



# Politechnika Wrocławska

---

## Sprawozdanie 2

Ćwiczenie 4. Oświetlenie scen

Krzysztof Zalewa

2.12.2024

# Spis treści

<b>1</b>	<b>Wstęp teoretyczny</b>	<b>2</b>
1.1	Temat1 . . . . .	2
1.1.1	Temat 1.1 . . . . .	2
<b>2</b>	<b>Zadanie laboratoryjne</b>	<b>2</b>
2.1	Opis działania programu . . . . .	2
2.2	Kod programu . . . . .	2
<b>3</b>	<b>Wnioski</b>	<b>12</b>
<b>4</b>	<b>Źródła</b>	<b>12</b>

## 1 Wstęp teoretyczny

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales...

### 1.1 Temat1

#### 1.1.1 Temat 1.1

## 2 Zadanie laboratoryjne

### 2.1 Treść zadania

### 2.2 Opis działania programu

Zgodnie z treścią zadania program rysuje 4 obiekty. Domyślnie jajko i czajnik rysowane są w kolorze czarnym. Jednakże jest możliwość zmiany koloru na losowy. Wyświetlone obiekty można obracać za pomocą klawiatury (Przycisk musi być wcisnięty i przytrzymany).

#### Kontrola obrotu:

**A D** - obrót po osi Y

**W S** - obrót po osi X

**Q E** - obrót po osi Z

**ESC** - Powrót do menu (okno konsolowe)

**Ruch myszy w osi X** - Obrót kamery w osi X

**Ruch myszy w osi Y** - Obrót kamery w osi Y

**Scroll up** - Przybliżenie obiektu

**Scroll down** - Oddalenie obiektu

### 2.3 Kod programu

```
1 #include <windows.h>
2 #include <iostream>
3 #include <GL/glu.h>
4 #include <vector>
5 #include <math.h>
6 #define FREEGLUT_STATIC
7 #include <GL/freeglut.h>
8 using namespace std;
```

```

9  HWND consoleWindow;
10 HWND glutWindow;
11
12 GLfloat deg = 0;
13 int sx =0,sy = 0,sz = 0;
14 bool spin = false;
15 bool drawTeapot = true;
16 bool color = false;
17 int eggMode = 0;
18 float totalRotationX = 0.0f,totalRotationY = 0.0f,totalRotationZ = 0.0f;
19 int radius = 6,lastX = 0,lastY = 0;
20 float cameraRotationX = 0.0f,cameraRotationY = 0.0f,cameraRotationZ = radius;
21 float phi = 0.0f;
22 float theta = 0.0f;
23 struct pointsRgb{
24     //Pozycja
25     float x = 0.0;
26     float y = 0.0;
27     float z = 0.0;
28     //Kolor
29     float r = 0.0;
30     float g = 0.0;
31     float b = 0.0;
32 }typedef pointsRgb;
33
34 class Egg{
35     private:
36     int density;
37     vector<vector<pointsRgb>> pointsMatrix;
38     float randFloat(){
39         return (float)rand()/(float)(RAND_MAX);
40     }
41     public:
42     Egg(int density ) : density(density){
43         pointsMatrix.resize(density,vector<pointsRgb>(density));
44     }
45     vector<vector<pointsRgb>> getPointsMatrix(){
46         return pointsMatrix;
47     }
48     void generateMatrix(float scale){
49         for(int u=0;u<(density);u++){
50             float _u = 0.5/((float)density-1);
51             _u *= u;
52             if(u==density-1){
53                 pointsMatrix[u][0].y = scale*((160*pow(_u,4)) - (320*pow(_u,3)) +
54                 ↪ (160 * pow(_u,2)) - 5);
55                 if(color){
56                     pointsMatrix[u][0].r = randFloat();
57                     pointsMatrix[u][0].g = randFloat();
58                     pointsMatrix[u][0].b = randFloat();
59                 }else{
60                     pointsMatrix[u][0].r = 0.0f;
61                     pointsMatrix[u][0].g = 0.0f;
62                     pointsMatrix[u][0].b = 0.0f;
63                 }
64             }
65             break;

