SPRAWOZDANIE

Zajęcia: Grafika komputerowa
Prowadzący: mgr inż. Mikołaj Grygiel

Laboratorium 5
26.02.2025
Temat: "Geometria trójwymiarowa OpenGL"
Wariant 6

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

1. Polecenie:

Stworzyć dwa obiekty przy użyciu OpenGL (w języku JavaScript). Po uruchomieniu zakończonego programu naciśnięcie jednego z klawiszy numerycznych 1 lub 2 spowoduje wybranie wyświetlanego obiektu. Program ustawia wartość zmiennej globalnej, objectNumber, aby powiedzieć, który obiekt ma zostać narysowany. Użytkownik może obracać obiekt za pomocą klawiszy strzałek, PageUp, PageDown i Home. Podprogram display() jest wywoływany, aby narysować obiekt.

Obiekt 1. Korkociąg wokół osi {x | y | z} zawierający N obrotów. Punkty są stopniowo powiększane. Ustalić aktualny kolor rysujący na {zielony | niebieski | brązowy | … }.

Obiekt 2. Pyramida, wykorzystując dwa wachlarze trójkątów oraz modelowanie hierarchiczne (najpierw tworzymy podprogramę rysowania jednego trójkonta; dalej wykorzystując przekształcenia geometryczne tworzymy pyramidę). Podstawą pyramidy jest wielokąt o N wierzchówkach.

2. Wprowadzane dane:

```
function rysujKorkociag(liczbaPetli, promienPodstawy, wysokoscNaPetle, segmentyNaPetle) {
    let calkowitaWysokosc = liczbaPetli * wysokoscNaPetle;
    let y_start = -calkowitaWysokosc / 2;

    glColor3f(0.0, 0.8, 0.0);

    glBegin(GL_LINE_STRIP);

    for (let i = 0; i <= liczbaPetli * segmentyNaPetle; i++) {
        let ulamek_petli = (i % segmentyNaPetle) / segmentyNaPetle;
        let ktora_petla = Math.floor(i / segmentyNaPetle);

        let kat = Math.PI * 2 * ulamek_petli;

        let x_wsp = Math.cos(kat) * promienPodstawy;
        let z_wsp = Math.sin(kat) * promienPodstawy;

        let aktualne_y = y_start + (ktora_petla + ulamek_petli) * wysokoscNaPetle;

        if (i === liczbaPetli * segmentyNaPetle) {
            aktualne_y = calkowitaWysokosc / 2;
        }

        glVertex3f(x_wsp, aktualne_y, z_wsp);
    }
    glEnd();
}
```

```
function rysujPiramide(liczbaScianBocznych, promienPodstawy) {
    let offset_y_podstawy = -promienPodstawy * 0.7;
    let offset_y_wierzcholka = promienPodstawy * 0.7;

    glBegin(GL_TRIANGLES);
    for (let i = 0; i < liczbaScianBocznych; i++) {
        let kat1 = 2 * Math.PI * i / liczbaScianBocznych;
        let kat2 = 2 * Math.PI * (i + 1) / liczbaScianBocznych;

        let x_podstawa_1 = promienPodstawy * Math.cos(kat1);
        let z_podstawa_1 = promienPodstawy * Math.sin(kat1);

        let x_podstawa_2 = promienPodstawy * Math.cos(kat2);
        let z_podstawa_2 = promienPodstawy * Math.sin(kat2);

        if (i % 2 === 0) {
            glColor3f(1.0, 0.0, 0.0);
        } else {
            glColor3f(0.0, 1.0, 0.0);
        }

        glVertex3f(0, offset_y_wierzcholka, 0);
        glVertex3f(x_podstawa_1, offset_y_podstawy, z_podstawa_1);
        glVertex3f(x_podstawa_2, offset_y_podstawy, z_podstawa_2);
    }
    glEnd();

    glColor3f(0.5, 0.5, 0.5);
    glBegin(GL_TRIANGLE_FAN);
    glVertex3f(0, offset_y_podstawy, 0);
    for (let i = 0; i <= liczbaScianBocznych; i++) {
        let kat = 2 * Math.PI * i / liczbaScianBocznych;
        glVertex3f(promienPodstawy * Math.cos(kat), offset_y_podstawy, promienPodstawy * Math.sin(kat));
    }
    glEnd();
}
```

```
function odswiezWyswietlanie() {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glLoadIdentity();

    glRotatef(obrotZ, 0, 0, 1);
    glRotatef(obrotY, 0, 1, 0);
    glRotatef(obrotX, 1, 0, 0);

    if (aktualnyKsztalt === 1) {
        rysujKorkociag(6, 0.3, 0.25, 30);
    } else {
        rysujPiramide(10, 0.8);
    }
}


function inicjalizacjaGrafiki() {
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(-1.2, 1.2, -1.2, 1.2, -2.5, 2.5);
    glMatrixMode(GL_MODELVIEW);
    glEnable(GL_DEPTH_TEST);
    glClearColor(0.9, 0.9, 0.9, 1);
}
```

```
function obslugaKlawiszy(event) {
    let kodKlawisza = event.keyCode;
    let czyPrzerysowac = true;

    switch (kodKlawisza) {
        case 37:
            obrotY -= 10;
            break;
        case 39:
            obrotY += 10;
            break;
        case 40:
            obrotX += 10;
            break;
        case 38:
            obrotX -= 10;
            break;
        case 33:
            obrotZ += 10;
            break;
        case 34:
            obrotZ -= 10;
            break;
        case 36:
            obrotX = 15;
            obrotY = -15;
            obrotZ = 0;
            break;
        case 49:
            aktualnyKsztalt = 1;
            break;
        case 50:
            aktualnyKsztalt = 2;
            break;
        default:
            czyPrzerysowac = false;
    }

    if ((kodKlawisza >=33 && kodKlawisza <= 40) || kodKlawisza === 32 || kodKlawisza === 49 || kodKlawisza === 50) {
        if (event.target === document.body) {
            event.preventDefault();
        }
    }

    if (czyPrzerysowac) {
        odswiezWyswietlanie();
    }
}
```

