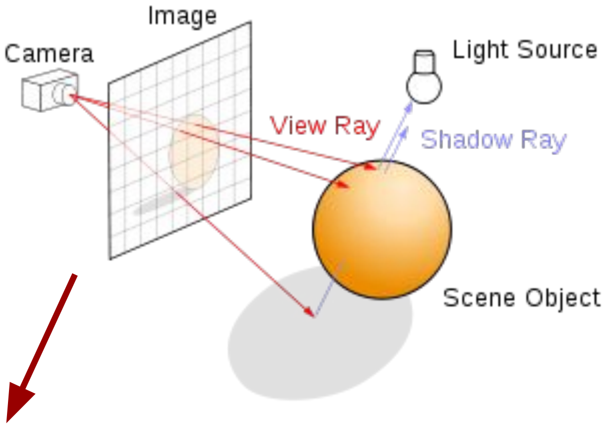
# Ray Tracer

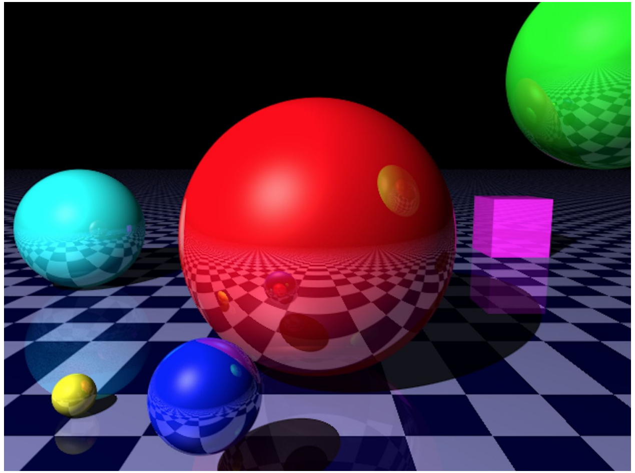
## Capturing the Path of Light

Kiet Tran and Ojashvi Rautela

# What is Ray Tracing?

Simulating how light works in real life:
**light source -> primary object -> *physical* interactions -> eye**

