CS405 Computer Graphics **Barış Ulaş Çukur**
Project 1 **29461**

Task 1: Getting model view from ChatGPT

Asking ChatGPT about the transformation matrix resulted with an array with (wrongly) calculated vertices (Figure 1.1). The full chat log with ChatGPT is available through this URL: https://chat.openai.com/share/9796d847-bdf9-4904-a037-3ff3d215d36e.

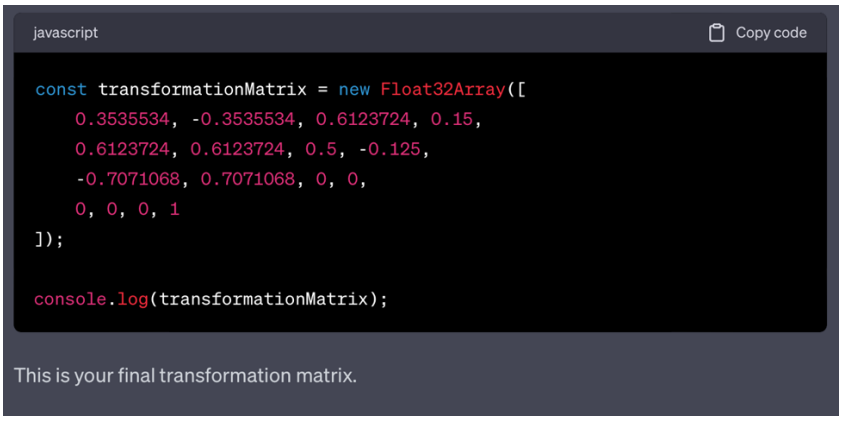

Figure 1.1: The resulting array from ChatGPT.

Rendering the calculated model from ChatGPT with WebGL reveals that the cube is distorted due to inaccurate offset calculations (Figure 1.2).

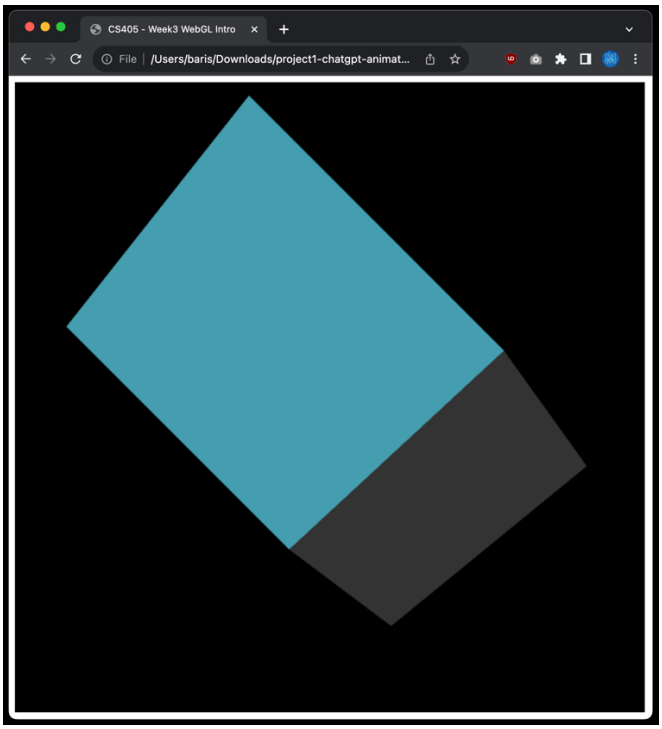

Figure 1.2: Rendering the distorted ChatGPT cube with WebGL.

Task 2: Calculating the matrix using the given methods

This task asks to transform the matrix using the given methods by the specific transformations given in transformation-prompt.txt file.

To accomplish the task, specific transformation matrices must be generated using given methods with following properties:

1) An identity matrix to initialize the cube.
2) A transformation matrix to move the cube +0.3 in x-axis and -0.25 in y-axis.
3) A scaling matrix to scale the cube 0.5 in x and y axes.
4) A rotation matrix to rotate the cube 30 degrees on x-axis.
5) A rotation matrix to rotate the cube 45 degrees on y-axis.
6) A rotation matrix to rotate the cube 60 degrees on z-axis.
7) Miscellaneous other matrices to multiply the resulting matrices.

The code works by creating the specified matrices and returning the final matrix after completing all the steps (Figure 2.1).

```
164  function getModelViewMatrix() {
165
166      const identityMatrix = createIdentityMatrix();
167      const translationMatrix = createTranslationMatrix(0.3, -0.25, 0);
168      const scalingMatrix = createScaleMatrix(0.5, 0.5, 1);
169      const rotationMatrixX = createRotationMatrix_X(30/180 * Math.PI);  // 30 degrees
170      const rotationMatrixY = createRotationMatrix_Y(45/180 * Math.PI);  // 45 degrees
171      const rotationMatrixZ = createRotationMatrix_Z(60/180 * Math.PI);  // 60 degrees
172
173      const m1 = multiplyMatrices(identityMatrix, translationMatrix);
174      const m2 = multiplyMatrices(m1, scalingMatrix);
175      const m3 = multiplyMatrices(m2, rotationMatrixX);
176      const m4 = multiplyMatrices(m3, rotationMatrixY);
177      const finalMatrix = multiplyMatrices(m4, rotationMatrixZ);
178
179      return finalMatrix;
180
181  }
```

Figure 2.1: getModelViewMatrix function applying the specified transformations to the matrix.

