**Reflection**

While working on this project, choosing a good image to recreate was a difficult choice since we would be using primitive shapes to model the objects instead of creating them in 3D modeling software. I ended up choosing the image I did because it showed a nice variety of objects with none of them being too complicated to create with these primitive shapes. Even though I had a general idea of how I wanted to model these shapes, choosing the best primitive shapes for them was another challenge as well.

The corn in the image was probably the object I spent the most time trying to figure out the best way to somewhat accurately model due to the change in the shape from small to large and back to small. I ended up using three cylinders to create the model for this object. One for the middle section which has a consistent circumference, and two more cylinders for each end. I modified the cylinder class so that a cylinder can be made with each end being a different size. This was useful to customize the cylinder to be able to make a wider variety of shapes including a cone where one end tapers down to a point. This doesn’t perfectly model the corn since it would have a rounded end instead of a flat end, but overall, the end result turned out pretty good. Adding a small sphere to each end could solve this problem but I was not able to figure out the calculations for a sphere in the time we had to work on this project.

The sausage was another object that made me wonder how I was going to replicate it. In the real world a sausage has a little bit of a curve to it, but without the use of 3D modeling software I was unable to find a good way to add this feature into the sausage. For this reason, I ended up making it straight out of a cylinder, with one end slightly smaller than the other to add a little more character to the object rather than just having a perfectly round straight stick. As with the corn, the sausage also has flat ends that could be made more realistic with a sphere on each end to bring it a more natural look.

The burgers were the easiest to implement as they are just wide, flattened cylinders. The plate and the main part of the skillet were also made with cylinders with another modification. This time I modified the cylinder to be drawn with one end circle, two end circles, or none. By leaving off the top circle and making it a little bigger than the bottom one, then flattening it to stick up just a little higher than the bottom circle I was able to create a pretty good effect that accurately models these objects. Then, for the skillet handle I made another long cylinder with one of the axis on the circle smaller than the other to create an oval shaped cylinder. After that I just had to rotate it and move it into place.

While it is possible to see that these objects are 3D even though the screen is 2D, adding movement to the scene allows anyone viewing the scene to be more immersed into the experience. Movement controls added to the scene are similar to those you would find in a PC video game with WSAD as the movement to the front, back, left, and right respectively. Since the camera is just flying through the 3D scene, the ability to move up and down with E and Q was also added. Lastly, mouse control was added so that you can point the camera in any direction without moving as well as the ability to zoom in or out with the mouse scroll wheel. Not related to camera movement, but pressing 1 or 2 will change the draw mode to either solid fill or wireframe so that all of the polygon skeletons can be seen with the press of a button.

While working on this project, several custom methods as well as classes were made and used to keep the code clean and allow for a much more complicated program without having everything in the main function. Some of the methods used in the main file are to initialize the GLFW window process, keep track of key presses, record mouse movement, and resize the window. This code along with an empty main function could easily be used to start another OpenGL project without the need to start from scratch. The custom classes include one for loading shader vertex and fragment source code files to make a shader program, a camera object that handles all of the movement on the screen, and classes to draw a plane and a cylinder complete with texture coordinates and calculated normals for each. All of these classes are designed to be able to be included in any of other project and can be used with no modification.