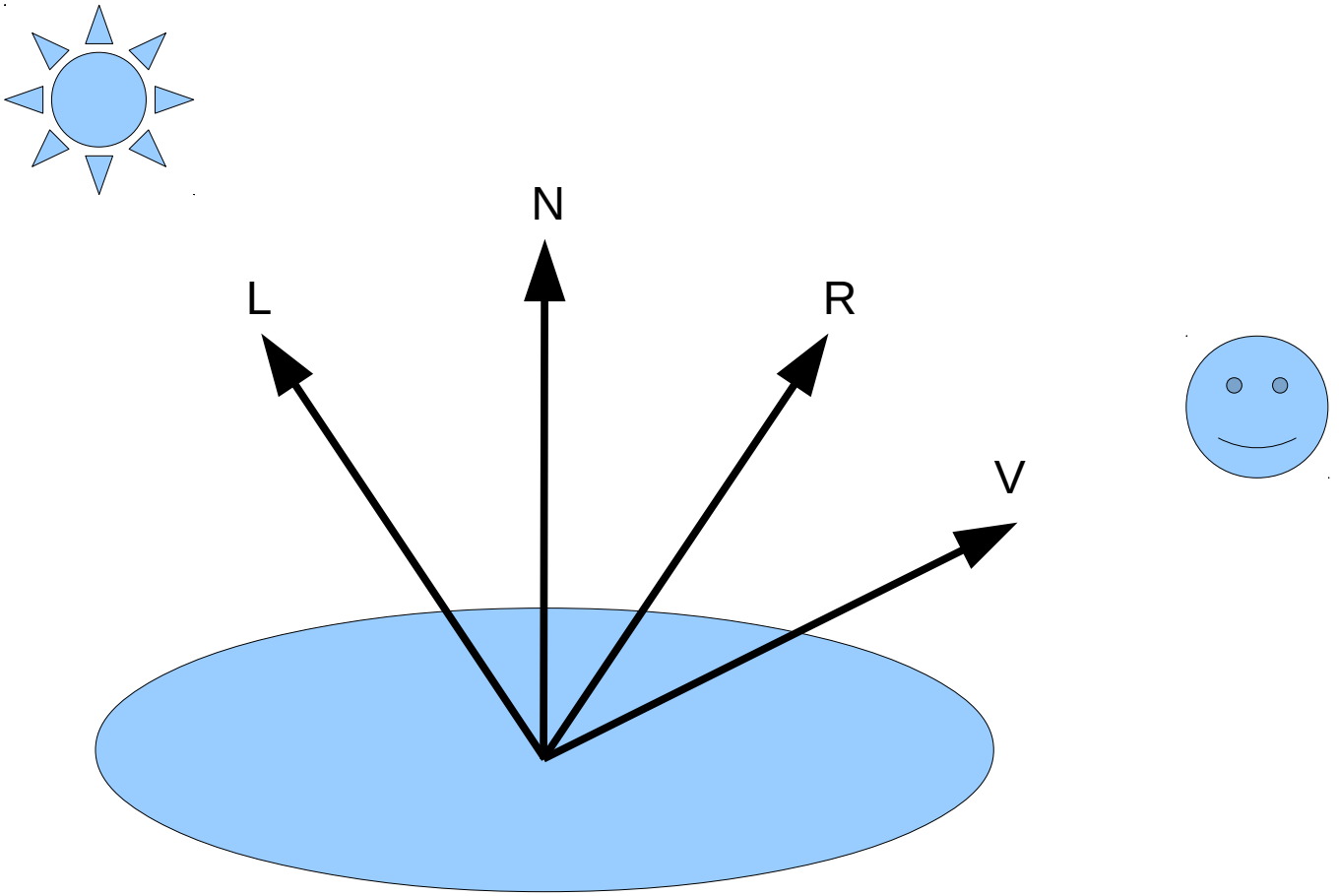
Computer Graphics

Raytracer 1

