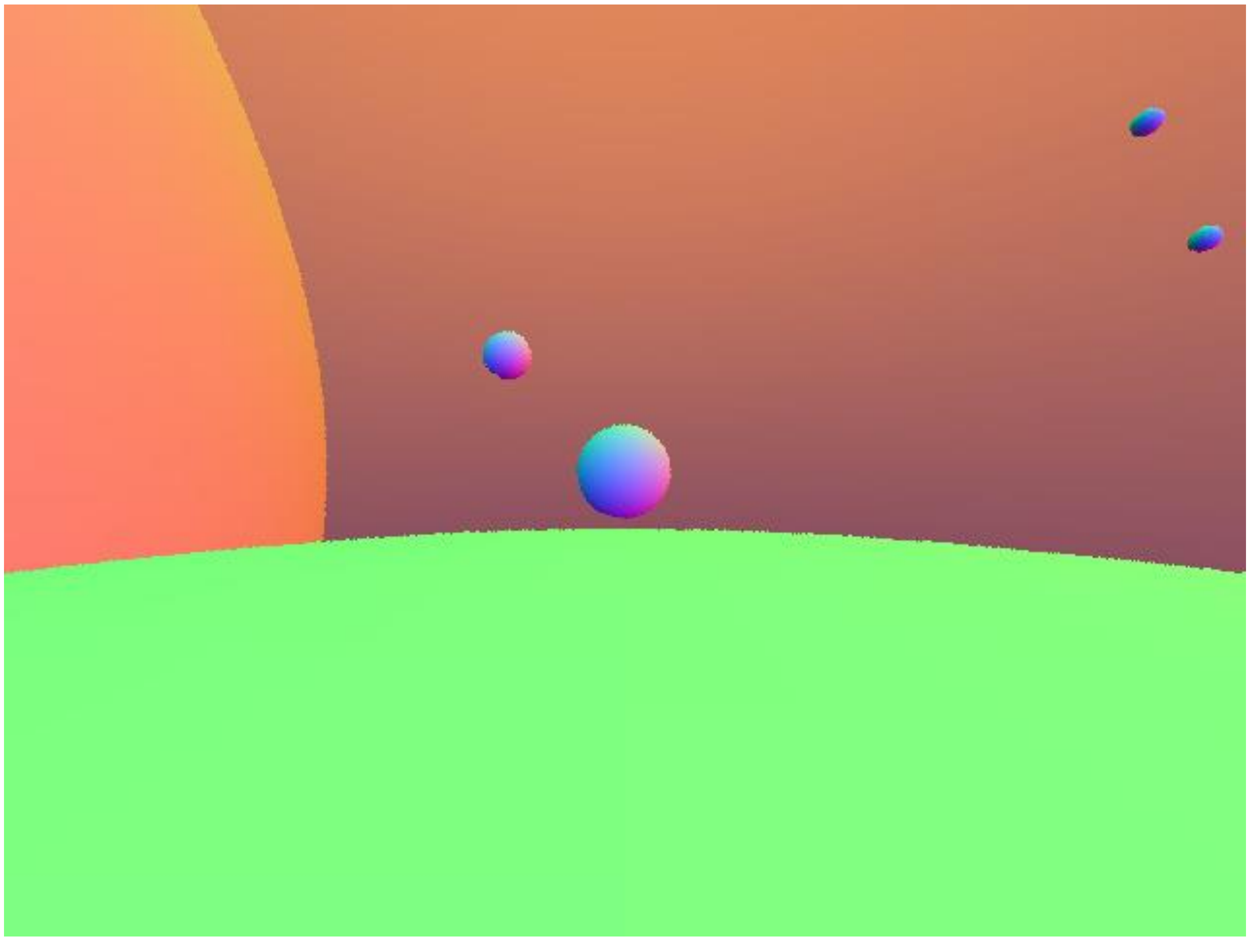
Cem Kaya 26440

Task 1 [Required]: Basic Scene (15Pt)

There are 6 spheres in these scenes located at :

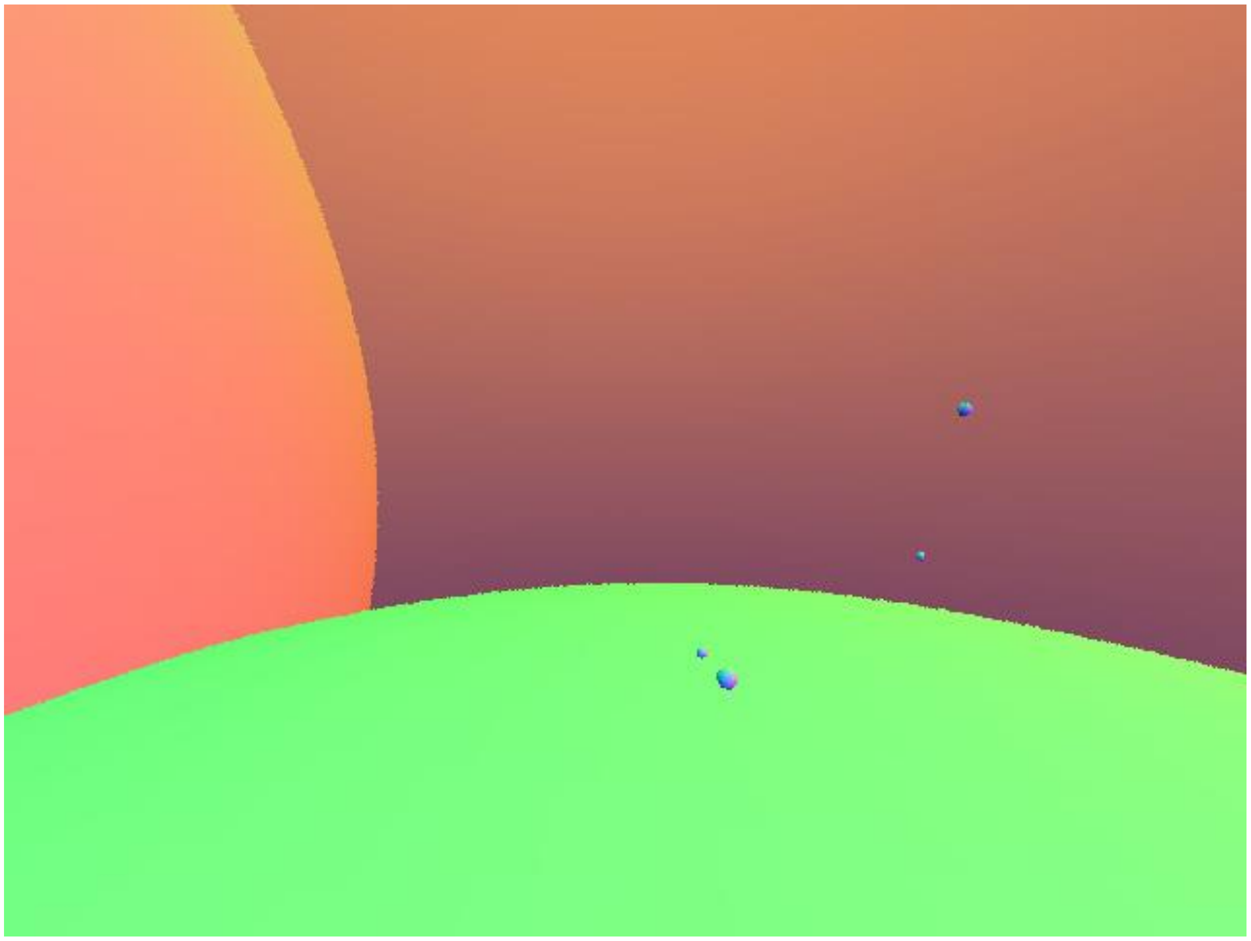
1. X: 0 Y: 0 Z: -1 with radius: 0.2
2. X: -0.5 Y:0.5 Z: -1 with radius :0.1
3. X:0 Y: -101 Z: -1 with radius:100
4. X:120 Y:0 Z: -1with radius: 100
5. X:5 Y:2 Z: -3 with radius: 0.1
6. X:9 Y:6 Z: -7 with radius :0.2

The first camera is in X:0 Y:0 Z: 1 and looking at X:0 Y:0 Z: -1 with VUP: X:0 Y:1 Z: 0
The second camera is in X: -2 Y: 4 Z: 8 and looking at X:0 Y:0 Z: -1 with VUP: X:0 Y:1 Z: 0
Outcome
1:



Cem Kaya 26440

2:



Task 2 [Optional]: Anti-Aliasing (10Pt)

I have tried random SS and grid SS for anti-aliasing and for low number of samples found the random SS better for the image quality sometimes. However, it was a bit slower from grid for N =2 or samples = 4. Beyond that as the sample number increases, they perform much closer, and the image is almost same. The difference between them is negligible. Both seem to increase the render time linearly.

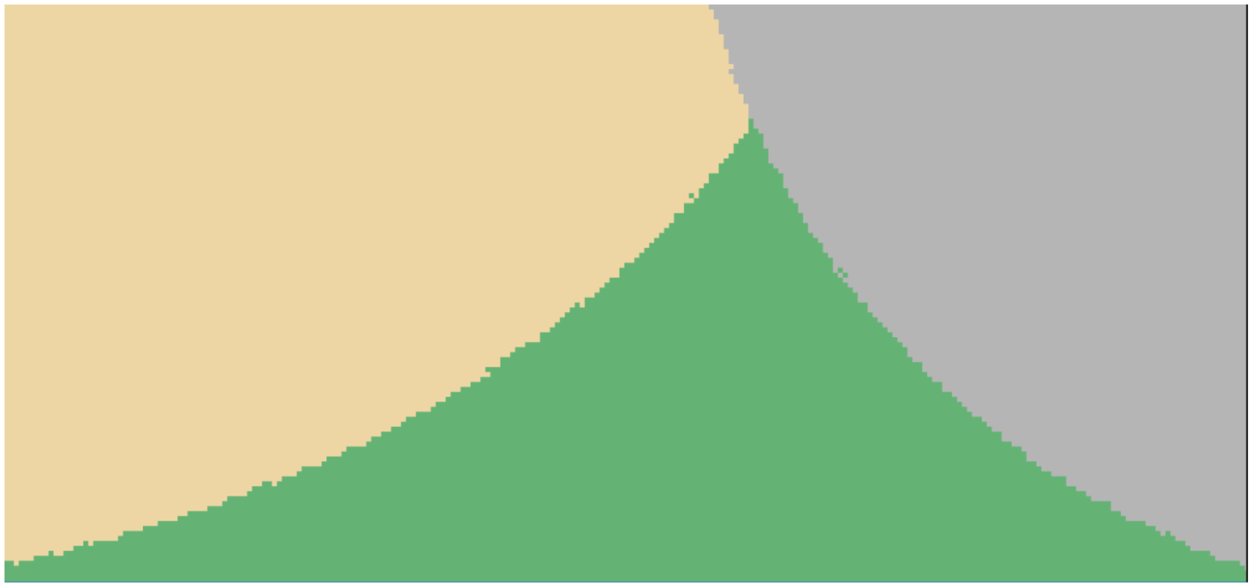
I am very interested in AA which are a type of post processing for especially DLSS1.0. if I had enough time I would like to try implementing such an AA with ML . However due to time constrains I could not.

The pictures below contains light emitting materials to avoid shadow acne and other artifacts due to low light.