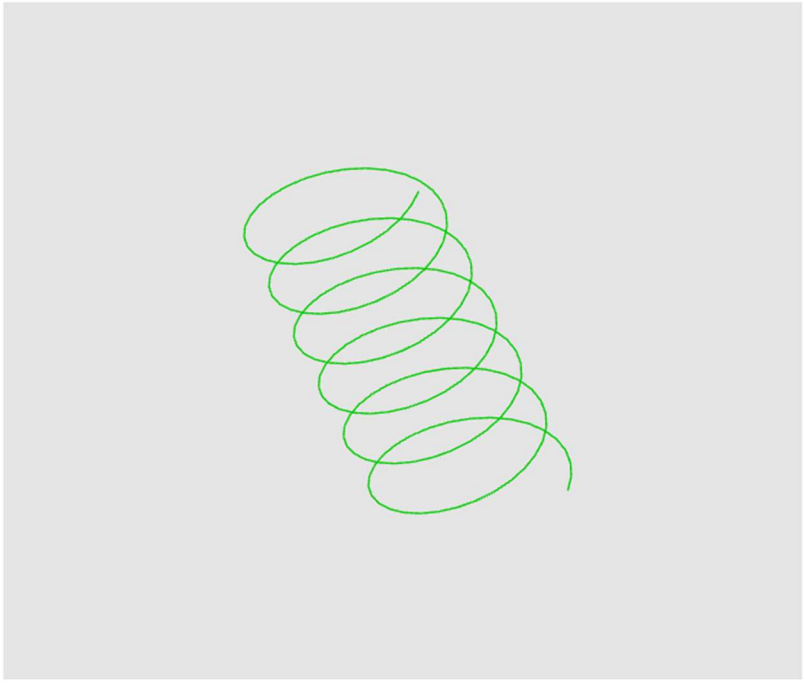
3. Wykorzystane komendy:

https://github.com/castehard33/Grafika_Komputerowa/tree/main/5%20Geometria%20tr%C3%B3jwymiarowa%20OpenGL