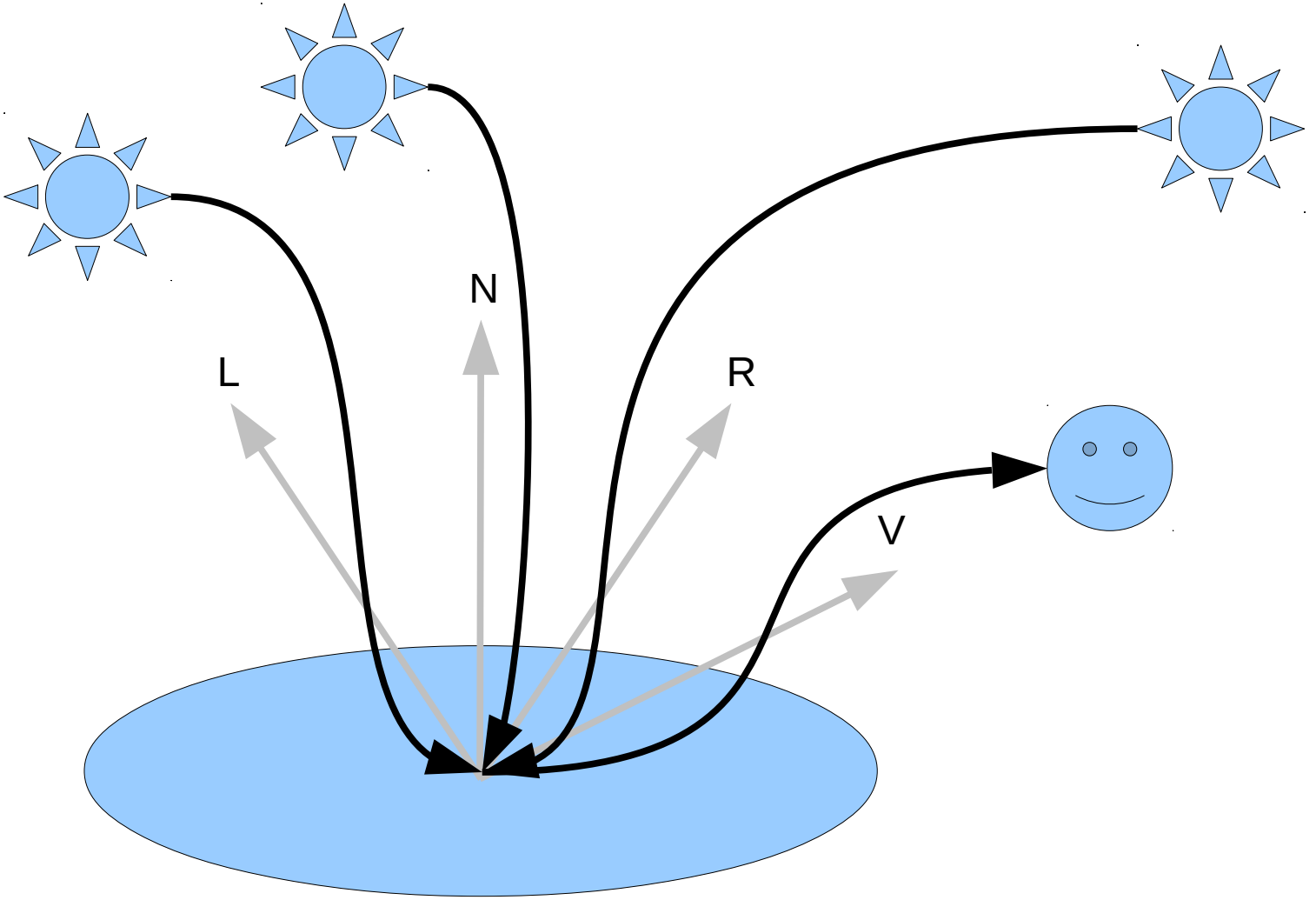
11 – 02 – 2015

Phong illumination



