Grafika Komputerowa - projekt 1 - kamera

Daniel Stańkowski, 311478

Cel projektu

Celem projektu jest stworzenie kamery rejestrującej bryły w przestrzeni wykorzystując dostępne narzędzia programistyczne. Kamera ma możliwość poruszania się we wszystkich kierunkach, robienia obrotów we wszystkich kierunkach oraz przybliżenia i oddalenia obrazu.

Rozwiązanie

W celu implementacji postawionego zadania skorzystałem z języka programowania python z dołączonymi bibliotekami: numpy i pygame. Pakiet numpy upraszcza działania na macierzach, a pygame służy wyświetlaniu kamery.

Przechowywanie informacji o wierzchołkach i krawędziach

Wszystkie operacje transformacji i rotacji przeprowadzane są na wszystkich punktach w układzie jednocześnie. Dla ułatwienia tych operacji wszystkie wierzchołki przechowywane są w jednej tablicy wektorów o długości 4. Każdy wektor składa się ze współrzędnych x, y, z i 1. Dodatkowa stała w postaci jedynki ułatwia przeprowadzanie transformacji punktów. Krawędzie przechowywane są w programie jako pary punktów i wykorzystywane są podczas rysowania układu.

W programie wyświetlane będą 4 prostopadłościany umieszczone obok siebie. Wczytywane będą one z plików, w którym każda z linii zawiera 6 liczb, 3 pierwsze liczby to współrzędne pierwszego punku krawędzi, a 3 kolejne to współrzędne drugiego punktu krawędzi.

Implementacja operacji ruchu i obrotu

Po przyciśnięciu odpowiedniego przycisku odpowiadającego za daną operację, wszystkie punkty poddawane są translacji lub rotacji. Dzięki temu cały układ zachowuje się w jednakowy sposób.

- Translacja

Do implementacji ruchów poziomych oraz pionowych została zastosowana macierz translacji, przez którą jest mnożony każdy wierzchołek układu

$$\begin{pmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Rotacja

Do implementacji obrotu kamery względem każdej z osi: x, y i z zostały zastosowane macierze rotacji, które w wyniku mnożenia z wierzchołkami dają nowy układ uwzględniający zastosowaną rotację.

Rotacja względem osi z:

$$\begin{pmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotacja względem osi x:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotacja względem osi y:

$$\begin{pmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Przybliżanie i oddalanie obiektów

Przybliżanie i oddalanie obiektów nie zmienia współrzędnych punktów, jest ono obliczane za pomocą zmiennej ogniskowej, przy pomocy której podczas mapowania obiektów trójwymiarowych na płaszczyznę kamery obliczana jest wielkość obiektów na ekranie.

Przykłady działania programu

Sterowanie:

- w, s, a, d, r, f – ruchy odpowiednio w przód, tył, lewo, prawo, górę i dół

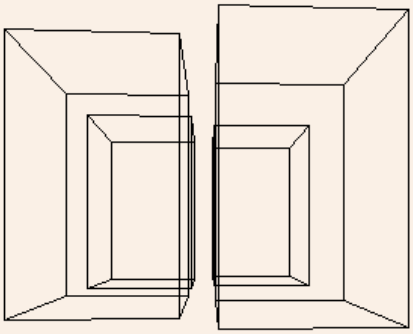
- arrow_up, arrow_down, arrow_left, arrow_right, e, q – rotacje odpowiednio w górę, w dół, w lewo, w prawo, zgodnie z ruchem wskazówek zegara i przeciwnie do ruchu wskazówek zegara