After completing this step and rendering the cube in WebGL, this time a correctly transformed cube appears (Figure 2.2).

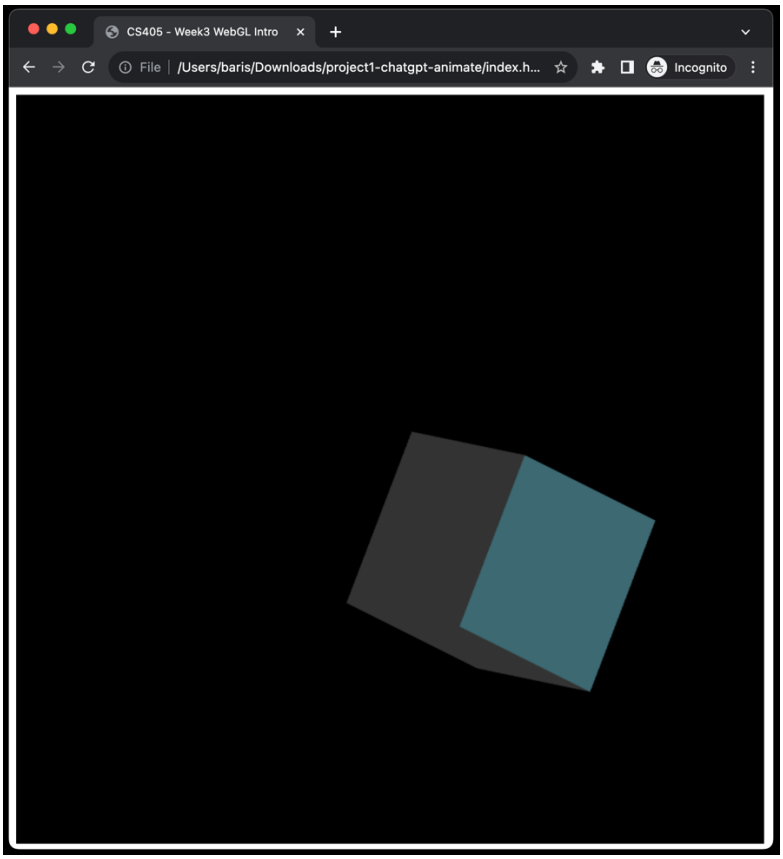

Figure 2.2: Rendering the cube after applying the specified transformations using the given methods.

Task 3: Animating the cube

Asking ChatGPT to animate the cube without the proper context resulted in erroneous answers. However, given the proper context (i.e., the source code of the full utils.js file), it successfully created a working getPeriodicMovement function (Figure 3.1). The chat log URL for this task is available through this URL: https://chat.openai.com/share/c3de61c1-a2dc-4a83-8894-b5d561cad63f.

```
function getPeriodicMovement(startTime) {
    const elapsed = (Date.now() - startTime) / 1000; // Elapsed time in seconds
    const period = 10; // Total period for one complete cycle (in seconds)
    const halfPeriod = period / 2; // Half of the period

    // Calculate the normalized time within the current period
    const normalizedTime = elapsed % period / period;

    // Interpolate between the identity matrix and the target matrix based on the normalized time
    let interpolatedMatrix = glMatrix.mat4.create();

    if (normalizedTime < 0.5) {
        // First 5 seconds: Transform from initial to target position
        const t = normalizedTime / 0.5; // Normalize time within the first half
        for (let i = 0; i < 16; i++) {
            interpolatedMatrix[i] = (1 - t) * createIdentityMatrix()[i] + t * getModelViewMatrix()[i];
        }
    } else {
        // Next 5 seconds: Transform from target to initial position
        const t = (normalizedTime - 0.5) / 0.5; // Normalize time within the second half
        for (let i = 0; i < 16; i++) {
            interpolatedMatrix[i] = (1 - t) * getModelViewMatrix()[i] + t * createIdentityMatrix()[i];
        }
    }

    return interpolatedMatrix;
}
```

Figure 3.1: The getPeriodicMovement function generated by ChatGPT.

The function basically creates cube matrices in between the identity matrix and the final matrix generated using the getModelViewMatrix function. It works in two phases, the animation of the cube moving to the destination (i.e., the transformed matrix generated by getModelViewMatrix function) and the animation of the cube reversing the action (moving to its initial position).

During the first 5 seconds, it creates an interpolated matrix using the linear interpolation method. This allows getting unique matrices for each frame of the animation. Similarly, during the last 5 seconds, it gets interpolated matrices using linear interpolation. However, this time it swaps the arguments to make the animation start from the transformed location instead of starting from the identity matrix.