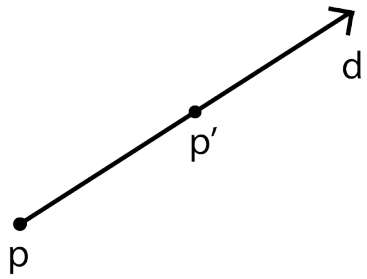
# Standalone Simple Ray Tracer

This document explains fundamental informations for better understanding of ray tracer mechanism.

## 1 - Math

### Ray

In geometry, a ray can be defined as a part of a line that has a fixed starting point but no end point. It can extend infinitely in one direction. So a ray must have a starting point p and a direction d as illustrated in the image.




$$p' = p + d*t$$

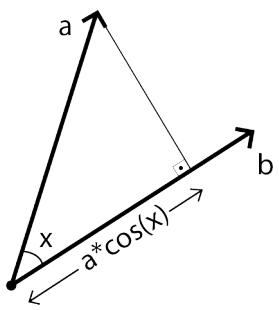
We can write the ray's equation as shown in the image. In this equation, t is a variant and p' is a point on the ray.

### Dot Product

Let a and b are two vectors, x is the angle between them, |a| and |b| are magnitudes of the vectors and "." is the dot product symbol. We can calculate the dot product of two vectors in this way;

$$a.b = |a| * |b| * \cos(x)$$

The first important point, if we look at the equation carefully, we can easily see that the dot product of two vectors is equal to multiplication of |a| and magnitude of projected b onto a or vice versa.



The second one, if we consider that these vectors are unit vectors, the equation will be like this;

$$a.b = 1 * 1 * \cos(x)$$
$$a.b = \cos(x)$$

So if we are using unit vectors, the dot product will give us information about the angle between the vectors. These are the two important informations about dot product.

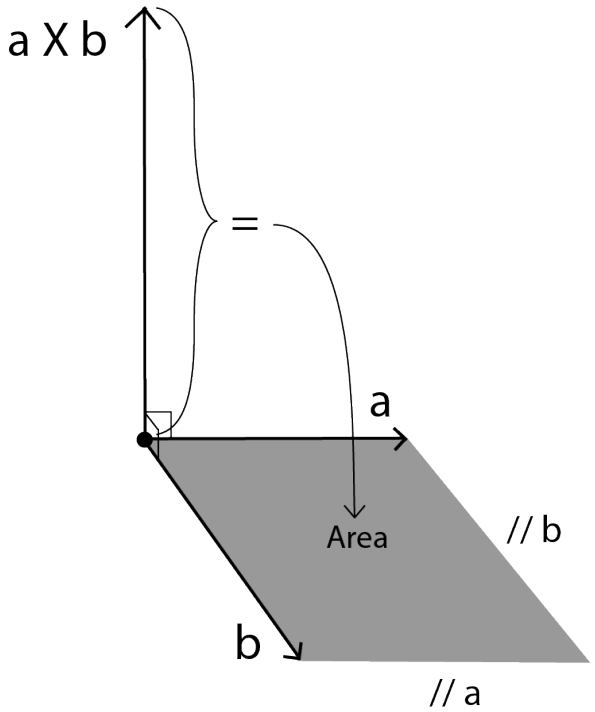
## Cross Product

Let a and b are two vectors, x is the angle between them, |a| and |b| are magnitudes of the vectors and "X" is the cross product symbol. We can calculate the cross product of two vectors in this way;

$$a \times b = |a| * |b| * \sin(x) * n$$

n is the perpendicular unit vector to both a and b.

As you can see from the equation, cross product of two vectors gives us another vector which is perpendicular to both vectors. The magnitude of the result vector is equal to area of parallelogram with vectors a and b for sides illustrated in the image;




Another important information about cross product, we multiply magnitudes and normal vector with sin(X). As the angle between vectors gets smaller, the magnitude of the result vector will get smaller. So if we use unit vectors for cross product;