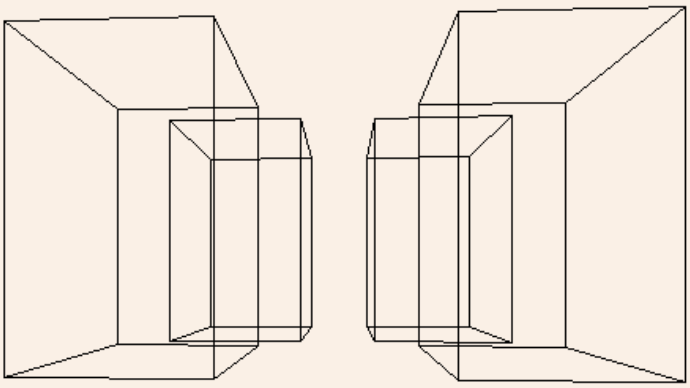
- -, = - oddalanie i przybliżanie

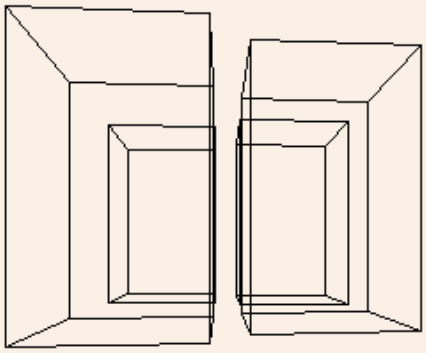
Test

Przeprowadziłem prosty test, otóż:

- Ustawiłem się po prawej stronie obiektów
- Wykonałem translację w prawo, żeby znaleźć się przed nimi
- Wykonałem rotację w lewo o 90 stopni
- Wykonałem translację w prawo, żeby znaleźć się obok obiektów
- Wykonałem rotację w lewo o 90 stopni
- Wykonałem translację w prawo aby ustawić się po środku obiektów