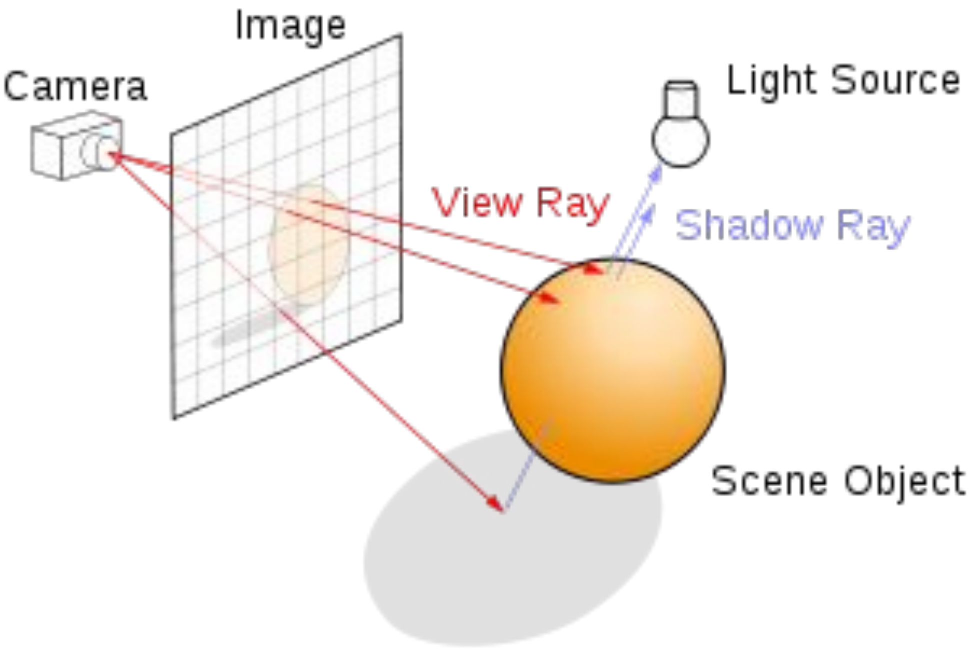




# What is Ray Tracing?

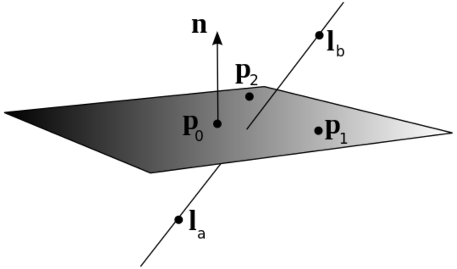
*Physical interactions:* **shadows, reflection, refraction** based on object properties such as **specular or diffusion** coefficients

**For every pixel in the image plane, check for primary and secondary ray-object intersections**



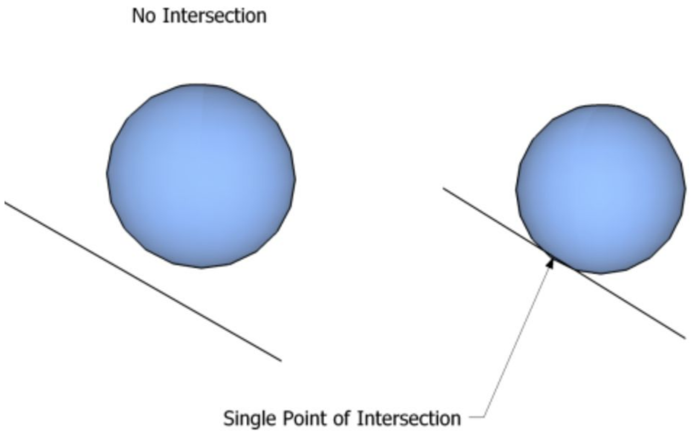
# Important Math : Object-Ray interactions

- **Plane Object and Ray (Line) intersections**
  - Plane Eq. for a set of points p
    - $(p - p_0).n = 0$
  - Point on a line (light ray)
    - ray_origin + (ray_dir * t)
    - Essentially the equation of a line
    - ray_origin = intercept; ray_dir  = slope
    - t = point of intersection
  - **Solve for t** = Substitute the eq. Of the line into the plane



# Important Math : Object-Ray interactions

- **Sphere Object and (Ray) Line intersections**
  - Sphere Eg.
    - $x^2 + y^2 + z^2 = r^2$
  - Point on ray line
    - ray_origin + (ray_dir * t)
  - **Solve for t** = Substitute the eq. of the line into the plane and



# Our Implementation : An Overview

## 1 Basic Scene

1. Vector, Color, Ray
2. Camera, Light, Objects
3. Ambient Light

## 2 Object Properties

1. Ray-Object interactions
2. Reflectivity, Transparency
3. Specular, Diffusion

## 3 Light Properties

1. Shadow
2. Reflection
3. Refraction

## 4 Anti-Aliasing

1. Averaging RGB components of n pixels around current
2. n = depth

# Program Structure (.cpp and .h files)

1. Reflection only
2. Reflection and Refraction

01 | **main.cpp** - create rays, render the scene via. anti-aliasing

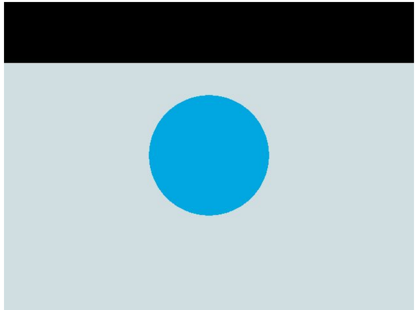
02 | **App.cpp -** getColorAt() is the main *ray tracing* function