```

```

64     }
65     for(int v=0;v<density;v++){
66         float _v = v/((float)density);
67         _v *= 2.0f;
68         pointsMatrix[u][v].x = scale*((-90*pow(_u,5) + 225*pow(_u,4) -
        ↪ 270*pow(_u,3) + 180*pow(_u,2) - 45*_u) * cos(M_PI*_v));
69         pointsMatrix[u][v].y = scale*(160*pow(_u,4) - 320*pow(_u,3) + 160
        ↪ * pow(_u,2) - 5);
70         pointsMatrix[u][v].z = scale*((-90*pow(_u,5) + 225*pow(_u,4) -
        ↪ 270*pow(_u,3) + 180*pow(_u,2) - 45*_u) * sin(M_PI*_v));
71         if(color){
72             pointsMatrix[u][v].r = randFloat();
73             pointsMatrix[u][v].g = randFloat();
74             pointsMatrix[u][v].b = randFloat();
75         }else{
76             pointsMatrix[u][v].r = 0.0f;
77             pointsMatrix[u][v].g = 0.0f;
78             pointsMatrix[u][v].b = 0.0f;
79         }
80     }
81 }
82 }
83 void draw(int model){
84     switch (model)
85     {
86     case 1:
87         glPointSize(5.0f);
88         glBegin(GL_POINTS);
89         for(int u=0;u<density-1;u++){
90             if(u==0){
91                 glColor3f(pointsMatrix[u][0].r,pointsMatrix[u][0].g,pointsMatr
92                 ↪ ix[u][0].b);
93                 glVertex3f(pointsMatrix[u][0].x,pointsMatrix[u][0].y,pointsMat
94                 ↪ rix[u][0].z);
95                 continue;
96             }
97             if(u==density-2){
98                 glColor3f(pointsMatrix[u+1][0].r,pointsMatrix[u+1][0].g,points
99                 ↪ Matrix[u+1][0].b);
100                 glVertex3f(pointsMatrix[u+1][0].x,pointsMatrix[u+1][0].y,point
101                 ↪ sMatrix[u+1][0].z);
102                 break;
103             }
104             for(int v=0;v<density;v++){
105                 glColor3f(pointsMatrix[u][v].r,pointsMatrix[u][v].g,pointsMatr
106                 ↪ ix[u][v].b);
107                 glVertex3f(pointsMatrix[u][v].x,pointsMatrix[u][v].y,pointsMat
108                 ↪ rix[u][v].z);
109             }
110         }
111     }
112     glEnd();
113     break;
114 case 2:
115     glBegin(GL_LINES);
116     for(int u=0;u<density-1;u++){
117         if(u==0){

```

```

111         for(int v=0;v<density;v++){
112             glColor3f(pointsMatrix[u][0].r,pointsMatrix[u][0].g,pointsM
113                 ↪ Matrix[u][0].b);
114             glVertex3f(pointsMatrix[u][0].x,pointsMatrix[u][0].y,point
115                 ↪ sMatrix[u][0].z);
116             glColor3f(pointsMatrix[u+1][v].r, pointsMatrix[u+1][v].g,
117                 ↪ pointsMatrix[u+1][v].b);
118             glVertex3f(pointsMatrix[u+1][v].x, pointsMatrix[u+1][v].y,
119                 ↪ pointsMatrix[u+1][v].z);
120         }
121         continue;
122     }
123     if(u==density-2){
124         for(int v=0;v<density;v++){
125             glColor3f(pointsMatrix[u+1][0].r,pointsMatrix[u+1][0].g,po
126                 ↪ intsMatrix[u+1][0].b);
127             glVertex3f(pointsMatrix[u+1][0].x,pointsMatrix[u+1][0].y,p
128                 ↪ ointsMatrix[u+1][0].z);
129             glColor3f(pointsMatrix[u][v].r, pointsMatrix[u][v].g,
130                 ↪ pointsMatrix[u][v].b);
131             glVertex3f(pointsMatrix[u][v].x, pointsMatrix[u][v].y,
132                 ↪ pointsMatrix[u][v].z);
133         }
134         break;
135     }
136     for(int v=0;v<density;v++){
137         int nextV = (v + 1) % density;
138         glColor3f(pointsMatrix[u][v].r,pointsMatrix[u][v].g,pointsMatr
139             ↪ ix[u][v].b);
140         glVertex3f(pointsMatrix[u][v].x,pointsMatrix[u][v].y,pointsMat
141             ↪ rix[u][v].z);
142         glColor3f(pointsMatrix[u+1][v].r, pointsMatrix[u+1][v].g,
143             ↪ pointsMatrix[u+1][v].b);
144         glVertex3f(pointsMatrix[u+1][v].x, pointsMatrix[u+1][v].y,
145             ↪ pointsMatrix[u+1][v].z);
146
147         glColor3f(pointsMatrix[u][v].r,pointsMatrix[u][v].g,pointsMatr
148             ↪ ix[u][v].b);
149         glVertex3f(pointsMatrix[u][v].x,pointsMatrix[u][v].y,pointsMat
150             ↪ rix[u][v].z);
151         glColor3f(pointsMatrix[u][nextV].r, pointsMatrix[u][nextV].g,
152             ↪ pointsMatrix[u][nextV].b);
153         glVertex3f(pointsMatrix[u][nextV].x, pointsMatrix[u][nextV].y,
154             ↪ pointsMatrix[u][nextV].z);
155     }
156 }
157 glEnd();
158 break;
159 case 3:
160     glBegin(GL_TRIANGLES);
161     for(int u=0;u<density-1;u++){
162         if(u==0){
163             for(int v=0;v<density;v++){
164                 int nextV = (v + 1) % density;