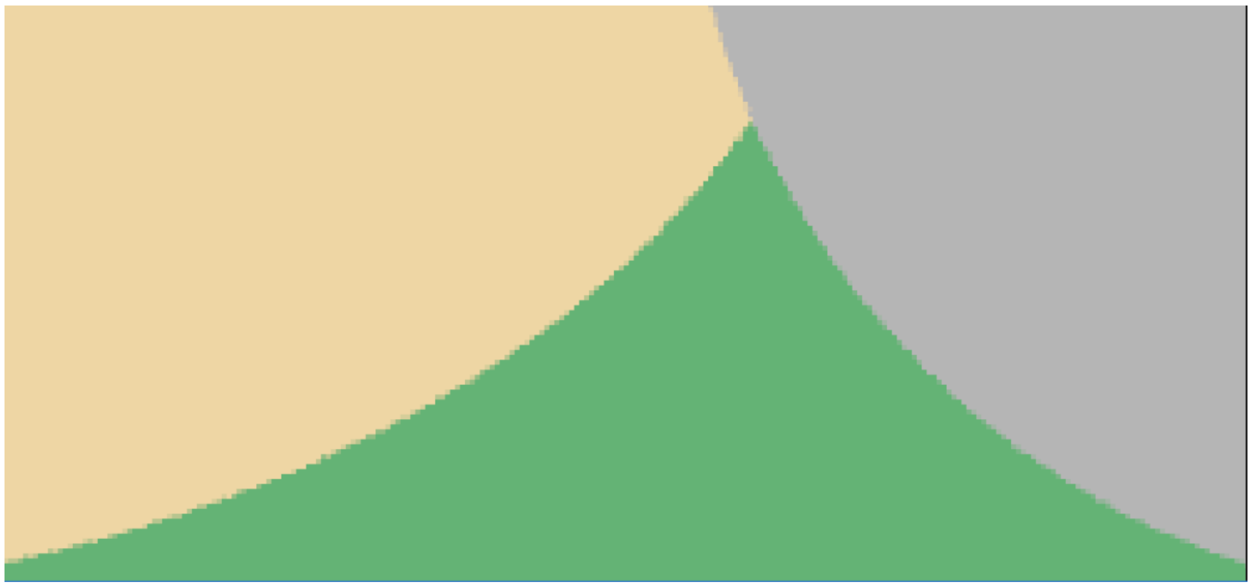
Outcome

Cem Kaya 26440

No AA:

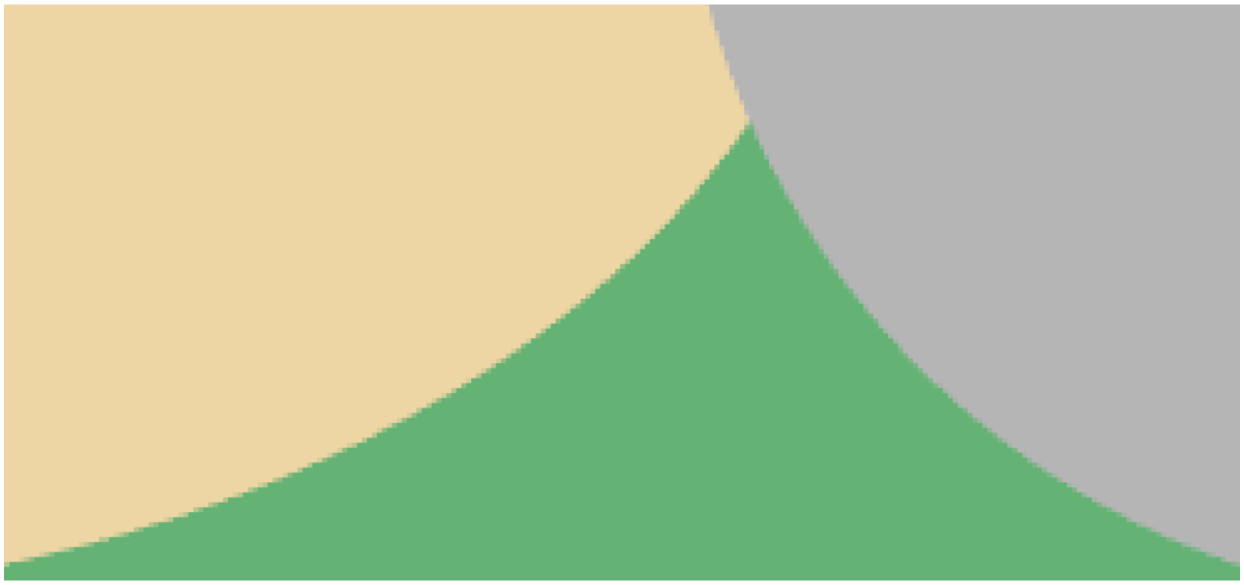


Random with 4 samples:

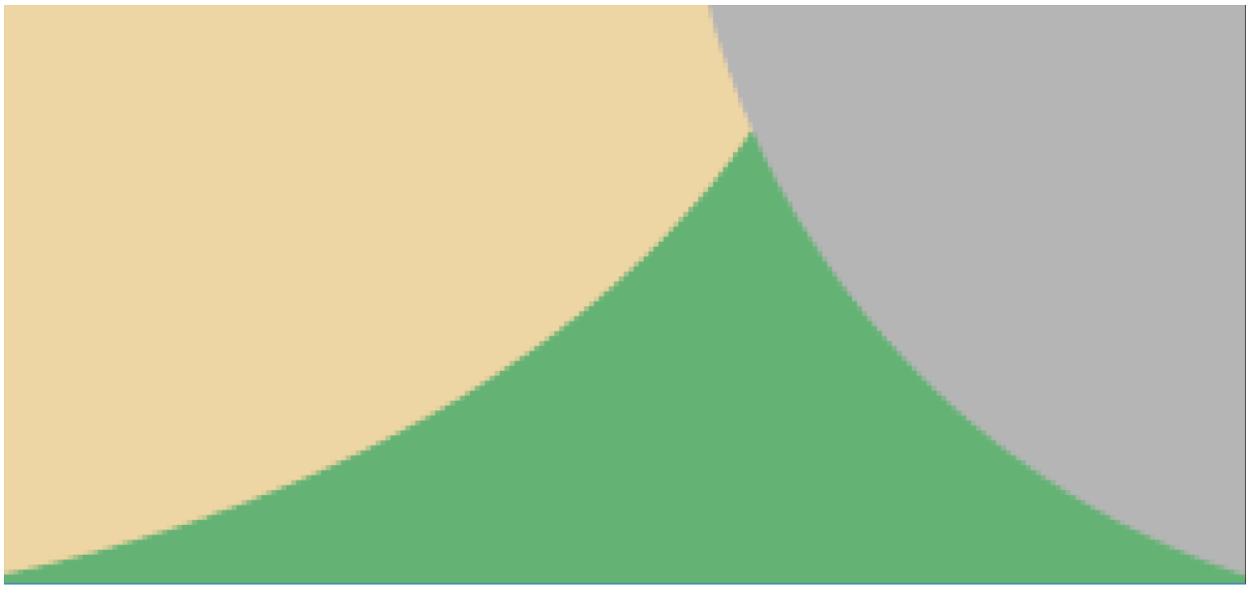


Cem Kaya 26440

Grid with 4 samples (n=2):



Random with 100 samples:



Cem Kaya 26440

Grid with 100 samples (n=10):

Task 3 [Optional]: More Shapes (5Pt)

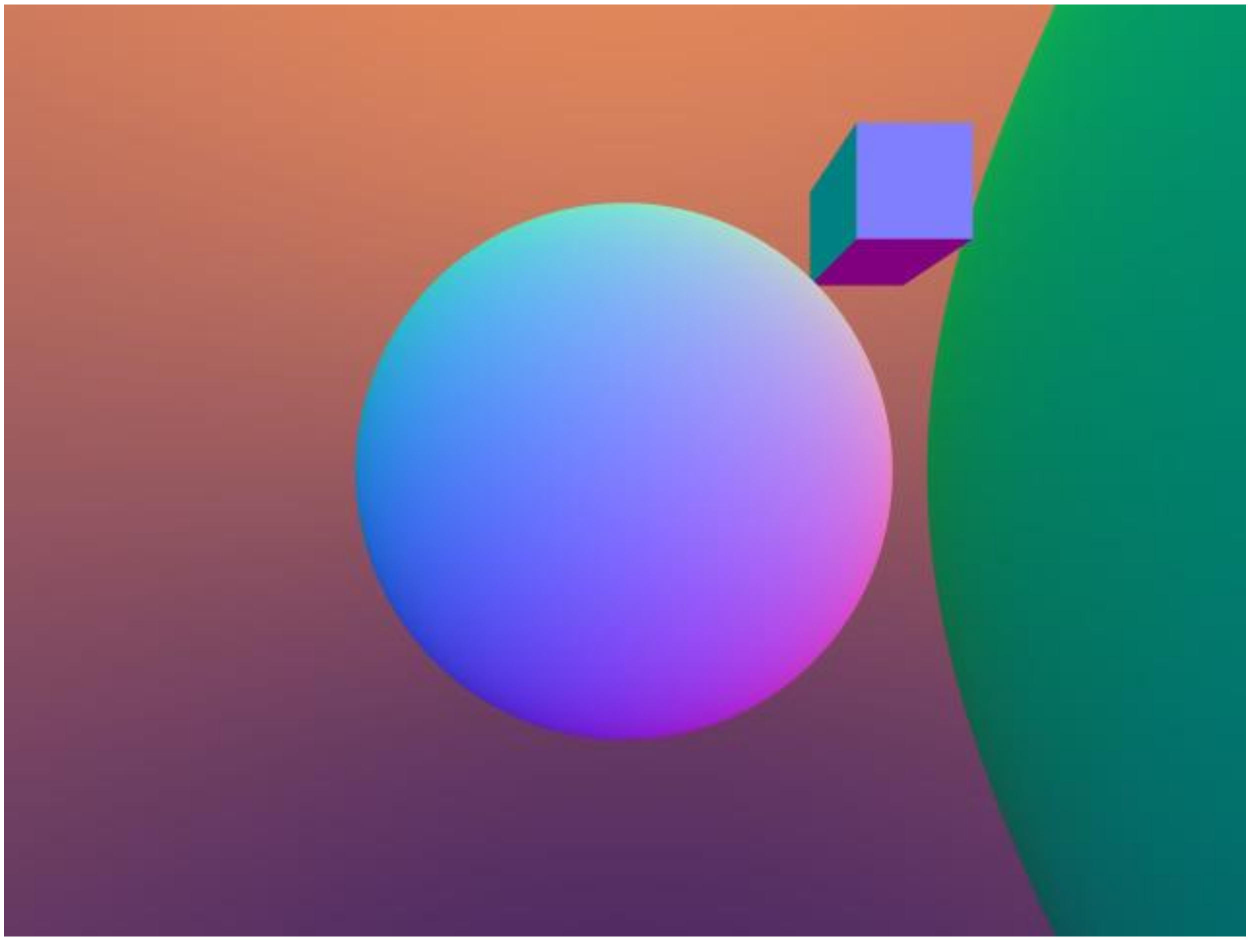
Instructions

I have implemented a cube and a triangle. Then I have implemented a mesh parser to convert any .obj to an array of triangles and render them. Then I have implemented a cube using the mash parser. This will be used in part7 for more interesting shapes. The cube was implemented by using AABB's from Ray tracing the next week for the base and then I have added normal vector by hand by calculating and deriving the directions. The triangle normal calculation is from Scratch a Pixel.

Outcome

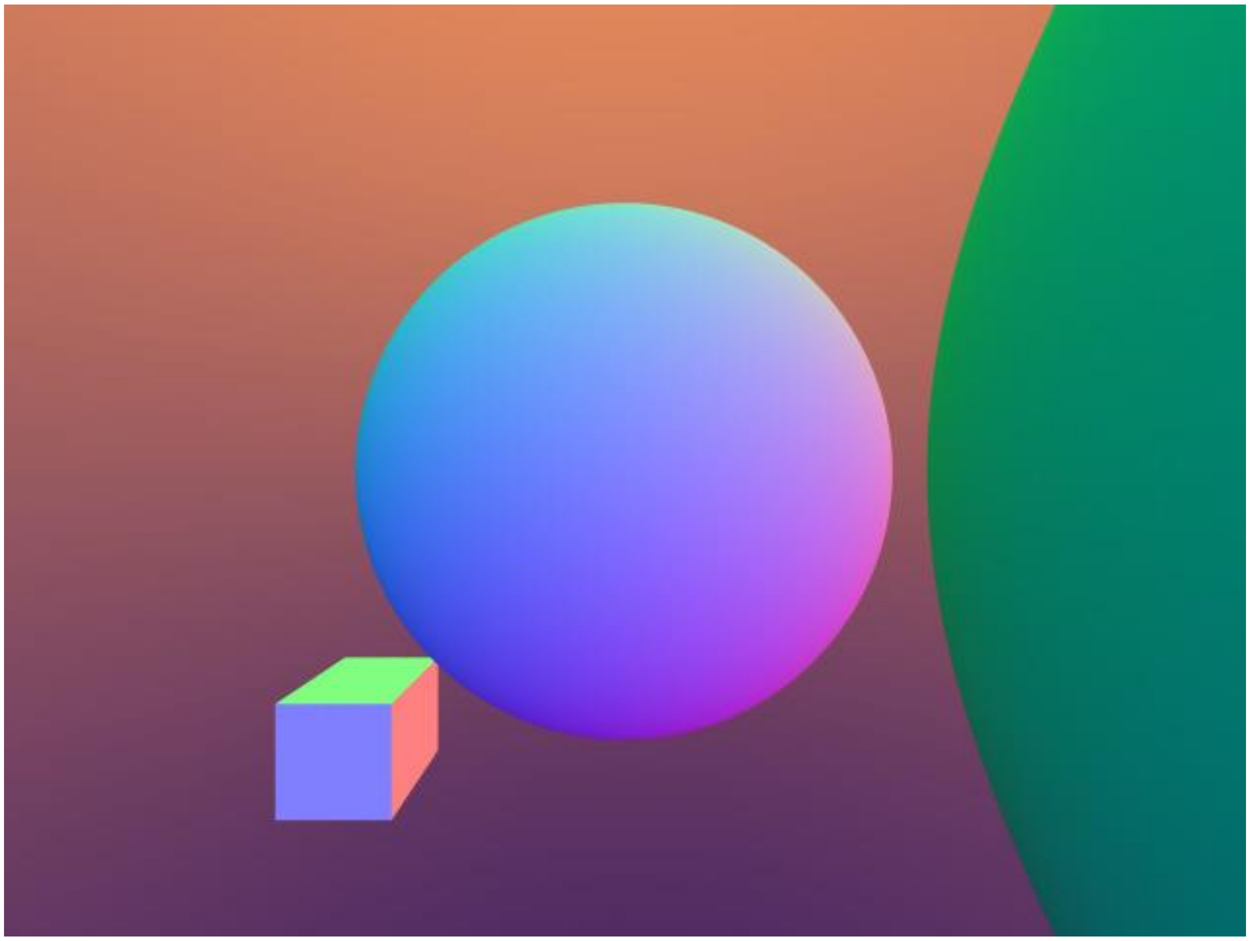
Cem Kaya 26440

CUBE normal1:



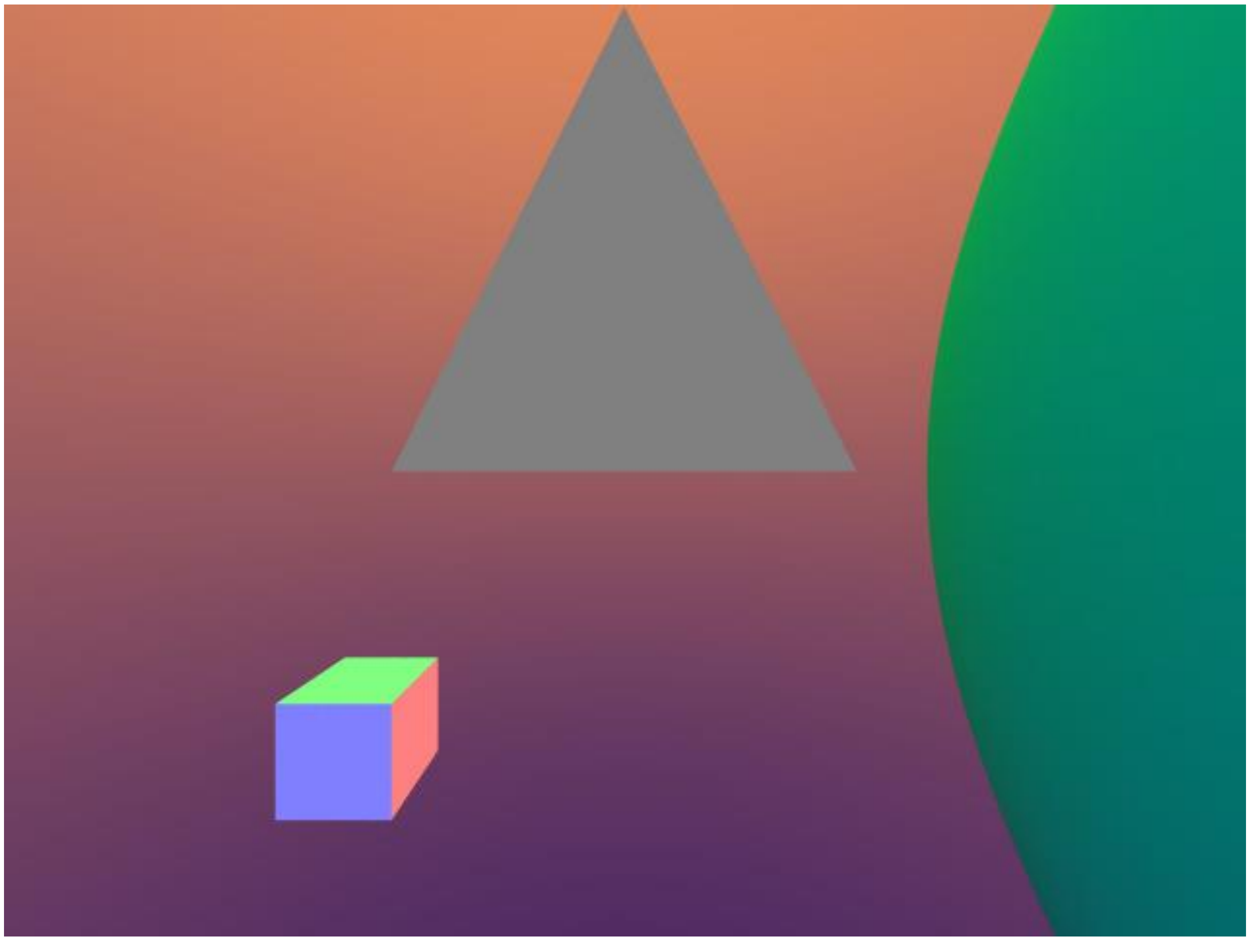
Cem Kaya 26440

CUBE normal1:



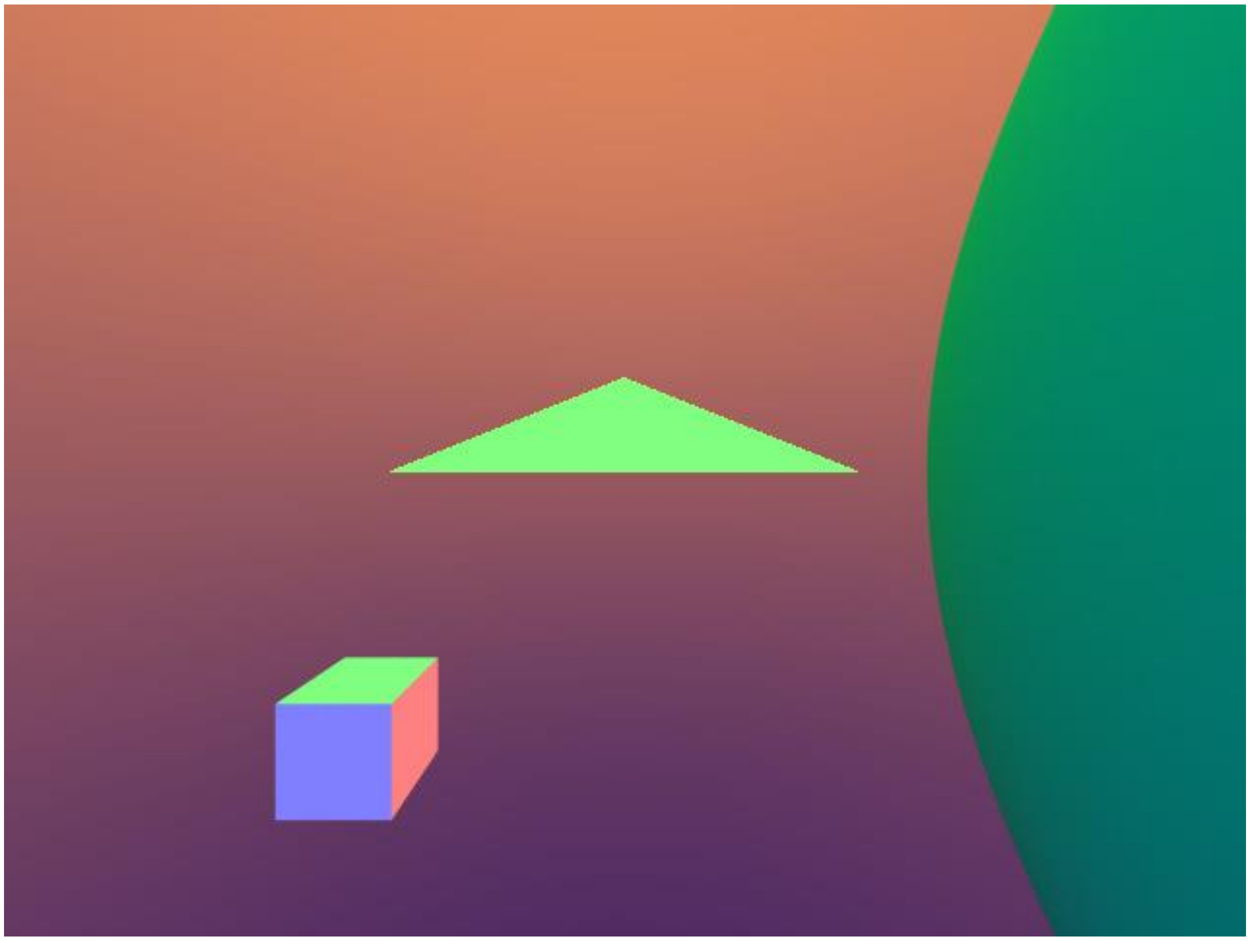
Cem Kaya 26440

Triangle :
Trig(point3(0.5, 0, -1), point3(-0.5, 0, -1), point3(0, 1, -1))



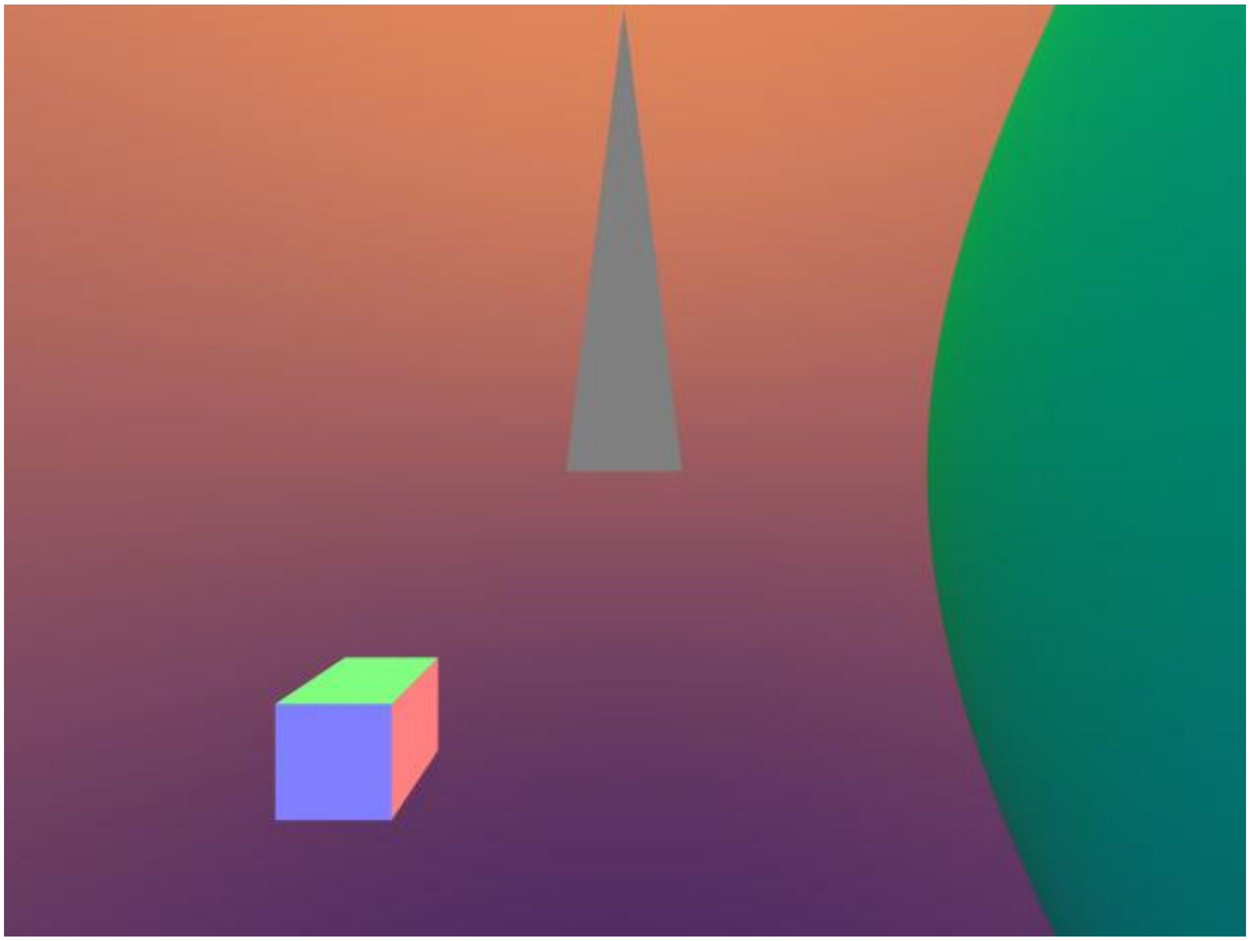
Cem Kaya 26440

Trig(point3(0.5, 0, -1), point3(-0.5, 0, -1), point3(0, 1, -5))



