Rendering

Anastasia-Danae Panagiotopoulou Email: s230274@dtu.dk

December 10, 2023

1 Introduction

Step into an exploration with me as I dive into the world of ray tracing. This project focuses on exploring torus interactions utilizing shading methods and texturing to elevate the visual aesthetics of the torus.

2 A bit of Math

2.1 Torus Geometry

In terms of geometry, torus is a surface of an object formed by revolving a circle in three-dimensional space about an axis that lies in the same plane as the circle. We only need two constants to define the torus those are:

- The radius of the circle R -the distance from the center of the hole inside the torus and the center of its tube.
- The radius of the tube.

The formula of the torus is given by:

$$\left(R - \sqrt{x^2 + z^2}\right)^2 + y^2 = r^2$$

2.2 Ray Tracing

Let's delve into the process of solving the quartic equation arising from the intersection of a ray and a torus. Consider our ray equation:

$$P(t) = O + t \times D \tag{1}$$

This parametric representation defines a straight line in 3D space, where:

- $P(t)$ is a point on the ray at parameter t.
- O represents the origin, the starting point of the ray, denoted as $O = (ox, oy, oz)$.
- t is a scalar parameter indicating the distance along the ray.
- D is the direction vector of the ray, expressed as $D = (dx, dy, dz)$.

Substituting this parametric ray equation into the torus equation, we obtain a quartic equation, which is solved through a systematic process. To simplify, we assume a, the coefficient of the highest degree, is 1 [Ska13]. This coefficient corresponds to $D^T D$. Further simplification involves determining coefficients (p, q, r) and transforming the quartic equation into a cubic one.

To handle scenarios where the coefficient of the third-degree variable is close to zero, indicative of a depressed quartic equation, we adjust coefficients appropriately and set a flag variable to be negative. Subsequently, we calculate the discriminant h and categorize the procedure into two cases based on its sign.

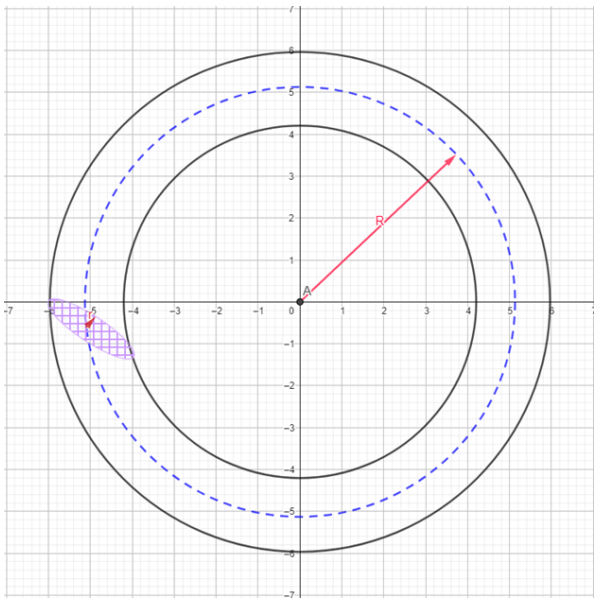

Figure 1: A representation of the torus in 2D.

- If h is non-negative, the cubic equation has four real solutions, two of which are identical. Utilizing Cardano's method, we find the two real roots and employ them to compute the two intersections or roots of the quartic equation.
- If h is negative, the cubic equation possesses three real distinct solutions, leading to the determination of four intersections in total.

In essence, the methodology for finding solutions to the quartic equation involves substitution, transforming it into a cubic equation. The cubic equation is then solved using Cardano's method, with the obtained real roots guiding the determination of the roots of the depressed quartic equation [HE95].

2.3 Finding the normal

Assuming Q to be the point at the center of the torus tube that is closest to the intersection point P, we can see the direction of the normal in 2. Having already calculated the intersection point P and knowing that the distance from Q to the origin of the torus A is equal to the radius of the circle -R, we find Q by converting the projection of point P, P, on the xy plane.

$$Q = R\frac{P'}{|P'|} \tag{2}$$

We process by finding the N vector from Q to P: $N = P - Q$ to be equal to:

$$N = (P_x, P_y, P_z) - \frac{R}{\sqrt{P_x^2 + P_y^2}}(P_x, P_y, 0) \tag{3}$$

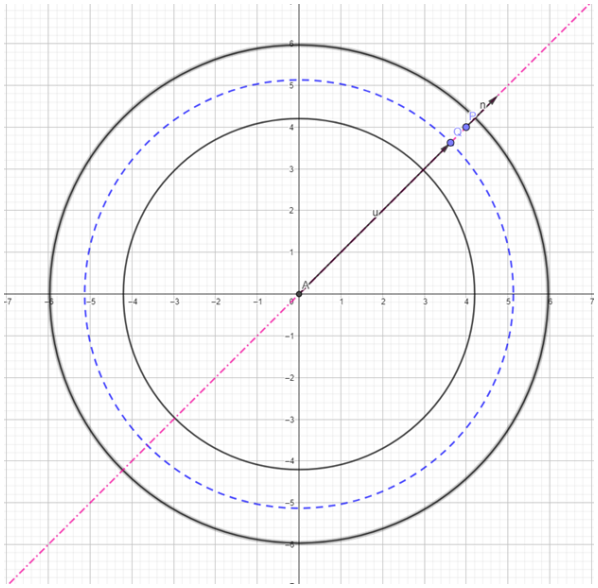

Figure 2: A representation of the normal.

