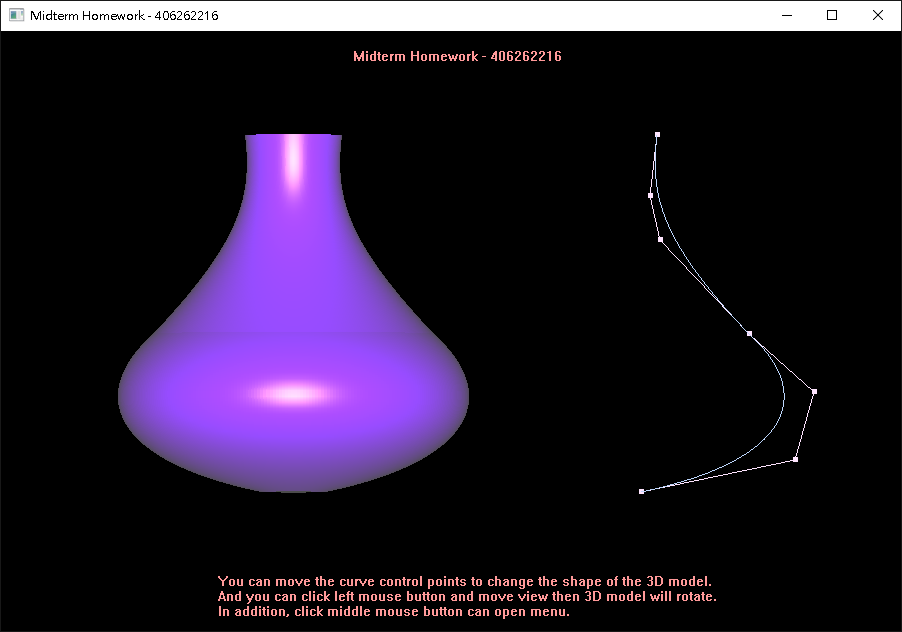
**橘色（Orange） 黃色（Yellow）**

**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. #include <cstdlib> 9. #include <cmath> 10. #include <cstdio> 11. #include <GL/glut.h> 12. #include <GL/glui.h> 13. #include <windows.h> 14. #define PI 3.14159265 15. #define MS 10.0 16. #define TN 3.5 17. #define FRAME 10 18. #define MAX\_CHAR 128 19. int type = 1; 20. int timer = 0; 21. int RainBowFuntion = 0; 22. int WindowId; 23. // 光源參數 24. GLfloat light0\_v[4] = {0.6, 0.6, 0.6, 1}; 25. // 設置控制點座標 26. 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}}}; 27. GLubyte image[64][64][3]; 28. 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))}}; 29. 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))}}; 30. 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))}}; 31. GLfloat thta[2] = {0.0, 0.0}; 32. float p[3], mp[3], interw, interh, mrgb[3] = {0.0, 0.4, 0.8}; 33. int pn = 7, ww, hh; 34. // 繪製字樣 35. void printString(const char\* str) 36. { 37. static int isFirstCall = 1; 38. static GLuint lists; 39. // 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 40. // 为每一个ASCII字符产生一个显示列表 41. if( isFirstCall ) { 42. isFirstCall = 0; 43. // 申請MAX\_CHAR個連續的顯示列表編號 44. lists = glGenLists(MAX\_CHAR); 45. // 把每個字符的繪製命令都裝到對應的顯示列表中 46. wglUseFontBitmaps(wglGetCurrentDC(), 0, MAX\_CHAR, lists); 47. } 48. // 調用每個字符對應的顯示列表，繪製每個字符 49. for(; \*str!='\0'; ++str) 50. glCallList(lists + \*str); 51. } 52. // 計算旋轉角度使用，搭配 display 53. void vmult(float m[3][3], float v[3], float r[3]) 54. { 55. int i, j; 56. for(i = 0; i < 3; ++i){ 57. for(j = 0, r[i] = 0.0; j < 3; ++j){ 58. r[i] += m[i][j] \* v[j]; 59. } 60. } 61. } 62. void display(void) 63. { 64. glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT); 65. // 初始化 Martix 66. glLoadIdentity(); 67. glPushMatrix(); 68. glTranslated(TN \* (float)ww / (float)hh, 0.0, 0.0); 70. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*產生貝式曲線 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ 71. glColor3f(0.2, 0.8, 0.3); 73. // 為曲線建立對應, u 的數值由 0 到 1 74. glMap1f(GL\_MAP1\_VERTEX\_3, 0.0, 1.0, 3, 4, &cpts[0][0][0]); 75. /\* 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)\*/ 76. glMapGrid1f(30, 0.0, 1.0); 77. /\* glMapGrid1f(網格中間的階段, u值起頭, u值結尾)\*/ 78. glEvalMesh1(GL\_LINE, 0, 30); 79. /\* glEvalMesh1(mode,i1,i2) 80. 對所有已經啟用的求值器應用定義一維網格。 81. mode參數可以是GL\_POINT、GL\_LINE。 82. 