```

```

150         glColor3f(pointsMatrix[u][0].r,pointsMatrix[u][0].g,points
        ↪ Matrix[u][0].b);
151         glVertex3f(pointsMatrix[u][0].x,pointsMatrix[u][0].y,point
        ↪ sMatrix[u][0].z);
152         glColor3f(pointsMatrix[u+1][nextV].r,pointsMatrix[u+1][nex
        ↪ tV].g,pointsMatrix[u+1][nextV].b);
153         glVertex3f(pointsMatrix[u+1][nextV].x,pointsMatrix[u+1][ne
        ↪ xtV].y,pointsMatrix[u+1][nextV].z);
154         glColor3f(pointsMatrix[u+1][v].r,pointsMatrix[u+1][v].g,po
        ↪ intsMatrix[u+1][v].b);
155         glVertex3f(pointsMatrix[u+1][v].x,pointsMatrix[u+1][v].y,p
        ↪ ointsMatrix[u+1][v].z);
156     }
157     continue;
158 }
159 if(u==density-2){
160     for(int v=0;v<density;v++){
161         int nextV = (v + 1) % density;
162         glColor3f(pointsMatrix[u+1][0].r,pointsMatrix[u+1][0].g,po
        ↪ intsMatrix[u+1][0].b);
163         glVertex3f(pointsMatrix[u+1][0].x,pointsMatrix[u+1][0].y,p
        ↪ ointsMatrix[u+1][0].z);
164         glColor3f(pointsMatrix[u][v].r,pointsMatrix[u][v].g,points
        ↪ Matrix[u][v].b);
165         glVertex3f(pointsMatrix[u][v].x,pointsMatrix[u][v].y,point
        ↪ sMatrix[u][v].z);
166         glColor3f(pointsMatrix[u][nextV].r,pointsMatrix[u][nextV].
        ↪ g,pointsMatrix[u][nextV].b);
167         glVertex3f(pointsMatrix[u][nextV].x,pointsMatrix[u][nextV].
        ↪ .y,pointsMatrix[u][nextV].z);
168     }
169     break;
170 }
171 for(int v=0;v<density;v++){
172     int nextV = (v + 1) % density;
173     glColor3f(pointsMatrix[u][v].r,pointsMatrix[u][v].g,pointsMatr
        ↪ ix[u][v].b);
174     glVertex3f(pointsMatrix[u][v].x,pointsMatrix[u][v].y,pointsMat
        ↪ rix[u][v].z);
175     glColor3f(pointsMatrix[u+1][nextV].r,pointsMatrix[u+1][nextV].
        ↪ g,pointsMatrix[u+1][nextV].b);
176     glVertex3f(pointsMatrix[u+1][nextV].x,
        ↪ pointsMatrix[u+1][nextV].y, pointsMatrix[u+1][nextV].z);
177     glColor3f(pointsMatrix[u+1][v].r,pointsMatrix[u+1][v].g,points
        ↪ Matrix[u+1][v].b);
178     glVertex3f(pointsMatrix[u+1][v].x, pointsMatrix[u+1][v].y,
        ↪ pointsMatrix[u+1][v].z);
179
180     glColor3f(pointsMatrix[u+1][nextV].r,pointsMatrix[u+1][nextV].
        ↪ g,pointsMatrix[u+1][nextV].b);
181     glVertex3f(pointsMatrix[u+1][nextV].x,
        ↪ pointsMatrix[u+1][nextV].y,
        ↪ pointsMatrix[u+1][nextV].z);
182     glColor3f(pointsMatrix[u][v].r,pointsMatrix[u][v].g,pointsMatr
        ↪ ix[u][v].b);

