SPRAWOZDANIE

Zajęcia: Grafika komputerowa
Prowadzący: mgr inż. Mikołaj Grygiel

Laboratorium 6
9.04.2025
Temat: "Światło i materiały w OpenGL"
Wariant 6

Bartłomiej Mędrzak
s61324
Informatyka I stopień,
stacjonarne,
4 semestr,
Gr.1A

1. Polecenie:

Celem jest stworzenie pyramidy z użyciem różnych materiałów okrelonych wariantem zadania i umieszczenie jej na „podstawie”. Użytkownik może obracać podstawę wokół osi Y, przeciągając mysz w poziomie. Scena wykorzystuje globalne światło otoczenia (ambient) oraz źródło światła o kształcie kuli z możliwością animacji obrotu wokół pyramidy.

Aby wykonać laboratorium w JavaScript polecane jest zapoznanie z plikami .html: four-lights-demo.html oraz materials-demo.html

2. Wprowadzane dane:

```
function uvSphere(radius, slices, stacks) {
    var i,j;
    for (j = 0; j < stacks; j++) {
        var latitude1 = (Math.PI/stacks) * j - Math.PI/2;
        var latitude2 = (Math.PI/stacks) * (j+1) - Math.PI/2;
        var sinLat1 = Math.sin(latitude1);
        var cosLat1 = Math.cos(latitude1);
        var sinLat2 = Math.sin(latitude2);
        var cosLat2 = Math.cos(latitude2);
        glBegin(GL_TRIANGLE_STRIP);
        for (i = 0; i <= slices; i++) {
            var longitude = (2*Math.PI/slices) * i;
            var sinLong = Math.sin(longitude);
            var cosLong = Math.cos(longitude);
            var x1 = cosLong * cosLat1;
            var y1 = sinLong * cosLat1;
            var z1 = sinLat1;
            var x2 = cosLong * cosLat2;
            var y2 = sinLong * cosLat2;
            var z2 = sinLat2;
            glNormal3d(x2,y2,z2);
        glVertex3d(radius*x2,radius*y2,radius*z2);
        glNormal3d(x1,y1,z1);
            glVertex3d(radius*x1,radius*y1,radius*z1);
        }
        glEnd();
    }
}
```

```
function lights() {
    glColor3d(0.5,0.5,0.5);
    var zero = [ 0, 0, 0, 1 ];
    glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, zero);

    if (viewpointLight.checked) glEnable(GL_LIGHT0); else glDisable(GL_LIGHT0);

    if (redLight.checked) {
        glMaterialfv(GL_FRONT_AND_BACK, GL_EMISSION, [0.5, 0, 0, 1]);
        glEnable(GL_LIGHT1);
    } else {
        glMaterialfv(GL_FRONT_AND_BACK, GL_EMISSION, zero);
        glDisable(GL_LIGHT1);
    }
    glPushMatrix(); glRotated(-frameNumber, 0, 1, 0); glTranslated(10, 7, 0);
    glLightfv(GL_LIGHT1, GL_POSITION, zero); uvSphere(0.5, 16, 8); glPopMatrix();

    if (greenLight.checked) {
        glMaterialfv(GL_FRONT_AND_BACK, GL_EMISSION, [0, 0.5, 0, 1]);
        glEnable(GL_LIGHT2);
    } else {
        glMaterialfv(GL_FRONT_AND_BACK, GL_EMISSION, zero);
        glDisable(GL_LIGHT2);
    }
    glPushMatrix(); glRotated((frameNumber+100)*0.8743, 0, 1, 0); glTranslated(9, 8, 0);
    glLightfv(GL_LIGHT2, GL_POSITION, zero); uvSphere(0.5, 16, 8); glPopMatrix();

    if (blueLight.checked) {
        glMaterialfv(GL_FRONT_AND_BACK, GL_EMISSION, [0, 0, 0.5, 1]);
        glEnable(GL_LIGHT3);
    } else {
