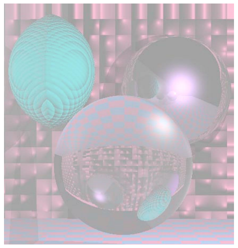


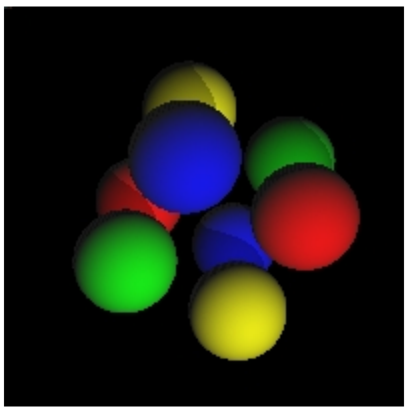
CS U540 Computer Graphics

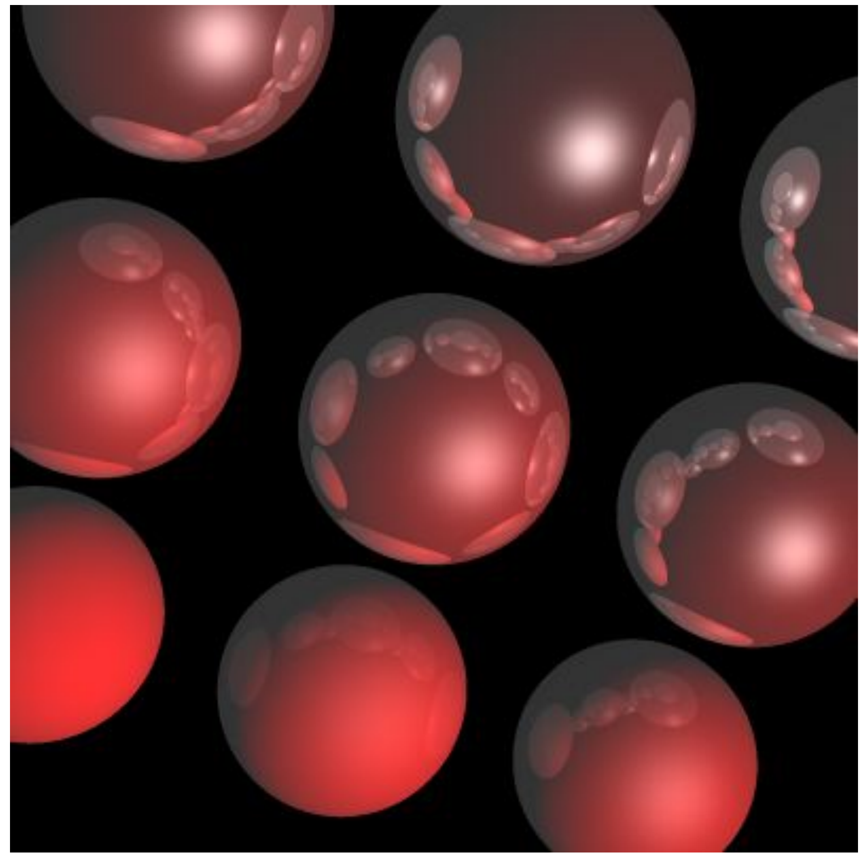
Prof. Harriet Fell
Fall 2011
Lecture 24 – November 2, 2011

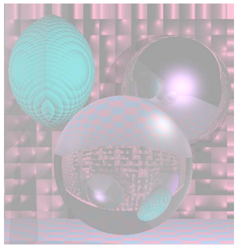
Today’s Topics

- Ray Casting

Ray Tracing
a World of Spheres

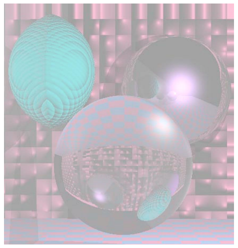




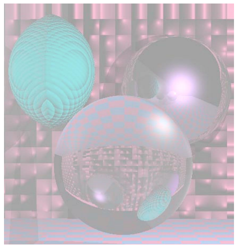


What is a Sphere

```
Vector3D    center;     // 3 doubles
double      radius;
double      R, G, B;    // for RGB colors between 0 and 1
double      kd;         // diffuse coeficient
double      ks;         // specular coeficient
(double     ka;         // ambient light coefficient)
```



```
-.01  .01   500 800 // transform theta phi mu distance
1 // antialias
1 // numlights
100 500 800 // Lx, Ly, Lz
9 // numspheres
//cx    cy  cz  radius R   G   B  ka  kd  ks  specExp  kgr  kt  pic
-100 -100   0  40     .9  0   0  .2  .9  .0     4      0    0   0
-100    0   0  40     .9  0   0  .2  .8  .1     8     .1    0   0
-100  100   0  40     .9  0   0  .2  .7  .2    12     .2    0   0
   0 -100   0  40     .9  0   0  .2  .6  .3    16     .3    0   0
   0    0   0  40     .9  0   0  .2  .5  .4    20     .4    0   0
   0  100   0  40     .9  0   0  .2  .4  .5    24     .5    0   0
 100 -100   0  40     .9  0   0  .2  .3  .6    28     .6    0   0
 100    0   0  40     .9  0   0  .2  .2  .7    32     .7    0   0
 100  100   0  40     .9  0   0  .2  .1  .8    36     .8    0   0
```

