**Three.js 3D Scene**

John Kucera

Prof. Amitava Karmaker

CMSC 405 6380 Computer Graphics (2218)

23 September 2021

**Included files**: Three3DScene.html

**Purpose of program**: To create a unique 3D graphics scene using Three.js that contains 6 different shapes, uses multiple light effects, is at least 640x480 pixels, and includes widgets that can turn on/off certain animation components.

**Descriptions of methods**:

* *createWorld()* function:
  + Sets up background color (BABY BLUE) and camera positioning. Camera is positioned and zoomed out enough to see all 6 shapes and is at a proper angle.
  + Sets up multiple lighting effects. They are mostly visible from the Torus Knot, Dodecahedron, and Rectangular Prism which are all white or gray.
    - **Viewpoint light**: WHITE. Shines in camera direction.
    - **Hemisphere and Ambient lights**: NIGHT BLUE. Shines on all shapes from all directions.
    - **Directional Lights**: MAGENTA, INDIGO, DARKRED. Shines from x, -y, and z directions respectively.
  + Creates 6 Three.js shapes and adds them to Object3D nodes:
    - **Rectangular Prism**: IRON GRAY. Tall in y component, sits in center of scene. Mesh Lambert Material. Rotates counterclockwise. Lighting effects are significantly visible on this shape.
    - **Octahedron**: GREEN. Next to the middle of the rectangular prism in the y component, revolves around it counterclockwise. Rotates in the +x direction.
    - **Tetrahedron**: NORMAL MATERIAL (Full spectrum). Revolves around rectangular prism clockwise in the x component. Rotates in the -y direction.
    - **Torus**: YELLOW. Mesh Phong Material, cyan blue specular, very shiny. Revolves around rectangular prism clockwise in the x component. Rotates in the -y and +x directions.
    - **Torus Knot**: WHITE. Mesh Phong Material, very shiny. Revolves around rectangular prism in the z component. Rotates in the +x direction. Looks Magenta/Indigo/Darkred from the Directional Lights.
    - **Dodecahedron**: WHITE. Mesh Phong Material, very shiny. Revolves around rectangular prism in the z component. Rotates in the +y direction. Looks Magenta/Indigo/Darkred from the Directional Lights.
* *render()* function: Renders scene for each animated frame.
* *updateForFrame()* function: Makes adjustments to objects for each frame, causing animation. All revolution and rotation for each shape is done in this method.
* *doFrame()* function: Calls animation functions if “animating” variable is true.
* *doAnimateRadio()* function: Responds to radio button for animating the whole scene. Makes “animating” variable true if respective radio button is clicked.
* *doKey()* function: Responds to user’s key press for rotating the scene. **User can use Arrow Keys, Page Up, Page Down, and Home keys for scene rotation.**
* *init()* function: Initializes WebGLRenderer, canvas, and HTML elements. Allows checkboxes to be responsive for controlling the animation of all or specific shapes.
* HTML code: Contains a header for the page, instructions on how to rotate the scene, radio buttons to turn all animation on or off, and checkboxes to turn animation on or off for certain shapes. Canvas is 780x700.

**Program running in Visual Studio Code, with HTML Page in Mozilla Firefox**:

*(For Javascript/HTML projects, I prefer using Visual Studio Code over Netbeans)*

Graphical user interface, text

Description automatically generated

Figure : Program running in Visual Studio Code with HTML Page in Mozilla Firefox

**Test Plan & Results**

|  |  |
| --- | --- |
| **Plan** | |
| After pressing Run and the HTML page is visible, first make sure that dimensions are 780x700, the widgets are in proper position, and that shapes are properly displayed in the canvas. Use the proper keys that will allow user to perform rotations: **Arrow keys, Page Up/Page Down, Home**. Test widget functionality and shape animation. Canvas should update for each transformation. | |
| **Test # + Description** | **Result** |
| **(SUCCESS)**  **Test 1: Initial page**. Page title, header, and other HTML text are as intended. With the radio buttons, “Animation Off” is the default setting. The checkboxes are all initially filled in. Canvas is 780x700. All Shapes are drawn, colored, and positioned perfectly. No animation is occurring yet. | Figure : Test 1 |
| **(SUCCESS)**  **Test 2: Rotate in +X and**  **-X direction**. Pressed RIGHT and LEFT ARROW keys and the canvas was updated to show the shapes translated in the +X and -X directions respectively. This figure on the right shows screenshots where pressed RIGHT then LEFT 20 times. | Figure : Test 2 |
| **(SUCCESS)**  **Test 3: Rotate in +Y and**  **-Y direction**. Pressed UP and DOWN ARROW keys and the canvas was updated to show the shapes translated in the +Y and -Y directions respectively. This figure on the right shows screenshots where I pressed UP then DOWN 20 times. | Figure : Test 3 |
| **(SUCCESS)**  **Test 4: Rotate in +Z and**  **-Z direction**. Pressed PGUP and PGDOWN keys and the canvas was updated to show the shapes translated in the +Z and -Z directions respectively. This figure on the right shows screenshots where I pressed PGUP then PGDOWN 20 times. | Figure : Test 4 |
| **(SUCCESS)**  **Test 5: Rotate back to initial position.** Pressed HOME key and the canvas was updated to show the shapes automatically rotated back to the initial position. This figure on the right shows a screenshot of when I pressed HOME. | Figure : Test 5 |
| **(SUCCESS)**  **Test 6: Animation On/ Animation Off Radio buttons.** Pressed “Animation On” Radio button and all shapes started performing animated transformations. When I press “Animation Off”, ALL animation is stopped in its current position. | Figure : Test 6 |
| **(SUCCESS)**  **Test 7: Rectangular Prism & Octahedron Animation Checkbox.** When checkbox is NOT checked, Rectangular Prism and Octahedron animation does not occur. When checkbox IS checked, rectangular prism and octahedron start revolving counterclockwise in the x component at the same speed. Octahedron rotates in +x direction. | Figure 8: Test 7 |
| **(SUCCESS)**  **Test 8: Tetrahedron Animation Checkbox.** When checkbox is NOT checked, Tetrahedron animation does not occur. When checkbox IS checked, Tetrahedron starts revolving clockwise in the x-component around the rectangular prism. Rotates in the -y direction. | Figure 9: Test 8 |
| **(SUCCESS)**  **Test 9: Torus Animation Checkbox.** When checkbox is NOT checked, Torus animation does not occur. When checkbox IS checked, Torus starts revolving clockwise in the x-component around the rectangular prism. Rotates in the -y and +x directions. | Figure 10: Test 9 |
| **(SUCCESS)**  **Test 10: Torus Knot & Dodecahedron Animation Checkbox.** When checkbox is NOT checked, Torus Knot and Dodecahedronanimation does not occur. When checkbox IS checked, Torus Knot and Dodecahedronstart revolving in the z component at the same speed around the rectangular prism. Torus knot rotates in the +x direction. Dodecahedron rotates in the +y direction. Since both shapes are WHITE, they reflect the directional lights MAGENTA, INDIGO, and DARKRED. | Figure 11: Test 10 Part 1    Figure 12: Test 10 Part 2 |