POLITECHNIKA WROCŁAWSKA

Instytut Informatyki, Automatyki i Robotyki

Zakład Systemów Komputerowych

Wprowadzenie do grafiki komputerowej

Kurs: INE4234L

Sprawozdanie z ćwiczenia nr 3

Open GL - teksturowanie powierzchni obiektów

Wykonał:	**Wojciech Wójcik, 235621**
Termin:	**PT/TP 8:00-11:00**
Data wykonania ćwiczenia:	**14.12.2018**
Data oddania sprawozdania:	**12.01.2019**
Ocena:	

Uwagi prowadzącego:

1. Cel ćwiczenia

Celem ćwiczenia jest pokazanie podstawowych technik teksturowania powierzchni obiektów z wykorzystaniem mechanizmów biblioteki OpenGL z rozszerzeniem GLUT. Na przykładach zilustrowana będzie droga od przeczytania obrazu tekstury, do nałożenia jaj fragmentów na poszczególne fragmenty modelu obiektu trójwymiarowego.

2. Zadanie do wykonania

Na zajęciach zostały teksturowane dwa modele: trójkąta, wielościanu i bardziej skomplikowanego modelu w postaci siatki trójkątów.

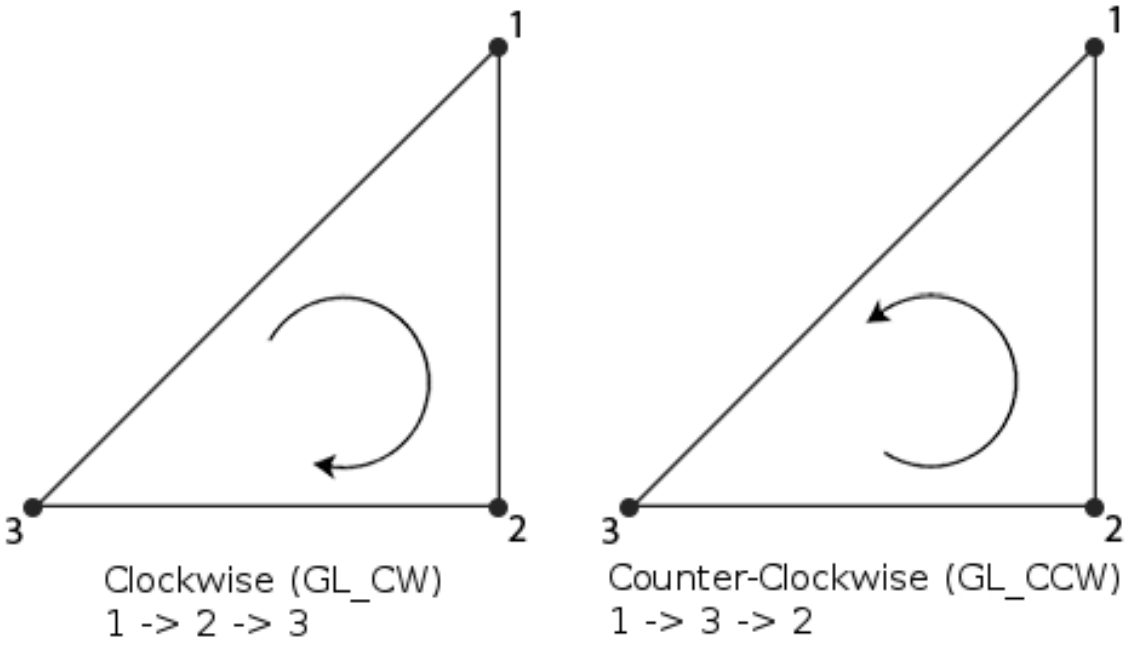
3. Realizacja zadania

3.1. Teksturowanie powierzchni wieloboku

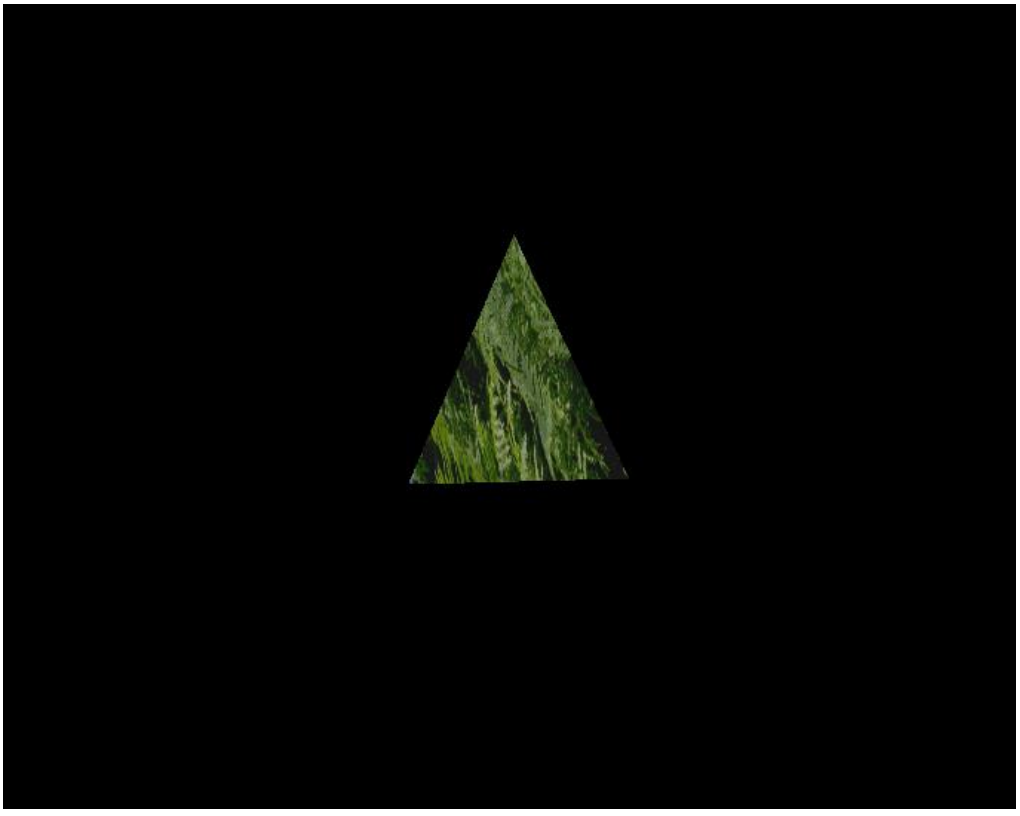
