POLITECHNIKA WROCŁAWSKA

Instytut Informatyki, Automatyki i Robotyki

Zakład Systemów Komputerowych

Wprowadzenie do grafiki komputerowej

Kurs: INE4234L

Sprawozdanie z ćwiczenia nr 3

Open GL - modelowanie obiektów 3-D

Wykonał:	**Wojciech Wójcik, 235621**
Termin:	**PT/TP 8:00-11:00**
Data wykonania ćwiczenia:	**16.11.2018**
Data oddania sprawozdania:	**2.01.2019**
Ocena:	

Uwagi prowadzącego:

1. Cel ćwiczenia

Ćwiczenie miało za zadanie pokazać jak można zrealizować prostą interakcję polegającą na sterowaniu ruchem obiektu lub obserwatora przy pomocy OpenGL oraz GLUT. Zostały wyjaśnione różnice między rzutem ortogonalnym i perspektywicznym .

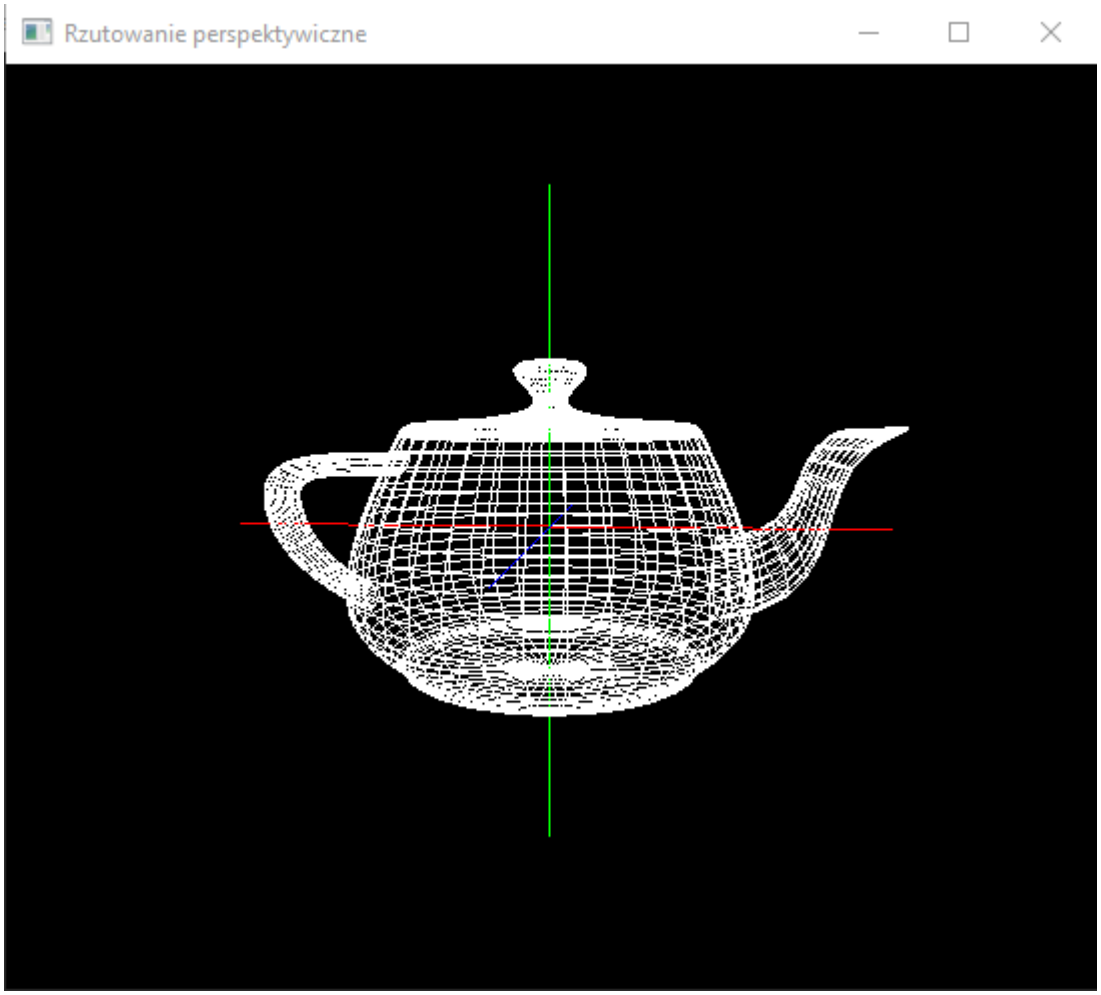
2. Zadanie do wykonania

Należało stworzyć program w którym można modyfikować położenie czajnika w osiach w osiach y i x oraz wykonywać zbliżanie i oddalanie

W kolejnym zadaniu interakcje i ruch należało realizować przez obrót kamery obserwatora wokół obiektu.