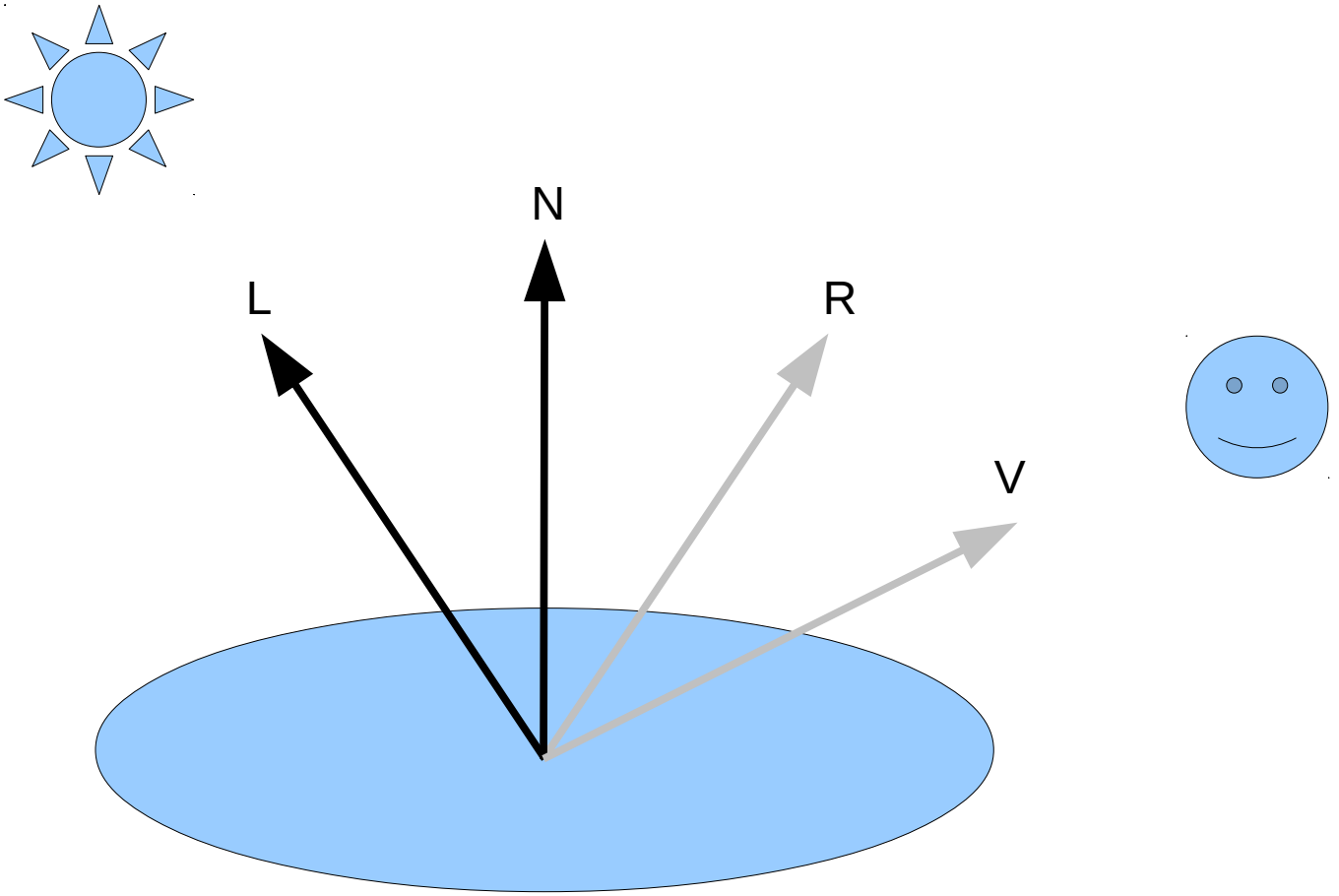
We use a simplified model, other possibilities exist!

