CS405 Project 1: 3D Animations using ChatGPT Report

Ulaş Yıldız - 24931

1. Introduction

In this project, it is aimed to apply 3D transformations to a cube object, calculate transformation matrices with the help of ChatGPT and combine these transformations with animations. In the project, the position and dimensions of the cube were changed by applying displacement, scaling and rotation operations respectively. In addition, transformation matrices were calculated using ChatGPT and the results were compared with manual calculations. Finally, based on these transformations, an animation was created in which a cube moves in a 10-second cycle.

2. Task 1: Calculate ModelView Matrix with ChatGPT

In this task, I calculated the model-view matrix of the cube using ChatGPT for transformation operations. I obtained the required transformation matrix by pasting the text from the transformation-prompt.txt file in the project folder into ChatGPT. The given transformations were respectively as follows:

- Translation: 0.3 units on x-axis, -0.25 units on y-axis,
- Scaling: 0.5 on x and y axes, z axis fixed at 1,
- Rotation:
    - 30 degrees around the x-axis,
    - 45 degrees around the y-axis,
    - 60 degrees around the z axis.

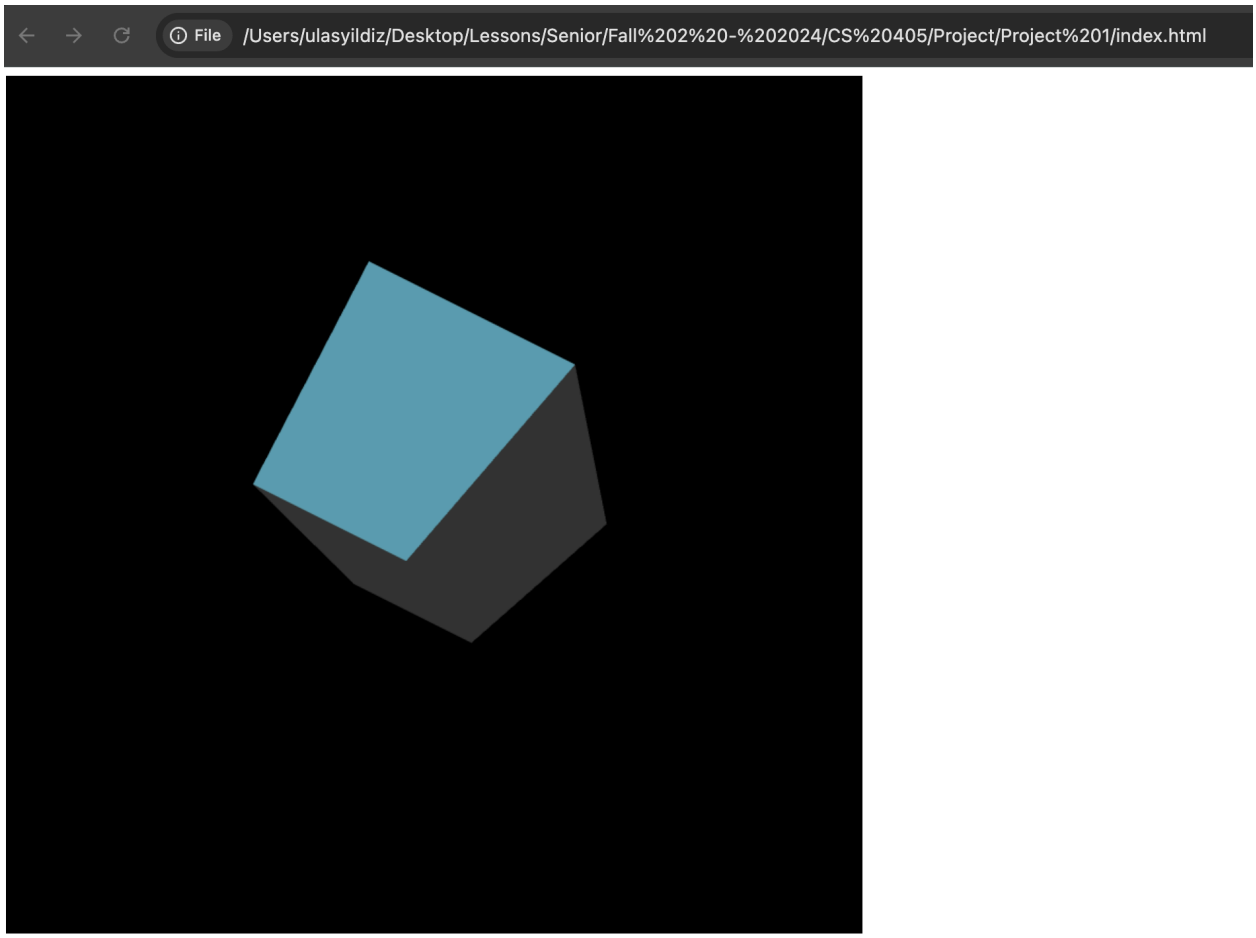
I asked ChatGPT to calculate these transformations and return the result in Float32Array format. I pasted the response into the getChatGPTModelViewMatrix() function.