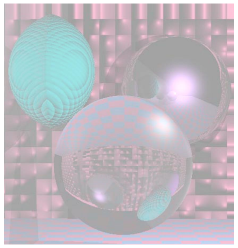


World of Spheres

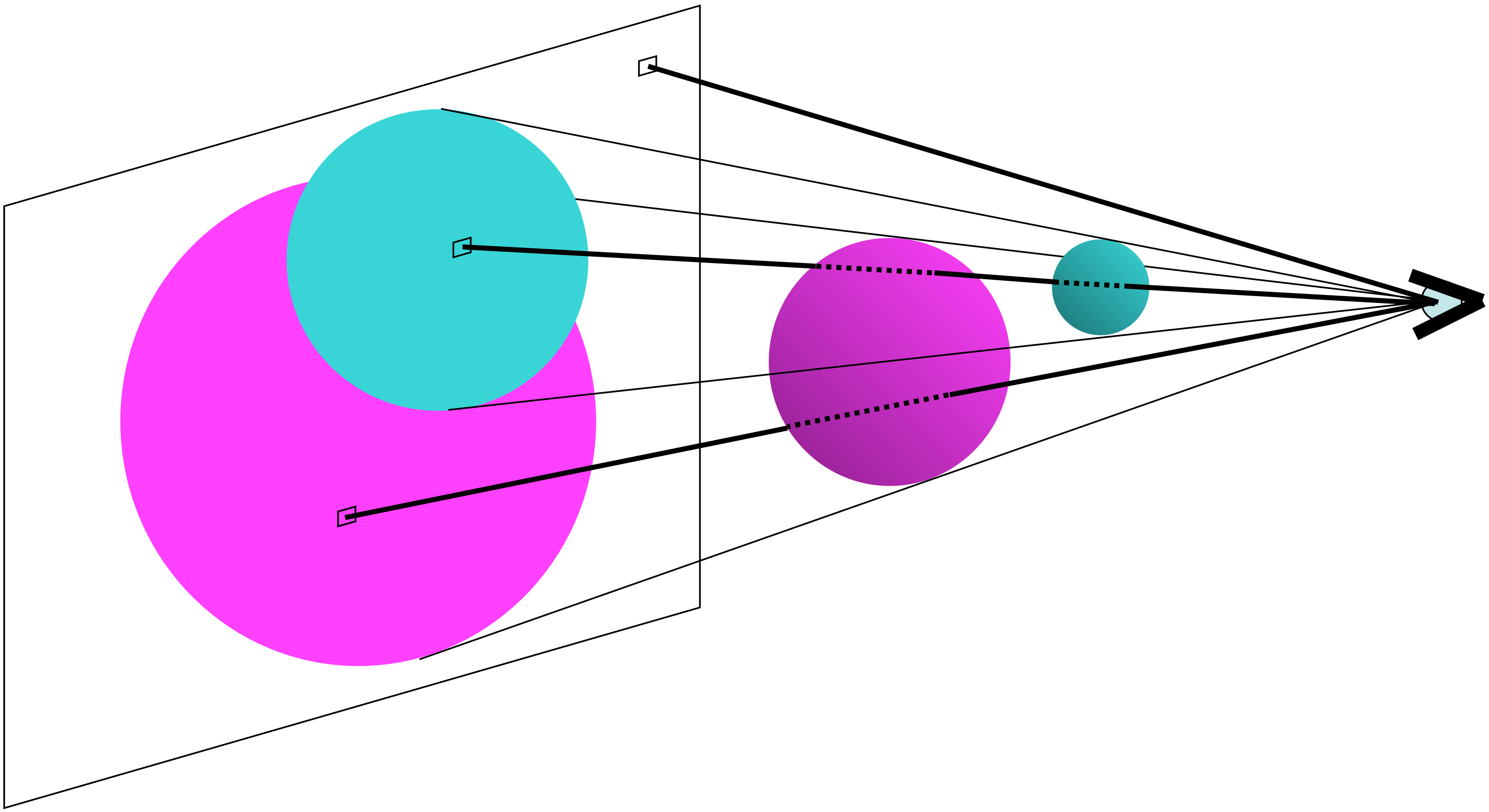
```
Vector3D VP;                    // the viewpoint
int numLights;
Vector3D theLights[5];          // up to 5 white lights
double ka;                      // ambient light coefficient
int numSpheres;
Sphere theSpheres[20];          // 20 sphere max

int ppmT[3];                    // ppm texture files
View sceneView;                 // transform data
double distance;                // view plane to VP
bool antialias;                 // if true antialias
```

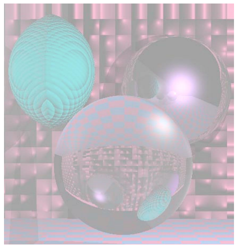
Simple Ray Tracing for Detecting Visible Surfaces

```
select window on viewplane and center of projection
for (each scanline in image) {
    for (each pixel in the scanline) {
        determine ray from center of projection
                through pixel;
        for (each object in scene) {
                if (object is intersected and
                    is closest considered thus far)
                        record intersection and object name;
        }
        set pixel's color to that of closest object intersected;
    }
}
```



