اونيۏرسيتي مليسيا ڤهڠ

UNIVERSITI MALAYSIA PAHANG

FACULTY OF COMPUTING

SESSION: 2022/2023 SEMESTER 2

BCM2053 COMPUTER GRAPHICS

MINI PROJECT

GROUP NAME:

CONTRACTOR

LECTURER'S NAME:

DR SURYANTI BINTI AWANG

Matric ID	Name	Section
CD22094	ARIFF HAKIMI AZANI BIN NOR AZMAN	01C
CD22097	NOR FAKHIRA IRDINA BINTI NOR FADHLI	01C
CD22100	AMIRAH SYAMIMI BINTI ARIFIN	01C
CD22102	NURATIKAH NURAIN BINTI SAHRIN	01C

QUESTION 1: What is your contribution to this Mini Project assessment?

ARIFF HAKIMI AZANI

I constructed the child of the object, which is the head of the truck, which includes the front window and the left and right windows. Correct the coding for the truck's head and body.

NOR FAKHIRA IRDINA

I developed all four tyres for the lorry. It is the lorry object's parent. I made the tyre out of a cube.

AMIRAH SYAMIMI

I constructed the truck’s body. This body is the child for that object. I constructed this truck’s body using the rectangle.

NURATIKAH NURAIN

I constructed the truck’s back cover. It is the grand child for the truck’s body. All the truck’s back cover constructed by using cubes.

QUESTION 2: Explain how you construct/develop your part of the project?

ARIFF HAKIMI AZANI

I started constructing the cube for the head using mat4.create and mat4.scale ([1, 1, 1.2]) to size it properly, and translated the size and position using mat4.translate ([0, 0, 2]). After that, I created the cube for the front window. I scaled the cube for the front window by using mat4.scale ([0.9, 0.4, 0.4]), and I translated the size and position to fit the front window by using mat4.translate ([0, 1, 1.7]). I created the cube for the left window. I scaled the cube for the front window by using mat4.scale ([0.5,1,2]), and I translated the size and position to fit the front window by using mat4.translate ([1.5,0,-0.9]). I created the cube for the right window. I scaled the cube for the front window by using mat4.scale ([0.5,1,1]), and I translated the size and position to fit the front window by using mat4.translate ([-7,0,0.05]).

NOR FAKHIRA IRDINA

First of all, I tried to create a tyre in JavaScript using a round shape. But it didn't work. So I built all four tyres using a cube shape.

Tyre1

- mat4.scale is [-0.2,-0.4,-0.4]
- mat4.translate is [5.5,2,4].

Tyre2

- mat4.scale is [-1,1,1]
- mat4.translate is [-0.3,0,-10].

Tyre3

- mat4.scale is [-1,1,1]
- mat4.translate is [-11.6,0,0].

Tyre4

- mat4.scale is [-1,1,1]
- mat4.translate is [-0.3,0,10].

AMIRAH SYAMIMI

First and foremost, I made a rectangular shape first. The total faces for that rectangular are 4. Then, I create a child function to rotate the y-axis. So, when click the child button to perform transformation, it will rotate and when click stop, the rotation will stop.

For the mat4.scale and mat4.translate, I use this coordinate:

- mat4.scale[0.95,1.2,2])
- mat4.translate[0,0.15,-0.4])

NURATIKAH NURAIN

First I made the center for the truck's back cover using a small cube for made the back cover rotate nicely. Then, I create a child function for the cube rotate to x-axis and I also create the solid cube without function for the truck's back cover. So, the center will rotate the back cover cube when the grand child button clicked. It will rotate and will stop when the stop button clicked.

Truck's back cover center:

- mat4.scale[0.1,0.1,0.1];
- mat4.translate[0,9,-8];

Truck's back cover:

- mat4.scale[9,9.8,1];
- Mat4.translate[0,-0.89,0];