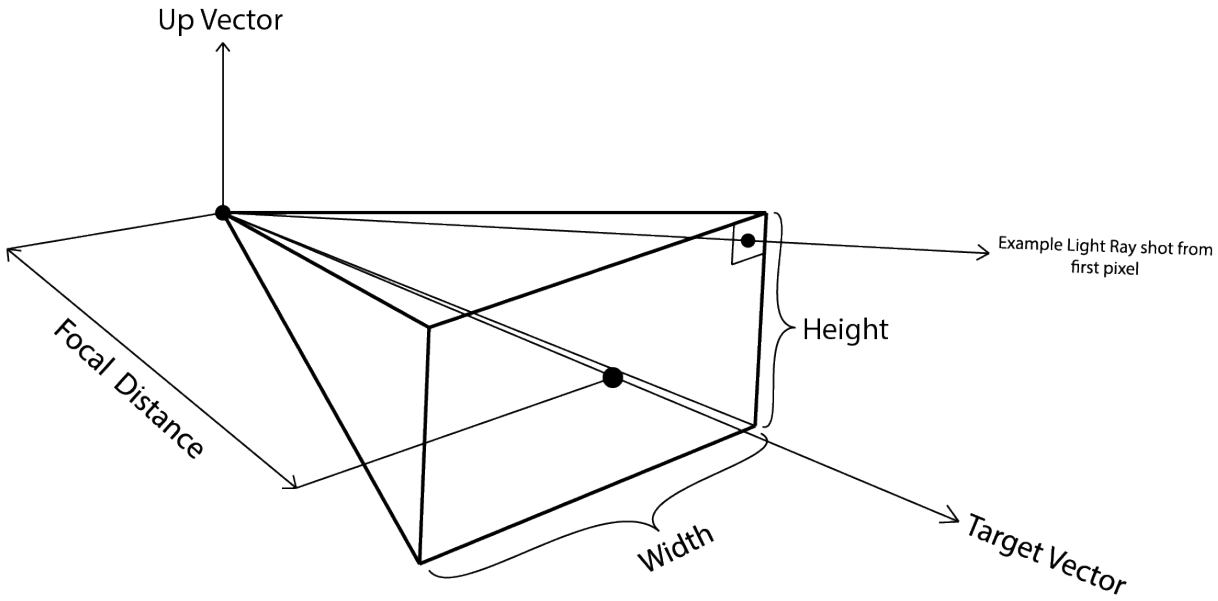
$$a \times b = 1 * 1 * \sin(x) * n$$

We will get this equation. As we did in the dot product, we can get information about the angle between vectors using cross product.

There are a lot of topic that we didn't mention in here like trigonometry, solving quadratic equation etc. We assume that you already know or familiar with the basic level math.

## 2 - Camera

Our camera model used in ray tracer is perspective model. You can see the general scheme of perspective camera model in the illustrated image;