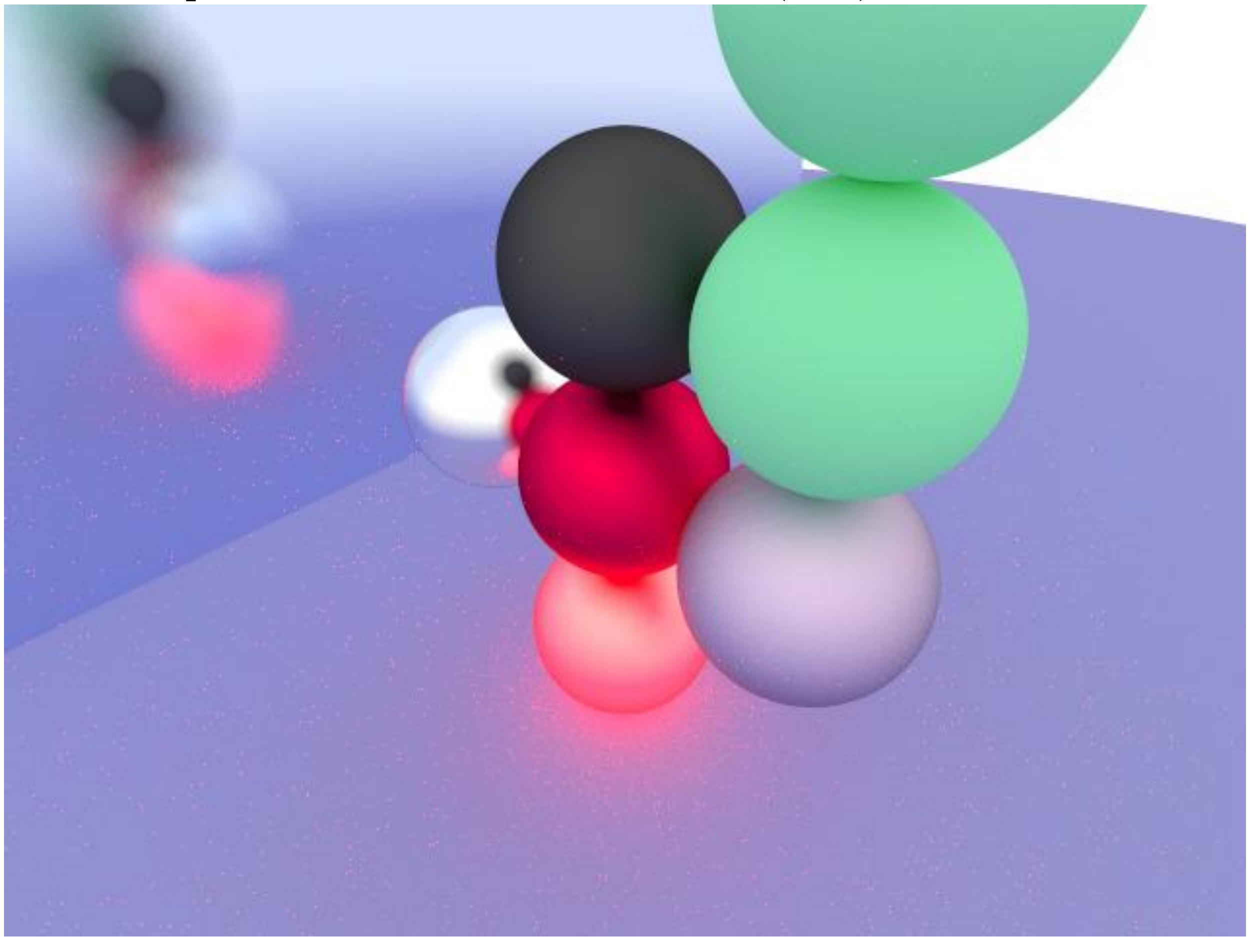
Cem Kaya 26440

Trig(point3(0.5, 0, -4), point3(-0.5, 0, -4), point3(0, 1, -1))



Cem Kaya 26440

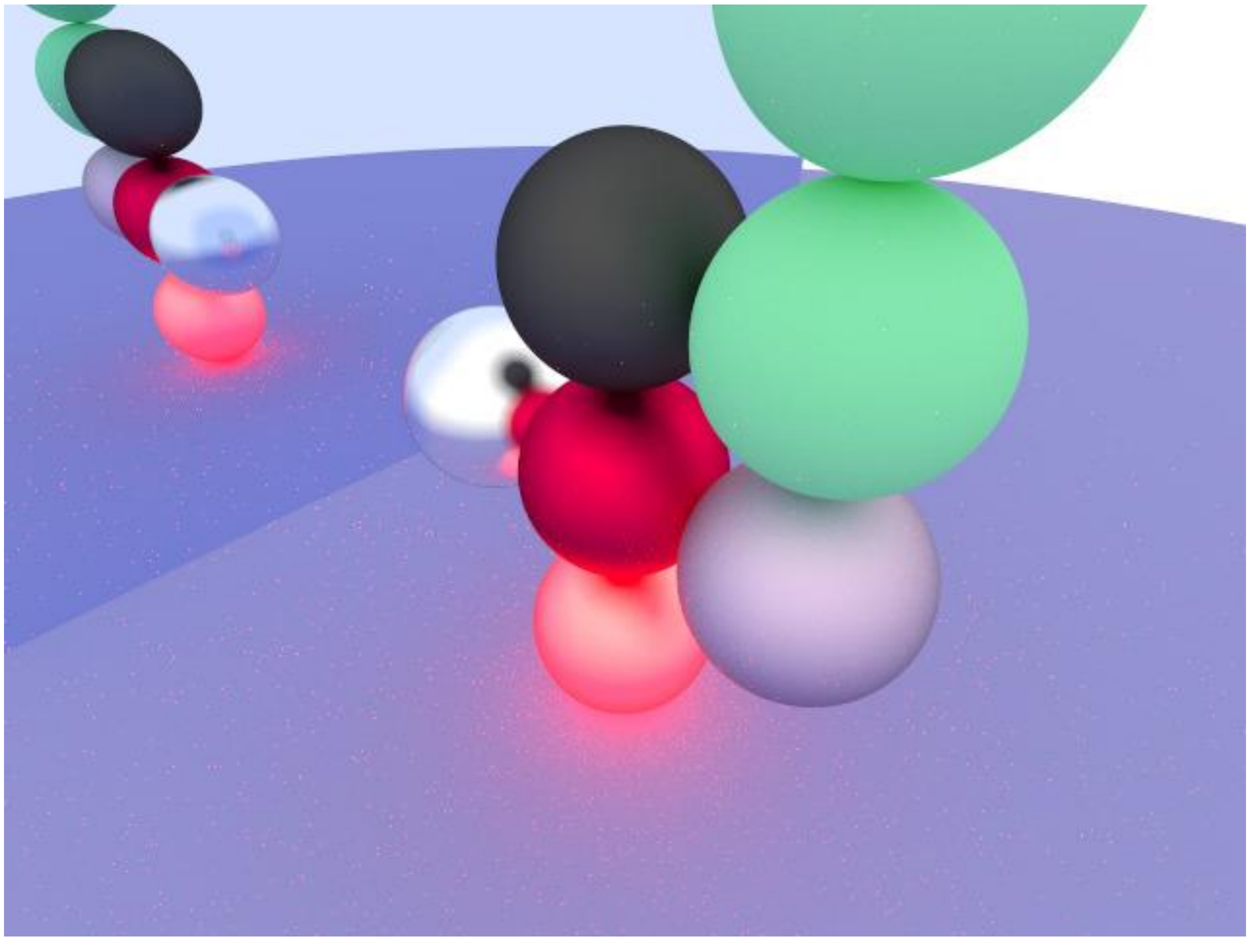
Task 4 [Required]: Diffuse and Metal Materials (25Pt)



This picture shows some metal some diffused sphere in front of a blurry

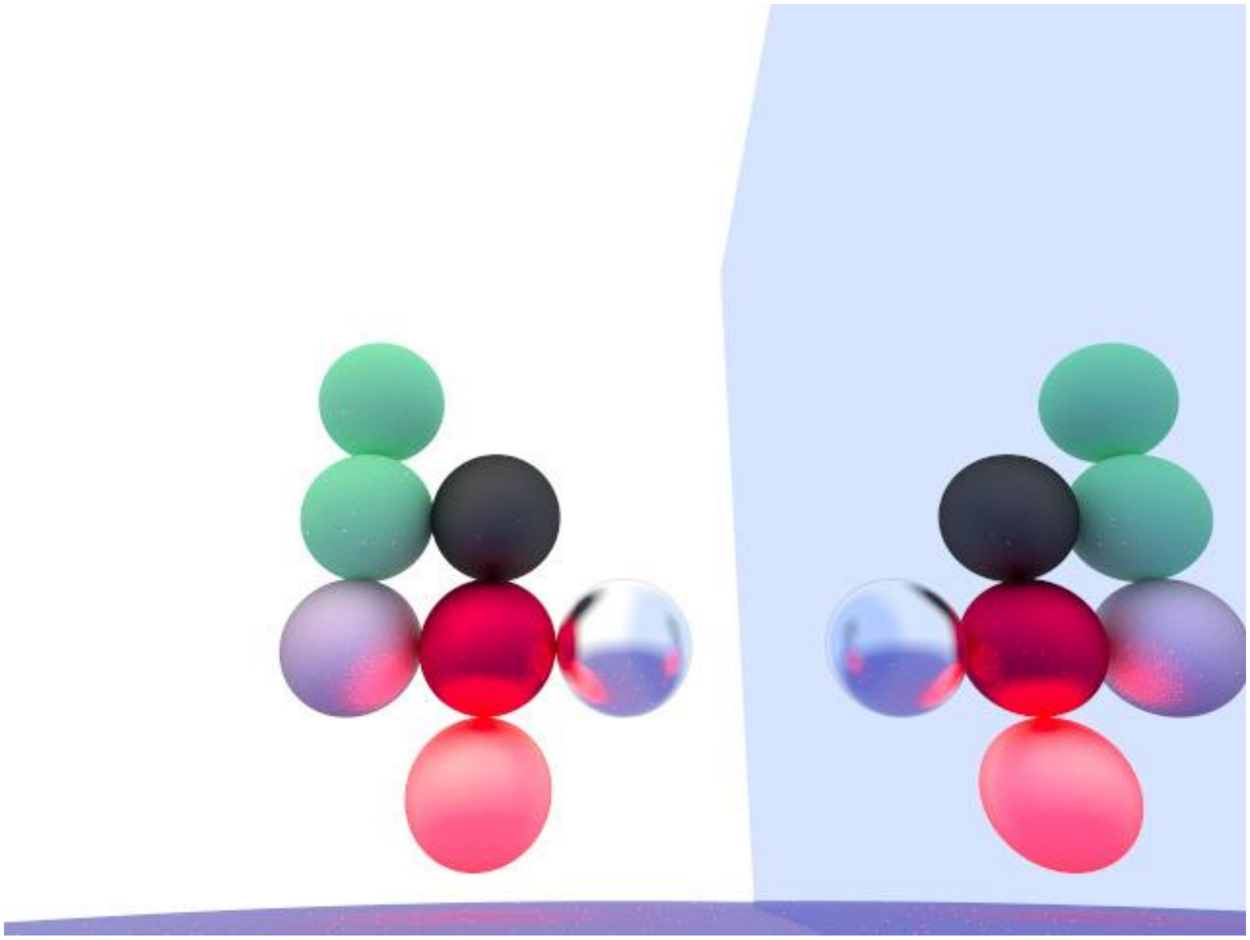
Cem Kaya 26440

mirror



Cem Kaya 26440

This picture shows some metal or diffused sphere in front of a mirror



```
auto m1 = new metal(color(0.1, 0.1, 0.1),0.1);

auto m2 = new metal(color(1, 1, 1),0.25);

auto m3 = new metal(color(05, 0.5, 0.5),0.55);

auto m4 = new metal(color(1, 0, 0.1),0.7);

auto m5 = new metal(color(0.1, 0.1, 0.1),0.8);

auto m6 = new metal(color(0.9, 0.7, 0.9),1);

auto m8 = new lambertian(color(0.3, 0.9, 0.5));

auto m7 = new lambertian(color(0.3, 0.9, 0.5));

auto m9 = new lambertian(color(0.7, 0.5, 0.5));

auto m10 = new lambertian(color(0.35, 0.35, 0.7));

auto miror= new metal(color(0.7, 0.8, 1), 0.0);
```