Phong illumination - Ambient



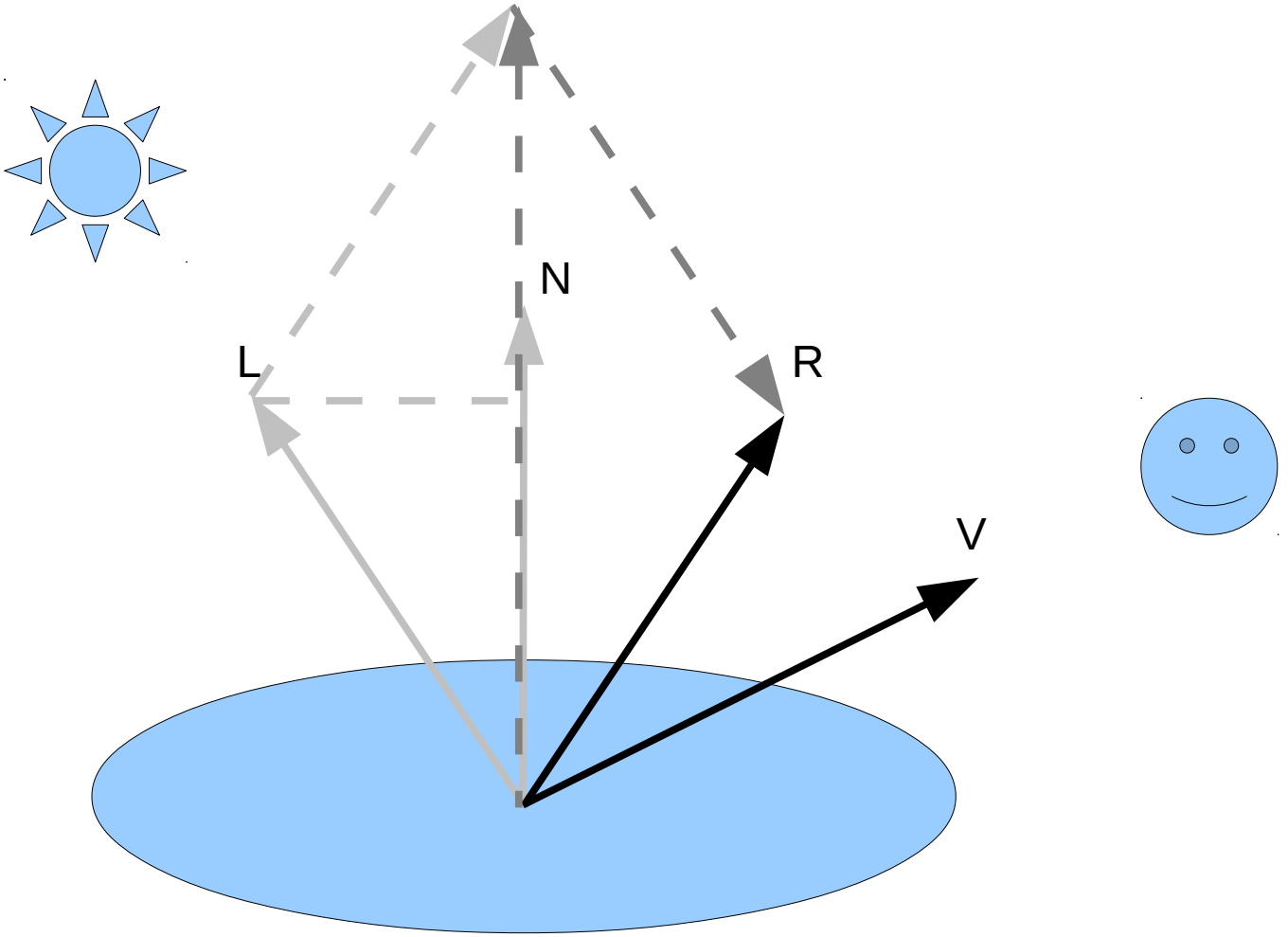
+ LightColor MaterialColor ka

Phong illumination - Diffuse



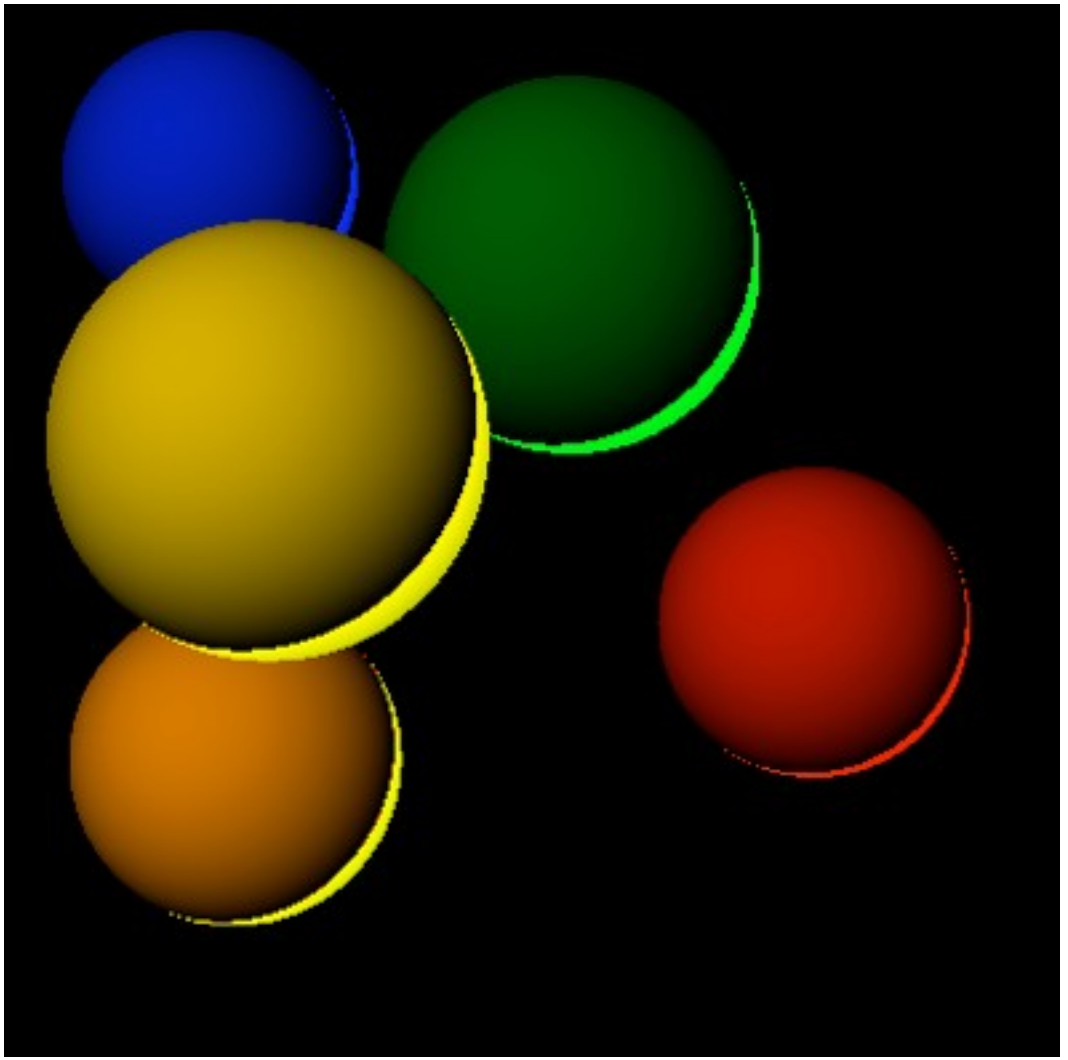
+ dot(L,N) LightColor MaterialColor kd