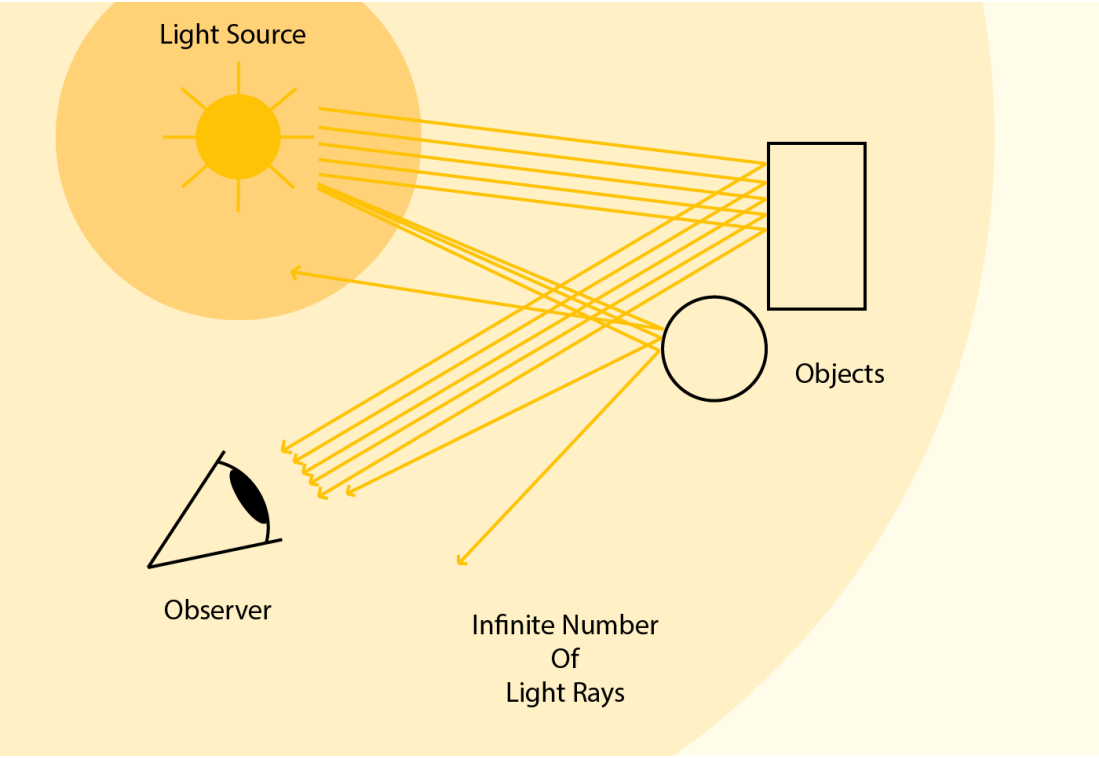



Normally there are two plane for perspective camera model, which are near plane and far plane but in our ray tracer we don't have any max distance to eliminate farther objects. So in the illustrated image, the drawn plane is near plane. Assume that our far plane is at infinite distance and we draw every object in our perspective no matter how far it is.

In this model, according to the given information in the image, we calculate each ray per pixel and prepare them for the next stage which is test them against the scene.

## 3 - How do we see objects?

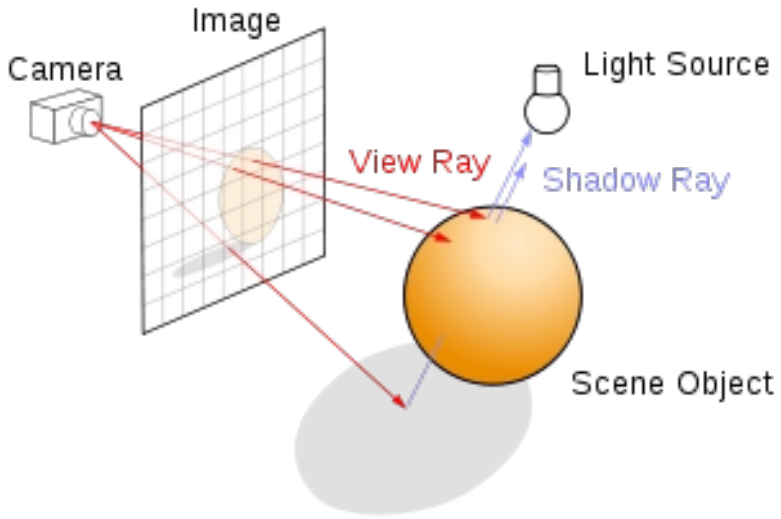
The images we see are made up of light reflected from the objects we look at. This light enters the eye through the cornea, which acts like a window at the front of the eye. The amount of light entering the eye is controlled by the pupil, which is surrounded by the iris(the colored part of the eye). Because the front part of the eye is curved, it bends the light, creating an upside down image on the retina. The brain eventually turns the image the right way up.




In real life, light sources shoots infinite number of light rays to environment. But because we have a finite resources, we can't simulate infinite number of light rays. For this reason, instead of shooting rays from light sources we will shoot finite number of light rays from our camera model then we will calculate each rays to find the corresponding color values and finally we will construct our final image by assigning color values to pixels.

## 4 - How Ray Tracing Works ?

As we said before, we have limited resources. Because of that, we will shoot and calculate light rays only visible by the camera. With this method we will have get rid of calculating non-visible light rays. With little math, we can easily prepare light rays from camera position to pixel positions. After preparation of light rays, we can test light rays against scene. In other words we can check each light ray for an intersection with objects in the scene. If there is an intersection, only thing we have to do is, find the intersection point, normal vector of that point, reflected ray vector from that point and lastly the vector from the intersection point to the light source. After find these informations, we will calculate color values for each pixels and draw the final image. So let's explain how we calculate color values for each pixels.