Pseudocode

ARIFF HAKIMI AZANI

```
BEGIN
        CALL FUNCTION(Main)
        DEFINE coordinate and axis size
        Initialize GL context

        Initialize WebGL context
        Define vertex shader
          // Define vertex attributes
          // Define uniform variables

          void main() {
           // Transform vertex position

        Define fragment shader
          void main() {
            // Set fragment color

        Compile and link shaders
          // Create and bind vertex buffer

        Define Truck geometry
          // Define ferris wheel vertices here
          let modelViewMatrix = mat4.create(); /// make the drawing centred at 0,0,0

                // DRAW Kepala Truck
                modelViewMatrix = mat4.create();
                mat4.scale([1,1,1.2]);
                mat4.translate([0,0,2]);
                mat4.rotateX(modelViewMatrix,modelViewMatrix,transformChildY);
                drawScene;
                gl.drawElements(gl.TRIANGLES, 36, gl.UNSIGNED_SHORT, 0);

                // DRAW Cermin depan
                colorsBuffer = initBuffers(gl,positionsBox,greyColors);
                modelViewMatrix = mat4.create();
                mat4.scale([0.9,0.4,0.4]);
                mat4.translate([0,1,1.7]);
                drawScene;
                gl.drawElements(gl.TRIANGLES, 36, gl.UNSIGNED_SHORT, 0);

                // DRAW Cermin kiri
```

```
colorsBuffer = initBuffers(gl,positionsBox,greyColors);
mat4.scale([0.5,1,2]);
mat4.translate([1.5,0,-0.9]);
drawScene;
gl.drawElements(gl.TRIANGLES, 36, gl.UNSIGNED_SHORT, 0);

// DRAW Cermin kanan
colorsBuffer = initBuffers(gl,positionsBox,greyColors);
mat4.scale([0.5,1,1]);
mat4.translate([-7,0,0.05]);
drawScene;
gl.drawElements(gl.TRIANGLES, 36, gl.UNSIGNED_SHORT, 0);
var rotateX = 0, rotateY = 0, rotateZ = 0;

// Mouse drag to rotateX & rotateY
  var prevx, prevy, canvas;
  var dragging = false;

  function doMouseDown(evt);
  function doMouseDrag(evt);
  function doMouseUp(evt);
  function resetRotation();

Load vertex data into the buffer
  // Define vertex attribute location and enable it
  // Specify how to interpret the vertex buffer data

 Set up matrices for transformation
  // Set up projection matrix
  // Set up view matrix

Main rendering loop
  // Clear the canvas
  // Update and set model matrix for truck
  // Apply transformation operations to the model matrix as needed
  // Request next frame

Start the rendering loop
  // Utility functions

  var transformChildZ = 0, transformGrandZ = 0;
  var rotationSpeed = 0.55, transparencyValue = 1;
  var translateZ = 0, scaleY = 1, rotateObject = 0;
  var revTransformGrandZ = 0;
  var reverseRotationSpeed = -0.513;
```

```
        // PARENT & CAMERA MOVEMENT functions

        var pMoveX = 0, pMoveY = 0, pMoveZ = 0;
        var cMoveX = 0, cMoveY = 1, cMoveZ = 10;

                function changeP1value();
                function changeC1value();
                function changeP2value();
                function changeC2value();
                function changeP3value();
                function changeC3value()

        // TRANSFORMATION functions

        var transformGrandBool = false, transformChildBool = false;
        var timerGrand = 0, timerChild = 0;

        // Transform Grandchild on/off switch
                function transformGrand()
        // Transform Grandchild by rotation on Z axis
                function transformGrandLoop()

        // Transform Child on/off switch
                function transformChild()
        // Transform Child by rotation on Z axis
                var rotateClockwise = false;
                function transformChildLoop()

END
```