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\*/ 84. glMap1f(GL\_MAP1\_VERTEX\_3, 0.0, 1.0, 3, 4, &cpts[0][3][0]); 85. glMapGrid1f(30, 0.0, 1.0); 86. glEvalMesh1(GL\_LINE, 0, 30); 88. // 設置控制點大小 89. glPointSize(5.0); 90. glColor3f(1.0, 1.0, 1.0); 91. glBegin(GL\_POINTS); 92. glVertex3fv(cpts[0][0]); 93. glVertex3fv(cpts[0][1]); 94. glVertex3fv(cpts[0][2]); 95. glVertex3fv(cpts[0][3]); 96. glVertex3fv(cpts[0][4]); 97. glVertex3fv(cpts[0][5]); 98. glVertex3fv(cpts[0][6]); 99. glEnd(); 101. // 將每個控制點連線 102. glColor3f(1.0, 1.0, 1.0); 103. glBegin(GL\_LINE\_STRIP); 104. { 105. for(int i = 0 ; i < 7 ; i++) 106. glVertex3fv(cpts[0][i]); 107. } 108. glEnd(); 109. // 繪製字 110. glColor3f(1.0f, 0.0f, 0.0f); 111. glRasterPos2f(-8.5f, 9.0f); 112. printString("Midterm Homework - 406262216"); 113. glRasterPos2f(-13.0f, -8.5f); 114. printString("You can move the curve control points to change the shape of the 3D model."); 115. glRasterPos2f(-13.0f, -9.0f); 116. printString("And you can click left mouse button and move view then 3D model will rotate."); 117. glRasterPos2f(-13.0f, -9.5f); 118. printString("In addition, click middle mouse button can open menu."); 119. // 模型顏色 120. glColor3f(1, 1, 1); 121. glBegin(GL\_LINE); 122. glVertex3fv(cpts[0][0]); 123. glVertex3fv(cpts[0][1]); 124. glEnd(); 125. glPopMatrix(); 126. glTranslated(-TN \* (float)ww / (float)hh, 0.0, 0.0); 127. // 讓模型能夠旋轉 128. glRotatef(thta[1], 1.0, 0.0, 0.0); 129. glRotatef(thta[2], 0.0, 1.0, 0.0); 130. glRotatef(thta[0], 0.0, 0.0, 1.0); 132. int i; 133. // 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 134. switch(type){ 135. case 1: 136. for(i = 0; i < 7; ++i) 137. vmult(r10, cpts[0][i], cpts[1][i]); 138. break; 139. case 2: 140. for(i = 0; i < 7; ++i) 141. vmult(r1, cpts[0][i], cpts[1][i]); 142. break; 143. case 3: 144. for(i = 0; i < 7; ++i) 145. vmult(r100, cpts[0][i], cpts[1][i]); 146. break; 147. } 149. // type 對應主選單點選的選項，決定呈現的方式：Line、Fill、Point 150. switch(type){ 151. case 1: 152. // Line 153. for(i = 0; i < 36; ++i){ 154. glRotatef(10.0, 0.0, 1.0, 0.0); 156. glMap2f(GL\_MAP2\_VERTEX\_3, 0.0, 1.0, 3, 4, 0.0, 1.0, 21, 2, &cpts[0][0][0]); 157. //glMapGrid2f() 從 a 到 b 的範圍對映一個包含 n 個點的網格 158. glMapGrid2f(30, // 產生網格數 159. 0.0, // u值下界 160. 1.0, // u值上界 161. 10, 162. 0.0, 163. 