POLITECHNIKA WROCŁAWSKA

Instytut Informatyki, Automatyki i Robotyki

Zakład Systemów Komputerowych

Wprowadzenie do grafiki komputerowej

Kurs: INE4234L

Sprawozdanie z ćwiczenia nr 3

Open GL – oświetlanie scen 3D

Wykonał:	**Wojciech Wójcik, 235621**
Termin:	**PT/TP 8:00-11:00**
Data wykonania ćwiczenia:	**30.11.2018**
Data oddania sprawozdania:	**5.01.2019**
Ocena:	

Uwagi prowadzącego:

1. Cel ćwiczenia

Celem ćwiczenia jest zapoznanie się z możliwościami oświetlenia obiektów w scenie 3D z wykorzystaniem OpenGL oraz GLUT. W ćwiczeniach została wykorzystana wiedza zdobyta we wcześniejszych ćwiczeniach. Zaprezentowano działanie wektorów normalnych oraz definicje parametrów światła oraz materiałów oświetlanego obiektu.

2. Zadanie do wykonania

Pierwsze zadanie są stosunkowo proste, należy odpowiednio zmodyfikować istniejący już program, tak aby dodać do sceny oświetlenie. Dotyczy to własnoręcznie wykonanego modelu jajka, którym można manipulować przy użyciu myszki.

Kolejnym zadaniem było dodanie dwóch źródeł światła do sceny z modelem jajka oraz umożliwienie manipulacji każdym z nich (jajko jest statyczne).

3. Realizacja zadania

3.1. Dodanie oświetlenia do sceny z modelem jajka.

Zgodnie z instrukcją zostały dodane wartości w funkcji `MyInit` tak aby poprawnie zainicjować parametry światła.

```
void MyInit(void)
{

	glClearColor(0.0f, 0.0f, 0.0f, 1.0f);

	GLfloat mat_ambient[] = { 1.0,1.0, 1.0, 1 };
	GLfloat mat_diffuse[] = { 1.0, 1.0, 1.0, 1 };
	GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
	GLfloat mat_shininess = { 20.0 };
	GLfloat light_position[] = { 0.0, 0.0, 30.0, 1.0 };
	GLfloat light_ambient[] = { 1.0, 0.0, 0.0, 1.0 };
	GLfloat light_diffuse[] = { 1.0, 0.0, 0.0, 1.0 };
	GLfloat light_specular[] = { 1.0, 1.0, 0.0, 1.0 };
	GLfloat att_constant = { 1.0 };
	GLfloat att_linear = { (GLfloat) 0.05 };
	GLfloat att_quadratic = { (GLfloat) 0.001 };

	glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
	glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient);
	glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
	glMaterialf(GL_FRONT, GL_SHININESS, mat_shininess);

	glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient);
	glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);
	glLightfv(GL_LIGHT0, GL_SPECULAR, light_specular);
	glLightfv(GL_LIGHT0, GL_POSITION, light_position);

	glLightf(GL_LIGHT0, GL_CONSTANT_ATTENUATION, att_constant);
	glLightf(GL_LIGHT0, GL_LINEAR_ATTENUATION, att_linear);
	glLightf(GL_LIGHT0, GL_QUADRATIC_ATTENUATION, att_quadratic);

	glShadeModel(GL_SMOOTH); // właczenie łagodnego cieniowania
	glEnable(GL_LIGHTING);   // właczenie systemu oświetlenia sceny
	glEnable(GL_LIGHT0);     // włączenie źródła o numerze 0
	glEnable(GL_DEPTH_TEST); // włączenie mechanizmu z-bufora
}
```

