

Grafika komputerowa i komunikacja człowiek - komputer

WebGL
20.01.2023r.

inż. Maciej Radecki
album: 253257

Prowadzący: dr inż. Jan Nikodem

Termin: Piątek TN, 16:25

1. Wstęp

WebGL to API rozszerzające język javascript o funkcje pozwalające na obsługę grafiki 3D w przeglądarce. W tym celu korzysta się z elementu HTML5 - canvas. WebGL rozszerza context elementu canvas o trzeci wymiar. Wiele instrukcji i rozwiązań jest identycznych lub zbliżonych do tych znanych z OpenGL.

2. Realizacja ćwiczenia

Struktura programu składa się z folderu głównego i dwóch podfolderów. W folderze głównym przechowywany jest plik index.html - to strona główna oraz w tym wypadku jedyna strona aplikacji webowej oraz pliki png z teksturami. Podfoldery to CSS i JS. Przechowywane są w nich odpowiednio pliki .css (arkusze stylów), .js (skrypty w języku javascript). Program jest skupiony na obsłudze WebGL, a nie na konstruowaniu strony internetowej, więc foldery CSS i JS nie są tutaj szczególnie istotne - arkusz stylów jest minimalistyczny, a plików .js jest niewiele i mogłyby być przechowywane w głównym katalogu projektu. Wszystkie pliki bazują na tych pokazanych w instrukcji laboratoryjnej.

Plik index.html przypisuje funkcję runWebGL() jako parametr onLoad dla body. Jednak w ostatecznym programie, funkcja ta powinna być przypisana do guzika. W związku z tym należało zmodyfikować plik index.html.

```
<!DOCTYPE html>
<html>
<head>
  <title>WebGL</title>
  <link rel="stylesheet" href="css/style.css">
</head>
<body>

  <div class="container">
    <div class="row justify-content-md-center">
      <form>
        <div class="form-group">
          <br/>
          <input type="checkbox" class="form-check-input" id="rotateX"> rotacja X
          <br/>
          <input type="checkbox" class="form-check-input" id="rotateY"> rotacja Y
          <br/>
          <input type="checkbox" class="form-check-input" id="rotateZ"> rotacja Z
          <br/>
          <button class="btn btn-success" onclick="runWebGL(); return false;">Zastosuj</button>
        </div>
      </form>

      <div class="form-group">
        <button class="btn btn-success" onclick = "changeTexture()">Zmień teksturę</button>
        <br>
        <button class="btn btn-success" onclick = "changeModel()">Zmień obiekt</button>
      </div>

      <div class="col-sm-8">
        <canvas id="glcanvas" width="500" height="300">
          Brak wsparcia dla elementu HTML5 typu canvas
        </canvas>
        <br/>
      </div>
    </div>
  </div>

  <script src="js/main.js"></script>
  <script src="js/matrix.js"></script>
  <script>runWebGL()</script>
</body>
</html>
```

Rysunek 1 - Zawartość pliku index.html

W ostatecznej wersji kodu z instrukcji program rysuje otekstutowany sześciąt. Należy zmienić rysowaną figurę na czworoscian. Oprócz tego, po dodaniu przypisania funkcji runWebGL do guzika, pojawił się nowy problem do rozwiązania - każde wciśnięcie guzika zwiększa prędkość rotacji narysowanego obiektu, choć prędkość powinna być stała.

W celu narysowania czworoscianu należy opisać 4 wierzchołki i połączyć je w trójkąty.

```
if (!model) { //czworoscian
    var triangleVertices = [
        1, 1, 1,      0, 0,
        -1, -1, 1,    1, 0,
        -1, 1, -1,    1, 1,
        1, -1, -1,    0, 1
    ]

    var triangleFaces = [
        0, 1, 2,
        3, 0, 1,
        3, 0, 2,
        3, 1, 2
    ];
}
```

Rysunek 2 - wierzchołki czworoscianu i dobranie trójkątów

Po odpowiednim dobraniu trójkątów, należy dopasować rozmiar bufora do ilości rysowanych elementów.

```
_triangleVertexBuffer = gl_ctx.createBuffer();
gl_ctx.bindBuffer(gl_ctx.ARRAY_BUFFER, _triangleVertexBuffer);
gl_ctx.bufferData(gl_ctx.ARRAY_BUFFER,
    new Float32Array(triangleVertices),
    gl_ctx.STATIC_DRAW
);

_triangleFacesBuffer = gl_ctx.createBuffer();
gl_ctx.bindBuffer(gl_ctx.ELEMENT_ARRAY_BUFFER, _triangleFacesBuffer);
gl_ctx.bufferData(gl_ctx.ELEMENT_ARRAY_BUFFER,
    new Uint16Array(triangleFaces),
    gl_ctx.STATIC_DRAW);
}
```

Rysunek 3 – Bufor

Animacja przyspieszała przy każdym kliknięciu w guzik, ponieważ po wciśnięciu guzika wywoływana jest komenda window.RequestAnimationFrame - pętla ta nie jest przerywana odpowiednim poleceniem, a więc po kolejnym wciśnięciu guzika uruchamiana jest równolegle kolejna pętla RequestAnimationFrame. Rozwiązaniem problemu okazało się przypisanie wartości zwracanej przez tę funkcję do zmiennej, w celu przekazania jej jako parametru funkcji window.CancelAnimationFrame, czyli funkcji przerywającej pętlę. Przy każdym wciśnięciu

guzika, ale przed rozpoczęciem kolejnej pętli animacji, sprawdzane jest requestId i jeśli jest już uruchomiona pętla RequestAnimationFrame, przerywa ją.

```
    requestId = window.requestAnimationFrame(animate);  
  };  
  if(requestId)  
    window.cancelAnimationFrame(requestId);  
  animate(0);
```

Rysunek 4 - kod rozwiązujący problem przyspieszającej animacji

Program poprawnie rysuje czworościan z nałożoną teksturą, pozwala na włączenie rotacji wobec wybranych osi, a animacja nie przyspiesza po kolejnych wciśnięciach guzika. Dodatkowymi opcjami są możliwość zmiany tekstury na rysowanym modelu, a także zmiana modelu na oryginalnie tworzony sześcián.



Rysunek 5 - stworzona strona internetowa

W przedstawionym programie zmodyfikowany został również arkusz stylów co pozwoliło na poprawę wyglądu stworzonej strony.

```

canvas#gcanvas {
  border: 1px solid #666660;
  background-color: #545469;
}

html {
  display: table;
  margin: auto;
}

body {
  display: table-cell;
  vertical-align: middle;
}

.btn-success{color:#fff;background-color:#28a745;border-color:#28a745}
.btn-success:hover{color:#fff;background-color:#218838;border-color:#1e7e34}
.btn-success:focus,
.btn-success:active{box-shadow:0 0 0 .2rem rgba(40,167,69,.5)}

.btn{display:inline-block;margin:2px; font-weight:400;text-align:center;white-space:nowrap;vertical-align:middle;-webkit-user-select:none;-moz-user-select:none;-ms-user-select:none;
user-select:none;border:1px solid transparent;padding:.375rem .75rem;font-size:1rem;line-height:1.5;border-radius:.25rem;transition:color .15s ease-in-out,background-color .15s
ease-in-out,box-shadow .15s ease-in-out}@media screen and (prefers-reduced-motion:reduce){.btn{transition:none}}

```

Rysunek 6 - arkusz stylów .css