1.0); 164. // 計算網格 165. glEvalMesh2(GL\_LINE, 0, 30, 0, 10); 167. glMap2f(GL\_MAP2\_VERTEX\_3, 0.0, 1.0, 3, 4, 0.0, 1.0, 21, 2, &cpts[0][3][0]); 168. glMapGrid2f(30, 0.0, 1.0, 10, 0.0, 1.0); 169. glEvalMesh2(GL\_LINE, 0, 30, 0, 10); 170. } 171. break; 172. case 2: 173. // Fill 174. for(i = 0; i < 360; ++i){ 175. glRotatef(1.0, 0.0, 1.0, 0.0); 177. glMap2f(GL\_MAP2\_VERTEX\_3, 0.0, 1.0, 3, 4, 0.0, 1.0, 21, 2, &cpts[0][0][0]); 178. glMapGrid2f(30, 0.0, 1.0, 10, 0.0, 1.0); 179. glEvalMesh2(GL\_FILL, 0, 30, 0, 10); 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\_FILL, 0, 30, 0, 10); 184. } 185. break; 186. case 3: 187. // Point 188. for(i = 0; i < 360; ++i){ 189. glEnable(GL\_POINT\_SIZE); 190. glEnable(GL\_TEXTURE\_2D); 191. glPointSize(0.5f); 192. glRotatef(1.0, 0.0, 1.0, 0.0); 194. glMap2f(GL\_MAP2\_VERTEX\_3, 0.0, 1.0, 3, 4, 0.0, 1.0, 21, 2, &cpts[0][0][0]); 195. glMapGrid2f(30, 0.0, 1.0, 10, 0.0, 1.0); 196. glEvalMesh2(GL\_POINT, 0, 30, 0, 10); 198. glMap2f(GL\_MAP2\_VERTEX\_3, 0.0, 1.0, 3, 4, 0.0, 1.0, 21, 2, &cpts[0][3][0]); 199. glMapGrid2f(30, 0.0, 1.0, 10, 0.0, 1.0); 200. glEvalMesh2(GL\_POINT, 0, 30, 0, 10); 201. } 202. break; 203. } 204. glFlush(); 205. glutSwapBuffers(); 206. } 207. // 設置滑鼠按鍵回調函式，取得滑鼠拖動時的位置 208. void myMotion(int x, int y){ 209. y = hh - y; 210. 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}; 211. int i; 212. if(pn == 7){ 213. for(i = 0; i < 2; ++i){ 214. if(mp[i] < tmp[i]) 215. --thta[i]; 216. else if(mp[i] > tmp[i]) 217. ++thta[i]; 218. if(thta[i] >= 360.0 || thta[i] <= -360.0) 219. thta[i] = 0.0; 220. } 221. } 222. else{ 223. for(i = 0; i < 3; ++i) 224. cpts[0][pn][i] = tmp[i]; 225. cpts[0][pn][0] -= TN \* (float)ww / (float)hh; 226. } 227. mp[0] = tmp[0]; 228. mp[1] = tmp[1]; 229. glutPostRedisplay(); 230. } 231. // myMouse 使用 232. float dis2p(float a[3], float b[3]){ 233. return sqrt((a[0] - b[0]) \* (a[0] - b[0]) + (a[1] - b[1]) \* (a[1] - b[1])); 234. } 235. // 用來換算滑鼠位置，搭配 dis2p 使用 236. void myMouse(int btn, int state, int x, int y){ 237. if(btn == GLUT\_LEFT\_BUTTON && state == GLUT\_DOWN){ 238. y = hh - y; 239. p[0] = mp[0] = 2.0 \* interw \* (float)x / (float)ww - interw - TN \* (float)ww / (float)hh; 240. p[1] = mp[1] = 2.0 \* interh \* (float)y / (float)hh - interh; 241. int i; 243. for(i = 0; i < 7; ++i){ 244. if(dis2p(cpts[0][i], p) <= 1.0){ 245. pn = i; 246. break; 247. } 248. } 249. } 250. else 251. pn = 7; 252. } 253. // 當應用程式視窗大小調動時調用的函式 254. void myReshape(int w, int h) 255. { 256. // w 代表 window 寬度，h 代表 window 高度 257. // 檢視設定，即告訴 openGL 渲染之後的圖形會繪製在哪 258. glViewport(0, // 視見區域的左下角在視窗中的 x 位置 259. 