Response from ChatGPT:

```
const transformationMatrix = new Float32Array([
      // you should paste the response of the chatGPT here:
      0.17677669, -0.28661165, 0.7391989,  0.3,
      0.30618623,  0.36959946, 0.2803301, -0.25,
     -0.35355338,  0.17677669, 0.61237246, 0,
      0,           0,          0,          1
  ]);
```

I added this matrix to the getChatGPTModelViewMatrix() function in the project and applied the transformations of the cube.

Here is a screenshot of the cube with the transformation matrix calculated by ChatGPT.



Shareable link to the conversation with ChatGPT: [Task 1 GPT Chat]

In this step, the result from ChatGPT was successfully integrated into the project and the new position and dimensions of the cube were determined by applying transformation operations.

3. Task 2: Calculate ModelView Matrix Manually

In this task, I manually calculated the transformation matrix of the cube using the getModelViewMatrix() function. This time I performed the transformation calculation done by ChatGPT manually. Manually, the following transformation operations were applied in order:

Given Transformations:

- Translation: 0.3 units on x-axis, -0.25 units on y-axis.
- Scaling: 0.5 on x and y axes, z axis fixed at 1.
- Rotation:
    - 30 degrees around the x-axis,
    - 45 degrees around the y-axis,
    - 60 degrees around the z-axis.

I calculated these transformations manually with the help of createTranslationMatrix(), createScaleMatrix() and createRotationMatrix_X(), createRotationMatrix_Y(), createRotationMatrix_Z() functions. The calculated transformation matrices were multiplied respectively and the final model-view matrix was obtained.

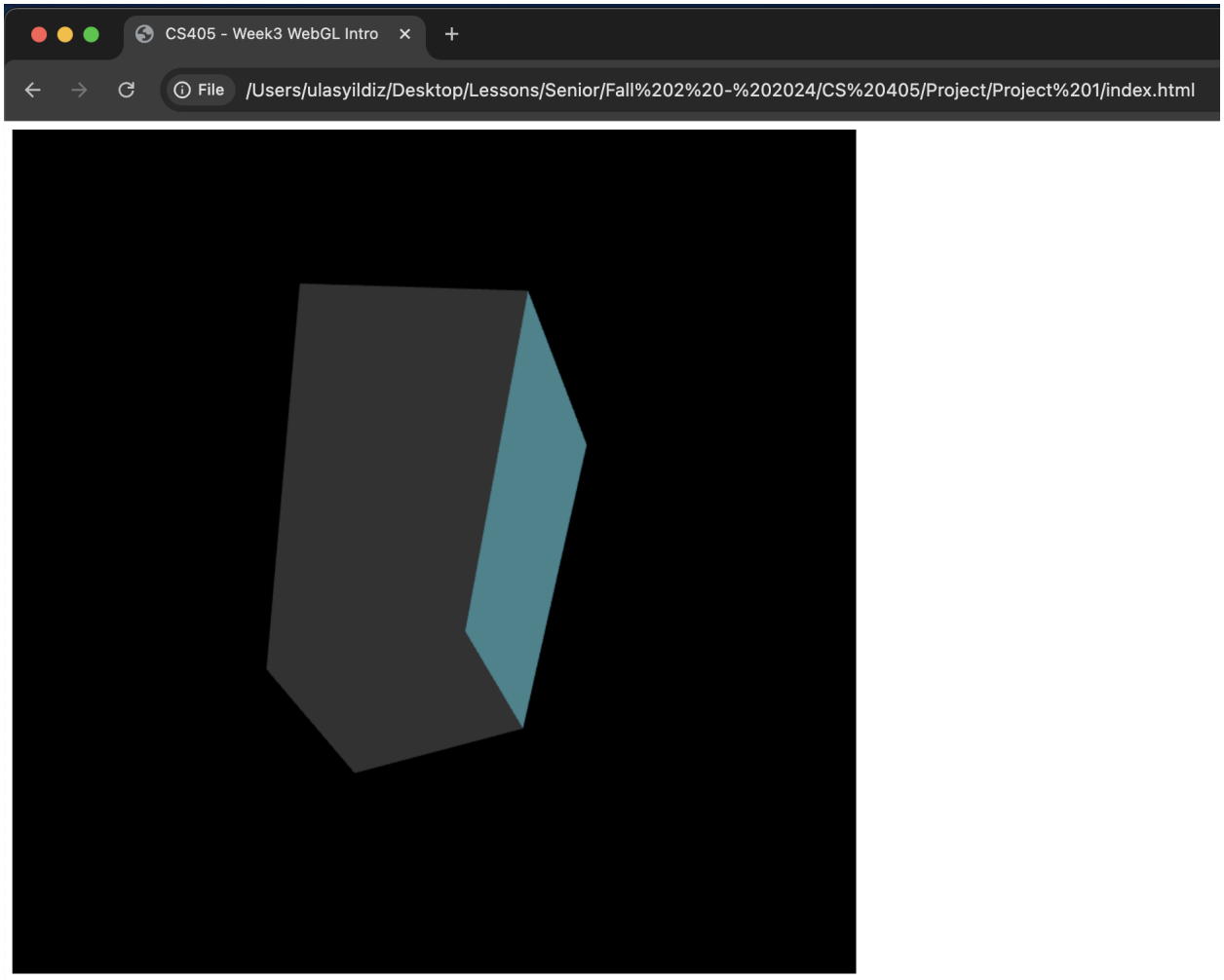
Code Based on Manual Calculation:

```
function getModelViewMatrix() {
   // calculate the model view matrix by using the transformation
   // methods and return the modelView matrix in this method
   const translationMatrix = createTranslationMatrix(0.3, -0.25, 0); // translation
   const scalingMatrix = createScaleMatrix(0.5, 0.5, 1); // scaling
   const rotationXMatrix = createRotationMatrix_X(Math.PI / 6); // 30 degrees
   const rotationYMatrix = createRotationMatrix_Y(Math.PI / 4); // 45 degrees
   const rotationZMatrix = createRotationMatrix_Z(Math.PI / 3); // 60 degrees

   let modelViewMatrix = createIdentityMatrix(); // start with the identity matrix
   modelViewMatrix = multiplyMatrices(modelViewMatrix, scalingMatrix); //  scaling
   modelViewMatrix = multiplyMatrices(modelViewMatrix, rotationXMatrix); // rotation X-axis
   modelViewMatrix = multiplyMatrices(modelViewMatrix, rotationYMatrix); // rotation Y-axis
   modelViewMatrix = multiplyMatrices(modelViewMatrix, rotationZMatrix); // rotation Z-axis
   modelViewMatrix = multiplyMatrices(modelViewMatrix, translationMatrix); // translation

   return getTransposeMatrix(modelViewMatrix);
}
```