        glMaterialfv(GL_FRONT_AND_BACK, GL_EMISSION, zero);
        glDisable(GL_LIGHT3);
    }
    glPushMatrix(); glRotated((frameNumber-100)*1.3057, 0, 1, 0); glTranslated(9.5, 7.5, 0);
    glLightfv(GL_LIGHT3, GL_POSITION, zero); uvSphere(0.5, 16, 8); glPopMatrix();

    if (rustLight.checked) {
        glMaterialfv(GL_FRONT_AND_BACK, GL_EMISSION, [0.8, 0.6, 0.1, 1]);
        glEnable(GL_LIGHT4);
    } else {
        glMaterialfv(GL_FRONT_AND_BACK, GL_EMISSION, zero);
        glDisable(GL_LIGHT4);
    }
    glPushMatrix(); glRotated((frameNumber+50)*0.7, 0, 1, 0); glTranslated(8, 6, 0);
    glLightfv(GL_LIGHT4, GL_POSITION, zero); uvSphere(0.5, 16, 8); glPopMatrix();

    glMaterialfv(GL_FRONT_AND_BACK, GL_EMISSION, zero);
}
```

```
function display() {
    glClearColor(0.1, 0.1, 0.1, 1);
    glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );

    camera.apply();

    glPushMatrix();
    glRotatef(sceneRotationY, 0, 1, 0);

        lights();

        if (ambientLight.checked) {
            glLightModelfv(GL_LIGHT_MODEL_AMBIENT, [0.15, 0.15, 0.15, 1] );
        } else {
            glLightModelfv(GL_LIGHT_MODEL_AMBIENT, [0, 0, 0, 1] );
        }

        var basePlatformRadius = 3.0;
        var basePlatformHeight = 0.3;

        if (drawBase.checked) {
            glPushAttrib(GL_ENABLE_BIT | GL_LIGHTING_BIT | GL_CURRENT_BIT);
            glDisable(GL_COLOR_MATERIAL);
            glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT,  [0.15, 0.07, 0.02, 1.0]);
            glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE,  [0.3, 0.15, 0.05, 1.0]);
            glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, [0.05, 0.02, 0.01, 1.0]);
            glMaterialf(GL_FRONT_AND_BACK, GL_SHININESS, 2);

            glPushMatrix();
                var cylinderCenterY = -(pyramidHeight / 2) - (basePlatformHeight / 2) - 0.05;
                glTranslated(0, cylinderCenterY , 0);
                glRotated(-90, 1, 0, 0);
                glScaled(basePlatformRadius, basePlatformRadius, basePlatformHeight / 2.0);
                drawCylinder(false);
            glPopMatrix();
            glPopAttrib();
        }

        if (pyramidModel) {
            glPushAttrib(GL_ENABLE_BIT | GL_LIGHTING_BIT | GL_CURRENT_BIT);
            glDisable(GL_COLOR_MATERIAL);

            glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT,  [0.02, 0.02, 0.1, 1.0]);
            glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE,  [0.1, 0.1, 0.4, 1.0]);
            glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, [0.2, 0.2, 0.5, 1.0]);
            glMaterialf(GL_FRONT_AND_BACK, GL_SHININESS, 15);

            glPushMatrix();
                var pyramidBaseY = -(pyramidHeight / 2) ;
                glTranslatef(0, pyramidBaseY + 1.2, 0);
                glRotatef(-90, 1, 0, 0);
                glsimDrawModel(pyramidModel);
            glPopMatrix();
            glPopAttrib();
        }
    glPopMatrix();
}
```