W celu naniesienia tekstury na obiekt w przestrzeni 3D należy wczytać plik tekstury do pamięci, co zostało to wykonane z pomocą instrukcji laboratoryjnej. Kolejnym krokiem było oświetlenie obiektu. Finalnie należało przy definiowaniu wierzchołków trójkąta podać odpowiednie koordynaty tekstury, tak aby przypisać im odpowiednie punkty „obrazu" tekstury, który zostanie nałożony zgodnie z nimi na obiekt. Funkcja odpowiadająca za rysowanie trójkąta wygląda następująco:

```
void trojkat()
{
	glBegin(GL_TRIANGLES);

	glTexCoord2f(0.0f, 0.0f);
	glVertex3f(0.0f, 1.0f, 0.0f);

	glTexCoord2f(0.0f, 1.0f);
	glVertex3f(-1.0f, -1.0f, 1.0f);

	glTexCoord2f(1.0f, 0.0f);
	glVertex3f(1.0f, -1.0f, 1.0f);

	glEnd();
}
```

Ważną rzeczą jaka została zauważona jest kolejność podawanych wierzchołków, gdyż trójkąt musi mieć jakoś określony „przód” i „tył”, aby tekstura została poprawnie nałożona (a nie odwrotnie). Domyślnie jest stosowana zasada podawania kolejności przeciwnej do wskazówek zegara, jednak możemy to zdefiniować komendą *void glFrontFace(GLenum mode);* W przypadku podania złej kolejności wierzchołków nie zobaczymy naszego trójkąta, elementy nie są renderowanie gdy widzimy je od „tyłu”. Pomaga to w optymalizacji bardziej złożonych scen.
Metody rysowania zostały pokazane na obrazku poniżej:



Narysowany trójkąt z teksturą:



3.2. Teksturowanie piramidy

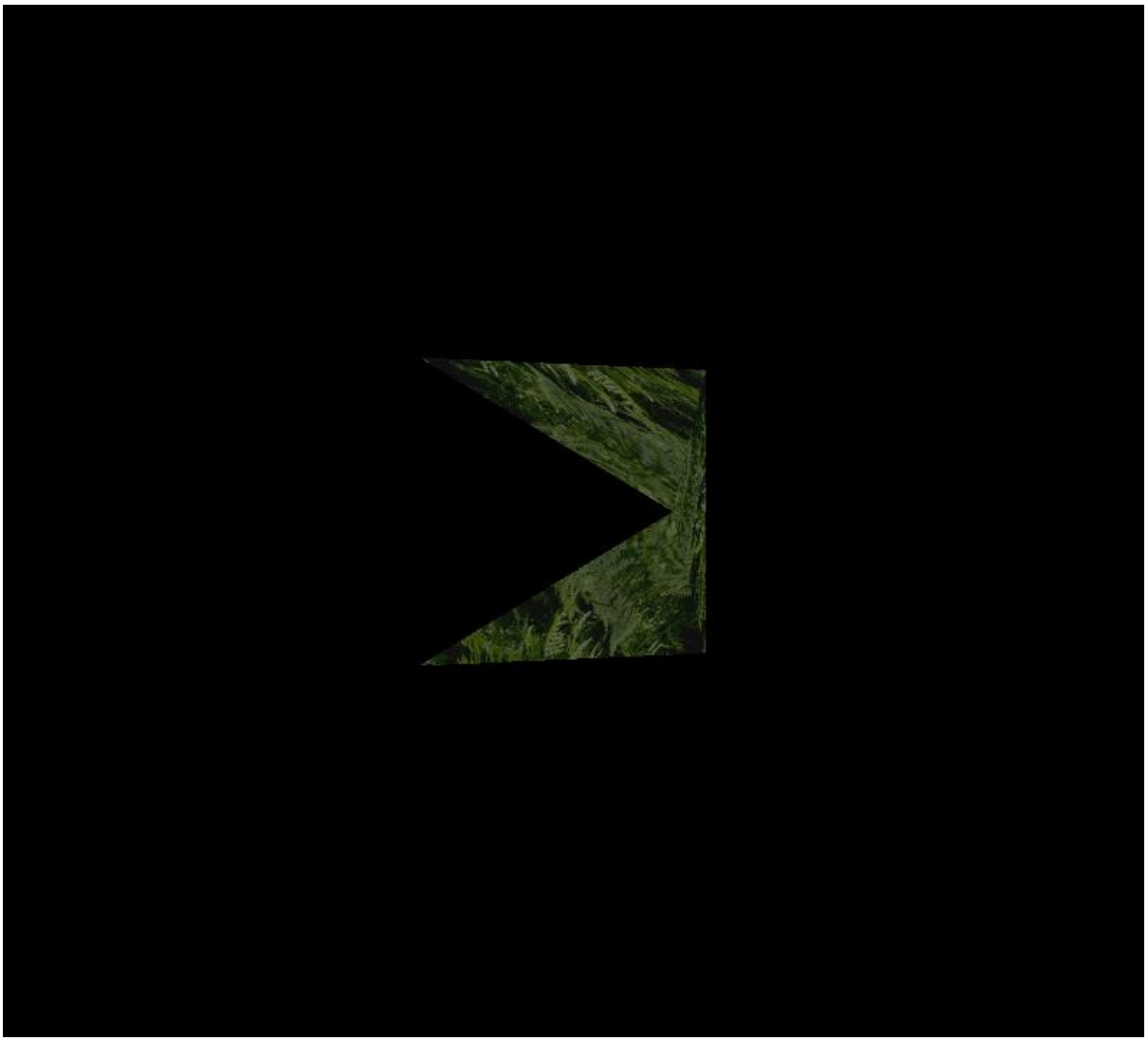
Kolejne zadanie składało się tak naprawdę z narysowania czterech trójkątów z odpowiednimi parametrami: koordynatami tekstury przypisanymi do wierzchołków, tak aby tekstura dwóch sąsiednich trójkątów pasowała do siebie, oraz by trójkąty były skierowane na zewnątrz obiektu, tak aby były widoczne. Funkcja odpowiadająca za rysowanie wygląda następująco (zostały dodane warunki pozwalające wyłączyć rysowanie poszczególnych ścian).

```
void piramida()
{
	glBegin(GL_TRIANGLES);

	if (mode1)
	{
		//n trojkat
		glTexCoord2f(0.0f, 0.0f);
		glVertex3f(0.0f, 1.0f, 0.0f);

		glTexCoord2f(0.0f, 1.0f);
		glVertex3f(-1.0f, -1.0f, 1.0f);

		glTexCoord2f(1.0f, 0.0f);
		glVertex3f(1.0f, -1.0f, 1.0f);
	}

	if (mode2)
	{
		//n trojkat
		glTexCoord2f(0.0f, 0.0f);
		glVertex3f(0.0f, 1.0f, 0.0f);

		glTexCoord2f(0.0f, 1.0f);
		glVertex3f(1.0f, -1.0f, 1.0f);

		glTexCoord2f(1.0f, 0.0f);
		glVertex3f(1.0f, -1.0f, -1.0f);
	}

	if (mode3)
	{