The main reason for the differences between Task 1 and Task 2 can be differences in conversion order and sensitivity:

- Conversion order: ChatGPT may have calculated the transformation operations in a different order, leading to different results. Changing the transformation order can lead to different results, especially when multiplying rotation matrices.
- Difference in precision: Differences in small decimal values may be due to differences in precision, especially in sine and cosine calculations.

The rendered version of the cube according to manual calculations is below:



4. Task 3: Animation of the Cube

In this task, I aimed to move the cube in a periodic cycle of 10 seconds using a manually calculated transformation matrix. In each 10-second cycle, the cube will move from the starting position to the target position according to the calculated transformation matrix in the first 5 seconds, and back to the starting position in the last 5 seconds. To realize this animation, the animation was created using the getPeriodicMovement() function.

Animation Details:

- First 5 seconds: The cube moves from the initial position (unit matrix) to the transformation matrix (modelViewMatrix calculated in Task 2).
- Last 5 seconds: The cube moves from the target position back to the starting position.
- Continuous loop: The animation repeats every 10 seconds.

This loop was calculated using linear interpolation (lerp). The lerp method allows for a smooth transition between two values. Throughout the animation, the transitions between both positions of the cube will be smooth and continuous.

Description of Animation:

- getPeriodicMovement() Function: The getPeriodicMovement() function calculates the model-view matrix of the cube according to the elapsed time and determines which position it will be in at any time.
- Time Management: With Date.now() we calculate the elapsed time and find out how long the animation has been running. The periodTime variable determines where we are during the 10-second loop (e.g., are we at 3 seconds or 7 seconds).
- Linear Interpolation (lerp): the t value determines at which stage the animation is. In the first 5 seconds, t increases from 0 to 1 and the cube moves from the starting position to the target position. In the next 5 seconds, t decreases from 1 to 0 and the cube moves from the target position back to the starting position.
- Matrix Multiplication: During each 10-second cycle, a smooth transition between two matrices is performed using the lerp method to allow the cube to transition between two positions.

In this task, an animation was successfully realized in which the cube object moves smoothly according to certain transformation matrices. With the getPeriodicMovement() function, the cube moves from the start position to the target position and back again every 10 seconds. Thus, a continuous animation was achieved. With this step, both the transformation process of the cube was completed and visualized with animation.

Shareable link to the conversation with ChatGPT: Task 3 GPT Chat

Shareable link to GitHub: Ulaş Yıldız GitHub