NOR FAKHIRA IRDINA

```
BEGIN
        CALL FUNCTION(Main)
        DEFINE coordinate and axis size
        Initialize GL context

        Initialize WebGL context
        Define vertex shader
          // Define vertex attributes
          // Define uniform variables

          void main() {
           // Transform vertex position

        Define fragment shader
          void main() {
            // Set fragment color

        Compile and link shaders
          // Create and bind vertex buffer

        Define Truck geometry
          // Define ferris wheel vertices here
          let modelViewMatrix = mat4.create(); /// make the drawing centred at 0,0,0

        // DRAW TAYAR1 (FAKHIRA)
        mat4.scale(modelViewMatrix,modelViewMatrix,[-0.2,-0.4,-0.4]);
        mat4.translate(modelViewMatrix,modelViewMatrix,[5.5,2,4]);
        drawScene(gl, programInfo, colorsBuffer, modelViewMatrix, projectionMatrix);
        gl.drawElements(gl.TRIANGLES, 36, gl.UNSIGNED_SHORT, 0);

        // DRAW TAYAR2 (FAKHIRA)
        mat4.scale(modelViewMatrix,modelViewMatrix,[-1,1,1]);
        mat4.translate(modelViewMatrix,modelViewMatrix,[-0.3,0,-10]);
        drawScene(gl, programInfo, colorsBuffer, modelViewMatrix, projectionMatrix);
        gl.drawElements(gl.TRIANGLES, 36, gl.UNSIGNED_SHORT, 0);

        // DRAW TAYAR3 (FAKHIRA)
        mat4.scale(modelViewMatrix,modelViewMatrix,[-1,1,1]);
        mat4.translate(modelViewMatrix,modelViewMatrix,[-11.6,0,0]);
        drawScene(gl, programInfo, colorsBuffer, modelViewMatrix, projectionMatrix);
        gl.drawElements(gl.TRIANGLES, 36, gl.UNSIGNED_SHORT, 0);
```

```
// DRAW TAYAR4 (FAKHIRA)
mat4.scale(modelViewMatrix,modelViewMatrix,[-1,1,1]);
mat4.translate(modelViewMatrix,modelViewMatrix,[-0.3,0,10]);
drawScene(gl, programInfo, colorsBuffer, modelViewMatrix, projectionMatrix);
gl.drawElements(gl.TRIANGLES, 36, gl.UNSIGNED_SHORT, 0);


// Mouse drag to rotateX & rotateY
  var prevx, prevy, canvas;
  var dragging = false;

  function doMouseDown(evt);
  function doMouseDrag(evt);
  function doMouseUp(evt);
  function resetRotation();

Load vertex data into the buffer
  // Define vertex attribute location and enable it
  // Specify how to interpret the vertex buffer data

 Set up matrices for transformation
  // Set up projection matrix
  // Set up view matrix

Main rendering loop
  // Clear the canvas
  // Update and set model matrix for truck
  // Apply transformation operations to the model matrix as needed
  // Request next frame

Start the rendering loop
  // Utility functions

  var transformChildZ = 0, transformGrandZ = 0;
  var rotationSpeed = 0.55, transparencyValue = 1;
  var translateZ = 0, scaleY = 1, rotateObject = 0;
  var revTransformGrandZ = 0;
  var reverseRotationSpeed = -0.513;

  // PARENT & CAMERA MOVEMENT functions

  var pMoveX = 0, pMoveY = 0, pMoveZ = 0;
  var cMoveX = 0, cMoveY = 1, cMoveZ = 10;

        function changeP1value();
        function changeC1value();
```

```
            function changeP2value();
            function changeC2value();
            function changeP3value();
            function changeC3value()

        // TRANSFORMATION functions

        var transformGrandBool = false, transformChildBool = false;
        var timerGrand = 0, timerChild = 0;

        // Transform Grandchild on/off switch
            function transformGrand()
        // Transform Grandchild by rotation on Z axis
            function transformGrandLoop()

        // Transform Child on/off switch
            function transformChild()
        // Transform Child by rotation on Z axis
            var rotateClockwise = false;
            function transformChildLoop()
```

END

AMIRAH SYAMIMI

```
BEGIN
        CALL FUNCTION(Main)
        DEFINE coordinate and axis size
        Initialize GL context

        Initialize WebGL context
        Define vertex shader
          // Define vertex attributes
          // Define uniform variables

          void main() {
           // Transform vertex position

        Define fragment shader
          void main() {
            // Set fragment color

        Compile and link shaders
          // Create and bind vertex buffer

        Define Truck geometry
          // Define ferris wheel vertices here
          let modelViewMatrix = mat4.create(); /// make the drawing centred at 0,0,0

       // DRAW badan lori
        mat4.scale(modelViewMatrix,modelViewMatrix,[0.95,1.2,2]);
        mat4.translate(modelViewMatrix,modelViewMatrix,[0,0.15,-0.4]);
        mat4.rotateX(modelViewMatrix,modelViewMatrix,transformChildY);
        drawScene(gl, programInfo, colorsBuffer2, modelViewMatrix, projectionMatrix);
        gl.drawElements(gl.TRIANGLES, 36, gl.UNSIGNED_SHORT, 0);


        // Mouse drag to rotateX & rotateY
          var prevx, prevy, canvas;
          var dragging = false;

          function doMouseDown(evt);
          function doMouseDrag(evt);
          function doMouseUp(evt);
          function resetRotation();

        Load vertex data into the buffer
```

```
  // Define vertex attribute location and enable it
  // Specify how to interpret the vertex buffer data

 Set up matrices for transformation
  // Set up projection matrix
  // Set up view matrix

Main rendering loop
  // Clear the canvas
  // Update and set model matrix for truck
  // Apply transformation operations to the model matrix as needed
  // Request next frame

Start the rendering loop
  // Utility functions

  var transformChildZ = 0, transformGrandZ = 0;
  var rotationSpeed = 0.55, transparencyValue = 1;
  var translateZ = 0, scaleY = 1, rotateObject = 0;
  var revTransformGrandZ = 0;
  var reverseRotationSpeed = -0.513;

  // PARENT & CAMERA MOVEMENT functions

  var pMoveX = 0, pMoveY = 0, pMoveZ = 0;
  var cMoveX = 0, cMoveY = 1, cMoveZ = 10;

        function changeP1value();
        function changeC1value();
        function changeP2value();
        function changeC2value();
        function changeP3value();
        function changeC3value()

  // TRANSFORMATION functions

  var transformGrandBool = false, transformChildBool = false;
  var timerGrand = 0, timerChild = 0;

  // Transform Grandchild on/off switch
        function transformGrand()
  // Transform Grandchild by rotation on Z axis
        function transformGrandLoop()

  // Transform Child on/off switch
        function transformChild()
```

```
        // Transform Child by rotation on Z axis
            var rotateClockwise = false;
            function transformChildLoop()

END
```