		//n trojkat
		glTexCoord2f(0.0f, 0.0f);
		glVertex3f(0.0f, 1.0f, 0.0f);

		glTexCoord2f(0.0f, 1.0f);
		glVertex3f(1.0f, -1.0f, -1.0f);

		glTexCoord2f(1.0f, 0.0f);
		glVertex3f(-1.0f, -1.0f, -1.0f);
	}
```

```
        if (mode4)
        {
                //n trojkat
                glTexCoord2f(0.0f, 0.0f);
                glVertex3f(0.0f, 1.0f, 0.0f);

                glTexCoord2f(0.0f, 1.0f);
                glVertex3f(-1.0f, -1.0f, -1.0f);

                glTexCoord2f(1.0f, 0.0f);
                glVertex3f(-1.0f, -1.0f, 1.0f);
        }


        glEnd();

        if (mode5)
        {
                glBegin(GL_QUADS);

                glTexCoord2f(0.0f, 0.0f);
                glVertex3f(-1.0f, -1.0f, -1.0f);
                glTexCoord2f(1.0f, 0.0f);
                glVertex3f(1.0f, -1.0f, -1.0f);
                glTexCoord2f(1.0f, 1.0f);
                glVertex3f(1.0f, -1.0f, 1.0f);
                glTexCoord2f(0.0f, 1.0f);
                glVertex3f(-1.0f, -1.0f, 1.0f);


                glEnd();
        }
}
```