```

```

183         glVertex3f(pointsMatrix[u][v].x,pointsMatrix[u][v].y,pointsMat_
        ↪   rix[u][v].z);
184         glColor3f(pointsMatrix[u][nextV].r,pointsMatrix[u][nextV].g,po_
        ↪   intsMatrix[u][nextV].b);
185         glVertex3f(pointsMatrix[u][nextV].x, pointsMatrix[u][nextV].y,
        ↪   pointsMatrix[u][nextV].z);
186     }
187 }
188 glEnd();
189 break;
190 }
191 }
192 ~Egg(){
193
194 }
195 };
196 Egg egg(20);
197 void toggleFocusToConsole() {
198     ShowWindow(glutWindow, SW_HIDE);
199     ShowWindow(consoleWindow, SW_SHOWNORMAL);
200     SetForegroundWindow(consoleWindow);
201 }
202
203 void toggleFocusToGLUT() {
204     ShowWindow(consoleWindow, SW_HIDE);
205     ShowWindow(glutWindow, SW_SHOWNORMAL);
206     SetForegroundWindow(glutWindow);
207 }
208 void animate(){
209     float rotationSpeed = 0.5f;
210     totalRotationX += rotationSpeed * sx;
211     totalRotationY += rotationSpeed * sy;
212     totalRotationZ += rotationSpeed * sz;
213     glutPostRedisplay();
214 }
215 void reset_rotation(){
216     totalRotationX = 0.0f;
217     totalRotationY = 0.0f;
218     totalRotationZ = 0.0f;
219     radius = 6;
220     cameraRotationX = 0.0f;
221     cameraRotationY = 0.0f;
222     cameraRotationZ = radius;
223     lastX = 0;
224     lastY = 0;
225 }
226 string bool_to_string(bool convert){
227     if(convert){
228         return "true";
229     }else{
230         return "false";
231     }
232 }
233 void printControls(){
234     cout<<"A D - obrot po osi Y\n";
235     cout<<"W S - obrot po osi X\n";

```

```

236     cout<<"Q E - obrot po osi Z\n";
237     cout<<"ESC - Powrot do menu (okno konsolowe)\n";
238     cout<<"Nalezy nacisnac i przytrzymac PPM\n";
239     cout<<"Ruch myszy w osi X - Obrot kamery w osi X\n";
240     cout<<"Ruch myszy w osi Y - Obrot kamery w osi Y\n";
241     cout<<"Scroll up - Przyblizenie obiektu\n";
242     cout<<"Scroll down - Oddalenie obiektu\n";
243     cout<<"Nacisnij Enter zeby kontynuowac\n"<<flush;
244     cin.get();
245     cin.get();
246 }
247 void menu(){
248     toggleFocusToConsole();
249     reset_rotation();
250     cout<<"=====\n";
251     cout<<"1. Narysuj czajnik\n";
252     cout<<"2. Narysuj jajko (punkty)\n";
253     cout<<"3. Narysuj jajko (linie)\n";
254     cout<<"4. Narysuj jajko (trojkaty) \n";
255     cout<<"5. Rysowanie w kolorze: "<<bool_to_string(color)<<"\n";
256     cout<<"6. Kontrola\n";
257     cout<<"7. Zakoncz program\n";
258     cout<<"> ";
259     int x;
260     cin>> x;
261     switch (x)
262     {
263     case 1:
264         drawTeapot = true;
265         break;
266     case 2:
267         drawTeapot = false;
268         eggMode = 1;
269         break;
270     case 3:
271         drawTeapot = false;
272         eggMode = 2;
273         break;
274     case 4:
275         drawTeapot = false;
276         eggMode = 3;
277         break;
278     case 5:
279         color=!color;
280         egg.generateMatrix(0.5f);
281         menu();
282         break;
283     case 6:
284         printControls();
285         menu();
286         break;
287     case 7:
288         exit(0);
289         break;
290     default:
291         cout<<"Podano nieporawny znak\n";

```



```

292         menu();
293         break;
294     }
295     toggleFocusToGLUT();
296     glutPostRedisplay();
297 }
298 void keyDown(u_char key,int x,int y){
299     switch (key)
300     {
301         case 'Q':
302         case 'q':
303             sz=1;
304             glutIdleFunc(animate);
305             break;
306         case 'E':
307         case 'e':
308             sz=-1;
309             glutIdleFunc(animate);
310             break;
311         case 'W':
312         case 'w':
313             sx=-1;
314             glutIdleFunc(animate);
315             break;
316         case 'S':
317         case 's':
318             sx=1;
319             glutIdleFunc(animate);
320             break;
321         case 'A':
322         case 'a':
323             sy=-1;
324             glutIdleFunc(animate);
325             break;
326         case 'D':
327         case 'd':
328             sy=1;
329             glutIdleFunc(animate);
330             break;
331         default:
332             break;
333     }
334 }
335 void keyUp(u_char key,int x,int y){
336     switch (key)
337     {
338         case 'E':
339         case 'Q':
340         case 'e':
341         case 'q':
342             sz=0;
343             break;
344         case 'W':
345         case 'S':
346         case 'w':
347         case 's':