Kolejnym i ostatnim krokiem była edycja funkcji `Egg`, tak aby dodać do modelu informacje o jego wektorach normalnych, co pozwoli na poprawną iluminacje powierzchni. Zostało to wykonane zgodnie z wzorami dostarczonymi w instrukcji laboratoryjnej:

$$x_u = \frac{\partial x(u,v)}{\partial u} = \left(-450u^4 + 900u^3 - 810u^2 + 360u - 45\right) \cdot \cos(\pi v)$$

$$x_v = \frac{\partial x(u,v)}{\partial v} = \pi \cdot \left(90u^5 - 225u^4 + 270u^3 - 180u^2 + 45u\right) \cdot \sin(\pi v)$$

$$y_u = \frac{\partial y(u,v)}{\partial u} = 640u^3 - 960u^2 + 320u$$

$$y_v = \frac{\partial y(u,v)}{\partial v} = 0$$

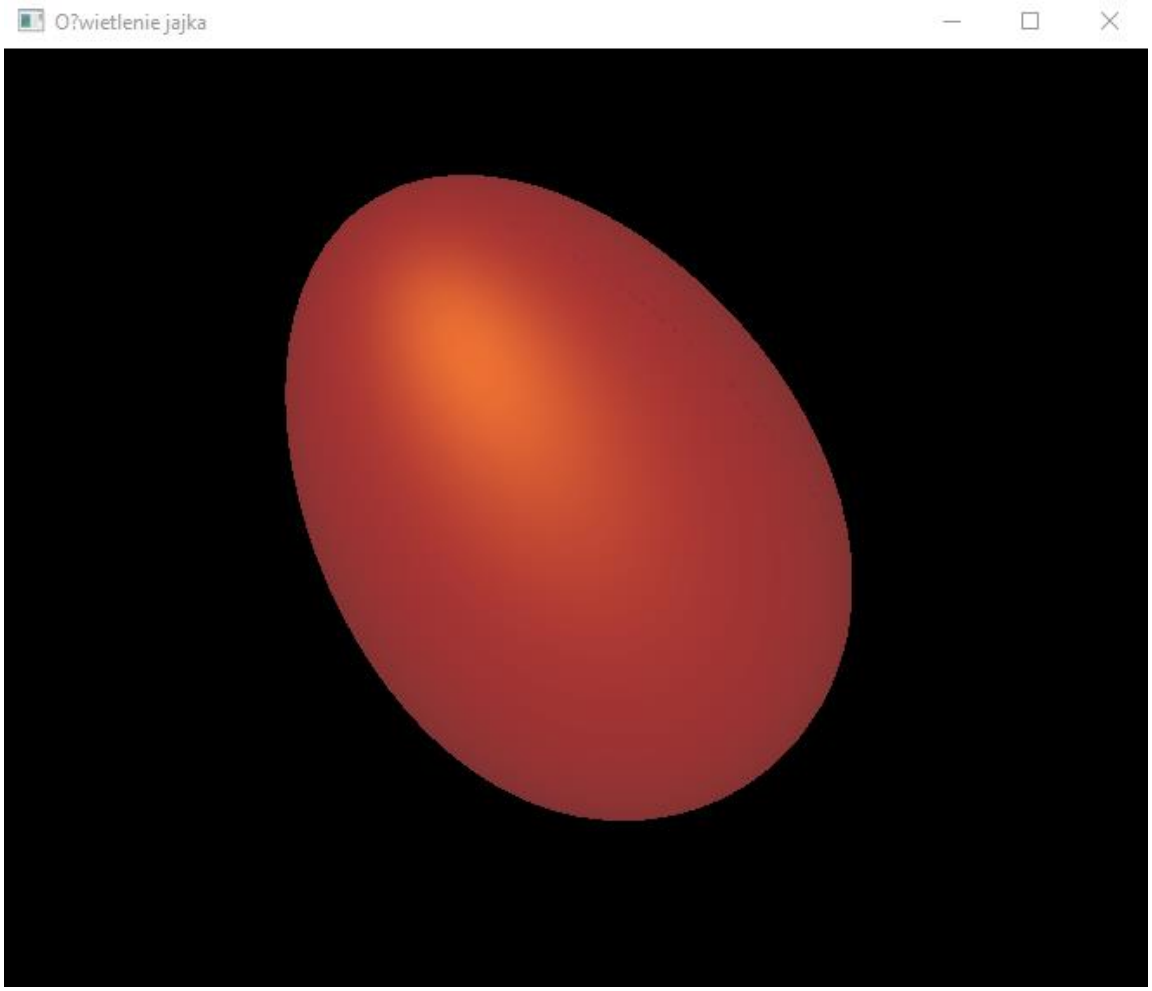
$$z_u = \frac{\partial z(u,v)}{\partial u} = \left(-450u^4 + 900u^3 - 810u^2 + 360u - 45\right) \cdot \sin(\pi v)$$

$$z_v = \frac{\partial z(u,v)}{\partial v} = -\pi \cdot \left(90u^5 - 225u^4 + 270u^3 - 180u^2 + 45u\right) \cdot \cos(\pi v)$$