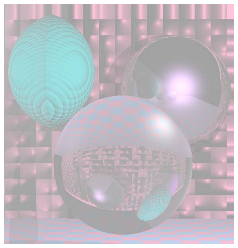
Ray Trace 1
Finding Visible Surfaces



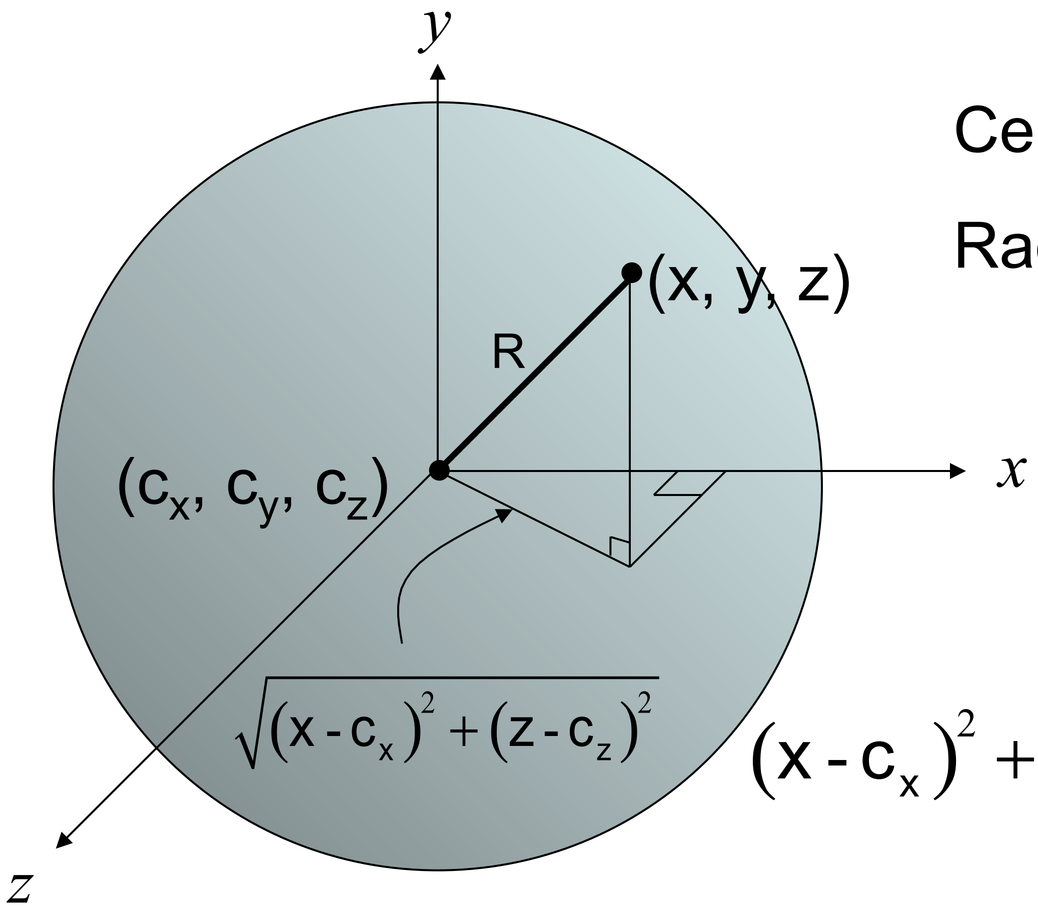


Ray-Sphere Intersection

- Given
  - Sphere
    - Center (c_x, c_y, c_z)
    - Radius, R
  - Ray from P_0 to P_1
    - $P_0 = (x_0, y_0, z_0)$ and $P_1 = (x_1, y_1, z_1)$
  - View Point
    - (V_x, V_y, V_z)
- Project to window from $(0,0,0)$ to $(w,h,0)$



Sphere Equation



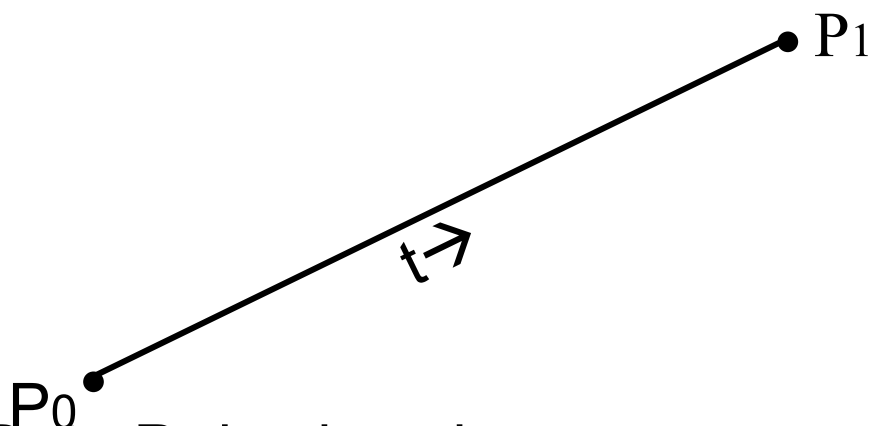
Center C = (c_x, c_y, c_z)

Radius R

$$(x - c_x)^2 + (y - c_y)^2 + (z - c_z)^2 = R^2$$

Ray Equation

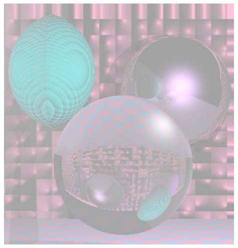
$P_0 = (x_0, y_0, z_0)$ and $P_1 = (x_1, y_1, z_1)$



The ray from P_0 to P_1 is given by:

$P(t) = (1 - t)P_0 + tP_1 \qquad 0 <= t <= 1$

$\quad = P_0 + t(P_1 - P_0)$



Intersection Equation

$P(t) = P_0 + t(P_1 - P_0)$ $\qquad 0 <= t <= 1$

is really three equations

$x(t) = x_0 + t(x_1 - x_0)$

$y(t) = y_0 + t(y_1 - y_0)$

$z(t) = z_0 + t(z_1 - z_0)$ $\qquad 0 <= t <= 1$

Substitute x(t), y(t), and z(t) for x, y, z, respectively in

$$(x - c_x)^2 + (y - c_y)^2 + (z - c_z)^2 = R^2$$

$$((x_0 + t(x_1 - x_0)) - c_x)^2 + ((y_0 + t(y_1 - y_0)_1) - c_y)^2 + ((z_0 + t(z_1 - z_0)) - c_z)^2 = R^2$$

Solving the Intersection Equation

$$\left(\left(x_0 + t(x_1 - x_0)\right) - c_x\right)^2 + \left(\left(y_0 + t(y_1 - y_0)_1\right) - c_y\right)^2 + \left(\left(z_0 + t(z_1 - z_0)\right) - c_z\right)^2 = R^2$$

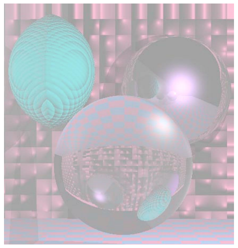
is a quadratic equation in variable t.

For a fixed pixel, VP, and sphere,

x_0, y_0, z_0, x_1, y_1, z_1, c_x, c_y, c_z, and R

are all constants.

We solve for t using the quadratic formula.