4. Wynik działania:

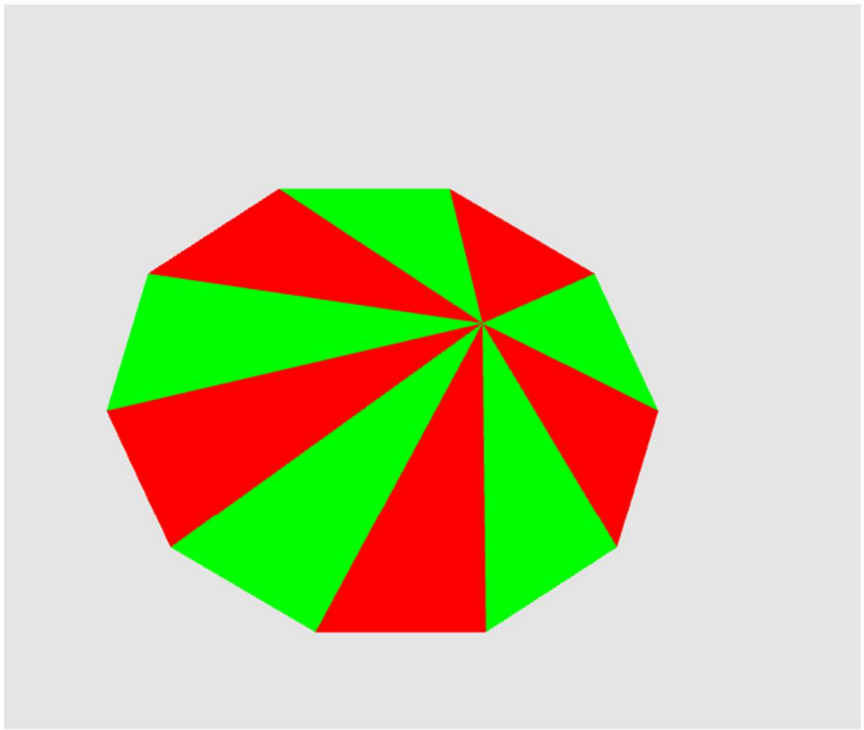
Dynamiczne Figury Trójwymiarowe - Wybór Klawiszami

Steruj obrotem obiektu używając klawiszy strzałek oraz Page Up/Page Down. Klawisz Home przywraca początkową orientację.
Naciśnij **1** aby wyświetlić Korkociąg (zielony), lub **2** aby wyświetlić Piramidę (czerwono-zielona).



Dynamiczne Figury Trójwymiarowe - Wybór Klawiszami

Steruj obrotem obiektu używając klawiszy strzałek oraz Page Up/Page Down. Klawisz Home przywraca początkową orientację.
Naciśnij **1** aby wyświetlić Korkociąg (zielony), lub **2** aby wyświetlić Piramidę (czerwono-zielona).



5. Wnioski:

Na podstawie otrzymanego wyniku można stwierdzić, że program poprawnie realizuje zadanie tworzenia i manipulowania dwoma różnymi obiektami trójwymiarowymi w środowisku symulującym OpenGL przy użyciu JavaScript. Zadanie pozwoliło na praktyczne zastosowanie podstawowych funkcji OpenGL, takich jak definiowanie geometrii za pomocą wierzchołków, rysowanie prymitywów (GL_LINE_STRIP, GL_TRIANGLE_FAN) oraz implementację transformacji geometrycznych do interaktywnego obracania obiektów. Możliwość dynamicznej zmiany wyświetlanego obiektu oraz jego orientacji za pomocą klawiatury potwierdza zrozumienie obsługi zdarzeń i zarządzania stanem w aplikacji graficznej, co jest kluczowe dla tworzenia interaktywnych wizualizacji.