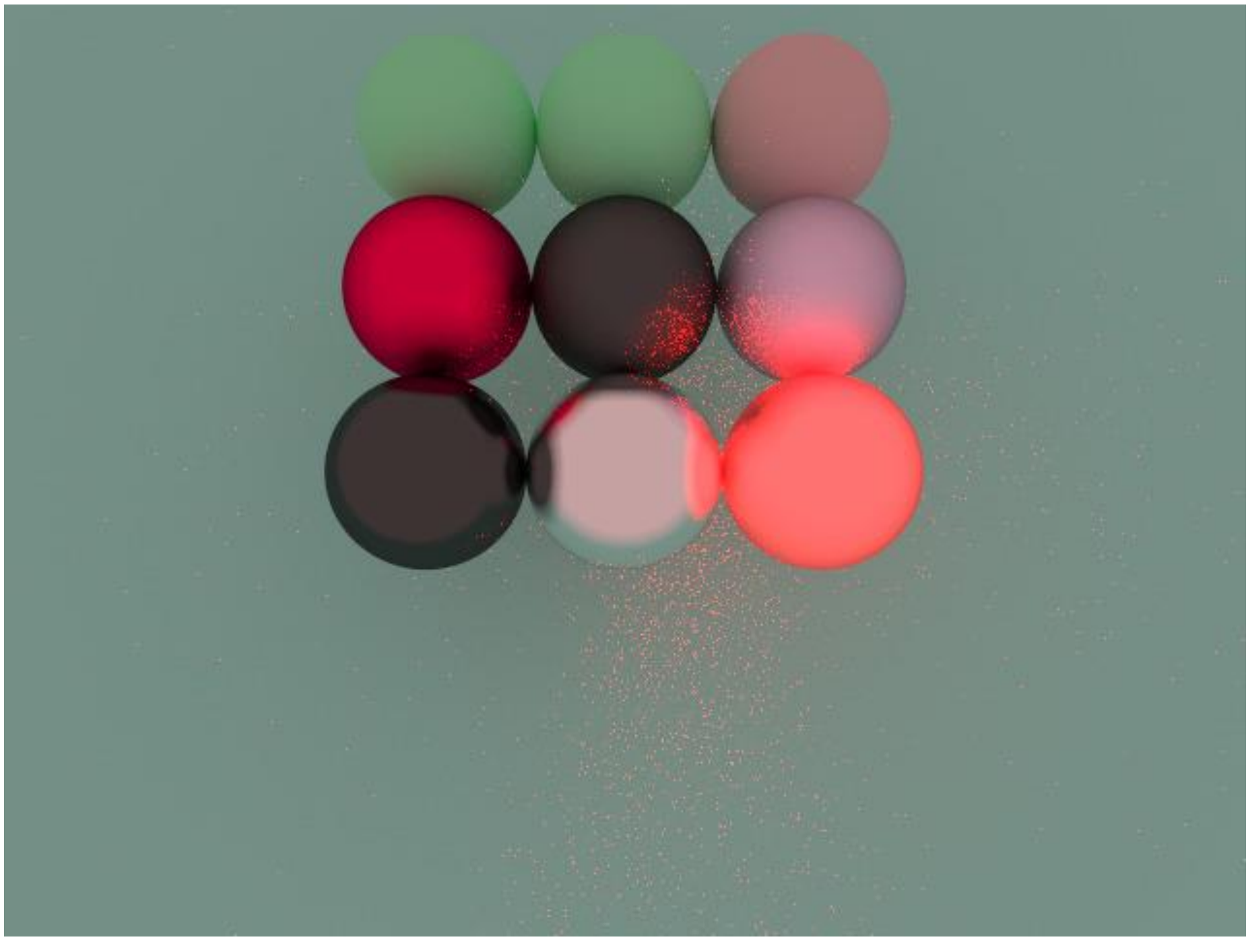
Cem Kaya 26440

```
world.add(new sphere(point3(0.0, -100.5, -1.0), 100.0, m10));



world.add(new cube(point3(-2, -6, -5), point3(-1, 5, 7), miror));
world.add(new sphere(point3(0, 1, 6), 0.5, m1));
world.add(new sphere(point3(0, 1, 6), 0.5, m2));
world.add(new sphere(point3(1, 0, 6), 0.5, m3));
world.add(new sphere(point3(1, 1, 6), 0.5, m4));
world.add(new sphere(point3(1, 2, 6), 0.5, m5));
world.add(new sphere(point3(2, 1, 6), 0.5, m6));
world.add(new sphere(point3(2, 2, 6), 0.5, m7));
world.add(new sphere(point3(2, 3, 6), 0.5, m8));
```

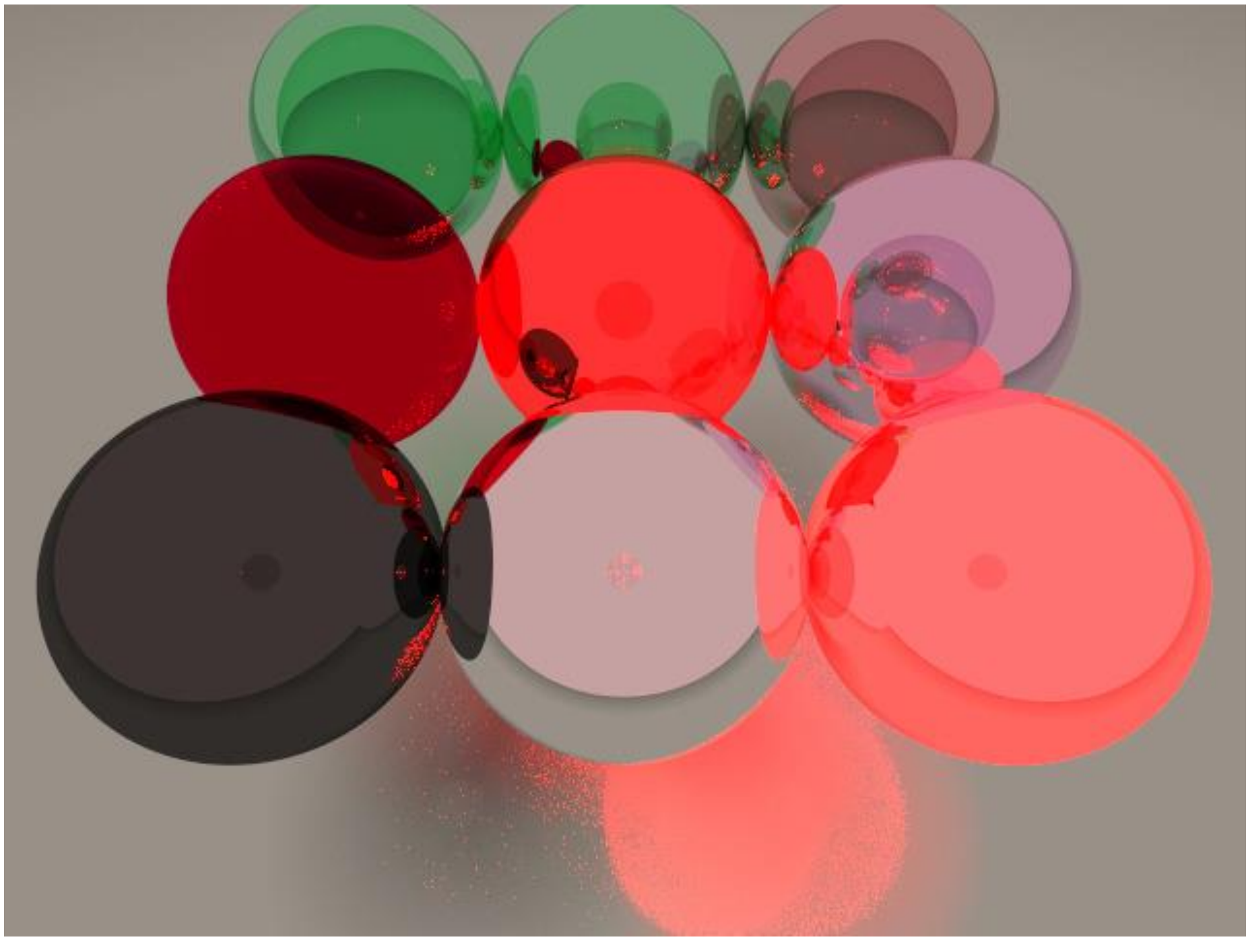
Cem Kaya 26440



Task 5 [Optional]: Refraction (10Pt)

I have added tint able dielectrics. Using this material coloured glass objects can be made. They refract or reflect depending on the angle and with the tint they absorb depending on the lights colour.