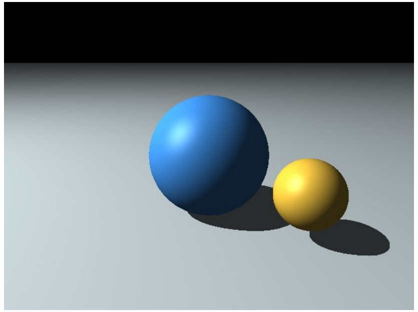
03 | **getColorAt() -** shadows, ambient, diffuse, reflection, refraction

04 | **Other classes:** simulate vectors, camera, light, rays, objects

# Milestones

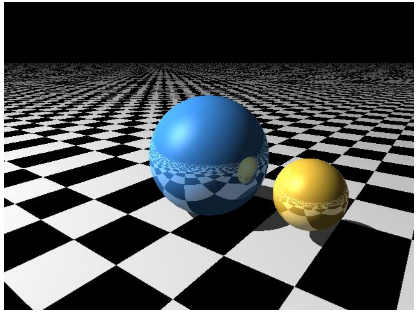
Rendered first scene with objects

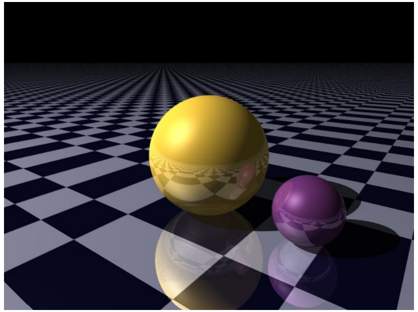




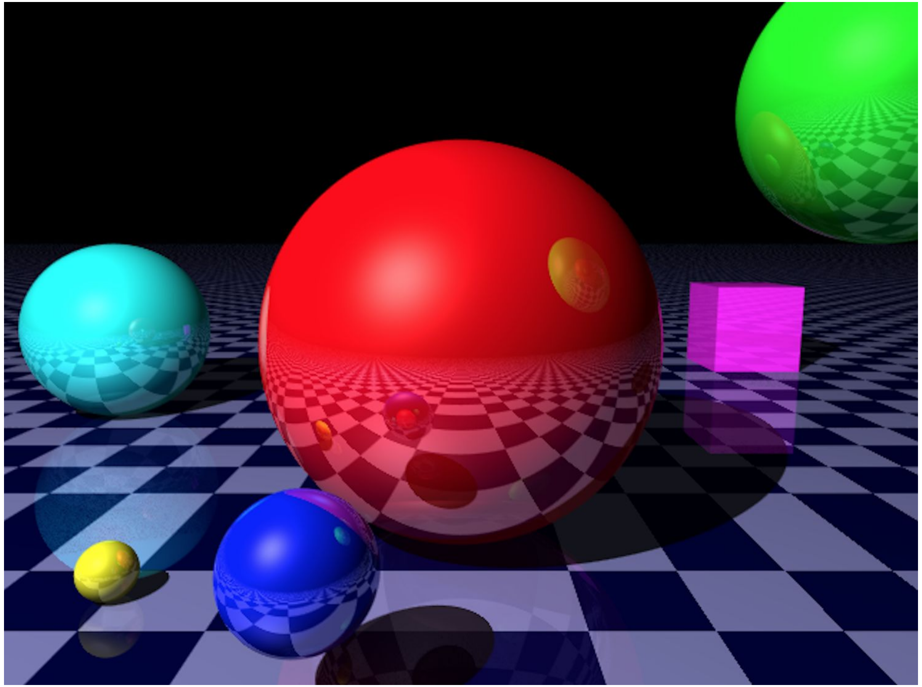
Shadow
(without anti-aliasing)

Reflection
(without anti-aliasing)

