Wydział Matematyki i Nauk Informacyjnych

Laboratorium systemów CAD/CAM

Rubik's Snake

Dokumentacja

Jakub Waszkiewicz

9 lipca 2019

1 Opis

Aplikacja służy do wirtualizacji modelu zabawki zwanej Rubik's Snake. Została stworzona w celach edukacyjnych i do rozwiązania problemu odwrotnego zabawki, czyli wyznaczenia jej ustawienia w celu umieszczenia ostatniej jej części w określonym miejscu w przestrzeni względem pierwszego elementu

2 Wymagania funkcjonalne

Poniższa tabela zawiera opis poszczególnych przypadków użycia

Nazwa	Opis
Wybór elementu	Użytkownik może wybrać jeden z elementów zabawki za pomocą rozwijanej listy. Wybrany element jest zaznaczony specjalnym kolorem
Obrót elementu	Użytkownik może zmienić pozycję elementu przez zmianę kąta jej obrotu względem dozwolonej osi
Odczytania pozycji ostatniego element	Użytkownik może zobaczyć, jaka jest pozycja ostatniego elementu w przestrzeni względem elementu pierwszego zabawki
Ustawienie pozycji ostatniego elementu	Użytkownik może wybrać, w jakim punkcie ma się znajdować ostatni element, aby wyznaczyć pozycję zabawki tak, aby spełniała ten warunek
Zmiana długości zabawki	Użytkownik może wybrać, z ilu elementów ma się składać zabawka

Tabela 1: Opis przypadków użycia

3 Wymagania niefunkcjonalne

Wymagania niefunkcjonalne zostały przedstawione w formie kategorii URPS:

- **Użyteczność**
  - Wszystkie funkcjonalności będą dostępne w formie klienckiej aplikacji webowej
- **Niezawodność**
  - Aplikacja będzie dostępna w trybie 24x7 przez co najmniej 99.9% czasu
- **Wydajność**
  - Aplikacja serwera będzie obsługiwała żądania, przetwarzała je i wysyłała odpowiedź do aplikacji klienckiej w czasie nie dłuższym niż minuta
- **Utrzymanie**

4 Architektura rozwiązania

4.1 Opis teoretyczny

Zakładamy, że w pozycji początkowej dla zabawki składającej się z n elementów pierwszy element znajduje się w początku układu współrzędnych, a wszystkie kolejne leżą wzdłuż osi OX. Wtedy ostatni element znajduje się w

$$\begin{pmatrix} 2(n-1) \\ 0 \\ 0 \end{pmatrix}$$

Aby wyznaczyć pozycję danego punktu po jego obrocie, należy przemnożyć go z lewej strony przez odpowiednią macierz obrotu - w tym celu musimy skorzystać z rozszerzonych współrzędnych o 4. współrzędną W zawsze równą 1. Dla dowolnej osi obrotu macierz wygląda następująco:

$$\begin{pmatrix} cos(\alpha) + u_z^2(1 - cos(\alpha)) & u_x u_y(1 - cos(\alpha)) - u_z sin(\alpha) & u_x u_z(1 - cos(\alpha)) + u_y sin(\alpha) & 0 \\ u_z u_z(1 - cos(\alpha)) + u_z sin(\alpha) & cos(\alpha) + u_y^2(1 - cos(\alpha)) & u_y u_z(1 - cos(\alpha)) - u_z sin(\alpha) & 0 \\ u_z u_x(1 - cos(\alpha)) - u_y sin(\alpha) & u_z u_y(1 - cos(\alpha)) + u_x sin(\alpha) & cos(\alpha) + u_z^2(1 - cos(\alpha)) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Gdzie α jest kątem obrotu, a u jest wersorem wyznaczającym oś obrotu. W naszym przypadku, w zależności od elementu, możliwe są jedynie 2 wartości tego wersora:

$$\begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \\ 0 \end{pmatrix} \text{ lub } \begin{pmatrix} \frac{-\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \\ 0 \end{pmatrix}$$

Powyższe macierze jednak służą do obrotu względem osi przechodzącej przez środek układu współrzędnych. Aby obrócić element względem osi przechodzącej przez dowolny punkt, należy najpierw (za pomocą macierzy translacji) 'przesunąć' ten punkt do początku układu współrzędnych, obrócić, a następnie przesunąć punkt do poprzedniej pozycji.