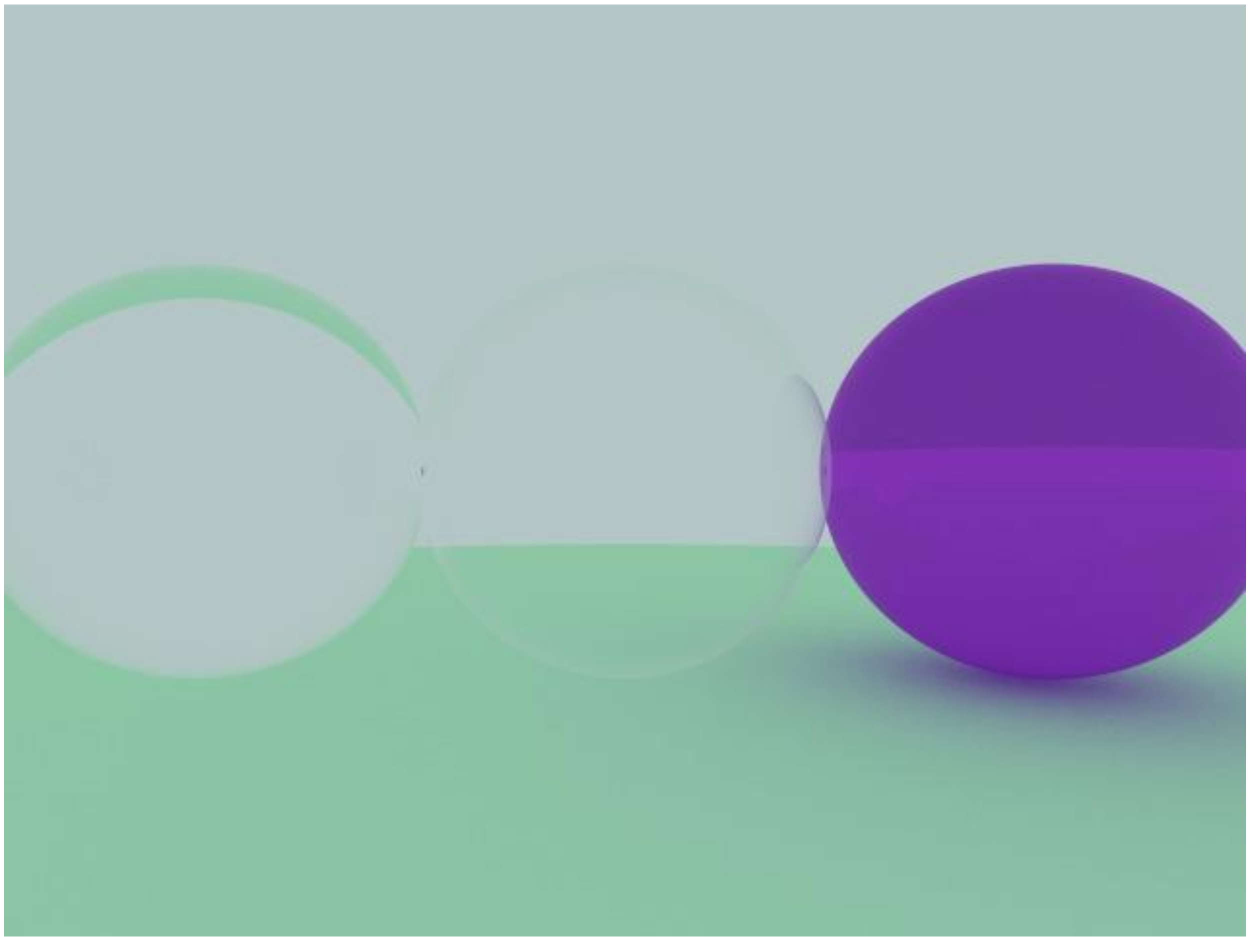
Cem Kaya 26440



Cem Kaya 26440

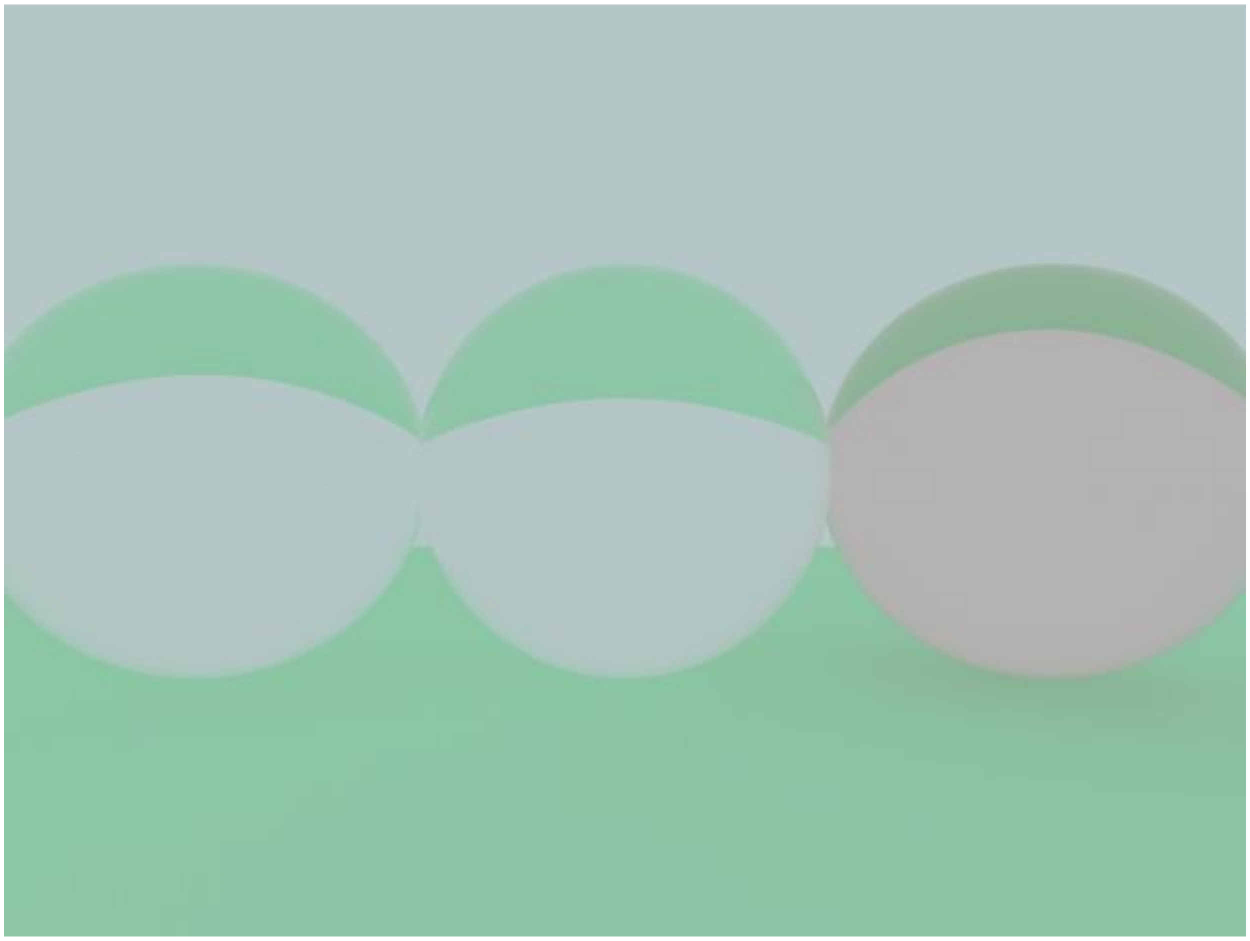
The ground is metal and the spheres are glass.

In this picture indexes of refraction are 1 1.14 2.5 the light acts interesting around ~1.14



The next picture has IR as fallows 1.4 1.3 1.2 the third is tinted 0.1

Cem Kaya 26440

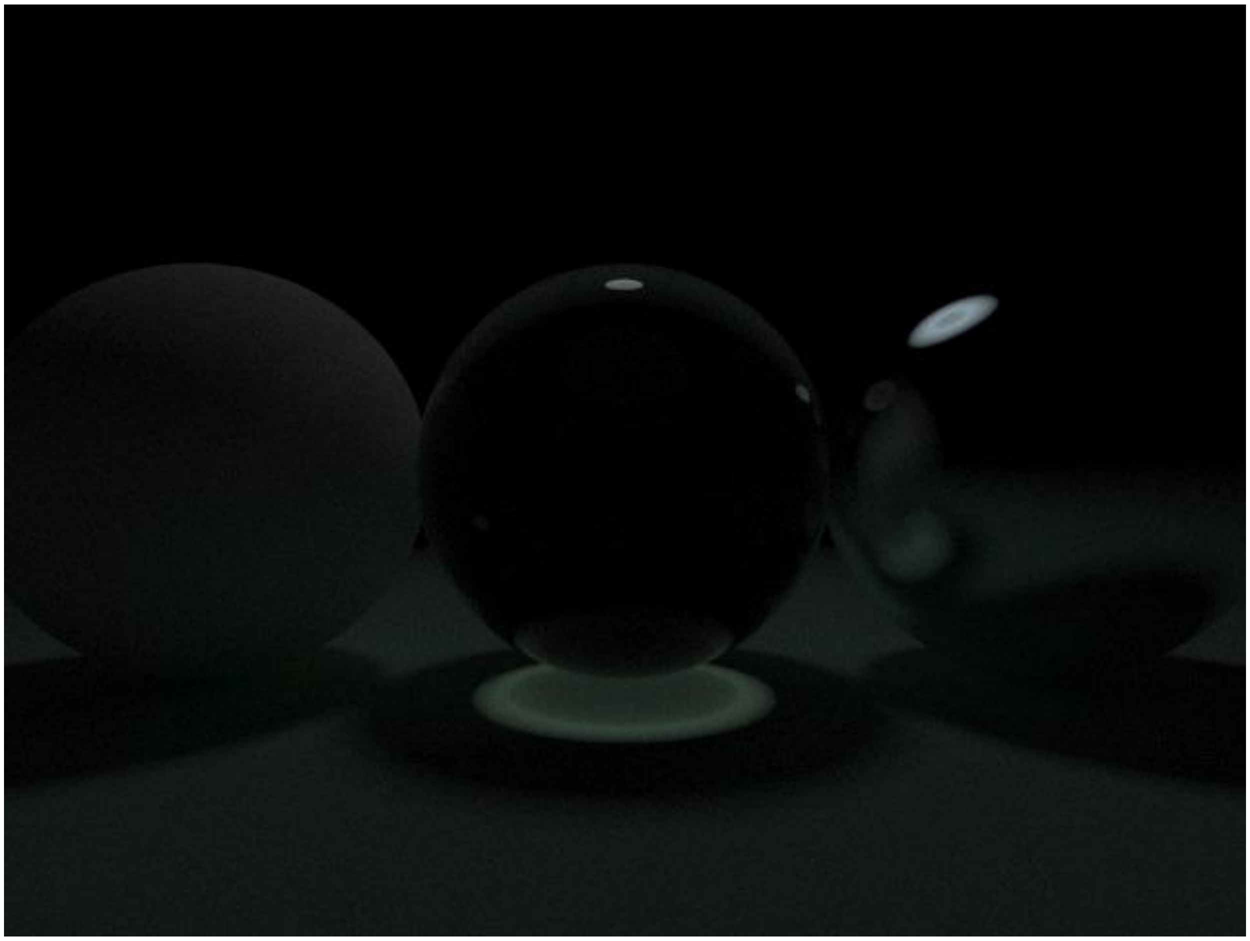


Task 6 [Required]: Lights (15Pt)

The point lite requires significant amount of sampling just to make its surroundings of it visible. (R smaller them 0.1)