The Quadratic Coefficients

$$\left(\left(x_0 + t\left(x_1 - x_0\right)\right) - c_x\right)^2 + \left(\left(y_0 + t\left(y_1 - y_0\right)_1\right) - c_y\right)^2 + \left(\left(z_0 + t\left(z_1 - z_0\right)\right) - c_z\right)^2 = R^2$$

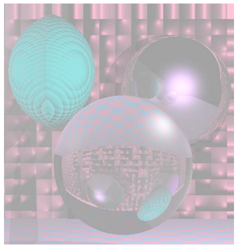
Set $d_x = x_1 - x_0$

$d_y = y_1 - y_0$

$d_z = z_1 - z_0$

Now find the the coefficients:

$$At^2 + Bt + C = 0$$



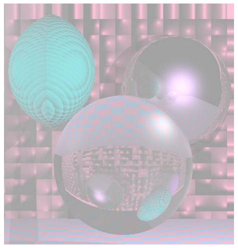
Computing Coefficients

$$\left(\left(x_0 + t(x_1 - x_0)\right) - c_x\right)^2 + \left(\left(y_0 + t(y_1 - y_0)\right) - c_y\right)^2 + \left(\left(z_0 + t(z_1 - z_0)\right) - c_z\right)^2 = R^2$$

$$\left((x_0 + td_x) - c_x\right)^2 + \left((y_0 + td_y) - c_y\right)^2 + \left(((z_0 + td_z) - c_z\right)^2 = R^2$$

$$(x_0 + td_x)^2 - 2c_x(x_0 + td_x) + c_x{}^2 +$$
$$(y_0 + td_y)^2 - 2c_y(y_0 + td_y) + c_y{}^2 +$$
$$(z_0 + td_z)^2 - 2c_z(z_0 + td_z) + c_z{}^2 - R^2 = 0$$

$$x_0{}^2 + 2x_0td_x + t^2d_x{}^2 - 2c_xx_0 - 2c_xtd_x + c_x{}^2 +$$
$$y_0{}^2 + 2y_0td_y + t^2d_y{}^2 - 2c_yy_0 - 2c_ytd_y + c_y{}^2 +$$
$$z_0{}^2 + 2z_0td_z + t^2d_z{}^2 - 2c_zz_0 - 2c_ztd_z + c_z{}^2 - R^2 = 0$$



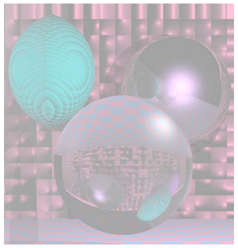
The Coefficients

$$x_0^2 + 2x_0td_x + t^2d_x^2 - 2c_xx_0 - 2c_xtd_x + c_x^2 +$$
$$y_0^2 + 2y_0td_y + t^2d_y^2 - 2c_yy_0 - 2c_ytd_y + c_y^2 +$$
$$z_0^2 + 2z_0td_z + t^2d_z^2 - 2c_zz_0 - 2c_ztd_z + c_z^2 - R^2 = 0$$

$$A = d_x^2 + d_y^2 + d_z^2$$

$$B = 2d_x(x_0 - c_x) + 2d_y(y_0 - c_y) + 2d_z(z_0 - c_z)$$

$$C = c_x^2 + c_y^2 + c_z^2 + x_0^2 + y_0^2 + z_0^2 + -2(c_xx_0 + c_yy_0 + c_zz_0) - R^2$$



Solving the Equation

$$At^2 + Bt + C = 0$$

$$\text{discriminant} = D(A,B,C) = B^2 - 4AC$$

$$D(A,B,C)\begin{cases} < 0 & \text{no intersection} \\ = 0 & \text{ray is tangent to the sphere} \\ > 0 & \text{ray intersects sphere in two points} \end{cases}$$

The intersection nearest P_0 is given by:

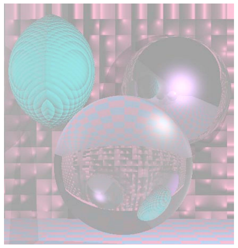
$$t = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

To find the coordinates of the intersection point:

$$x = x_0 + td_x$$

$$y = y_0 + td_y$$

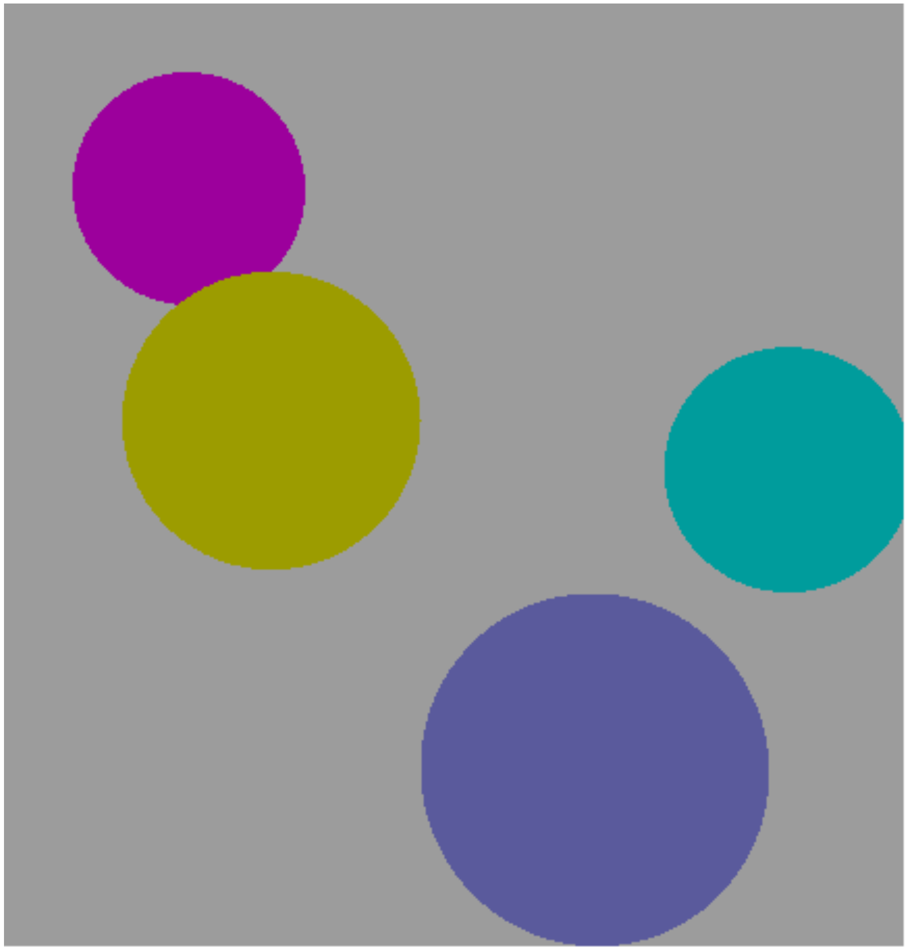
$$z = z_0 + td_z$$



First Lighting Model

- Ambient light is a global constant.
  - Ambient Light = $k_a (A_R, A_G, A_B)$
  - k_a is in the “World of Spheres”
  - $0 \le k_a \le 1$
  - (A_R, A_G, A_B) = average of the light sources
    - $(A_R, A_G, A_B) = (1, 1, 1)$ for white light
- Color of object $S = (S_R, S_G, S_B)$
- Visible Color of an object S with only ambient light
  - $C_S = k_a (A_R S_R, A_G S_G, A_B S_B)$
- For white light
  - $C_S = k_a (S_R, S_G, S_B)$

Visible Surfaces Ambient Light



Text

```
View Point: (200, 200,1000)
Light     : (750,   0,2000)


SPHERES

Center :   (100, 100,100)
Radius :      50
RED:       0.50
GREEN:     0.00
BLUE:      0.50

Center :   (150, 200,300)
Radius :      50
RED:       0.50
GREEN:     0.50
BLUE:      0.00

Center :   (350, 220,150)
Radius :      50
RED:       0.00
GREEN:     0.50
BLUE:      0.50

Center :   (250, 300,400)
Radius :      50
RED:       0.25
GREEN:     0.25
BLUE:      0.50

Only ambient light
```

Second Lighting Model

- Point source light $L = (L_R, L_G, L_B)$ at (L_x, L_y, L_z)
- Ambient light is also present.
- Color at **point p** on an object S with ambient & diffuse reflection

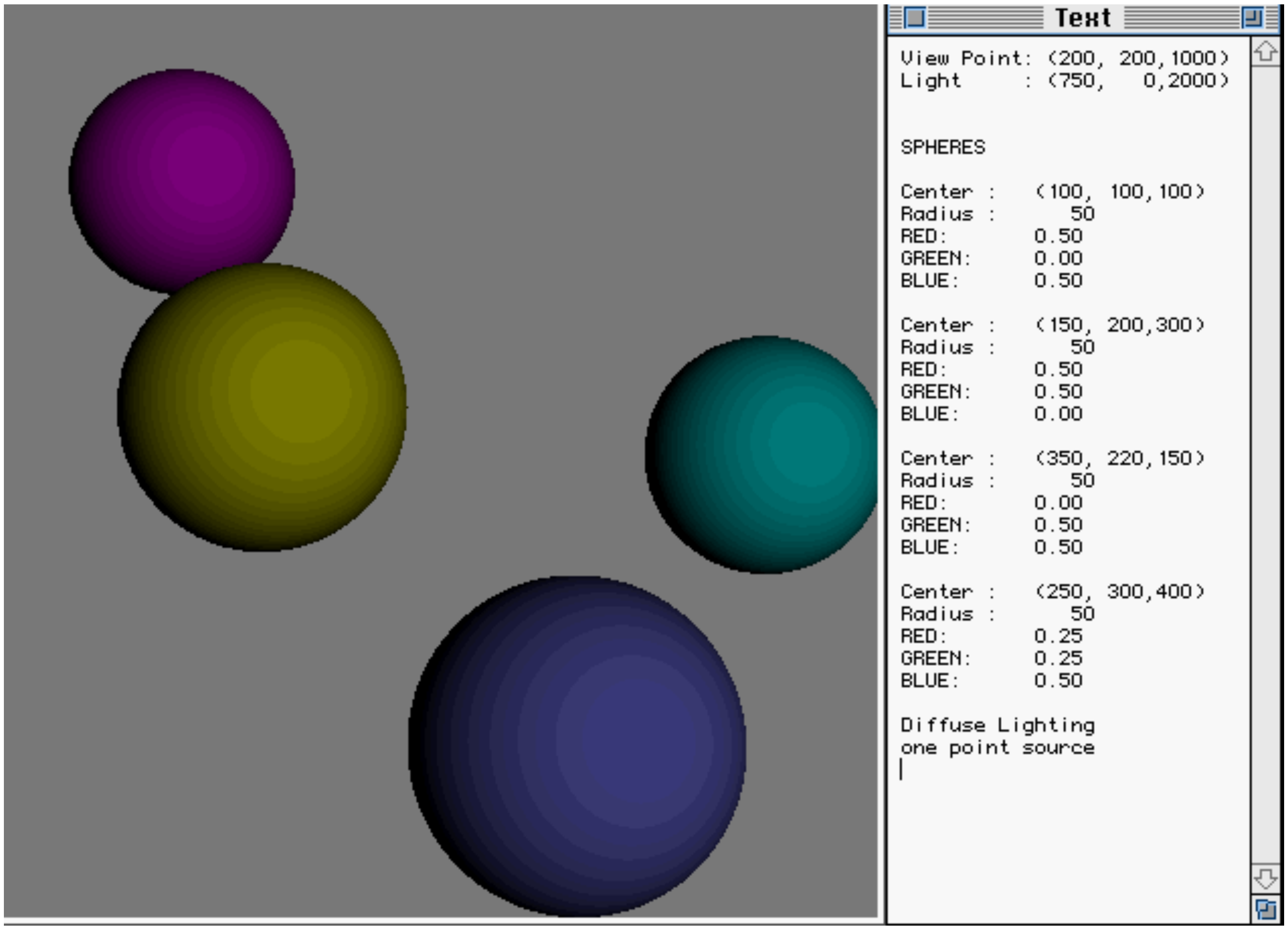
$$C_p = k_a (A_R S_R, A_G S_G, A_B S_B) + k_d k_p (L_R S_R, L_G S_G, L_B S_B)$$