Do funkcji `Egg` została dodana zmienna `vektorNorm`, która jest trójwymiarową tablicą przetrzymującą informacje o wektorach normalnych. Po wyznaczeniu wektorów normalnych zostały one dodane do pętli odpowiadającej za rysowanie modelu jajka.

```
for (int i = 0; i < n - 1; i++) {
	for (int j = 0; j < n - 1; j++) {

		glColor3f(255.0f, 255.0f, 255.0f);
		glBegin(GL_TRIANGLES);
		glNormal3fv(vektorNorm[i][j + 1]);
		glVertex3fv(vectors3d[i][j + 1]);

		glNormal3fv(vektorNorm[i + 1][j]);
		glVertex3fv(vectors3d[i + 1][j]);

		glNormal3fv(vektorNorm[i + 1][j + 1]);
		glVertex3fv(vectors3d[i + 1][j + 1]);
		glEnd();


		glBegin(GL_TRIANGLES);
		glNormal3fv(vektorNorm[i][j]);
		glVertex3fv(vectors3d[i][j]);

		glNormal3fv(vektorNorm[i + 1][j]);
		glVertex3fv(vectors3d[i + 1][j]);

		glNormal3fv(vektorNorm[i][j + 1]);
		glVertex3fv(vectors3d[i][j + 1]);
		glEnd();

	}
}
```

Efekt końcowy prezentuje się następująco:

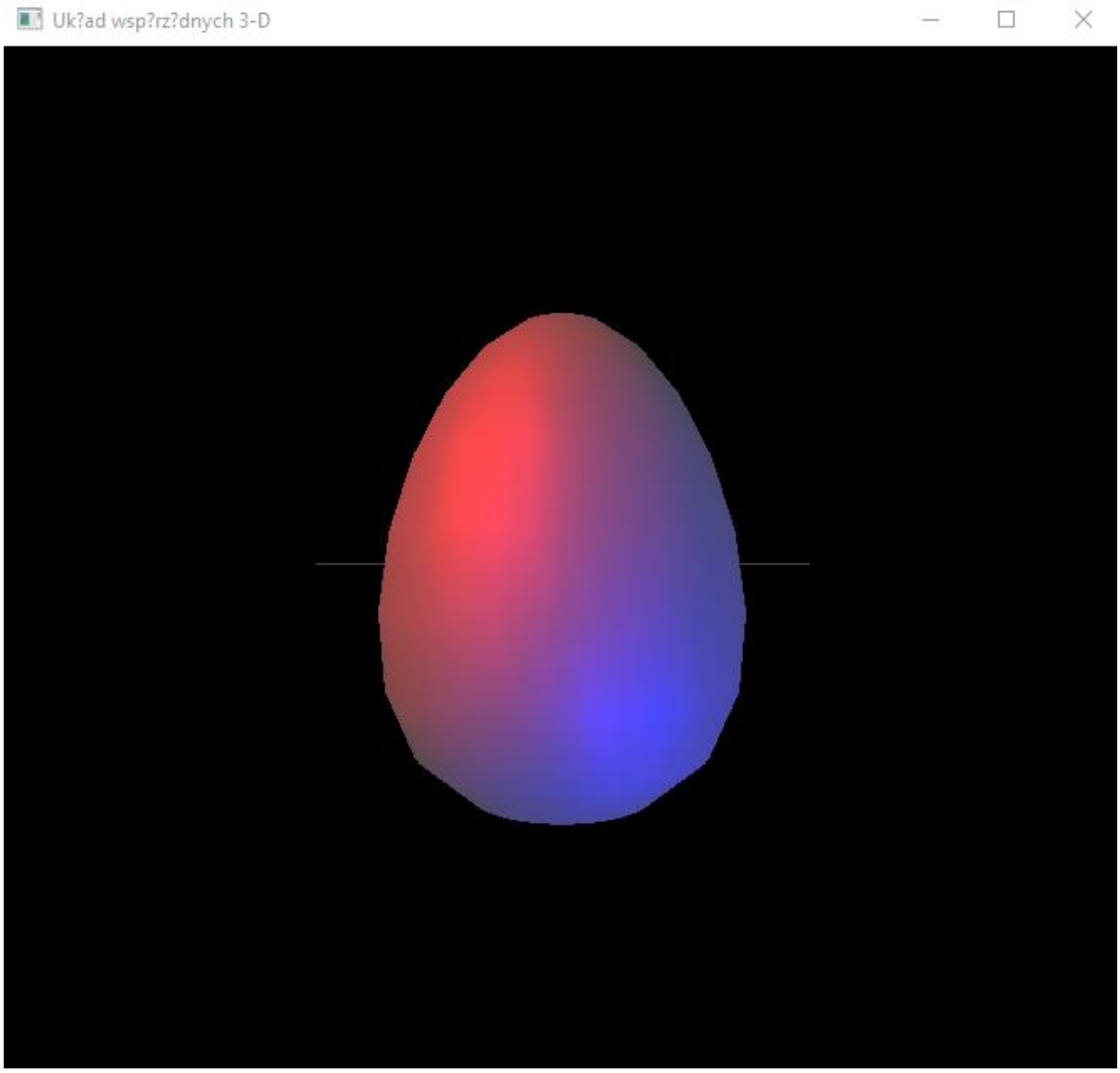


3.2. Oświetlenie modelu jajka dwoma źródłami światła.

Zgodnie z poprzednią instrukcją laboratoryjną została zaimplementowana mechanika obracania elementu wokół punktu, lecz tym razem dotyczyło to położenia 2 źródeł światła. Funkcja MyInit prezentuje wyliczanie położenia źródeł świateł.

```
void RenderScene(void)
{

	glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);


	glLoadIdentity();

	if (status == 1)
	{
		theta1 += delta1_x * pix2angle / 180 * M_PI;
		theta1_y += delta1_y * pix2angle / 180 * M_PI;
	}

	if (status == 2)
	{
		theta2 += delta2_x * pix2angle / 180 * M_PI;
		theta2_y += delta2_y * pix2angle / 180 * M_PI;
	}

	GLfloat up = 1.0;

	GLfloat x1 = viewer[2] * cos(-theta1) * cos(-theta1_y);
	GLfloat y1 = viewer[2] * sin(-theta1_y);
	GLfloat z1 = viewer[2] * sin(-theta1) * cos(-theta1_y);

	GLfloat x2 = viewer[2] * cos(-theta2) * cos(-theta2_y);
	GLfloat y2 = viewer[2] * sin(-theta2_y);
	GLfloat z2 = viewer[2] * sin(-theta2) * cos(-theta2_y);

	GLfloat light_position[] = { x1, y1, z1, 1.0 };
	GLfloat light_position2[] = { x2, y2, z2, 1.0 };
	glLightfv(GL_LIGHT0, GL_POSITION, light_position);
	glLightfv(GL_LIGHT1, GL_POSITION, light_position2);

	gluLookAt(0.0, 0.0, -15.0, 0.0, 0.0, 0.0, 0.0, up, 0.0);

	Axes();

	glColor3f(1.0f, 1.0f, 1.0f);
	glTranslated(0.0, -5.0, 0.0);

	Egg();
	glFlush();
	glutSwapBuffers();

}
```

Również została zmodyfikowana odpowiednio funkcja odpowiedzialna za przechwytywanie informacji z myszki aby umożliwić manipulację źródłami światła. Uzyskany efekt prezentuje się następująco:



4. Wnioski

Głównym problemem ćwiczenia było poprawne zaimplementowanie obliczenia wektorów normalnych, wymagane było przypomnienie jak dokładnie działa funkcja tworząca model jajka. Zostały zdobyte podstawowe informację o oświetlaniu sceny, materiałach oraz wyznaczaniu wektorów normalnych. Ponadto zostały udoskonalone umiejętności interakcji z użytkownikiem.