0, // 視見區域的左下角在視窗中的 y 位置 260. w, // 視見區域的寬度 261. h); // 視見區域的高度 262. // 聲明接下來要做什麼 263. glMatrixMode(GL\_PROJECTION); 264. glLoadIdentity(); 266. if (w <= h){ 267. // glOrtho(left, right, bottom, top, near, far) 268. glOrtho(-FRAME, // 視景體左面座標（left） 269. FRAME, // 視景體右面座標（right） 270. -FRAME \* ((GLfloat) h / (GLfloat) w), // （buttom） 271. FRAME \* ((GLfloat) h / (GLfloat) w), // （top） 272. -FRAME, // (near） 273. FRAME); //（far） 274. interw = FRAME; 275. interh = FRAME \* (GLfloat) h / (GLfloat) w; 276. } 277. else{ 278. glOrtho(-FRAME \* (GLfloat) w / (GLfloat) h, FRAME \* (GLfloat) w / (GLfloat) h, -FRAME, FRAME, -FRAME, FRAME); 279. interw = FRAME \* (GLfloat) w / (GLfloat) h; 280. interh = FRAME; 281. } 283. // 聲明接下來要做什麼 284. glMatrixMode(GL\_MODELVIEW); 285. ww = w; 286. hh = h; 287. } 288. // 七彩變換選單，改變 RainBowFunction 數值來開關效果 289. void RainbowShowMenu(int index) 290. { 291. if(index == 1) 292. RainBowFuntion = 1; 293. if(index == 2) 294. RainBowFuntion = 0; 295. } 296. // 顏色選單，直接改變light0\_v array 的數值 297. void ColorMenu(int index){ 298. RainBowFuntion = 0; 299. switch(index){ 300. case 1: //White 301. light0\_v[0] = 1.000000; 302. light0\_v[1] = 1.000000; 303. light0\_v[2] = 1.000000; 304. light0\_v[3] = 1.000000; 305. break; 306. case 2: //Red 307. light0\_v[0] = 1.000000; 308. light0\_v[1] = 0.000000; 309. light0\_v[2] = 0.000000; 310. light0\_v[3] = 0.000000; 311. break; 312. case 3: //Orange 313. light0\_v[0] = 1.000000; 314. light0\_v[1] = 0.560976; 315. light0\_v[2] = 0.000000; 316. light0\_v[3] = 1.000000; 317. break; 318. case 4: //Yellow 319. light0\_v[0] = 1.000000; 320. light0\_v[1] = 1.000000; 321. light0\_v[2] = 0.000000; 322. light0\_v[3] = 0.000000; 323. break; 324. case 5: //Green 325. light0\_v[0] = 0.000000; 326. light0\_v[1] = 1.000000; 327. light0\_v[2] = 0.000000; 328. light0\_v[3] = 0.000000; 329. break; 330. case 6: //Blue 331. light0\_v[0] = 0.000000; 332. light0\_v[1] = 0.560976; 333. light0\_v[2] = 1.000000; 334. light0\_v[3] = 1.000000; 335. break; 336. case 7: //DarkBlue 337. light0\_v[0] = 0.000000; 338. light0\_v[1] = 0.000000; 339. light0\_v[2] = 1.000000; 340. light0\_v[3] = 1.000000; 341. break; 342. case 8: //Purple 343. light0\_v[0] = 0.414634; 344. light0\_v[1] = 0.000000; 345. light0\_v[2] = 1.000000; 346. light0\_v[3] = 1.000000; 347. break; 348. } 349. glLightfv(GL\_LIGHT0, GL\_DIFFUSE, light0\_v); 350. glutPostRedisplay(); 351. } 352. // 主選單 