3. Realizacja zadania

3.1. Obrót czajnika



Pierwszą częścią zadania było wykrywanie naciśnietego klawisza myszy. Została użyta do tego funkcja *Mouse*. W niej jest porównywana zmienna *btn* do `GLUT_RIGHT_BUTTON` oraz `GLUT_LEFT_BUTTON.` Aby zapamiętać który przycisk został wciśnięty została użyta zmienna status, która jest wykorzystywana w funkcji renderowania sceny. Funkcja wygląda następująco:

```
void Mouse(int btn, int state, int x, int y)
{
	if (btn == GLUT_LEFT_BUTTON && state == GLUT_DOWN)
	{
		x_pos_old = x;
		status = 1;
		y_pos_old = y;
	}
	else if (btn == GLUT_RIGHT_BUTTON && state == GLUT_DOWN)
	{
		x_pos_old = x;
		y_pos_old = y;
		status = 2;
	}
	else
	status = 0;          // nie został wciśnięty żaden klawisz
 }
```

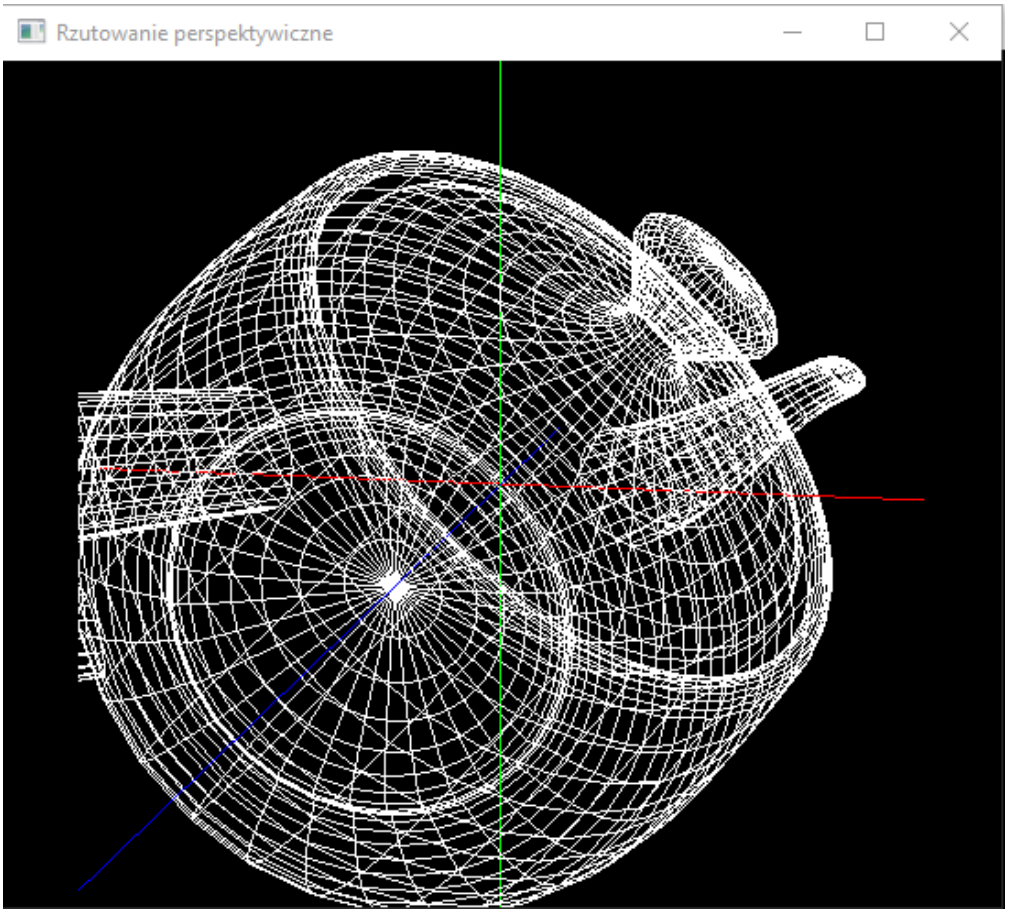
Dodatkowo, aby wykonać zadanie należało dodać do funkcji *Motion* zadeklarowaną wcześniej zmienną *delta_y* w której zapamiętywana jest zmiana współrzędnych osi Y.