Phong illumination - Specular

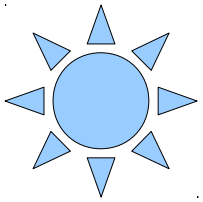


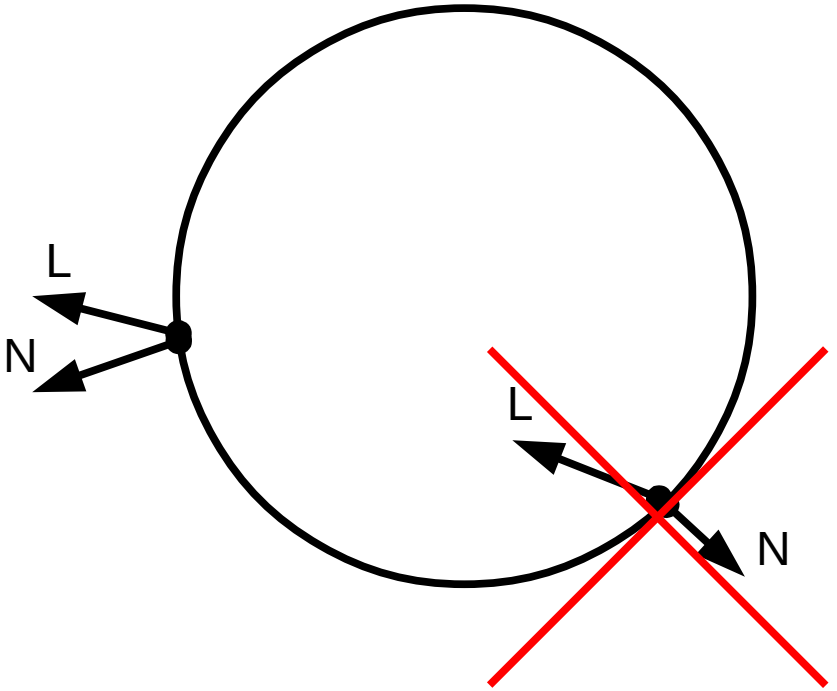
+ dot(R,V)^n LightColor ks

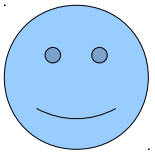
What's wrong?



This is wrong!