- For white light, $L = (1, 1, 1)$

$$C_p = k_a (S_R, S_G, S_B) + k_d k_p (S_R, S_G, S_B)$$

- k_p depends on the **point p** on the object and (L_x, L_y, L_z)
- k_d depends on the object (sphere)
- k_a is global
- $k_a + k_d \leq 1$

Diffuse Light

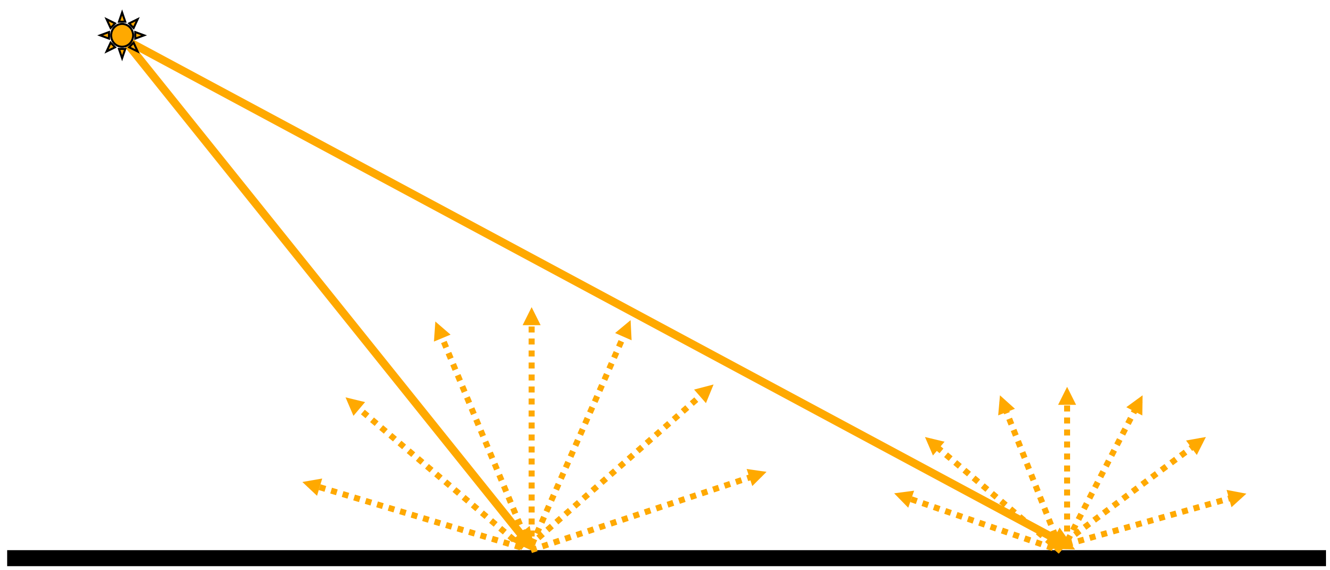


Lambertian Reflection Model
Diffuse Shading

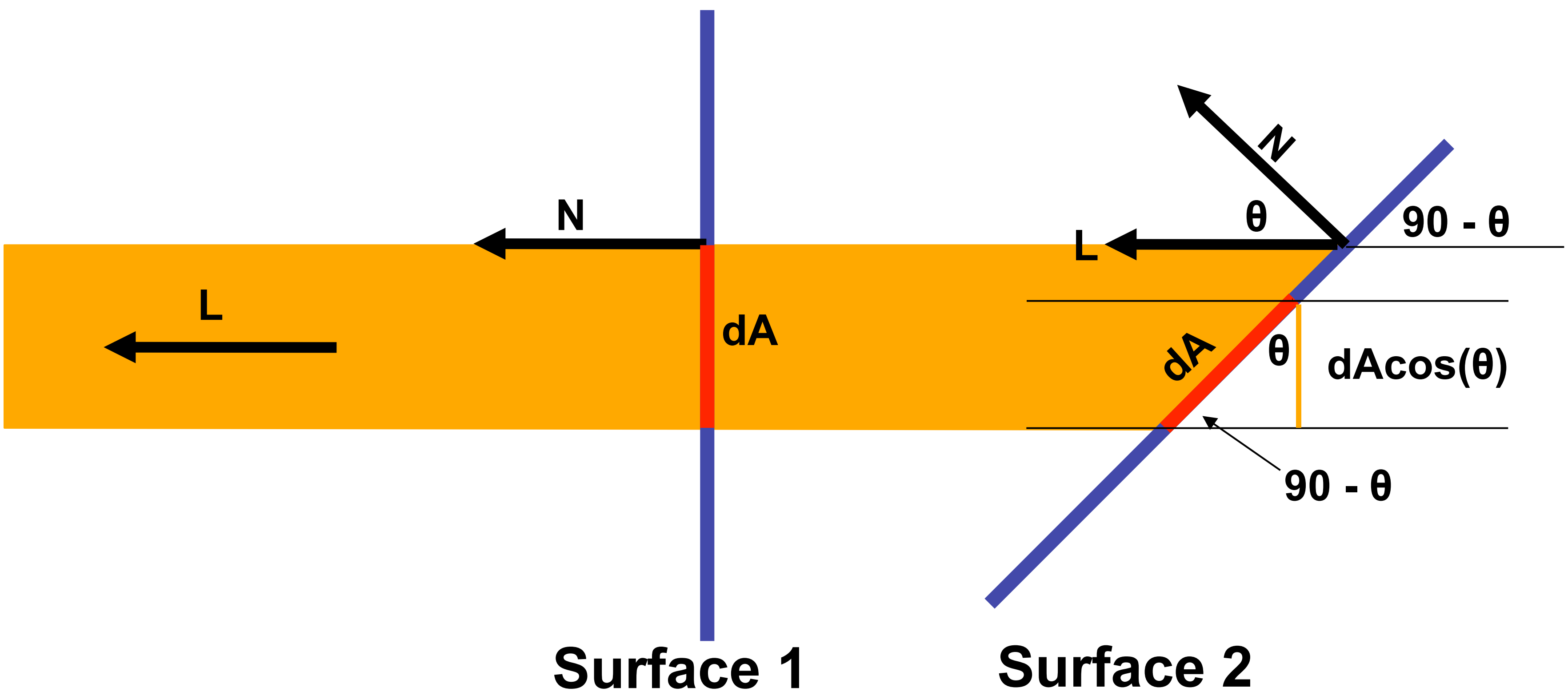
- For matte (non-shiny) objects
- Examples
  - Matte paper, newsprint
  - Unpolished wood
  - Unpolished stones
- Color at a point on a matte object does not change with viewpoint.

