Radiosity Rendering using WebGL

Utkarsh Agarwal
IMT2018082
International Institute of Information Technology, Bangalore
utkarsh.agarwal@iiitb.ac.in

I. Introduction

The goal of the project is to understand radiosity and how it works , and implement the same.

II. What is Radiosity ?

Radiosity is a rendering technique which considers light as energy and simulates the distribution of energy in a scene, this method takes all the surfaces as lambrent surfaces and can produce results which are view independent.

This means that once that scene is rendered and stored , even if we move the camera the amount of computation needed will not be much, but this also means it will only work on static scenes.

III. Radiosity Equation

The basic form of the radiosity equation is :-

$$B_i = E_i + p_i \Sigma B_j F_{ij} \tag{1}$$

In Equation 1, B_i is the Radiosity of patch i, E_i is the emission of patch i, p_i is the reflectivity of patch i, B_j is the Radiosity of patch j, and F_{ij} is the form factor between patches i and j. This equation is computed for each patch i.

So the general work flow for computing radiosity is

1) Compute the scene into patches
2) Compute the form factor
3) Solve the equation
4) Display the scene

Now there are two ways of doing this

1) We compute the form factor and solve the equation
2) Or gather all the light through several passes, and update it after every pass

In this implementation we will be seeing the second method, of gathering and updating the light after every pass.

IV. Implementation

Logic : Assuming each patch as light source which emits and absorbs light, we iterate over each patch and render image which it sees and then assign it a radiosity color according to that image.

Code : We store an array of colors called the radiosity colors which helps us keep track of the iterations, after each iteration we update the color buffer and iterate over again.

This goes hand in hand with the color extractor, we use the output of the intermediate rendering to get the color , this rendering is done by taking a specific vertex as the camera position and generating the image.

V. Conclusion

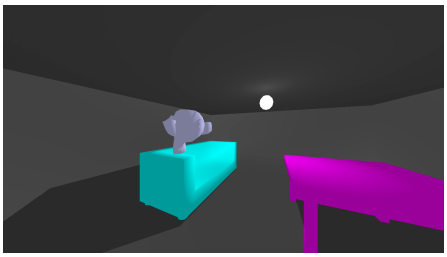

Figure 1. Rendered output

Over here we can see that on the monkey head the color of the table is tinted, and also we can see the light is bleeding on the roof, and soft shadows on the monkey head.

VI. Bibliography

1) Webgl documentation
2) Indigo CS
3) Stack Overflow