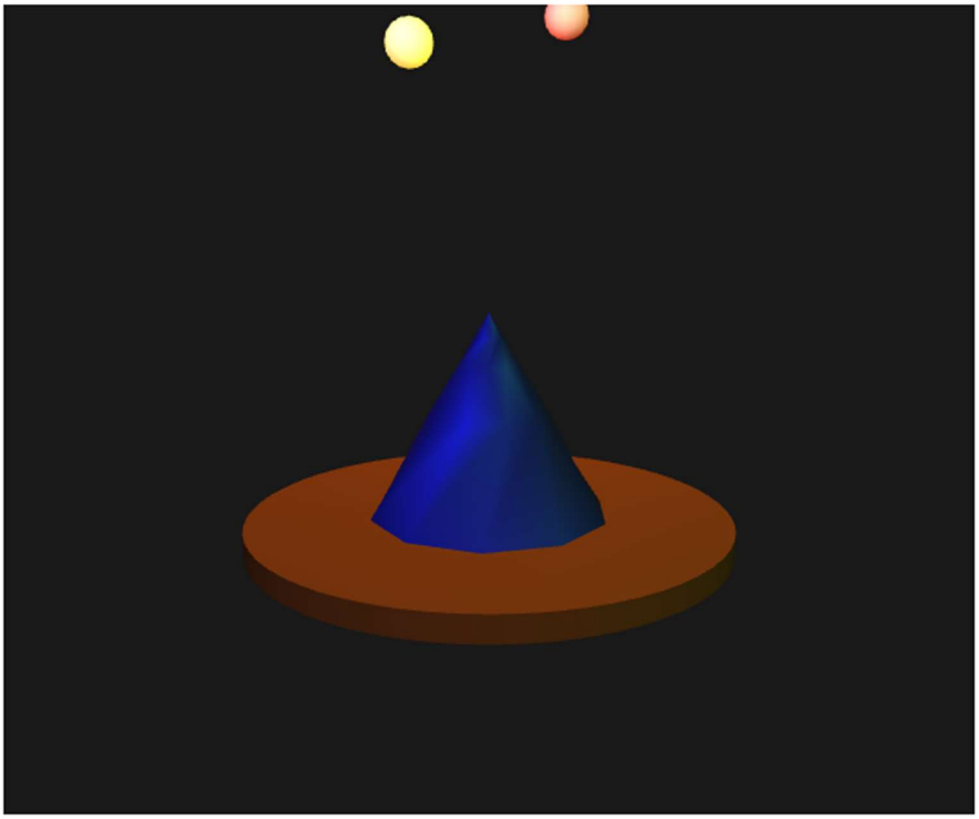
3. Wykorzystane komendy:

https://github.com/castehard33/Grafika_Komputerowa/tree/main/6%20Swiat%C5%82o%20i%20materia%C5%82y%20w%20OpenGL

4. Wynik działania:

Przy animacji widać 4 źródła światła

Pyramid on Base Demo



- [ ] Animate Lights
- [x] Draw Base
- [x] Global Ambient
- [x] Viewpoint Light
- [x] Red Light
- [x] Green Light
- [x] Blue Light
- [x] Yellow Light

5. Wnioski:

Realizacja zadania pozwoliła na praktyczne zapoznanie się z koncepcjami oświetlenia i materiałów w grafice komputerowej z wykorzystaniem API symulującego OpenGL (glsim.js).

Dobór odpowiednich parametrów materiałów (składowe ambient, diffuse, specular, shininess) ma kluczowe znaczenie dla realistycznego odwzorowania wyglądu obiektów, co było widoczne przy próbach uzyskania np. "brązu" dla podstawy czy specyficznego koloru dla piramidy.

Zrozumienie działania różnych typów świateł (ambient, kierunkowe, punktowe) i ich interakcji z materiałami obiektów jest fundamentalne do osiągnięcia pożądanych efektów wizualnych, co zademonstrowano poprzez możliwość włączania/wyłączania poszczególnych źródeł światła.

Praca z hierarchią transformacji (stos macierzy glPushMatrix/glPopMatrix) okazała się niezbędna do prawidłowego pozycjonowania i animowania poszczególnych elementów sceny niezależnie od siebie.

Laboratorium stanowiło dobre wprowadzenie do podstawowych technik renderowania 3D i interakcji użytkownika, które są fundamentem bardziej zaawansowanych aplikacji graficznych.