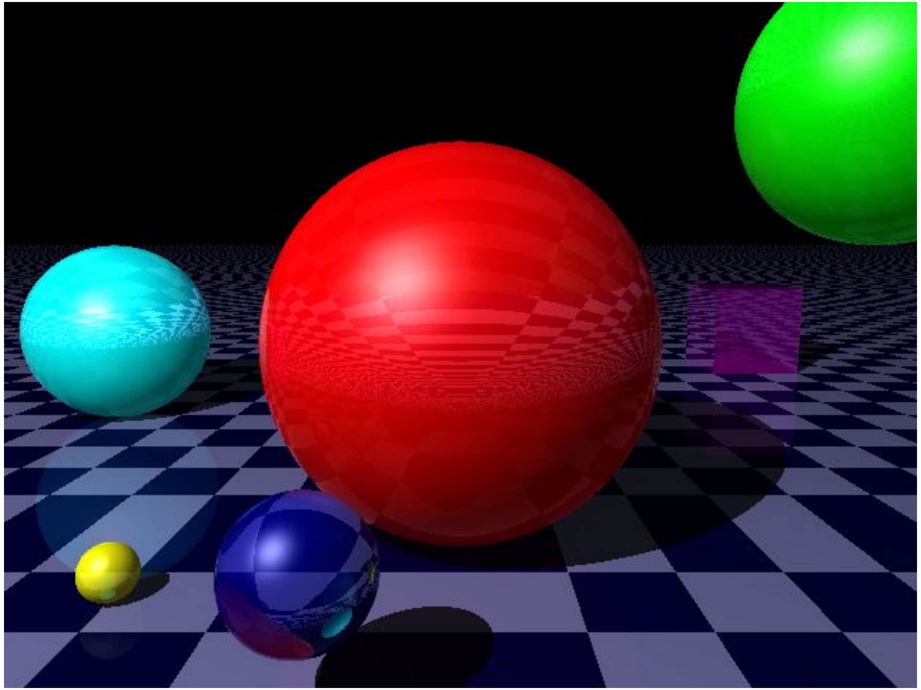




Anti-aliasing

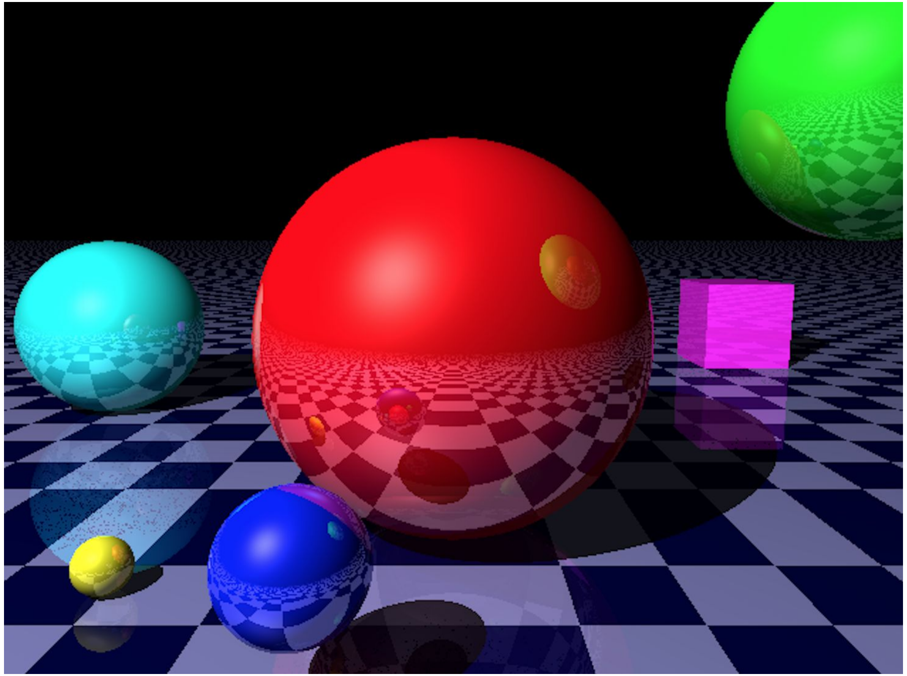


Reflection + Shadows



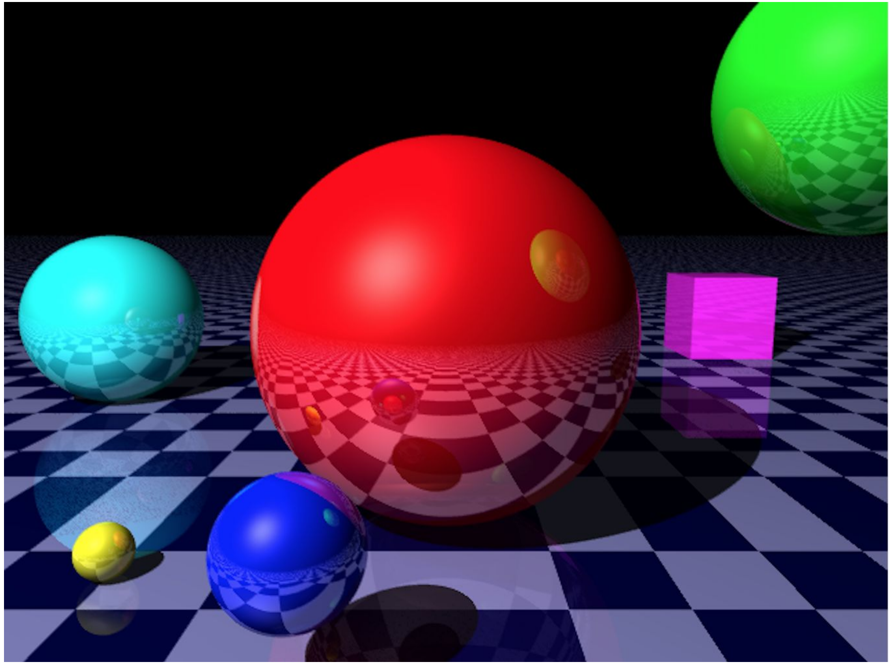
Reflection + Refraction + Shadows

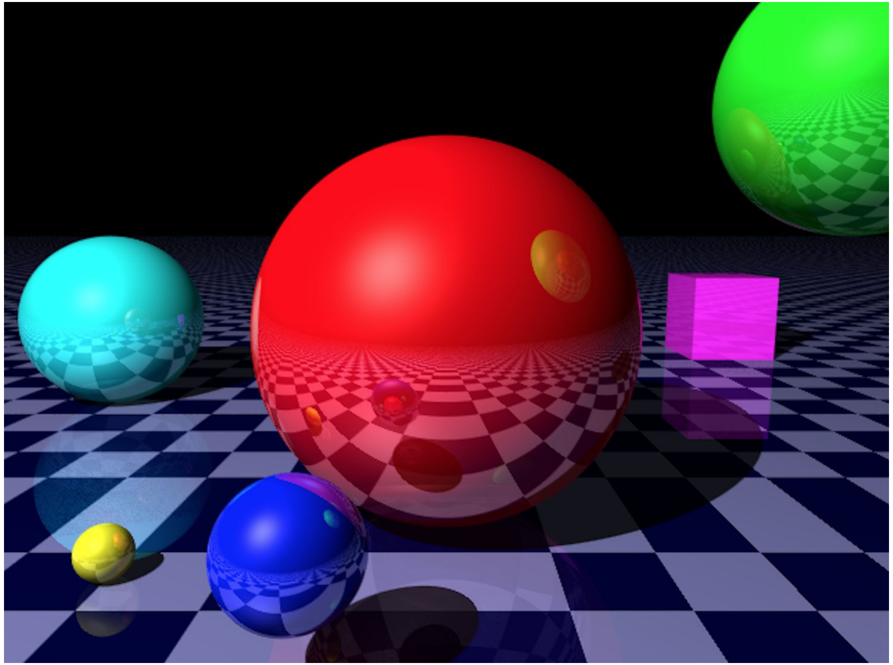
# Anti-Aliasing



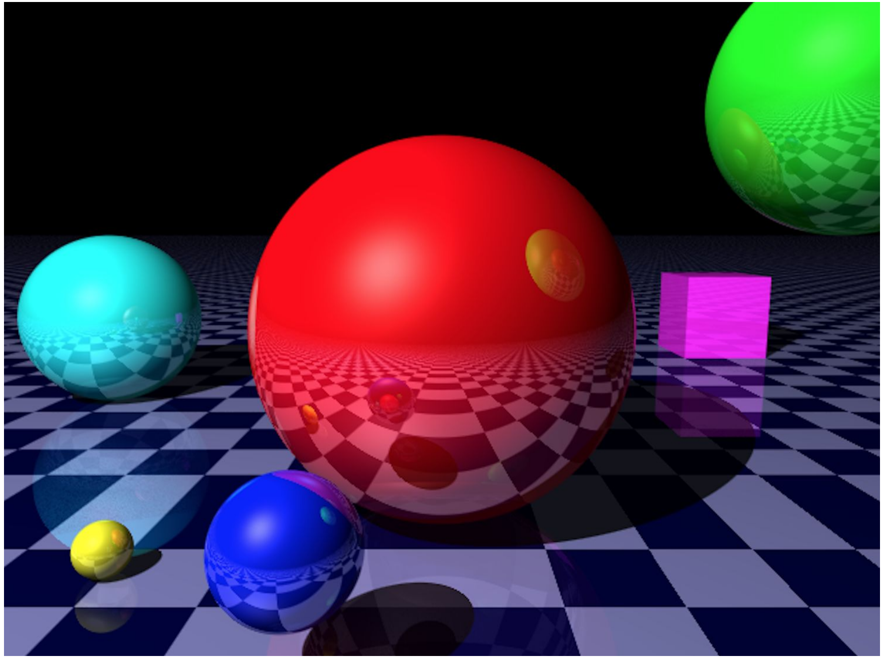