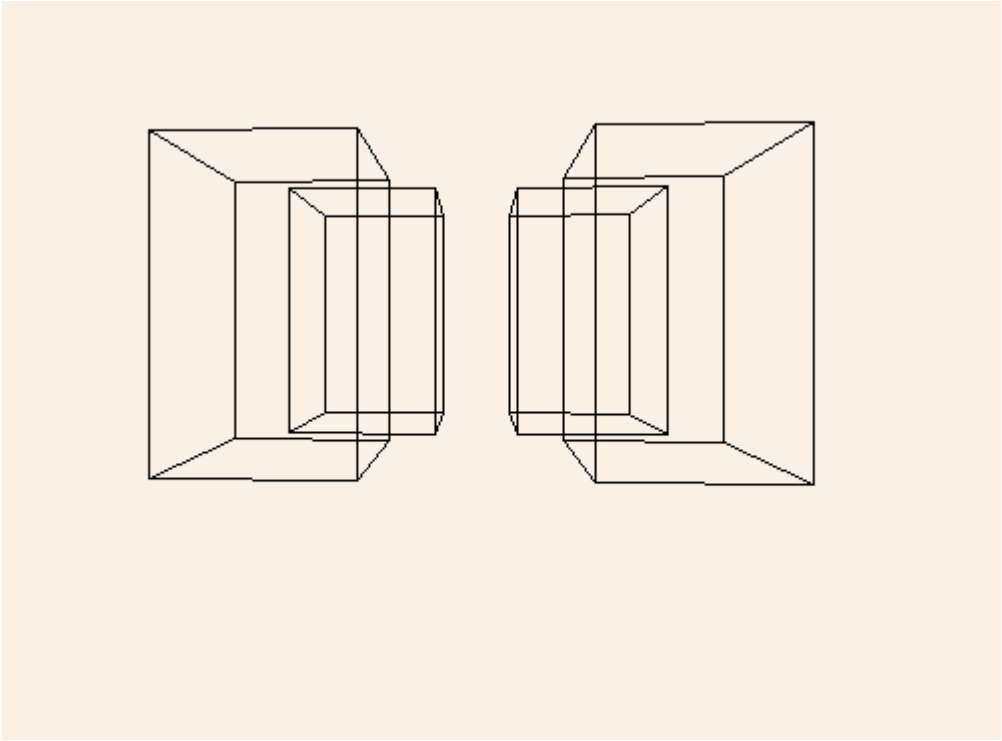


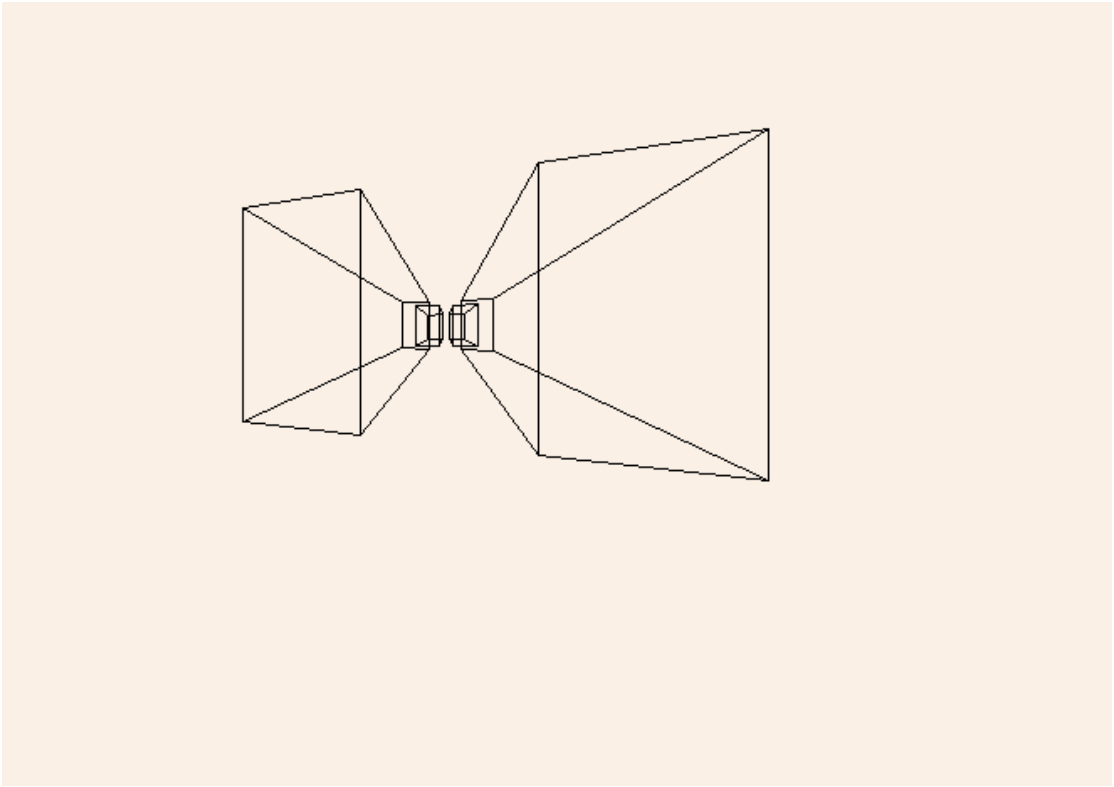
Pierwszy i ostatni obraz nie wyglądają tak samo, dlatego że scena nie jest symetryczna po jednej stronie prawy prostopadłościan jest większy bo jest bliżej po drugiej lewy jest większy. Jednak wszystko wygląda jakby renderowało się poprawnie. Podobnie możemy przetestować inne rotacje i translacje.

Test zoom

Przebieg testu:

- Odzoomowanie
- Translacja do przodu





Widać ewidentną różnicę między translacją, a zoomem.

Wnioski

Realizacja projektu pozwoliła mi na zagłębienie się w tematykę reprezentacji obiektów przestrzennych, wyświetlania ich oraz przeprowadzania transformacji tych obiektów.

Dowiedziałem się, że zmiana położenia kamery na dowolny punkt w przestrzeni jest złożeniem wielu transformacji. Poznałem macierze transformacji i nauczyłem się wykorzystywać je do zmiany położenia punktów w przestrzeni.

Dowiedziałem się jak generować przybliżenie na etapie rzutowania punktów przestrzennych na płaszczyznę.

Źródła

https://www.javatpoint.com/computer-graphics-3d-rotation-about-arbitrary-axis

https://www.javatpoint.com/computer-graphics-3d-transformations