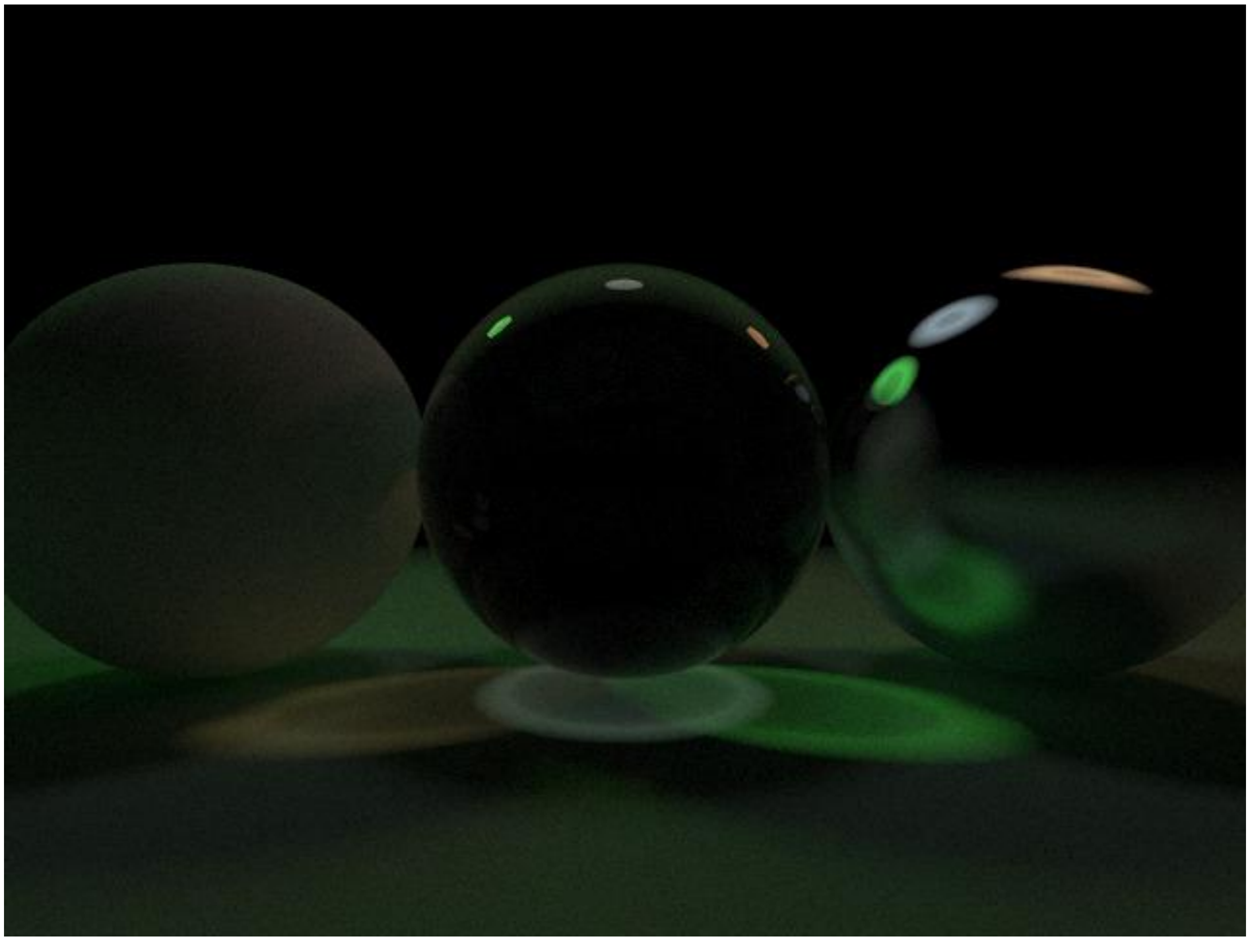
The

Cem Kaya 26440

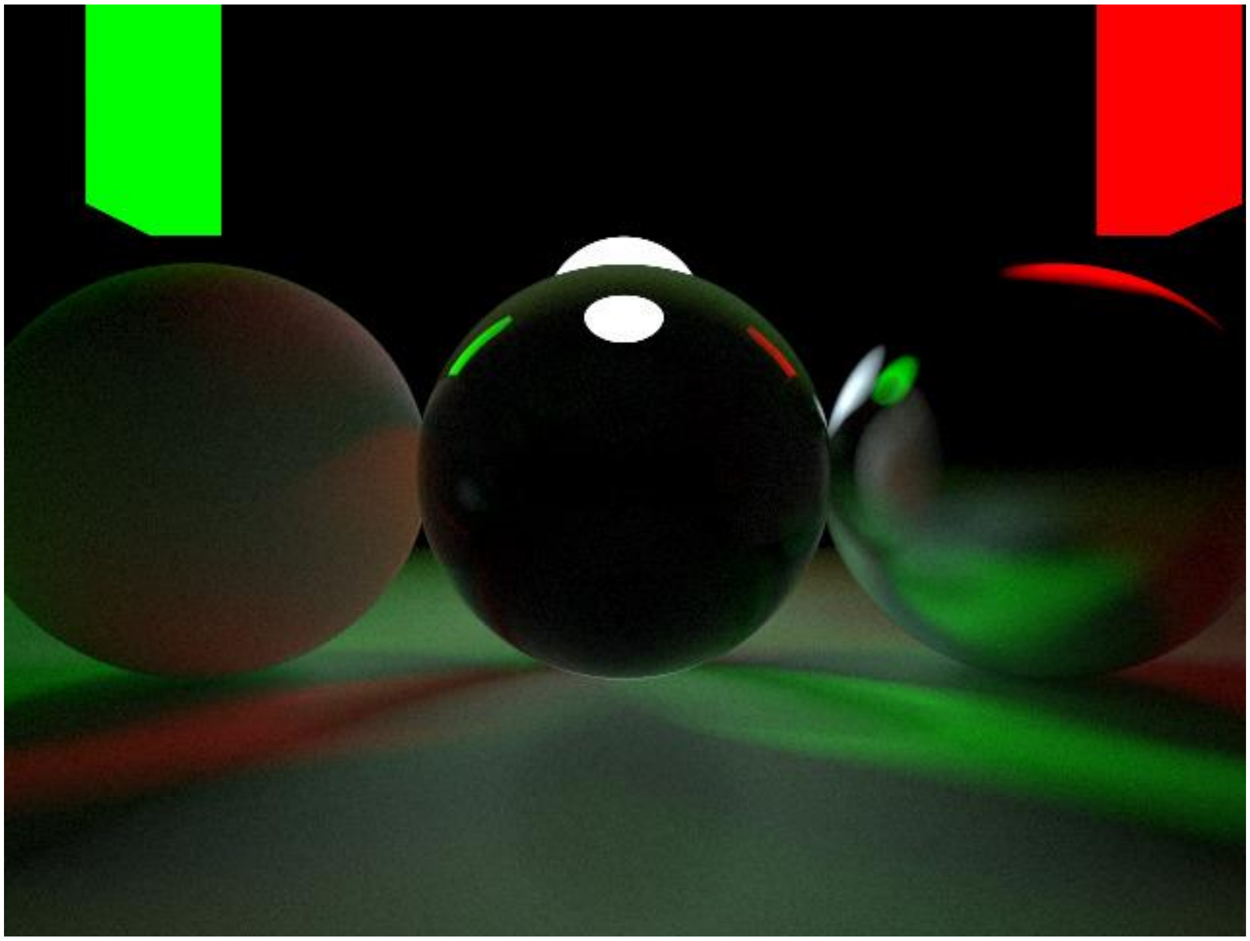


Cem Kaya 26440

The light is above the middle sphere 2 units and back 2 units even though it isn’t seeable the light refraction clearly shows its
location.

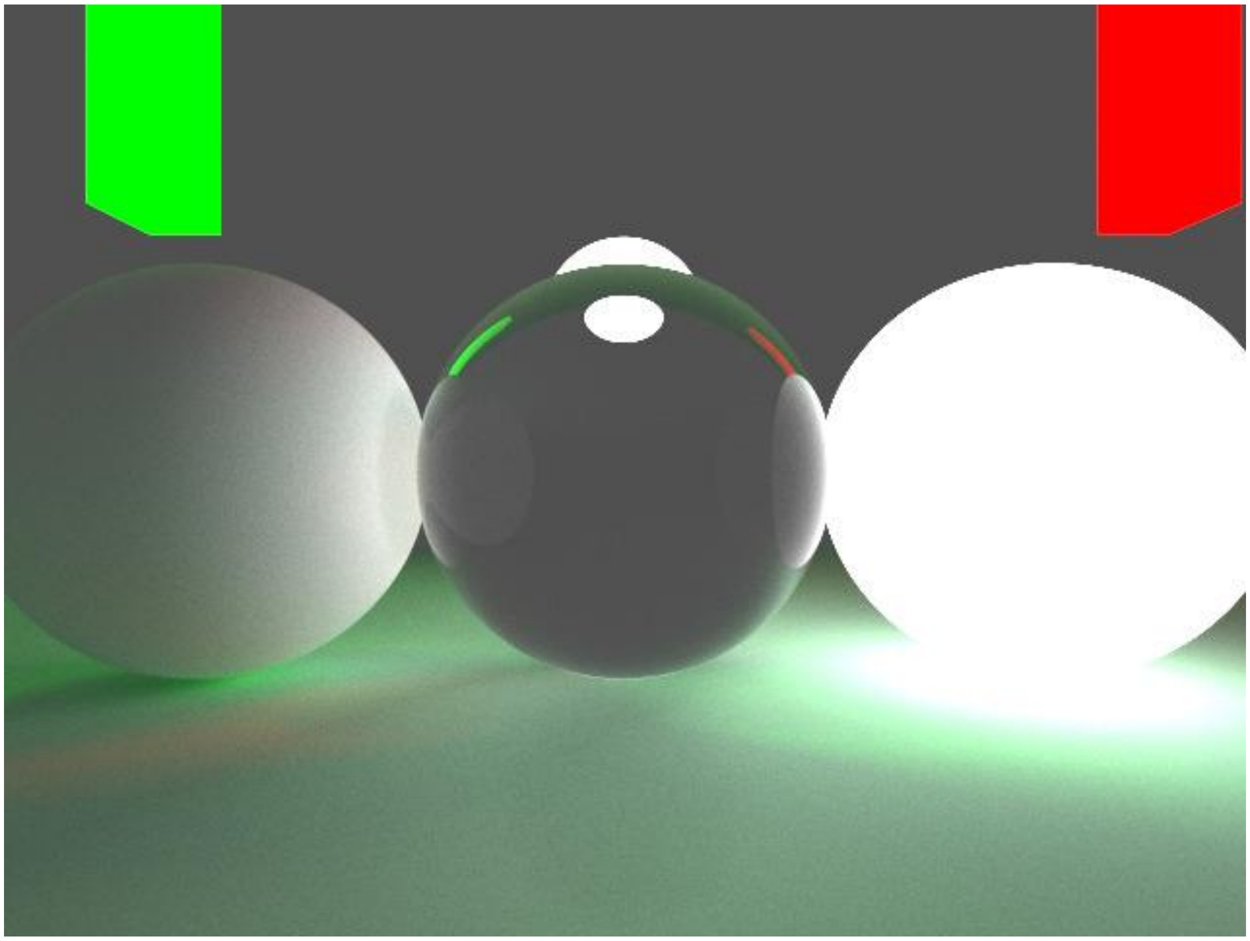


There are 3 light in this one is just above and back of the middl e sphere and other are obove of other spheres



Lets bring the lights down and make their unique colour more visible

Cem Kaya 26440



Task 7 [Optional]: Let’s get creative! (10Pt)

Instructions

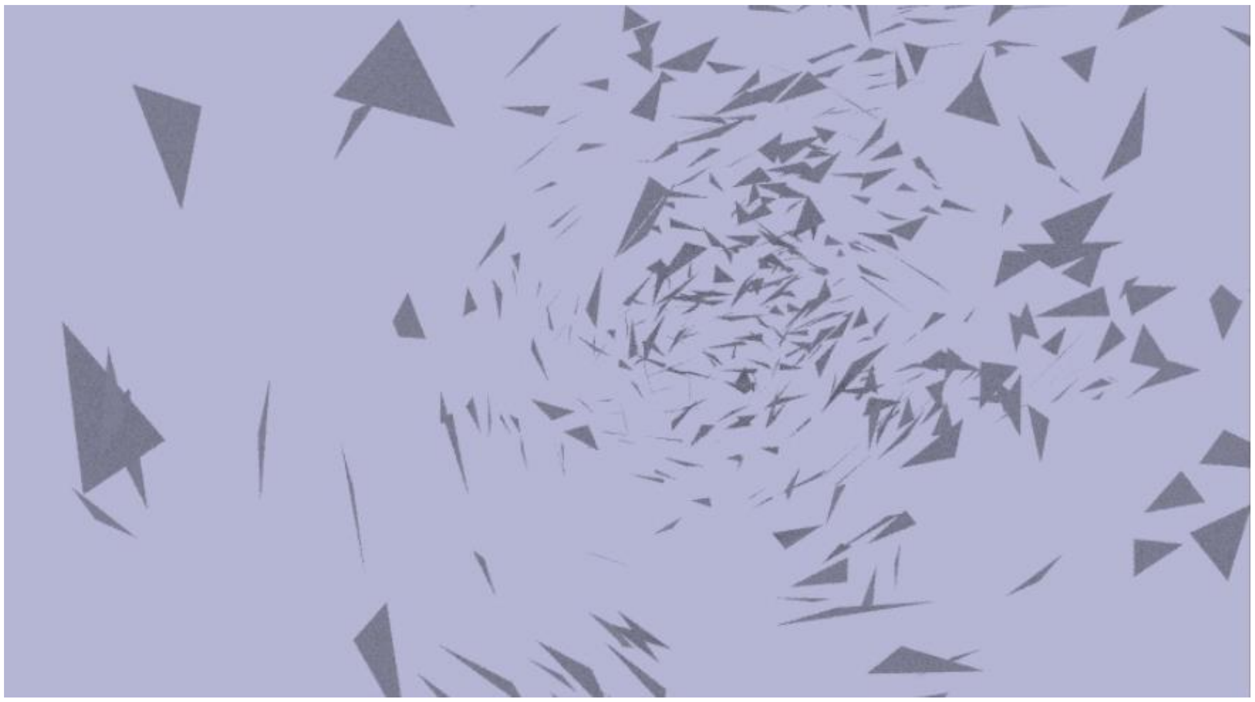
The following render Is 2560 × 1080 because that is my monitors resolution. It contains a edited version of a real life scanned T-rex skull. This model is my favourite 3D printed model which I printed. However before show the render I will show the steps which i had to take before I could achieve this render. Since the coordinates of the model is determined and I didn’t implement a way to move the model I had to work with the given coordinates. This was a very big challenge for me since just viewing a low resolution render took 2-3 minutes and this way of composing a scene isn’t user friendly. It made me respect old CGI even more.