353. void MainMenu(int index){ 354. if(index == 4) 355. exit(0); 356. type = index; 357. glutPostRedisplay(); 358. } 359. // 當 RainBowFuntion == 1 時，七彩變化效果開啟，隨時間改變light0\_v array 的值 360. void RainbowShow(int value) 361. { 362. if(RainBowFuntion == 1) 363. { 364. timer = timer + 1; 365. // 紅 366. if(timer % 7 == 0) 367. { 368. light0\_v[0] = 1.000000; 369. light0\_v[1] = 0.000000; 370. light0\_v[2] = 0.000000; 371. light0\_v[3] = 0.000000; 372. } 373. // 橙 374. if(timer % 7 == 1) 375. { 376. light0\_v[0] = 1.000000; 377. light0\_v[1] = 0.560976; 378. light0\_v[2] = 0.000000; 379. light0\_v[3] = 1.000000; 380. } 381. if(timer % 7 == 2) 382. { 383. light0\_v[0] = 1.000000; 384. light0\_v[1] = 1.000000; 385. light0\_v[2] = 0.000000; 386. light0\_v[3] = 0.000000; 387. } 388. if(timer % 7 == 3) 389. { 390. light0\_v[0] = 0.000000; 391. light0\_v[1] = 1.000000; 392. light0\_v[2] = 0.000000; 393. light0\_v[3] = 0.000000; 394. } 395. if(timer % 7 == 4) 396. { 397. light0\_v[0] = 0.000000; 398. light0\_v[1] = 0.560976; 399. light0\_v[2] = 1.000000; 400. light0\_v[3] = 1.000000; 401. } 402. if(timer % 7 == 5) 403. { 404. light0\_v[0] = 0.000000; 405. light0\_v[1] = 0.000000; 406. light0\_v[2] = 1.000000; 407. light0\_v[3] = 1.000000; 408. } 409. if(timer % 7 == 6) 410. { 411. light0\_v[0] = 0.414634; 412. light0\_v[1] = 0.000000; 413. light0\_v[2] = 1.000000; 414. light0\_v[3] = 1.000000; 415. } 416. } 417. glLightfv(GL\_LIGHT0, GL\_DIFFUSE, light0\_v); 418. glutPostRedisplay(); 419. glutTimerFunc(60,RainbowShow,1); 420. } 421. // 光源、材質、背景顏色等的初始化函式 422. void myinit() 423. { 424. GLfloat mat\_specular[]={1.0, 1.0, 1.0, 1.0}; 425. GLfloat mat\_diffuse[]={1.0, 1.0, 1.0, 1.0}; 426. GLfloat mat\_ambient[]={0.8, 0.8, 0.8, 1.0}; 427. GLfloat mat\_shininess={50.0}; 429. GLfloat light\_specular[]={0.6, 0.6, 0.6, 1.0}; 430. GLfloat light\_diffuse[]={0.6, 0.6, 0.6, 1.0}; 431. GLfloat light\_ambient[]={0.1, 0.1, 0.1, 1.0}; 433. // 設置光源 glLightfv(光源編號, 光源特性, 參數數據) 434. glLightfv(GL\_LIGHT0, GL\_AMBIENT, light\_ambient); // GL\_AMBIENT 設置環境光屬性 435. glLightfv(GL\_LIGHT0, GL\_DIFFUSE, light\_diffuse); // GL\_DIFFUSE 設置散射光屬性 436. glLightfv(GL\_LIGHT0, GL\_SPECULAR, light\_specular); // GL\_SPECULAR 設置鏡面反射光屬性 438. // 設置材質 決定打光出來的效果、對應選單的color 439. glMaterialfv(GL\_FRONT, GL\_SPECULAR, mat\_specular); 440. glMaterialfv(GL\_FRONT, GL\_AMBIENT, mat\_ambient); 441. glMaterialfv(GL\_FRONT, GL\_DIFFUSE, mat\_diffuse); 