
Human-Computer Interaction

ICG ToDo Liste Übung 6, WiSe 2023/24

Aufgabe 0 (Vorbereitung)

In dieser Übung soll nun die Anwendung der letzten Wochen interaktiv gestaltet werden, nutzen Sie daher als Basis für diese Übung Ihr Projekt von letzter Woche.

- **Aufgabe:**

- Falls noch nicht geschehen, laden Sie die Bibliothek glMatrix aus Moodle herunter und fügen Sie diese in Ihr Projekt ein.

- **Tipps:**

- Achten Sie darauf, die Bibliothek dem restlichen JavaScript Code bekannt zu machen, in dem Sie diese in die Index.html einbinden.
- Die Dokumentation für glMatrix ist unter <http://glmatrix.net/docs/> zu finden.

Aufgabe 1 (Wiederholung)

- **Aufgabe:**

- Falls noch nicht geschehen, setzen Sie die View-Matrix aus dem Vertexshader vom Anwendungsprogramm aus.

- **Wichtige Funktionen:**

```
gl.getUniformLocation(...);
gl.uniformMatrix[1234][fi][v](...);
```

Aufgabe 2

- **Aufgabe:**

- Schreiben Sie mit Hilfe von `requestAnimationFrame` eine Renderschleife, welche die Objekte kontinuierlich rendert.

- **Tipps:**

- Sie können `requestAnimationFrame` eine Funktion wie folgt übergeben:

```
let lastTimestamp;

...

function render(timestamp) {
  if (lastTimestamp == undefined) {
    lastTimestamp = timestamp;
  }

  const elapsed = timestamp - lastTimestamp;

  // render here!
  ...
}

function main() {
  // init here
  ...

  window.requestAnimationFrame(render);
}
```

Aufgabe 3

- **Aufgabe:**

- Die vorgegebene View-Matrix soll nun durch eine Matrix ersetzt werden, welche die Kamera um die Insel rotiert. Hierzu soll zur Berechnung der View-Matrix die Funktion `mat4.lookAt(out, eye, center, up)` verwendet werden. Weiter sollen die Kamerakoordinaten abhängig von der Zeit verschoben werden, um eine Rotation zu erhalten.
- Nutzen Sie für die Kamera den Koordinatenvektor

$$p = (3 \cdot \sin(0.001 \cdot \text{time}), 1, 3 \cdot \cos(0.001 \cdot \text{time}))$$

als neue Position.

- **Tipps:**

- Da das Konzept der Projektionsmatrizen noch nicht behandelt wurde, können Sie folgenden Programmcode als Startpunkt für die Aufgabe verwenden.

```
...
let modelViewProjection;
let modelMatrix;
let viewMatrix;
let projectionMatrix;

function main() {
  ...

  viewMatrix = mat4.create();
  projectionMatrix = mat4.create();
  modelViewProjection = mat4.create();

  mat4.perspective(projectionMatrix, 90.0, 1.0, 0.0001, 1000.0)
}

function render(timestamp) {
  if (lastTimestamp == undefined) {
    lastTimestamp = timestamp;
  }

  const elapsed = timestamp - lastTimestamp;

  // Code zur rotation hier!
  // Schreiben Sie Ihr Ergebniss dann in die Variable viewMatrix

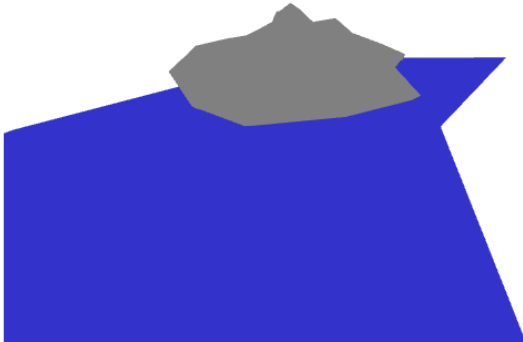
  mat4.multiply(modelViewProjection, projectionMatrix, viewMatrix)

  viewLoc = gl.getUniformLocation(program, "viewMatrix");
  gl.uniformMatrix4fv(viewLoc, false, modelViewProjection);

  // Ab hier rendern
  ...
}
```

- Überlegen Sie sich, welchen einzelnen Parameter Sie bei `mat4.lookAt(out, eye, center, up)` verändern müssen.
- Denken Sie auch daran, dass die Geschwindigkeit der Rotation abhängig von der Zeit ist, um zu schnelle Rotationen zu vermeiden.

Aufgabe 4



```
Inspektor  Konsole  Debugger  Netzwerkanalyse  Stilbearbeitung  Laufzeitanalyse  Speicher  Web-Speicher  Barrierefreiheit  Anwendung
Ausgabe filtern
Coordinates: (229,207)
115
100
103
Coordinates: (228,207)
100
115
103
115
100
103
102
110
Coordinates: (228,209)
Coordinates: (150,327)
Coordinates: (3,634)
Coordinates: (2432,4)
Coordinates: (2442,32)
Coordinates: (2473,78)
Coordinates: (2494,94)
Coordinates: (2503,98)
Coordinates: (2514,100)
Coordinates: (2518,89)
```

- **Aufgabe:**

- Schreiben Sie nun eine Funktion, welche die Mouse-Events abfängt und in die Konsole ausgibt. Implementieren Sie danach eine weitere Funktion, welche den Keycode der auf der Tastatur gedrückten Taste ausgibt.

- **Tipps:**

- Nutzen Sie für die Aufgabe die JavaScript Events 'keydown' und 'mousemove'.
- Um diese dem Browser bekannt zu machen nutzen Sie die Funktion `window.addEventListener(...)`.

Aufgabe 5

- **Aufgabe:**
 - Kombinieren Sie nun Aufgabe 3 und Aufgabe 4 um die Rotation abhängig von der horizontalen Mausbewegung der Nutzenden zu machen.
- **Tipps:**
 - Passen Sie mit den in Aufgabe 4 ausgelesenen Werten den LookAt Vektor der Kamera an.

Aufgabe 6 (Optional)

- Fügen Sie weitere Bewegungsmöglichkeiten, wie Beispielsweise eine Vorwärtsbewegung der Kamera durch drücken einer Taste oder das Zoomen mit Hilfe des Mausekzes ein.