Computer Graphics

Anastasia-Danae Panagiotopoulou Email: s230274@dtu.dk

December 3, 2023

1 Introduction

The objective of this project is building a WebGL page that allows the user to alter the texture of a 3D object through interactions on the Document Object Model (DOM). First, I went through 3D modeling in Blender making a 3D object and painting some textures for it. After setting up the common environment in WebGl, I focused on adjusting the OBJ.js file, a critical parser for .obj files, so that it can handle applying textures. Finally, I made a simple menu for the user to interact with the page and change the textures of the 3D model.

2 3D Modelling

For the first step of the project the goal was to create a nice 3D model that was also easy to texture. I chose a cupcake as a model, envisioning the opportunity to apply different textures like changing flavors and toppings.The model consists of three meshes—a sphere for the filling, a star for the topping, and a circle for the cup. Figure 1 showcases the final cupcake model with a specific texture applied. To offer diverse texture options, I used smart UV mapping and worked with texture painting. Figure 2 displays a texture map for a cupcake featuring chocolate filling, vanilla topping, and a charming pink cup.

3 Parsing a 3D model

Creating a parser module is the first step on handling 3D models using the WebGL framework. At its core, the OBJDoc object manages various properties, including material information (mtls), object details (objects), vertex data (vertices), normal vectors (normals), and texture coordinates (textures). Parsing begins with the file being split into lines, and each line is systematically processed. Any comments on the .obj file are cleverly skipped using the StringParser. When the 'mtllib' command is encountered, it signals the inclusion of Material

Figure 1: A 3D cupcake modeled using Blender.

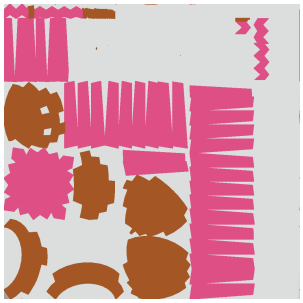

Figure 2: A texture map for my 3D cupcake.

Template Library (MTL) files. The onReadMTLFile function systematically processes MTL files, extracting color information for ambient (Ka), diffuse (Kd), and specular (Ks) or other components that associate with the visual representation of the model. Object names ('o' or 'g' commands) lead to the creation of new objects - in our case we have 3 different ones that count for each mesh of the model. Vertex ('v'), texture ('vt'), and normal ('vn') information is then extracted and stored in their respective arrays.
Enhancing the parser to include texture handling requires an extension of the code that identifies faces ('f' command). Each face, associated with a material name, is connected to texture coordinates ('vt') indices during parsing. That, coupled with loading distinct indices for textures ensures the precise mapping of textures onto the 3D model. All data essential for the representation of the 3D model are gathered in the getDrawningInfo creating a dataset that can be used later during the rendering process.

4 Setting up WebGL and Interactions with DOM

The last thing on the list is the configuration of a WebGL environment and the integration of the Document Object Model (DOM). The HTML file establishes the structure of the webpage. It includes a canvas element for rendering WebGL graphics, dropdowns for flavor and topping selection, a button for user interaction, and images for textures. The JavaScript files include WebGL shader scripts, utility functions, and the main application script. WebGL is initialized with a canvas element, and necessary extensions are checked. Object loading and shader setup are performed, including the definition of the rendering pipeline. User interactions are facilitated through dropdown menus and a button. Event listeners are attached to the flavor and topping dropdowns, enabling dynamic updates based on user selections. A toggle button allows users to start and stop the rotation of the 3D model.

5 References

- **Book:** Angel, E., & Shreiner, D. (2014). *Interactive Computer Graphics: A Top-Down Approach with WebGL, 7th Edition.* Pearson.
- **Web Resource:** WebGL Fundamentals. (n.d.). Retrieved from `https://webglfundamentals.org/webgl/lessons/webgl-fundamentals.html#toc`
- **Book:** Matsuda, K., & Lea, R. (2013). *WebGL Programming Guide: Interactive 3D Graphics Programming with WebGL.* Addison-Wesley.