442. glMaterialf(GL\_FRONT, GL\_SHININESS, mat\_shininess); 444. // 控制繪製指定兩點間其他點顏色的過度模式 445. glShadeModel(GL\_SMOOTH); 446. glEnable(GL\_LIGHTING); 447. glEnable(GL\_LIGHT0); 448. glEnable(GL\_LIGHT1); 449. glColorMaterial(GL\_FRONT\_AND\_BACK, GL\_AMBIENT\_AND\_DIFFUSE); 450. glEnable(GL\_COLOR\_MATERIAL); 452. // 將視窗背景顏色設置為黑色 453. glClearColor(0, 0, 0, 0); 454. glColor3f(0.0, 0.0, 0.0); 455. glEnable(GL\_TEXTURE\_2D); 456. //glTexImage2D(GL\_TEXTURE\_2D,0,3,64,64,0,GL\_RGB,GL\_UNSIGNED\_BYTE, image); 457. glTexParameterf(GL\_TEXTURE\_2D,GL\_TEXTURE\_WRAP\_S,GL\_REPEAT); 458. glTexParameterf(GL\_TEXTURE\_2D,GL\_TEXTURE\_WRAP\_T,GL\_REPEAT); 459. glTexParameterf(GL\_TEXTURE\_2D,GL\_TEXTURE\_MAG\_FILTER,GL\_NEAREST); 460. glTexParameterf(GL\_TEXTURE\_2D,GL\_TEXTURE\_MIN\_FILTER,GL\_NEAREST); 461. } 462. int main(int argc, char \*\*argv) 463. { 464. glutInit(&argc, argv); 465. glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB | GLUT\_DEPTH); 466. glutInitWindowSize(900, 600); 467. WindowId = glutCreateWindow("Midterm Homework - 406262216 "); 468. myinit(); 469. // 創建 Color 子選單 470. int Color\_Menu; 471. Color\_Menu = glutCreateMenu(ColorMenu); 472. glutAddMenuEntry("White", 1); 473. glutAddMenuEntry("Red", 2); 474. glutAddMenuEntry("Orange", 3); 475. glutAddMenuEntry("Yellow", 4); 476. glutAddMenuEntry("Green", 5); 477. glutAddMenuEntry("Blue", 6); 478. glutAddMenuEntry("Indigo", 7); 479. glutAddMenuEntry("Purple", 8); 480. int RainbowShow\_Menu; 481. RainbowShow\_Menu = glutCreateMenu(RainbowShowMenu); 482. glutAddMenuEntry("Open", 1); 483. glutAddMenuEntry("Close", 2); 484. // 創建主選單 485. glutCreateMenu(MainMenu); 486. glutAddMenuEntry("Line", 1); 487. glutAddMenuEntry("Fill", 2); 488. glutAddMenuEntry("Point", 3); 489. glutAddSubMenu("Colors", Color\_Menu); 490. glutAddSubMenu("Rainbow Show", RainbowShow\_Menu); 491. glutAddMenuEntry("exit", 4); 492. glutAttachMenu(GLUT\_MIDDLE\_BUTTON); 493. // 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 494. glutReshapeFunc(myReshape); 495. // OpenGL的繪圖動作都寫在其所指定的自定函式 display 496. glutDisplayFunc(display); 497. glutMouseFunc (myMouse); 498. glutMotionFunc(myMotion); 499. glutTimerFunc(33,RainbowShow,1); 501. glEnable(GL\_DEPTH\_TEST); 502. glEnable(GL\_MAP1\_VERTEX\_3); 503. glEnable(GL\_MAP2\_VERTEX\_3); 504. glEnable(GL\_AUTO\_NORMAL); 505. glEnable(GL\_MAP2\_TEXTURE\_COORD\_2); 506. glutMainLoop(); 508. return 0; 509. } |