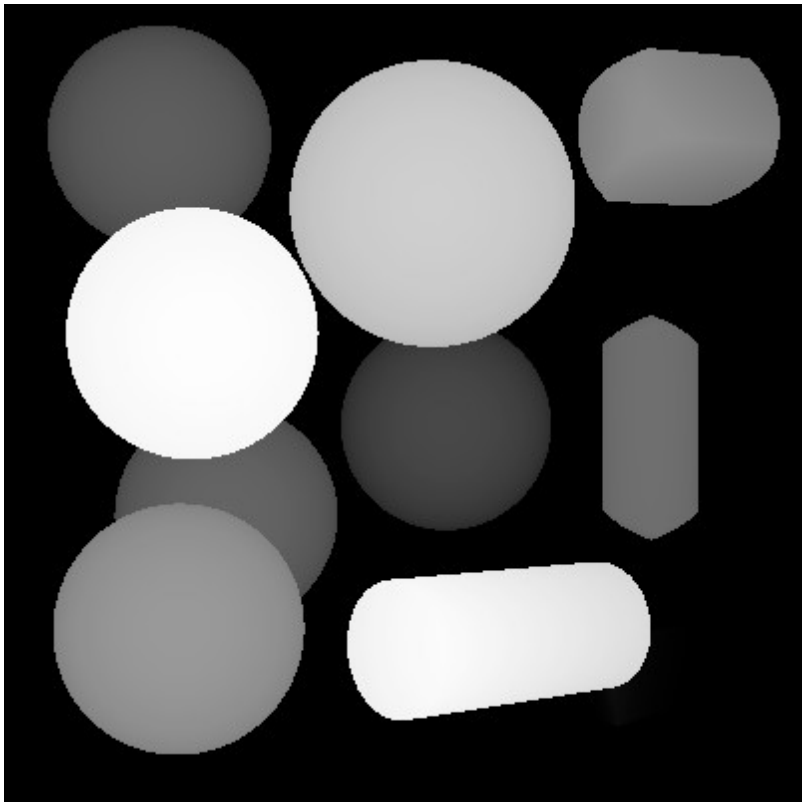






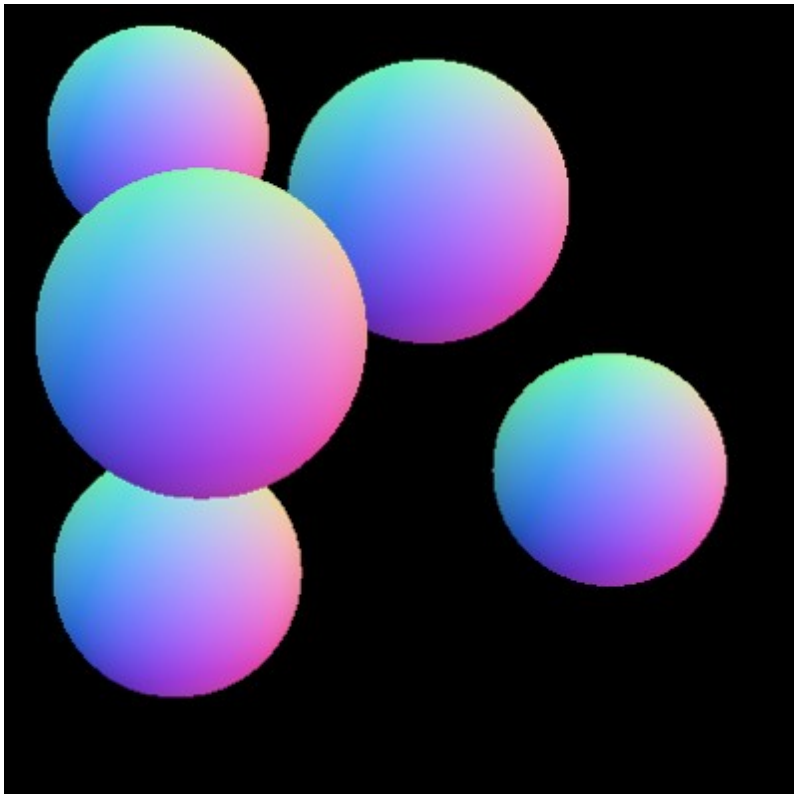
Raytracer: z-buffer

- Create z-buffer image
- Gray-scale to encode depth
- Should be configurable!