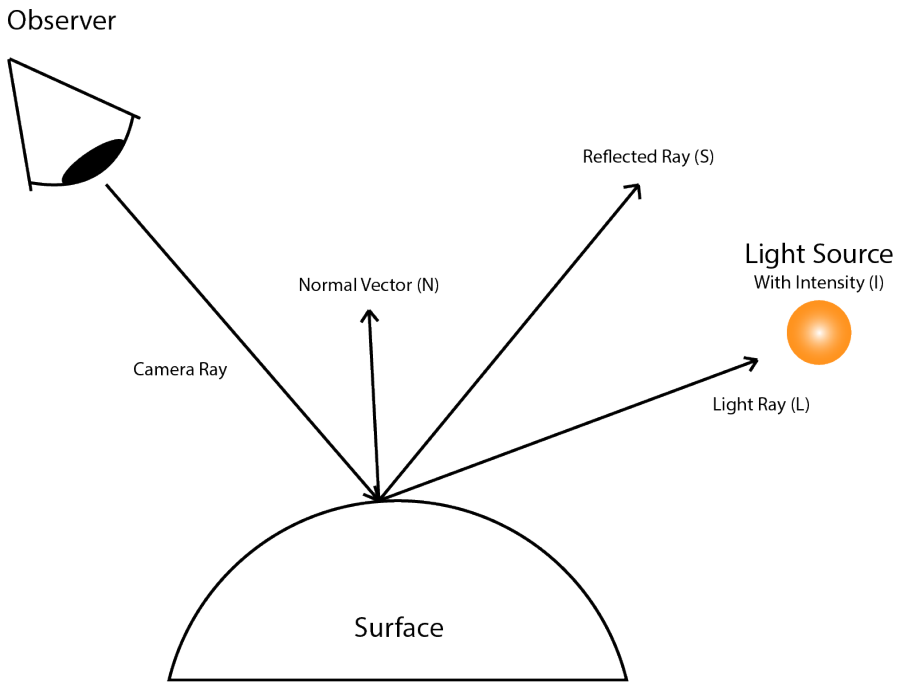

## Light

Each light source has its own intensity, own mechanism and own color. For example, point lights illuminates a spherical area around itself. The intensity decreases as the distance between the object and the light source increases. To find out how much illumination we have to apply on object, we need to procees these two steps;

1 - Find out if the intersection point is visible by the light source or not.

2 - If it isn't visible by the light source then skip the contribution of this light source on this point otherwise find the contribution of this light source on this point.

These two steps are calculated separately for each light source in the scene then illumination values are accumulated for find the final illumination amount on the related intersection point.



## Material

Each object has its own behaviour against the light source. And this behaviour is determined by the material of the object. For example, metallic objects reflect their environment but plastic objects can't do that. Normally, to calculate the illumination on the surface according to the material information is a little bit cumbersome. Because there is no object such that the material information is same in every piece of surface and also unstriated. But for the sake of simplicity, we have constructed our "Material" struct like that. With this struct, to calculate illumination of a point on a surface, we use this formula;

Illumination on a point;

$$(kd * (N.S) * I + ks * (S.L)^{ns} * I + ka * I)*(1 - kr) + kr * Cr$$

kd: diffuse coefficient
ks: specular coefficient
ns: specular disperse coefficient
ka: ambient coefficient (calculated if and only if the light type is ambient)
kr: reflection coefficient
Cr: contribution of the reflected ray
N, S, L and I are explained in the image above

## Shadow

If there is no illumination or less illumination on a surface comparing to environment, we see shadows. We are already using the ray tracing method to render our scene, so there is only one thing we need to do for rendering shadows.

Before calculating illumination,

1 - test the L ray illustrated in the image above against the scene
2 - check if there is any intersection with other objects
3 - if there is any intersection, this means that something blocking that light source so skip this light source, don't add its contribution to the intersection point.
4 - if there is no intersection, this means that our intersection point see that light source. So add its contribution to the intersection point.

By applying these steps, intersection points where the shadows must be located will get less illumination and there will be darker areas(shadows) in the final image.

I think it's better to stop here otherwise there is no limits for theoretical informations and I don't want to strangle you with unnecessary informations. I tried my best and exlained source codes with comments, so if you understood these informations you can skip directly to the source code.