NURATIKAH NURAIN BINTI SAHRIN

```
BEGIN
    CALL FUNCTION(Main)
    DEFINE coordinate and axis size
    Initialize GL context

    Initialize WebGL context
    Define vertex shader
      // Define vertex attributes
      // Define uniform variables

      void main() {
       // Transform vertex position

    Define fragment shader
      void main() {
        // Set fragment color

    Compile and link shaders
      // Create and bind vertex buffer

    Define Truck geometry
      // Define ferris wheel vertices here
      let modelViewMatrix = mat4.create(); /// make the drawing centred at 0,0,0

    // replace colorsBuffer with new color (the "redColors")
    colorsBuffer = initBuffers(gl,positionsBox,redColors);

    // Change rotation pivot to back towards -z axis a bit
    mat4.translate(modelViewMatrix,modelViewMatrix,[0,0,-0.5]);
    // z=0.3: Change back pivot
    mat4.translate(modelViewMatrix,modelViewMatrix,[0,0,0.3]);

     // DRAW CENTER TUTUP ATAS
          mat4.scale(modelViewMatrix,modelViewMatrix,[0.1,0.1,0.1]);
```

```
      mat4.translate(modelViewMatrix,modelViewMatrix,[0,9,-8]);
 // Rotate cover (grandchild) on button clicked
      mat4.rotateX(modelViewMatrix,modelViewMatrix,transformGrandZ);
      drawScene(gl, programInfo, colorsBuffer, modelViewMatrix, projectionMatrix);
      gl.drawElements(gl.TRIANGLES, 36, gl.UNSIGNED_SHORT, 0);



// Mouse drag to rotateX & rotateY
  var prevx, prevy, canvas;
  var dragging = false;

  function doMouseDown(evt);
  function doMouseDrag(evt);
  function doMouseUp(evt);
  function resetRotation();

Load vertex data into the buffer
  // Define vertex attribute location and enable it
  // Specify how to interpret the vertex buffer data

 Set up matrices for transformation
  // Set up projection matrix
  // Set up view matrix

Main rendering loop
  // Clear the canvas
  // Update and set model matrix for truck
  // Apply transformation operations to the model matrix as needed
  // Request next frame

Start the rendering loop
  // Utility functions

  var transformChildZ = 0, transformGrandZ = 0;
  var rotationSpeed = 0.55, transparencyValue = 1;
  var translateZ = 0, scaleY = 1, rotateObject = 0;
  var revTransformGrandZ = 0;
  var reverseRotationSpeed = -0.513;

  // PARENT & CAMERA MOVEMENT functions

  var pMoveX = 0, pMoveY = 0, pMoveZ = 0;
  var cMoveX = 0, cMoveY = 1, cMoveZ = 10;

        function changeP1value();
```

```
        function changeC1value();
        function changeP2value();
        function changeC2value();
        function changeP3value();
        function changeC3value()

    // TRANSFORMATION functions

    var transformGrandBool = false, transformChildBool = false;
    var timerGrand = 0, timerChild = 0;

    // Transform Grandchild on/off switch
        function transformGrand()
    // Transform Grandchild by rotation on Z axis
        function transformGrandLoop()

    // Transform Child on/off switch
        function transformChild()
    // Transform Child by rotation on Z axis
        var rotateClockwise = false;
        function transformChildLoop()
```

END