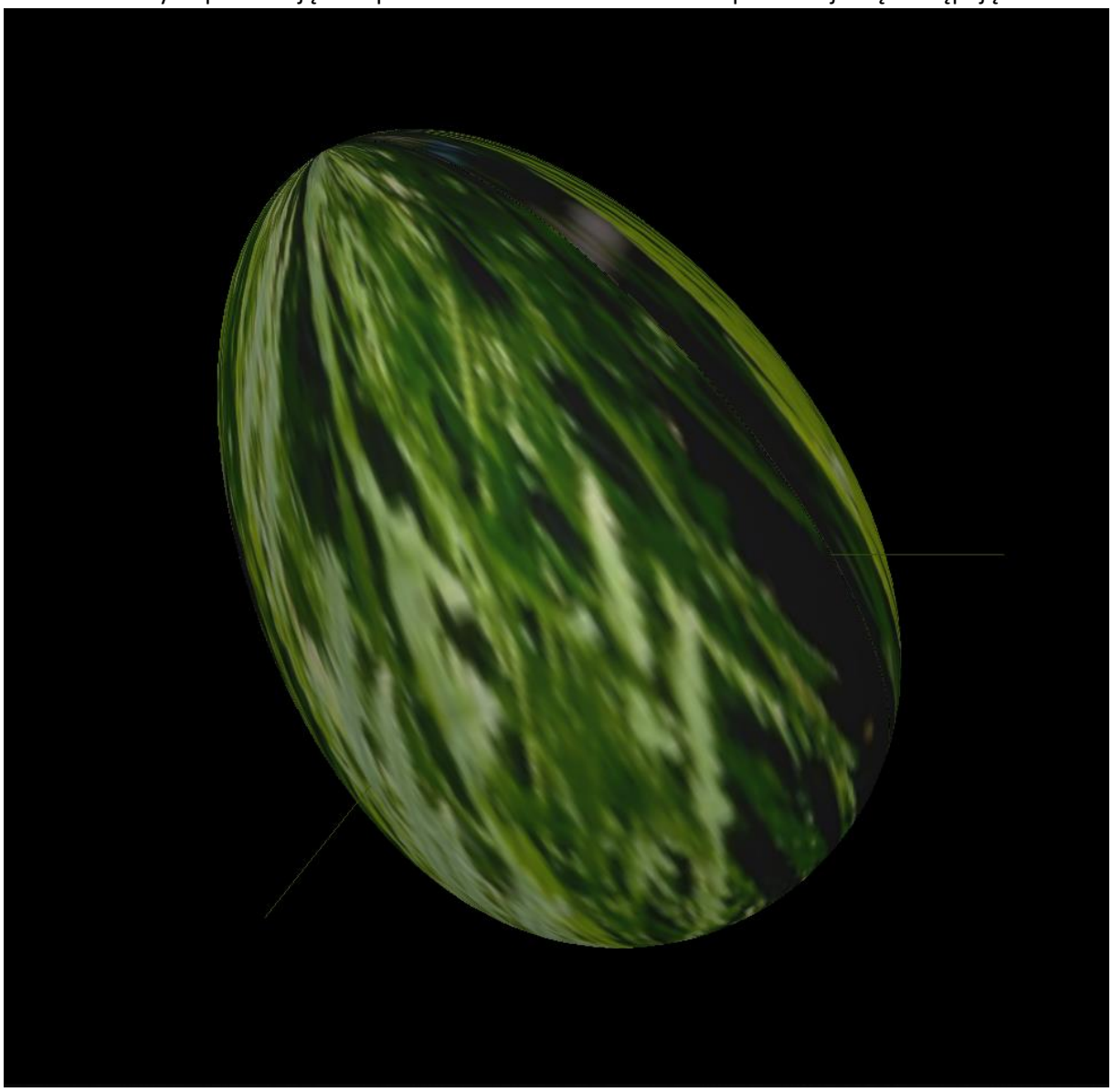
Efekt jest następujący:

Przykład z wyłączoną jedną scianą:



3.3. Teksturowanie jajka

Ostaniem zadaniem było wyposażenie jajka w teksturę. Modyfikacji kodu poddano głównie logikę kolejności rysowania trójkątów. Została dodana tablica w której zostały wyliczone koordynaty tekstury odpowiadające odpowiednim wierzchołkom. Efekt prezentuje się następująco:



4. Wnioski

Przy tym laboratorium bardzo pomogło mi doświadczenie pozyskane w silniku Unity 3D, w którym obowiązują podobne zasady teksturowania i rysowania obiektów (meshy) z elementów podstawowych jakimi są trójkąty. Najcięższą częścią zadania była odpowiednia modyfikacja funkcji rysowania jajka. Wcześniej nie trzeba było myśleć o kolejności w jakiej podawało się wierzchołki trójkąta, teraz była to kluczowa rzecz.