No Anti-Aliasing

Anti-Aliasing Depth = 5





Anti-Aliasing Depth = 10



Anti-Aliasing Depth = 20

# Technical Challenges

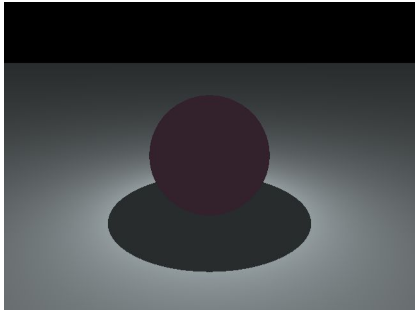
01 | Ray-Object Intersection

02 | Shadow Implementation

03 | Shadow Acne

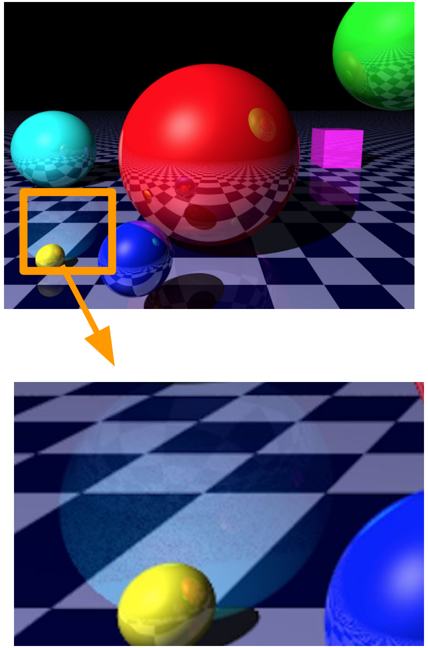
04 | Refraction

02



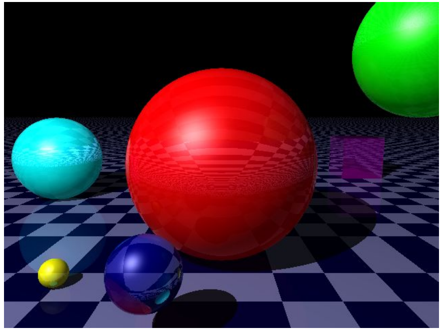
Buggy Shadow Implementation

03



Shadow Acne

04



Unrealistic Refraction

# Future Work

01 | Removing Shadow Acnes

02 | Implement transparency

03 | Add an Interactive component

# Thank you!