Physics of Lambertian Reflection

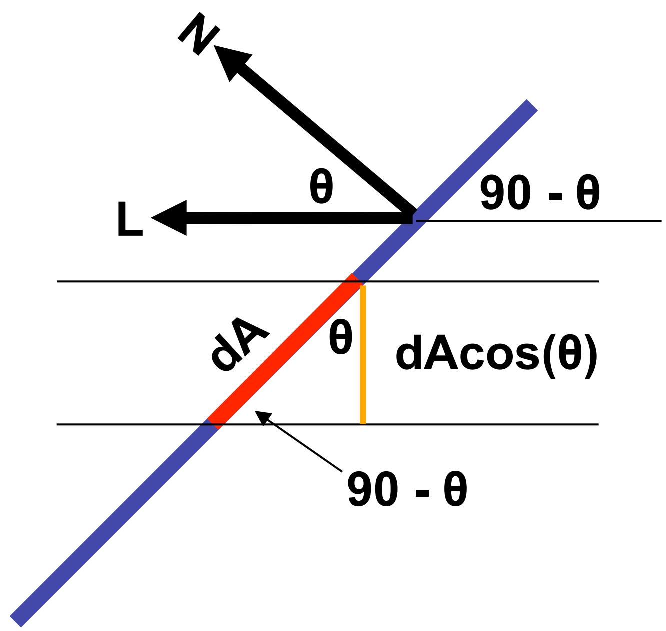
- Incoming light is partially absorbed and partially transmitted equally in all directions