```

```

348         sx=0;
349         break;
350     case 'A':
351     case 'D':
352     case 'd':
353     case 'a':
354         sy=0;
355         break;
356     case 27:
357         menu();
358         break;
359     default:
360         break;
361 }
362 if (sx == 0 && sy == 0 && sz == 0) {
363     glutIdleFunc(nullptr);
364 }
365 }
366 void mouse(int x, int y){
367     float sensitivity =0.75f;
368     float phi = sensitivity*((2.0f * y / 400) - 1.0f);
369     float theta = sensitivity*((2.0f * (400 - x) / 400) - 1.0f);
370     float maxPhi = 1.75f; // Restrict phi range to avoid gimbal lock (approx ±85
    ↪ degrees)
371     if (phi > maxPhi){
372         phi = maxPhi;
373     }
374     if (phi < -maxPhi){
375         phi = -maxPhi;
376     }
377     cameraRotationX = radius*cos(theta)*cos(phi);
378     cameraRotationY = radius*sin(phi);
379     cameraRotationZ = radius*sin(theta)*cos(phi);
380     lastX = x;
381     lastY = y;
382     glutPostRedisplay();
383 }
384 void mouseWheel(int button, int dir, int x, int y){
385     if (dir > 0){
386         radius -= 1;
387     }else{
388         radius += 1;
389     }
390     if(radius>=10){
391         radius=10;
392     }
393     if(radius<=1){
394         radius=1;
395     }
396     glutPostRedisplay();
397 }
398 void display() {
399     GLfloat lPos[] = {0,4,0,1}; //x,y,z, czy światło jest odległe
400     GLfloat col[] = {1,0,0,1};
401     glLoadIdentity();
402     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

```

```

403 //glLightfv(GL_LIGHT0, GL_POSITION, lPos);
404 gluLookAt(cameraRotationX, cameraRotationY, cameraRotationZ, 0, 0, 0, 1, 0); //Ustaw
    ↪ ienie kamery
405 glRotatef(totalRotationX, 1.0f, 0.0f, 0.0f);
406 glRotatef(totalRotationY, 0.0f, 1.0f, 0.0f);
407 glRotatef(totalRotationZ, 0.0f, 0.0f, 1.0f);
408 if(drawTeapot){
409     glutWireTeapot(1);
410 }else{
411     glShadeModel(GL_FLAT);
412     egg.draw(eggMode);
413 }
414 glutSwapBuffers();
415 }
416 void Init() {
417     egg.generateMatrix(0.5f);
418     glEnable(GL_DEPTH_TEST); //bez tego frontalna sciana nadpisuje tylnia
419     glClearColor(1.0f, 1.0f, 1.0f, 1.0f);
420     glMatrixMode(GL_PROJECTION);
421     glLoadIdentity();
422     glFrustum(-1, 1, -1, 1, 2, 10);
423     glMatrixMode(GL_MODELVIEW);
424     // // Włącza culling, czyli pomijanie tylnych ścianek
425     // glEnable(GL_CULL_FACE);
426     // // Ustawia kierunek frontowych ścianek jako przeciwny do ruchu wskazówek
    ↪ zegara
427     // glFrontFace(GL_CW);
428     // // Ustawia pomijanie tylnych ścianek
429     glCullFace(GL_BACK);
430     //glEnable(GL_LIGHTING); //Włączenie oświetlenia
431     //glEnable(GL_LIGHT0); //Dodanie źródła światła
432 }
433 int main(int argc, char** argv){
434     consoleWindow = GetConsoleWindow();
435     glutInit(&argc, argv);
436     glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
437     glutInitWindowSize(800, 800);
438     glutCreateWindow("Lab 3 - Czajnik i Jajko");
439     glutWindow = FindWindowW(NULL, L"Lab 3 - Czajnik i Jajko");
440     Init();
441     glutDisplayFunc(display);
442     glutIdleFunc(nullptr);
443     glutKeyboardFunc(keyDown);
444     glutKeyboardUpFunc(keyUp);
445     glutMotionFunc(mouse);
446     glutMouseWheelFunc(mouseWheel);
447     menu();
448
449     glutMainLoop();
450     system("pause");
451     return 0;
452 }

```

Fragment kodu 1: Fragment kodu z programu

### 3 Wnioski

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales...

### 4 Źródła

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales...