TED UNIVERSITY

Project Report

CMPE 360

Project 5

AUTHORS:

Kerem Güven 63556047646

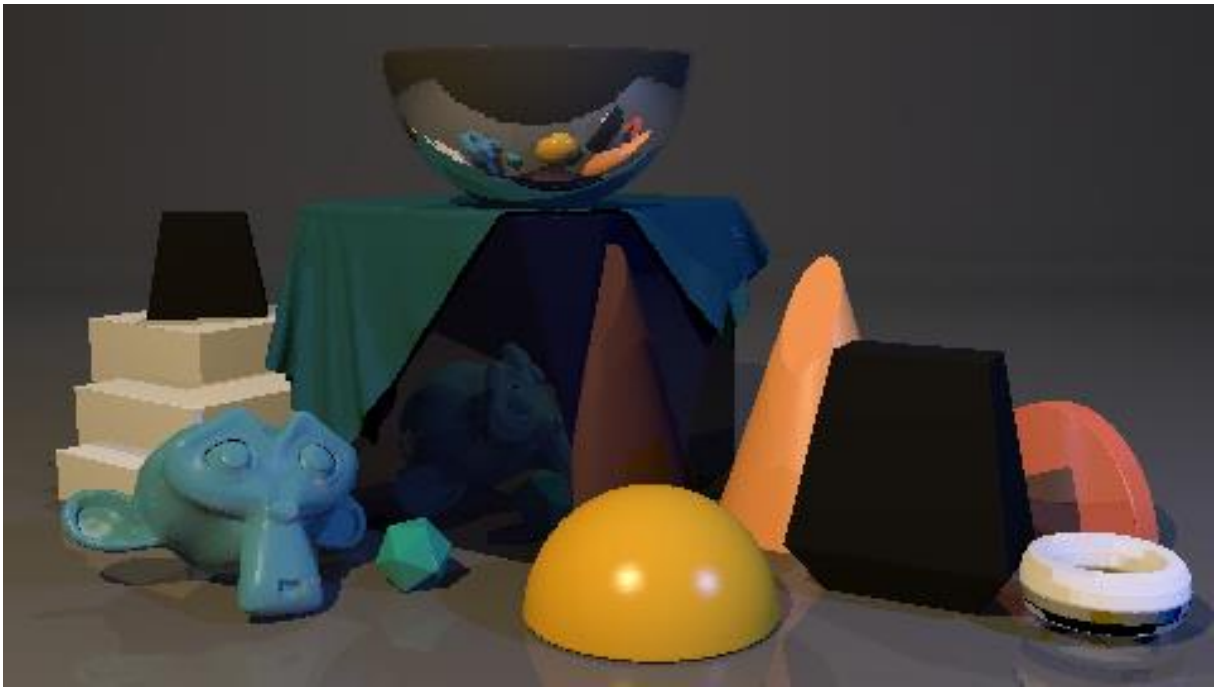
Batuhan İşcan 11863132364

DATE: 11/12/2023

Part 1-

REFLECTION

a) Save the image with reflection and add the screenshot of it on your pdf file



b) Explain your process and effect of reflection

If the surface of the object is reflective as reflectivity is a property of Blender and bpy, a ray that is reflected is cast in pursuit of other objects to return colour values to the camera. This is done by calculating the angle between the ray that makes contact with the surface and the normal vector of that surface to make the reflected ray have it's reflection angle the same with the already explained incident angle.

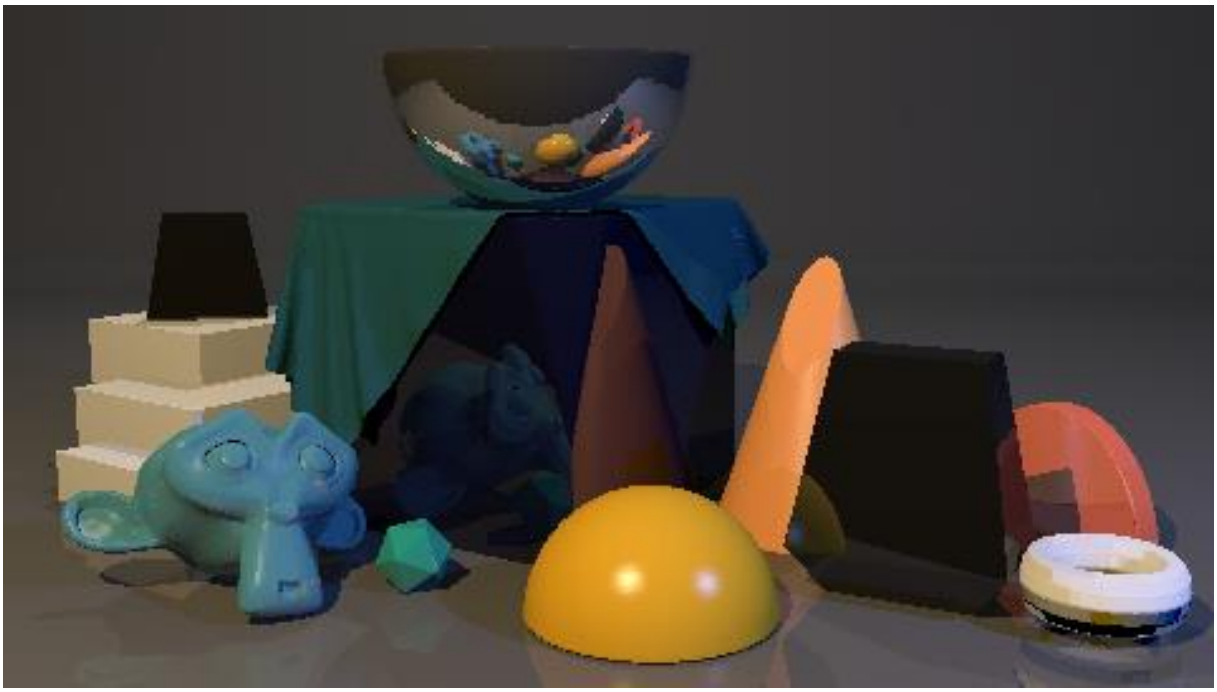
In order to cast the ray, the same Python function that is being used to run this TODO 5 part is used so it is recursively called by putting the condition that the depth element that is configured through the "Output Properties" is decremented and this section of code only works when the depth's value is greater than 0.

To circumvent the phenomenon called self-occlusion, the new ray is cast from a point that is close, yet not the same exact point on the direction of the new ray. Using this method, the ray couldn't collide with the same point it was already on.

Part 2-

FRESNEL

a) Save the image with fresnel and add the screenshot of it on your pdf file



b) B. Explain your process and effect of fresnel

The addition of Fresnel made the reflections on the ground more defined and also created reflections of other objects on objects that are supposed to be “shiny”. This effect was caused by Schlick’s approximation.

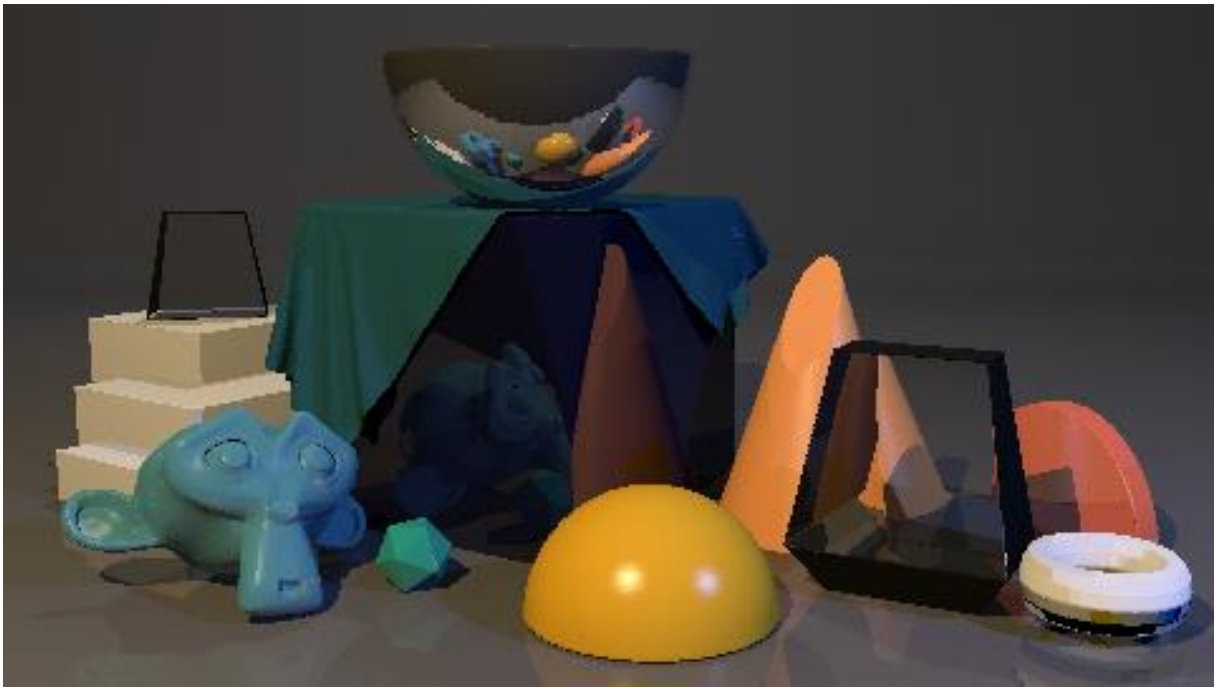
Schlick’s approximation uses the incident angle of the ray to the surface’s normal vector and the materials’ refractive indices to calculate a value for reflection. While the colour is not modified in this section, since the recursive ray tracing used the material’s reflectivity by default, we just make this calculation in case the material has the property “mat.use_fresnel”.

On the matter of how the incidence angle is used, the cosine of the angle between the ray and the surface’s normal vector is subtracted from 1. In this code, the angle is found by the dot product of the two vectors and the cosine is calculated with the numpy library’s function np.cos(Ø). Since the incidence angle is by definition $0 < Ø < 90$ degrees, it must take a value between 0 and 1. The exponent on this value makes it get smaller and in turn, decrease it’s reflectivity. By that standard, angles that make the cosine closer to 0 should make that point more reflective.

Part 3-

TRANSMISSION

a) Save the image with transmission and add the screenshot of it on your pdf file



b) Explain your process and effect of transmission

The transmission part is called from within the recursive ray tracing part and it shares it's condition of having depth greater than 0 to be used. That aside, transmission is the phenomenon of light rays passing through transparent or translucent objects.

For this to occur, the material of the object must allow transmission and so, the material that the ray hits must have "mat.transmission" value greater than 0. These are all properties that bpy makes possible to use in scripting in Blender.

According to the Snell's Law, light will get refracted while passing through objects, according to their refractive index, a value found by the relation between the light's speed inside different objects. Since the air's accepted refractive index is 1, according to whether the ray is inside an object or not, it will use the multiplicative inverse of the proportions of the indices, the denominator being the refractive index of the object that the ray made contact with.

After calculating the direction of this transmitted ray, the self-occlusion is counter-acted by changing the ray's origin by a bit and the transmitted ray is recursively cast. Since the ray is casted

from within the RT_trace_ray() function, rather than casting the ray ourselves, we use this function. This function will then check if the ray hit a reflective or transmissive material and will decrement the depth each time a new ray must be cast. Finally it will return the colours of each point that the ray hit.

Using the returned colour values and multiplying it with the material's transmission property, it is multiplied again with 1 substracted with the objects reflectivity that was calculated in Part 2. With the insight that if the material is reflective, what is not being reflected from a ray must have been transmitted, this mathematical operation's result is added to the colour of the pixel.