Zatem dla obrotu danego elementu możemy wyznaczyć współrzędne ostatniego punktu jako:

$$p' = M_{trans1} M_{ob} M_{trans2} p$$

Gdzie:

- p - początkowa pozycja punktu,
- p' - nowa pozycja punktu,
- M_{trans1}, M_{trans2} - odpowiednie macierze translacji
- M_{ob} - macierz obrotu

Dzięki temu możemy wyznaczyć pozycję ostatniego punktu przy dowolnym ciągu kątów obrotu $\alpha_1 \dots \alpha_n$. Bardzo ważna jest tu kolejność mnożeń przekształceń - zaczynamy od przemnożenia punktu początkowego przez macierze przekształcenia po obrocie ostatniego punktu. Gdybyśmy zmienili kolejność, to nie wiadomo by było, w jakiej pozycji znajduje się następny element, który musimy obrócić. Przy tej kolejności mamy pewność, że jest on wciąż w pozycji początkowej w odległości i od początku układu, gdzie i jest indeksem tego elementu.

Ponieważ nie jesteśmy w stanie jawnie zapisać w postaci jednej macierzy całego równania, nie możemy w prosty sposób znaleźć konfiguracji kątów dla danej pozycji ostatniego punktu. Dlatego też problem ten należy rozwiązać **iteracyjną metodą Newtona** do znalezienia najbliższego faktycznemu rozwiązaniu przybliżenia. Jedną iterację tej metody możemy zapisać wzorem:

$$F'(x_k)p = -F(x_k)$$

Gdzie:

- x_k - aktualne przybliżenie (wektor kątów)
- p - wektor poprawki (kolejne przybliżenie x_{k+1} jest równe $x_k + p$)
- $F(x_k)$ - wektor funkcji (w naszym przypadku jest to pozycja ostatniego punktu dla wektora kątów x_k)
- $F'(x_k)$ - macierz pochodnych, której i-ta kolumna jest wektorem pochodnych cząstkowych funkcji po i-tym elemencie wektora x_k

Nie jesteśmy w stanie wyznaczyć dokładnej wartości pochodnej w danym punkcie, dlatego też korzystamy z przybliżenia tej pochodnej na podstawie szeregu Taylora

$$f'(x) \approx \frac{-3f(x) + 4f(x+h) - f(x+2h)}{2h}$$

Gdzie h jest pewną małą wartością. Wzór ten stosujemy dla każdego elementu wektora funkcji.

Niestety macierz $F'(x_k)$ nie musi być kwadratowa (zabawka nie ma zawsze 4 elementów, a punkty mają zawsze 3 współrzędne). Dlatego też do rozwiązania powyższego układu równań liniowych należy wykorzystać metodę **rozkładu macierzy QR**.

4.2 Wykorzystane technologie

Aplikacja kliencka została napisana przy użyciu języka **JavaScript** oraz **HTML** z wykorzystaniem biblioteki **jQuery**.

Do wizualizacji modelu zabawki wykorzystano gotową bibliotekę z silnikiem graficznym **ThreeJS**. Poszczególne elementy strony zostały wystylizowane przy użyciu biblioteki **Bootstrap**.

Aplikacja została udostępniona publicznie przy użyciu serwisu **AWS S3** jako statyczna strona.

Do rozwiązania problemu odwrotnego został wykorzystany język **C#** wraz z **.NET Core**. Do obliczeń użyto biblioteki **Math.NET Numerics**.

Serwis do wykonywania obliczeń do wyznaczenia kątów obrotu zabawki został udostępniony za pomocą serwisu **Azure Functions**.