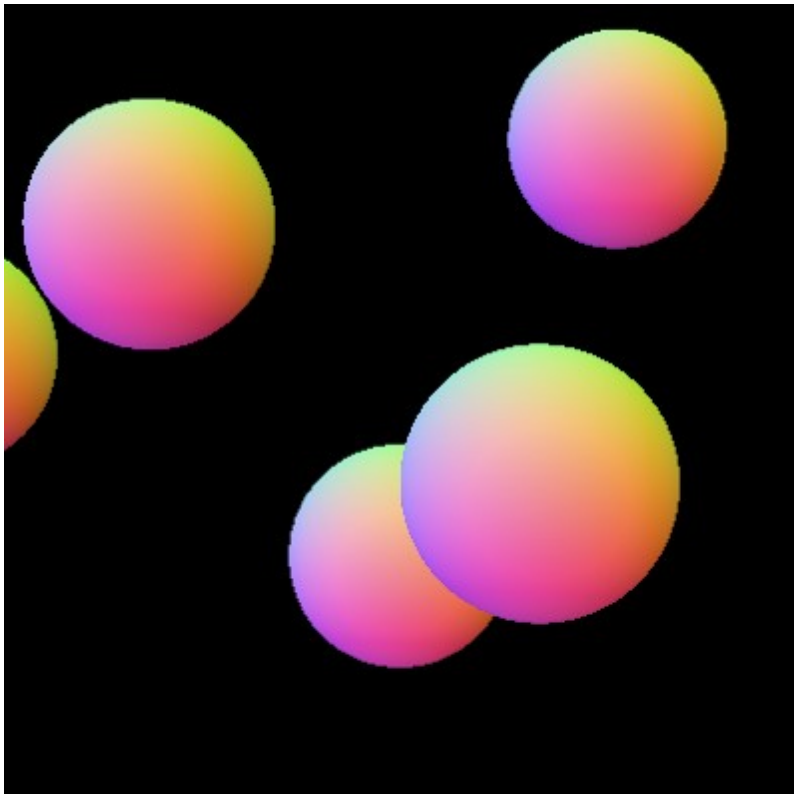
Raytracer: normal buffer

- Normal buffer: visual representation of normals
- Map [-1,1] to range of possible colors
- Again: configurable!



Raytracer: normal buffer

- Normal buffer: visual representation of normals
- Map [-1,1] to range of possible colors
- Again: configurable!



Raytracer: extra geometry

- Quad
- Plane
- Cylinder
- Cone
- Triangle
- Torus

  (or variations such, e.g.: semi spheres)

Ray tracer framework

- C++ framework
- Yaml files → less verbose xml
- Must:
  - Run on normal lab computers
  - Read the provided input files
  - Compile and run from a script/Makefile

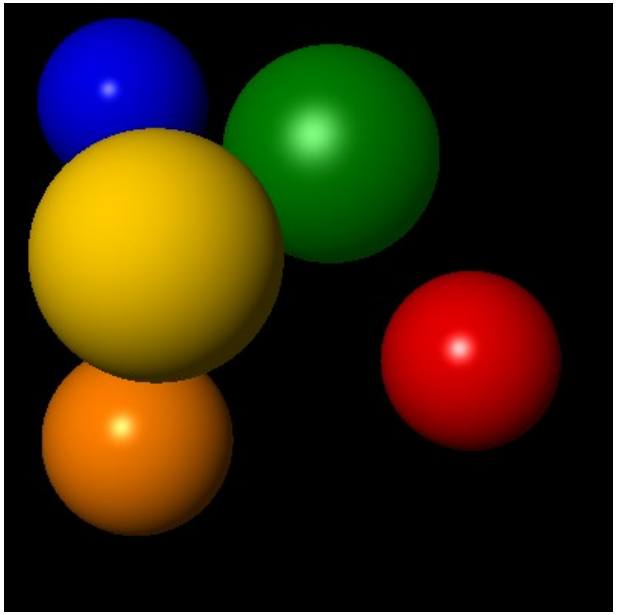
Building and running

- Compile `$ make`
- Run `$ ./ray scene01.yaml`
- Clean `$ make clean`

Assignments

1) Ray tracer with spheres

- Implement Sphere::intersect
  - Hint: look up dot product
- Phong lighting
  - Should match example output



- 2) Ray tracer normal/z-buffer
  - Use provided scene files to test
  - Use the EXACT format described
  - Extra geometry types