Uniwersytet Bielsko-Bialski

Sprawozdanie

Zajęcia: Grafika Komputerowa
(Ćwiczenia laboratoryjne)
Prowadzący: prof dr. hab. Vasyl Martsenyuk

Laboratoria nr: 10
Temat ćwiczenia:

ZadanieWebGL

Maksymilian Wójcik
Imformatyka I stopnia
niestacjonarne
4 semestr
gr. 1A

1.Polecenie:

Program w lab11.html pokazuje wiele ruchomych czerwonych kwadratów, które odbijają się od krawędzi płótna. Płótno wypełnia cały obszar zawartości przeglądarki internetowej. Kwadraty odpowiadają również myszą: Jeśli klikniesz lewym przyciskiem myszy lub klikniesz lewym przyciskiem myszy i przeciągniesz na płótnie, cały kwadrat będzie kierowany w stronę pozycji myszy. Jeśli klikniesz lewym przyciskiem myszy, dane punktów zostaną ponownie zainicjowane, więc zaczną się od środka. Możesz wstrzymać i ponownie uruchomić animację, naciskając spację...

2. Wykorzystane komendy:

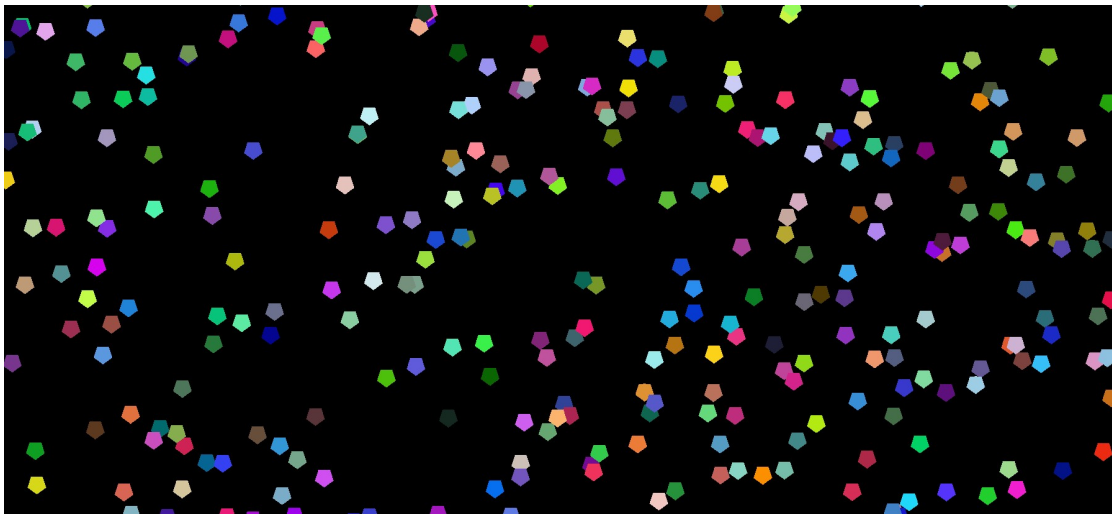
```
function render() {
  gl.clear(gl.COLOR_BUFFER_BIT);

  gl.bindBuffer(gl.ARRAY_BUFFER, vertexCoordsBuffer);
  gl.bufferData(gl.ARRAY_BUFFER, positions, gl.STREAM_DRAW);
  gl.vertexAttribPointer(vertexCoordsLoc, 2, gl.FLOAT, false, 0, 0);

  if (isColorRandom) {
    gl.enableVertexAttribArray(vertexColorLoc);
  } else {
    gl.disableVertexAttribArray(vertexColorLoc);
    gl.vertexAttrib3f(vertexColorLoc, 0.7, 0.2, 0.5);
  }

  gl.drawArrays(gl.POINTS, 0, POINT_COUNT);
  if (gl.getError() != gl.NO_ERROR) {
    console.log("During render, a GL error has been detected.");
  }
}
```

3.Wynik działania:



4.Wnioski

Różnorodność kolorów i stylów: Wprowadzenie losowych kolorów i różnych stylów rysowania punktów znacząco zwiększa wizualną atrakcyjność i elastyczność programu. Umożliwienie użytkownikowi przełączania między jednolitymi i losowymi kolorami oraz różnymi stylami punktów sprawia, że interakcja z aplikacją jest bardziej angażująca i dynamiczna.

Zaawansowane shaderowanie: Dodanie nowych zmiennych atrybutów i jednolitych do shaderów pozwala na zaawansowane manipulacje wizualne. Zmienne atrybutów dla kolorów i jednolite dla stylów umożliwiają tworzenie złożonych efektów wizualnych bez konieczności zmiany geometrii obiektów.

Interakcja użytkownika: Reagowanie na klawisze i kliknięcia myszą pozwala użytkownikom na bezpośrednie wpływanie na stan i wygląd aplikacji. Przykładowo, umożliwienie użytkownikowi zatrzymania i wznowienia animacji za pomocą spacji oraz kierowania kwadratami w stronę kursora myszy sprawia, że program staje się bardziej interaktywny i intuicyjny.

Optymalizacja i wydajność: Użycie WebGL do renderowania wielu obiektów jednocześnie, a także efektywne zarządzanie buforami i atrybutami, jest kluczowe dla utrzymania płynności animacji. Dodatkowo, korzystanie z funkcji gl.BLEND i gl.blendFunc dla przezroczystości alfa pozwala na tworzenie płynnych przejść i efektów wizualnych bez znacznego wpływu na wydajność.

Elastyczność i rozszerzalność: Projektowanie kodu w sposób modularny, z oddzielnymi funkcjami dla inicjalizacji, aktualizacji i renderowania, umożliwia łatwą modyfikację i rozbudowę programu. Dodanie nowych funkcji, takich jak dodatkowe style punktów czy

nowe typy interakcji, jest prostsze dzięki jasno określonym strukturom kodu.

Nauka shaderów i WebGL: Implementacja tego rodzaju funkcjonalności jest świetnym ćwiczeniem w nauce shaderów i programowania WebGL. Pozwala na zrozumienie, jak działa pipeline renderowania, jak tworzyć i zarządzać shaderami oraz jak efektywnie wykorzystać możliwości WebGL do tworzenia złożonych efektów graficznych.

Kod źródłowy: https://github.com/mwojcik123/UBB-GK-MW