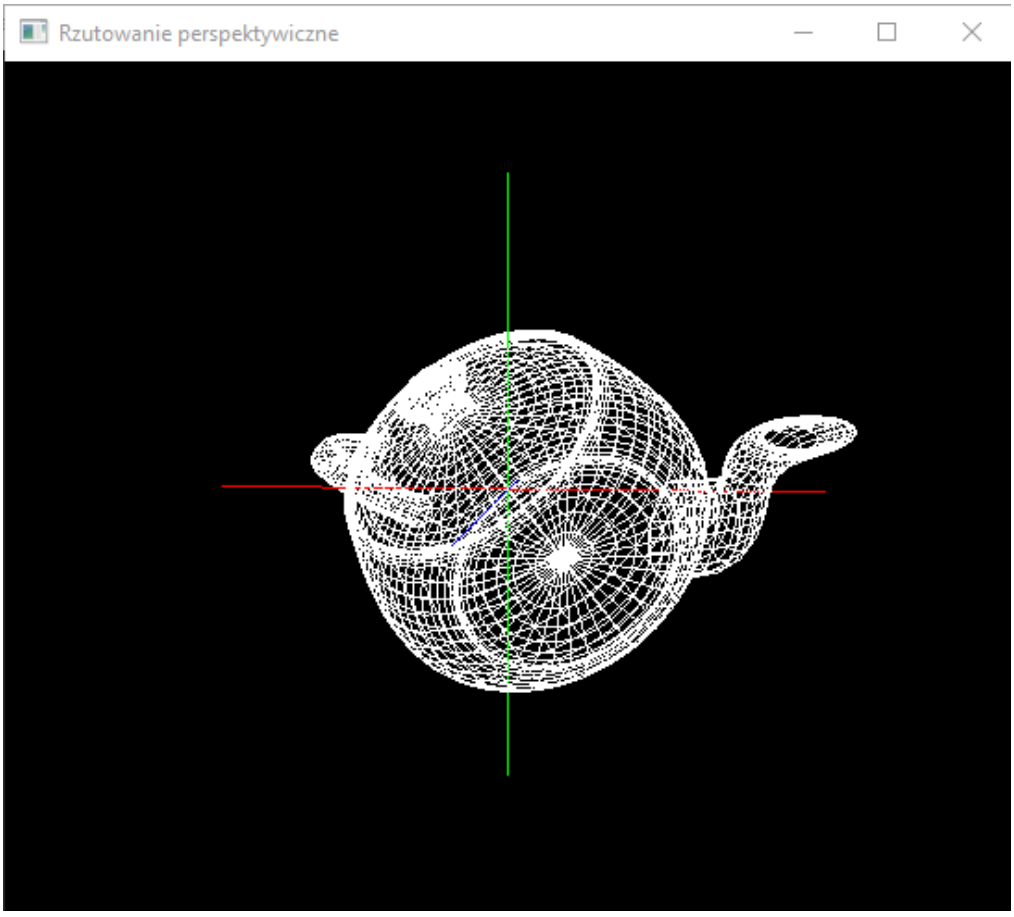
```
void Motion(GLsizei x, GLsizei y)
{

	delta_x = x - x_pos_old;     // obliczenie różnicy położenia kursora myszy
	x_pos_old = x;            // podstawienie bieżącego położenia jako poprzednie

	delta_y = y - y_pos_old;
	y_pos_old = y;

	glutPostRedisplay();     // przerysowanie obrazu sceny
}
```

Końcową częścią zadania była modyfikacja kodu funkcji *RenderScene.* Wymagane było sprawdzanie czy lewy lub prawy klawisz został wciśnięty i odpowiednia modyfikacja parametrów.

```
	if (status == 1)                    // jeśli lewy klawisz myszy wciśnięty
	{
		theta += delta_x * pix2angle;
		thetaY += delta_y * pix2angle;
	}
	if (status == 2)                    // jeśli prawy klawisz myszy wciśnięty
	{
		viewer[2] += delta_x * pix2angle;

		if (viewer[2] > 100)
			viewer[2] = 100;
		else if (viewer[2] < 5)
			viewer[2] = 5;
	}
```

Efekt dokonanych modyfikacji:





3.2. Obracanie obserwatora na przykładzie jajka

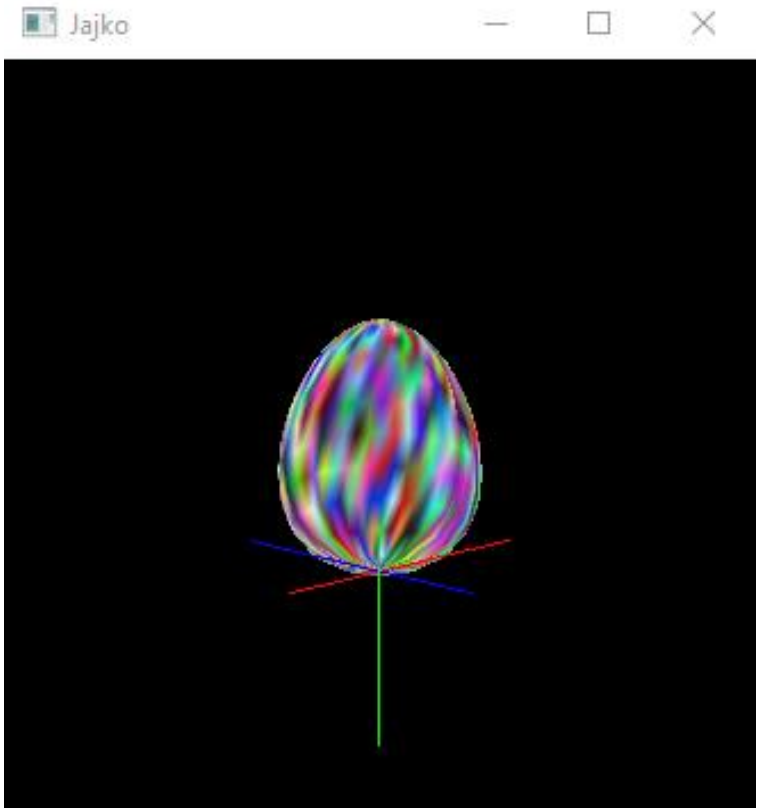
Kolejnym zadaniem było napisanie programu zmieniającego położenie obserwatora, do tego wykorzystano zmienne *theta*[0] i *thetaY.* Współrzędne obserwatora obliczane są zgodnie ze wzorami podanymi w instrukcji.

```
viewer[0] = R * cos(theta[0])*cos(thetaY);
viewer[1] = R * sin(thetaY);
viewer[2] = R * sin(theta[0])*cos(thetaY);
```

Poniżej fragment funkcji *RenderScene()* odpowiedzialny za ustawienie obserwatora:

```
if (cos(thetaY)>0)
	{
		gluLookAt(viewer[0], viewer[1], viewer[2], 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
		//pierwsze 3 argumenty określają współrzędne obserwatora

	}
	else
	{
		gluLookAt(viewer[0], viewer[1], viewer[2], 0.0, 0.0, 0.0, 0.0, -1.0,
0.0);		//pierwsze 3 argumenty określają współrzędne obserwatora

	}



	if (status == 1)                    // jeśli lewy klawisz myszy wcięnięty
	{
		theta[0] += delta_x * pix2angle * 0.05;   // modyfikacja kąta obrotu o
		kat proporcjonalny do różnicy położeń kursora myszy
		thetaY += delta_y * pix2angle* 0.05;
	}
	if (status == 2)                    // jeśli prawy klawisz myszy wcięnięty
	{
		R += delta_y/2;

		if (R > 32)
			R = 32;
		else if (R < 1)
			R = 8;
	}

	viewer[0] = R * cos(theta[0])*cos(thetaY);
	viewer[1] = R * sin(thetaY);
	viewer[2] = R * sin(theta[0])*cos(thetaY);

	if (thetaY == MOJE_PI)
	{
		thetaY = MOJE_PI + MOJE_PI / 2;
	}
```

Działający program z funkcją zmiany pozycji obserwatora.



4. Wnioski

Podczas pracy nad zadaniami napotkano problem związany z wartościami jakie przyjmują funkcje sinusoidalne, przez co niemożliwe było obrócenie kamery „do góry nogami”. Problem został rozwiązany poprzez prosty warunek `if (cos(thetaY)>0)` który odpowiednio ustawiał parametry funkcji *gluLookAt()* .

Te dwa ćwiczenia pozwoliły również nabyć podstawową wiedzę pozwalającą na interakcję z użytkownikiem w przestrzeni trójwymiarowej i pozwoliły na zapoznanie się z różnicami między rzutem perspektywicznym i ortogonalnym.