SPRAWOZDANIE

Zajęcia: Grafika Komputerowa
Prowadzący: mgr inż. Mikołaj Grygiel

Laboratorium: 11
Data: 14.05.2025
Temat: “Grafika 3D w bibliotece WebGL/GLSL”

Wariant: 14

Illia Bryka,
Informatyka I stopień,
stacjonarne,
4 semestr,
Gr.1a

Zadanie 1

1. Polecenie:

Plik lab12.html pokazuje mały sześcian, który można obrócić, przeciągając myszą na płótnie. Zadaniem jest zastąpienie sześcianu dużym wiatrakiem siedzącym na prostokątnej podstawie, jak pokazano na rysunku. Łopatki wiatraka powinny obracać się po włączeniu animacji. Każda łopatka wiatraka powinna być zbudowana z dwóch stożków. (Dodanie czajniczka, który znajduje się na podstawie, jest konieczne dla uzyskania oceny "5")

Program zawiera trzy zmienne instancji reprezentujące podstawowe obiekty: cube, cone, cylinder. Te zmienne mają metody instancji cube.render(), cone.render(), cylinder.render(), które można wywołać w celu narysowania obiektów. Obiekty nietransformowane mają rozmiar 1 we wszystkich trzech kierunkach i mają swój środek na (0,0,0). Oś stożka i oś cylindra są wyrównane wzdłuż osi Z. Wszystkie obiekty na scenie powinny być przekształconymi wersjami podstawowych obiektów (lub podstawowego obiektu czajnika).

2. Wykorzystane komendy:

Do wykonania zadania należało zmodyfikować kod:

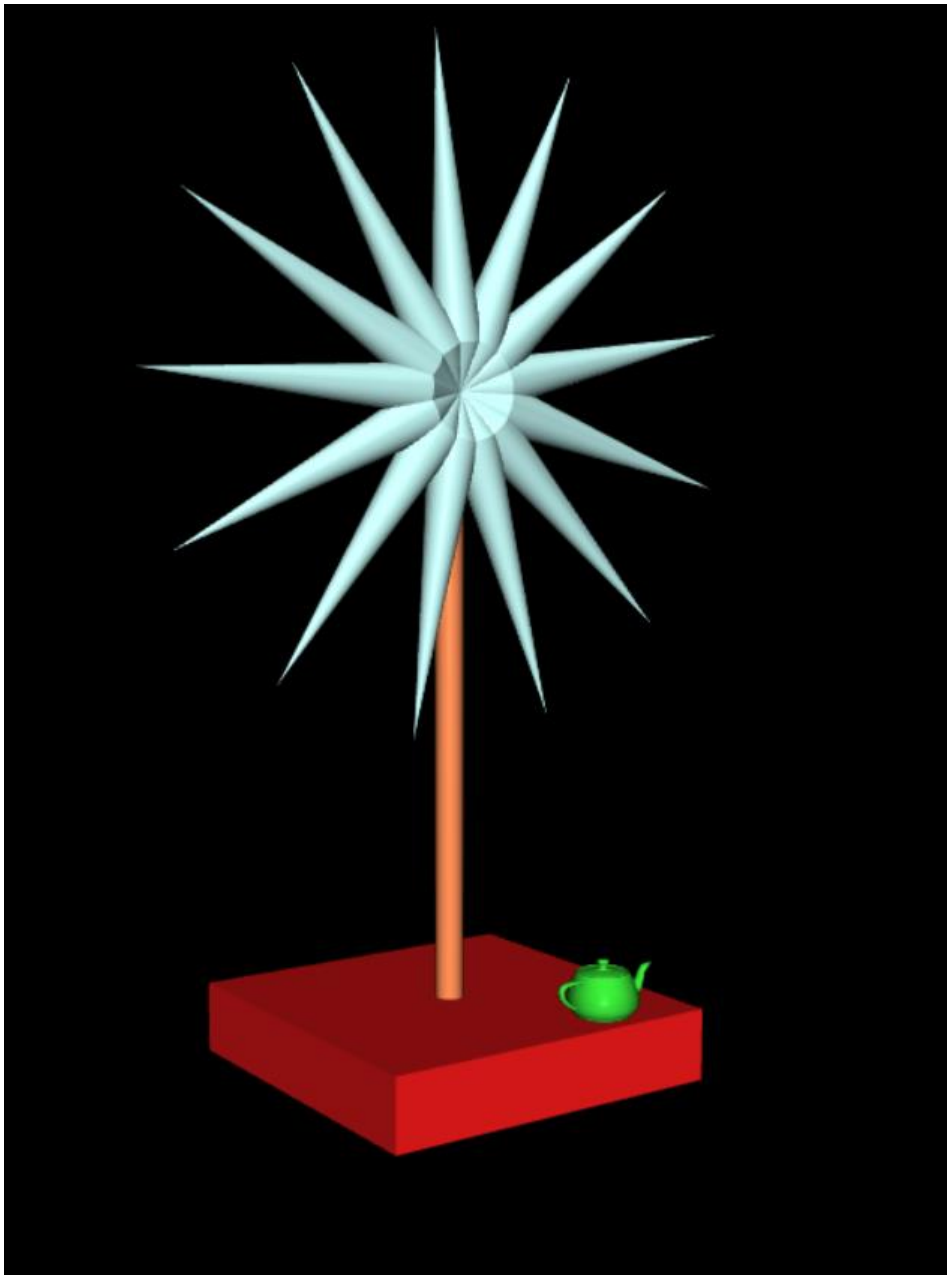
```
function draw() {
    gl.clearColor(0, 0, 0, 1);
    gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);

    mat4.perspective(projection, Math.PI / 4, 1, 1, 50);
    gl.uniformMatrix4fv(u_projection, false, projection);

    mat4.lookAt(modelview, [0, 0, 25], [0, 0, 0], [0, 1, 0]);

    mat4.rotateX(modelview, modelview, rotateX);
    mat4.rotateY(modelview, modelview, rotateY);

    // Podstawka
    pushMatrix();
    currentColor = [0.999, 0.111, 0.111];
    mat4.translate(modelview, modelview, [0, -5, 0]);
    mat4.scale(modelview, modelview, [5, 1, 5]);
    cube.render();
    popMatrix();

    // Trzonek
    pushMatrix();
    currentColor = [0.999, 0.555, 0.333];
    mat4.translate(modelview, modelview, [0, 0, 0]);
    mat4.rotateX(modelview, modelview, Math.PI * 0.5);
    mat4.scale(modelview, modelview, [0.4, 0.4, 10]);
    cylinder.render();
    popMatrix();

    // Dzbanek
    pushMatrix();
    currentColor = [0.222, 0.888, 0.222];
    mat4.translate(modelview, modelview, [1.2, -4.1, 2.0]);
    mat4.scale(modelview, modelview, [0.05, 0.05, 0.05]);
    teapot.render();
    popMatrix();
```

```
// łopatki wiatraka
pushMatrix();
mat4.translate(modelview, modelview, [-2.9, 2, 0.2]);
pushMatrix();
mat4.translate(modelview, modelview, [2.9, 2, 0]);
mat4.rotate(modelview, modelview, rotateEachFrame, [0, 0, 1]);
mat4.translate(modelview, modelview, [2.9, -2, 0]);

for (let i = 0; i < 13; i++) {
    pushMatrix();
    mat4.translate(modelview, modelview, [-3, 1.95, 0]);
    mat4.rotateZ(modelview, modelview, i * (360 / 13) * (Math.PI / 180));
    mat4.rotateY(modelview, modelview, Math.PI);

    pushMatrix();

    currentColor = [200 / 255, 1, 252 / 255];
    mat4.rotateY(modelview, modelview, Math.PI / 2);
    mat4.translate(modelview, modelview, [0, 0, 2.7]);
    mat4.scale(modelview, modelview, [0.7, 0.7, 4.1]);
    cone.render();
    popMatrix();

    pushMatrix();
    currentColor = [200 / 255, 1, 252 / 255];
    mat4.translate(modelview, modelview, [0.3, 0, 0]);
    mat4.rotateY(modelview, modelview, Math.PI / 2);
    mat4.rotateX(modelview, modelview, Math.PI);
    mat4.scale(modelview, modelview, [0.7, 0.7, 0.7]);
    cone.render();
    popMatrix();

    popMatrix();
}
```

```
function frame() {
    if (animating) {
        rotateEachFrame += Math.PI * 0.01;
        frameNumber += 1;
        draw();

        requestAnimationFrame(frame);
    }
}
```

Link do Repozytorium:
https://github.com/bebrabimba/Grafika-Komputerowa/tree/main/Lab12

3. Wyniki



Wnioski:

Biblioteka WebGL/GLSL to biblioteka podobna do Three.js. Swoje możliwości zawdzięcza bibliotece OpenGL, ponieważ to na niej bazuje. Daje ona spore możliwości budowania i zarządzania strukturami, a w tym ich animowanie