3 Shading

For illuminating the torus we compute the shading for a torus based in a function called *directionallight*. By creating a directional light source, this function orchestrates the shading process by calculating both ambient and diffuse lighting contributions. We choose to add lambertian reflectance for a layer of realism for the torus. The orientation of the light source adapts to the torus's rotation making the lighting effect more authentic. Figure 3 shows the resulted shaded torus without applied texture.

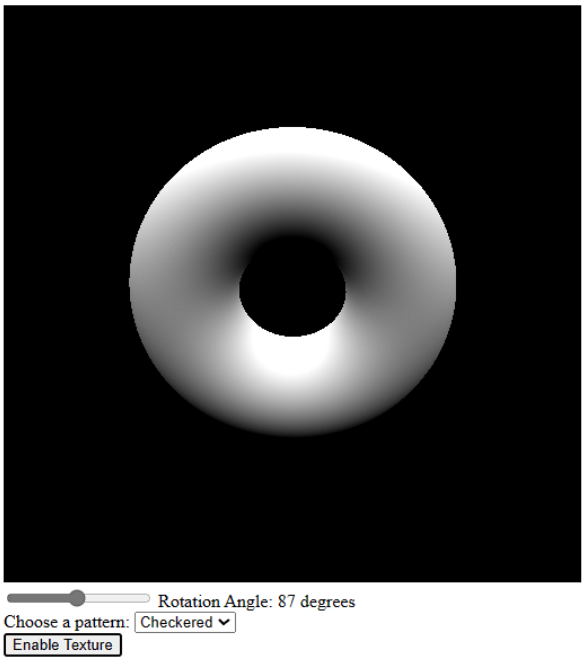

Figure 3: A representation of the shaded torus.

4 Texture Mapping

For making the torus more interesting, we apply texture mapping on its surface using the function *torusTextureCoordinates*. This function ensures an accurate representation by employing the azimuthal and polar angles (phi and theta) calculated from the torus position. The *loadtexture* function facilitates the loading and creation of textures, converting external image files into a format suitable for texturing. The user interface includes a menu that enables the selection of different patterns such as 'checkered,' 'geometric,' and 'waves.' The chosen pattern is then processed, and the corresponding texture is applied to the torus surface in real-time. The texture.sampler configuration further refines the rendering process. The sampler includes settings for addressing modes, such as 'repeat,' and sampling filters, including 'nearest' for both minification and magnification. A toggleable 'Texture' button allows users to seamlessly enable or disable texture rendering, providing an interactive and customizable experience. Figure 4 shows the resulted shaded torus when applying the checkered choice for texturing.

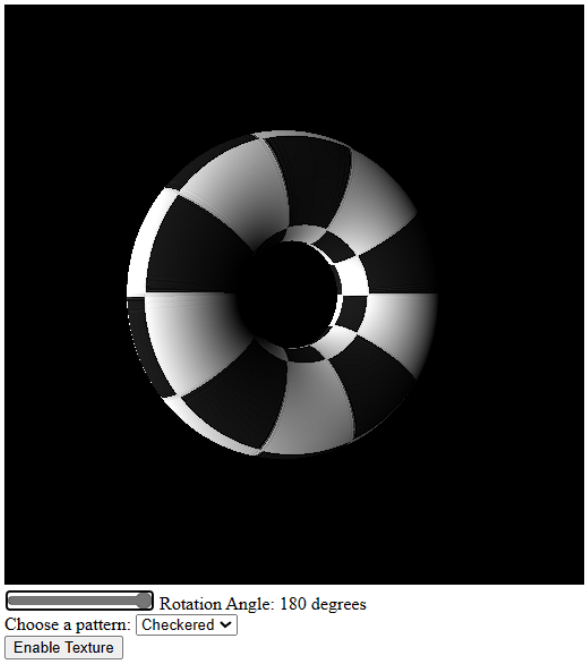

Figure 4: A representation of the shaded torus.

References

[HE95] Don Herbison-Evans. *Solving Quartics and Cubics for Graphics*, pages 3–15. 12 1995.

[Ska13] Vaclav Skala. A new torus bounding for line-torus intersection. In *Proceedings of the 2013 International Conference on Applied Mathematics and Computational Methods in Engineering*, pages 225–230, CZ 306 14 Plzen, Czech Republic, 2013. Department of Computer Science and Engineering, Faculty of Applied Sciences, University of West Bohemia, Rhodos, Greece. AMCME 2013 Conference on Applied Mathematics and Computational Methods in Engineering.