**Sprawozdanie z projektu 1  
„Wirtualna Kamera 3D” w ramach zajęć  
z Grafiki Komputerowej**

Maciej Karabin, 325483

**1. Wstęp i cel projektu**

Projekt polegał na stworzeniu prostej wizualizacji 3D za pomocą wirtualnej kamery. Celem zadania było zademonstrowanie podstawowych technik związanych z grafiką komputerową, takich jak:

* Rzutowanie perspektywiczne punktów z przestrzeni trójwymiarowej na ekran.
* Operacje geometryczne: translacja, obrót oraz przemieszczenie kamery.
* Wyświetlanie krawędziowych modeli obiektów w przestrzeni trójwymiarowej.

Cel został osiągnięty przy użyciu wyłącznie podstawowych metod graficznych, bez użycia OpenGL. W efekcie powstała aplikacja umożliwiająca interaktywną eksplorację przestrzeni trójwymiarowej.

**2. Technologie i biblioteki wykorzystane w projekcie**

Projekt został wykonany w języku **Python** z użyciem następujących bibliotek:

* **Tkinter** – graficzna biblioteka do stworzenia interfejsu użytkownika oraz renderowania na ekranie.
* **math** – biblioteka matematyczna do wykonywania operacji geometrycznych i transformacji punktów (funkcje trygonometryczne, obroty).

**3. Podstawowe założenia logiczne projektu**

Projekt opiera się na następujących założeniach:

* Statyczne modele: scenę tworzą obiekty statycznie określone jako zestawy punktów (krawędzi), które nie mogą być edytowane w trakcie działania aplikacji.
* Proste modelowanie geometryczne: wykorzystano wyłącznie liniowe modele geometryczne (odcinki), bez obiektów krzywoliniowych lub powierzchniowych.
* Brak eliminacji niewidocznych krawędzi: wszystkie linie są rysowane bez względu na ich faktyczną widoczność.
* Prosty model kamery: kamera może się poruszać w przestrzeni w dowolnym kierunku oraz obracać w kątach RPY (roll, pitch, yaw)

**4. Logiczny opis struktury projektu**

Projekt logicznie podzielony jest na trzy główne komponenty:

**4.1 Model Kamery (camera.py)**

Kamera jest podstawowym elementem wizualizacji, który definiuje punkt widzenia użytkownika. Jest opisana przez:

* Pozycję w przestrzeni 3D (współrzędne XYZ).
* Orientację przestrzenną, określoną trzema kątami:
  + Yaw – obrót poziomy (prawo/lewo),
  + Pitch – obrót pionowy (góra/dół),
  + Roll – obrót wokół osi patrzenia (clockwise/counter-clockwise).
* Wektor kierunku patrzenia (forward vector) – kierunek, w którym zwrócona jest kamera.
* Wektor kierunku prawego boku kamery (right vector) – pomocniczy przy przemieszczaniu kamery prawo/lewo względem kierunku patrzenia.

Dzięki temu kamera może przemieszczać się swobodnie po scenie i obserwować ją z dowolnej perspektywy.

**4.2 Renderowanie sceny (renderer.py)**

Ważnym aspektem projektu jest proces projekcji punktów przestrzeni 3D na płaszczyznę ekranu. W projekcie użyto prostej projekcji perspektywicznej. Logika tego procesu obejmuje następujące etapy:

1. Przesunięcie punktów względem pozycji kamery (przestrzeń globalna → przestrzeń kamery).
2. Rotacje punktów (obrót wokół osi yaw, pitch, roll):
   * Każdy punkt jest rotowany względem kamery, aby uwzględnić jej aktualną orientację.
   * Wykorzystano do tego celu standardowe macierze rotacji wokół trzech osi.
3. Projekcja perspektywiczna na ekran:
   * Punkty transformowane są zgodnie z zasadami rzutu perspektywicznego (im obiekt jest dalej, tym mniejszy jest jego obraz na ekranie).
   * Punkty znajdujące się za kamerą są pomijane i nie są wyświetlane.

W rezultacie uzyskany jest efekt realistycznej perspektywy.

**4.3 Obsługa GUI i interaktywność (main.py oraz renderer.py)**

Interfejs użytkownika zaimplementowano za pomocą biblioteki Tkinter. Logicznie realizuje on:

* Wyświetlenie sceny – każdorazowo, gdy kamera zmieni położenie lub orientację, scena jest na nowo renderowana.
* Sterowanie kamerą – użytkownik za pomocą klawiatury zmienia położenie i orientację kamery. Efektem jest dynamiczna, interaktywna eksploracja sceny 3D.
* Informacje o sterowaniu – na ekranie wyświetlany jest opis dostępnych klawiszy sterujących.

Dzięki temu użytkownik może wygodnie eksplorować scenę, obserwując efekty zastosowanych algorytmów graficznych.

**5. Podsumowanie**

Projekt ten jasno pokazuje logikę działania kamery w grafice trójwymiarowej, podstawowe operacje geometryczne oraz zasady projekcji perspektywicznej. Projekt skutecznie demonstruje fundamenty grafiki komputerowej oraz sposób ich praktycznego wykorzystania w aplikacji interaktywnej.

Użyte podejście logiczne pozwala zrozumieć jak zbudować i obsłużyć prosty system wizualizacji 3D od podstaw, bez ukrytych mechanizmów sprzętowych czy bibliotek graficznych.

**6. Bibliografia**

<https://anzeljg.github.io/rin2/book2/2405/docs/tkinter/canvas.html>

<https://stackoverflow.com/questions/59685562/live-3d-view-pov-ray-and-tkinter>

<https://jsfiddle.net/PoutineErable/w3z0mtes/69/>

<https://www.zacobria.com/universal-robots-knowledge-base-tech-support-forum-hints-tips-cb2-cb3/index.php/python-code-example-of-converting-rpyeuler-angles-to-rotation-vectorangle-axis-for-universal-robots/>