MODEL: https://www.thingiverse.com/thing:308335

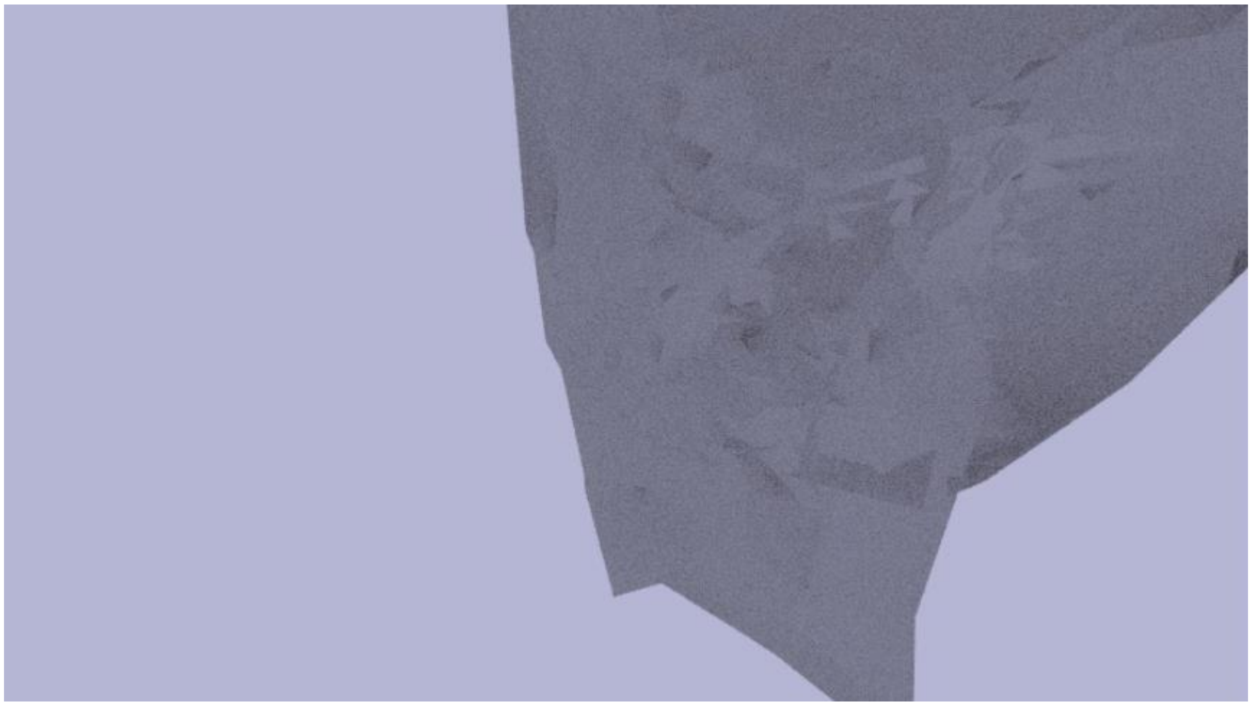
I have edited it to make it usable

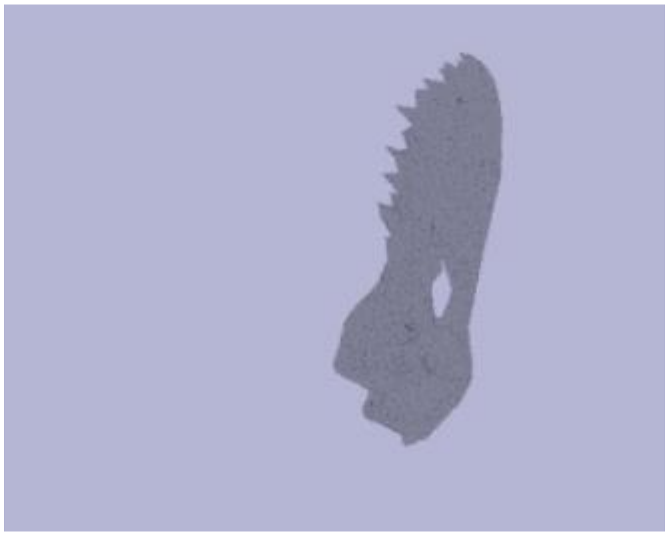
This is where I started from:

Cem Kaya 26440



Cem Kaya 26440





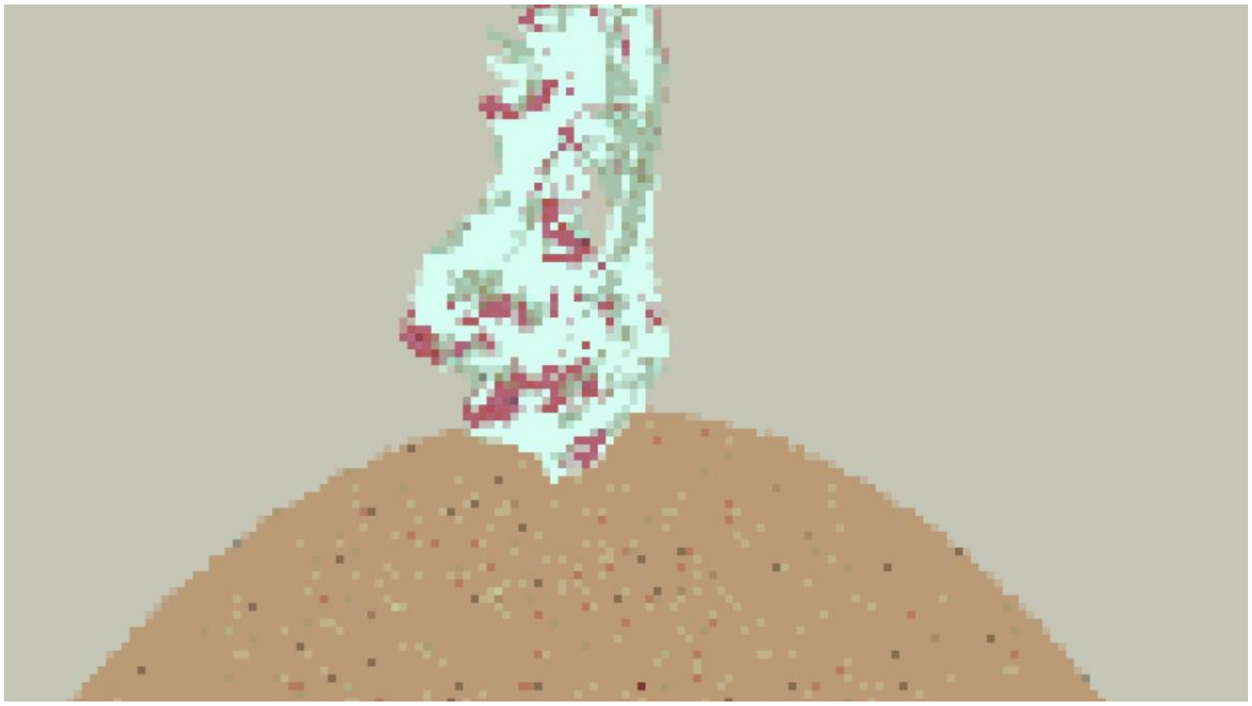
NOTE: i dont use the normal data in the model and assume that model has the vertexes of triangles in a spesific order which would indicate the normals. However this was partially wrong for this model.

Cem Kaya 26440

Cem Kaya 26440

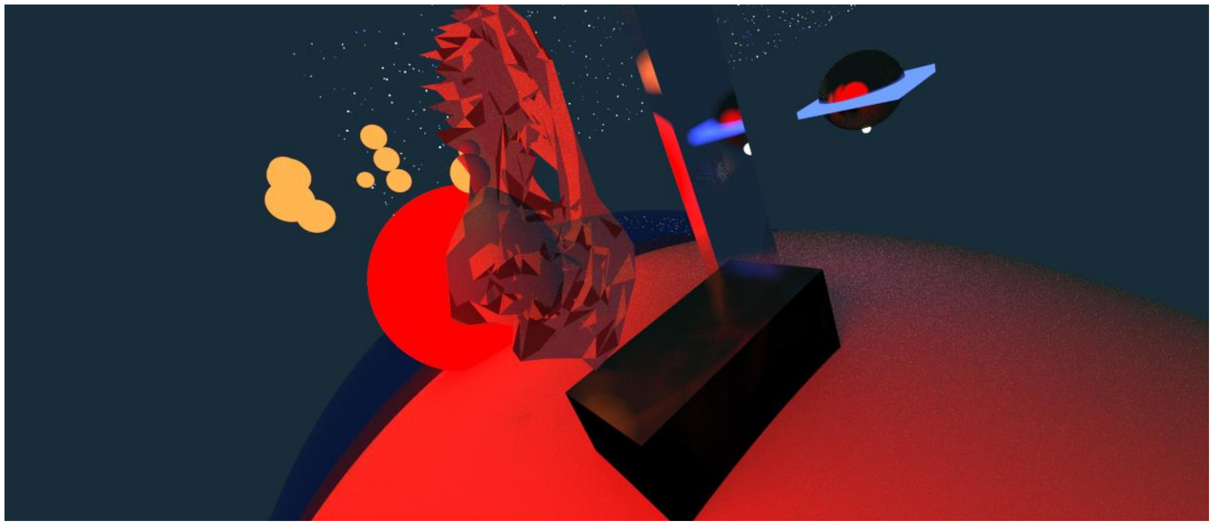
Trying to add texture by using different materials

Cem Kaya 26440



This is the final render:
It took over 11 hours

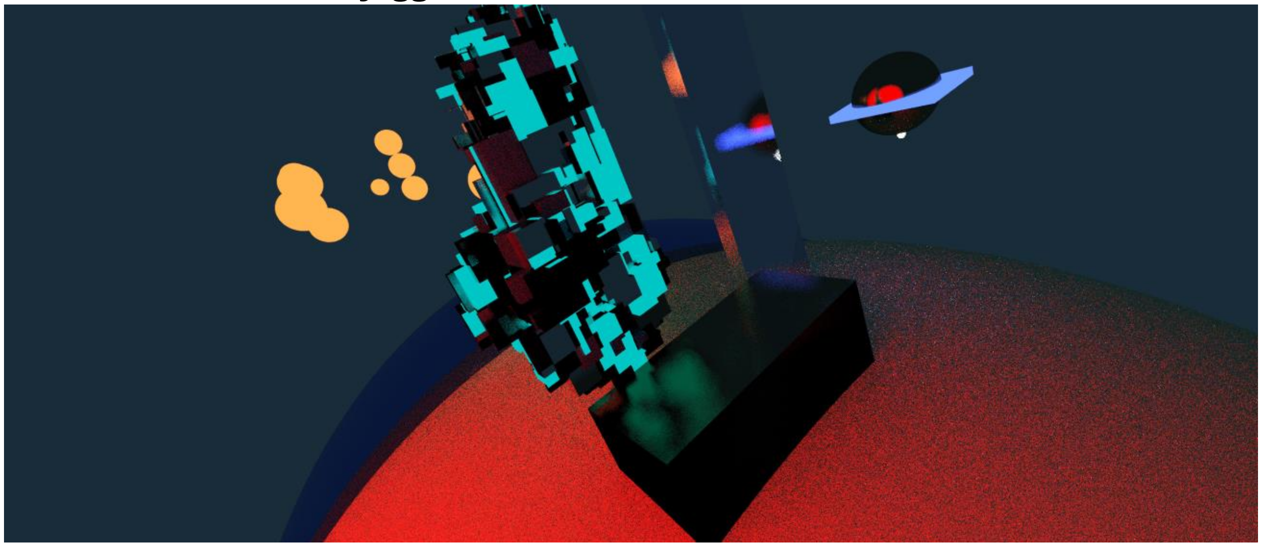
Re-extinction:



This picture depicts extinction of alien dinosaurs on an asteroid with the milky way in the background.

Cem Kaya 26440

I had another idea while I was waiting for the picture above rendering instead of using triangles what if I used cubes for each pair of point on the triangle to make the model more jagged. This render took 2 hours



No explosion in this one the meteor   in the first picture was not good looking any way

Task 8 [Required]: Questions (10Pt)

Instructions

Answer the following questions:

- The approximate time complexity of ray tracing on models with triangles.

IT is O(n^depth) n being number of triangles and depth being the number of ray bounces
However there are more variables which affect the time complexity then these 2
O(n^d*p*s) p is the pixel count and s is sample count d is ray bounce depth count

Cem Kaya 26440

- What is the difference between pre-processing and computing the image? Why?

  Depending on what we mean by pre-processing we are composing the elements and the objects which will be in the image for example: assigning the materials, determining the locations of objects, and parsing meshes. This may include creation of acceleration data structures and scene graphs.

  When we are computing/rendering the image we use these pre-set upped environment and structures to physically simulate the real world to capture the light data as a camara or an eye would.

- What are the critical parameters for the ray tracer algorithm's performance?
  1. Number of pixels
  2. Sampling amount
  3. Ray bounce depth
  4. Number of objects/ triangles
- Imagine you are a systems engineer at Pixar. There is a new super-resolution in 6000×6000 pixels and you have to estimate the maximum rendering time per frame. Assume that the scenes are static, so you are not going to spend any CPU on animation, scene hierarchy, etc. The average scene has 500 objects with a total of 5.000.000 triangles to check intersection with.

  I assume we don't use any optimization methods such as AABB's or parallel processing.
  This makes the number of hitable objects 2.500.000.000. we will have to check every possible hit per pixel and per ray bounce. No ray bounce depth or sampling number is given I assume 1000 for sampling and 10 for the depth . For the worst case every ray will hit every triangle and they will bounce 10 times for each pixal.
  My computer calculates 2764800 ~ (2560X1080) pixel image with 500 triangles and ~4000 other hittable objects and ray bounce depth of 10 with 200 samples per pixel with the speed of 8K pixel per minute

Num pixal	Object num	depth	Sample	Pixel per minute
2764800	4500	10	200	8K
36000000	2.500.000.000	100	1000	0.0028800

Pixar has 5 times of samples so if this was added to my renderer pixel per minute: 1.6K
Pixar has 555555 times more objects then my render so it will be 555555 times slower pixel per minute: 0.0028800

So one pixel of Pixar's render would take 347 minutes in my renderer on my computer.

There are 36000000 pixels so it would take 347*36000000= 12492000000 minutes

Cem Kaya 26440

23 751.3629 years

The assumption of 10 depth and 1000 samples is slowing the render however my path tracer is not optimised. To improve the performance ewe can use AABB's to check an object rather then checking every triangle. However, this doesn't affect the worst-case time. We can parallelise this render for example my renderer is single threaded on a Cpu while I have 12 cores and 24 threads. Just multithreading can improve the performance by a factor of ~20. Furthered more we can parallelise the render on gpU. There is a reason why octane and other renderers use GPU. Even further, we can distribute the render over multiple machines for speed up.