Geometry of Lambert's Law



$\cos(\theta) = N \bullet L$

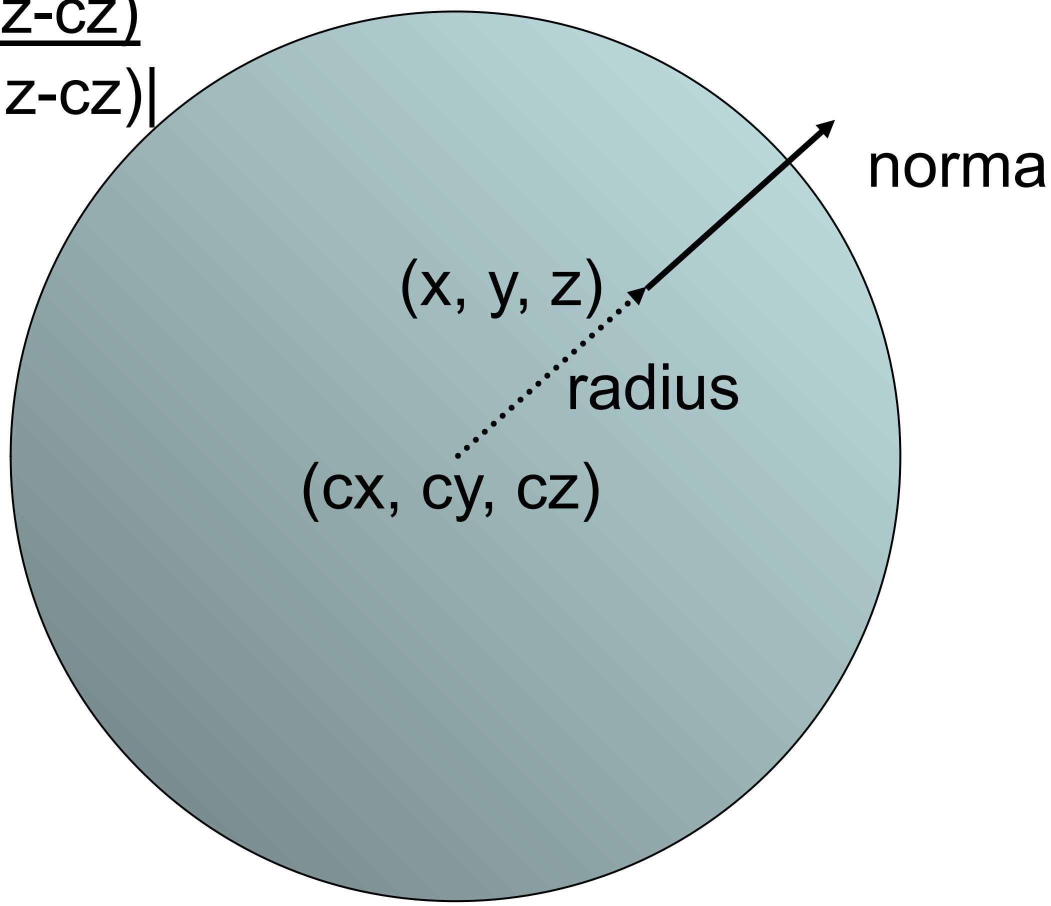


Surface 2

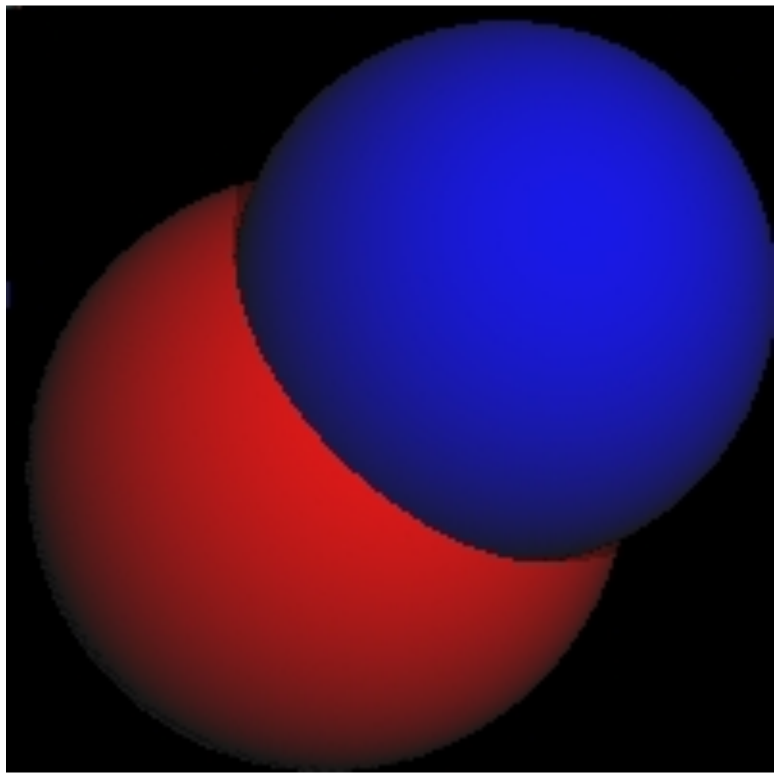
$$Cp = ka\ (SR, SG, SB) + kd\ \mathbf{N \bullet L}\ (SR, SG, SB)$$

Finding N

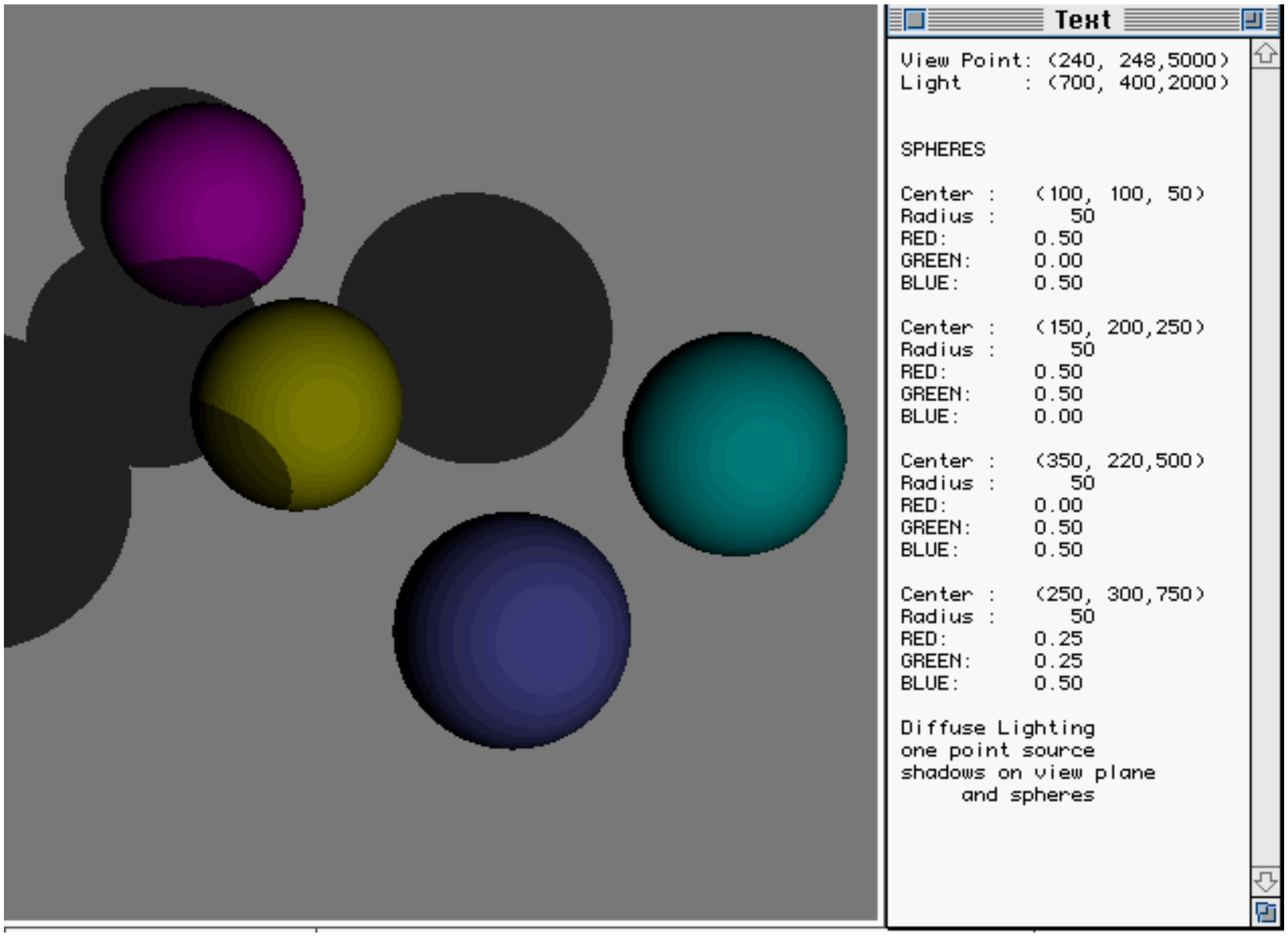
$$\mathbf{N} = \frac{(x-cx,\ y-cy,\ z-cz)}{|(x-cx,\ y-cy,\ z-cz)|}$$



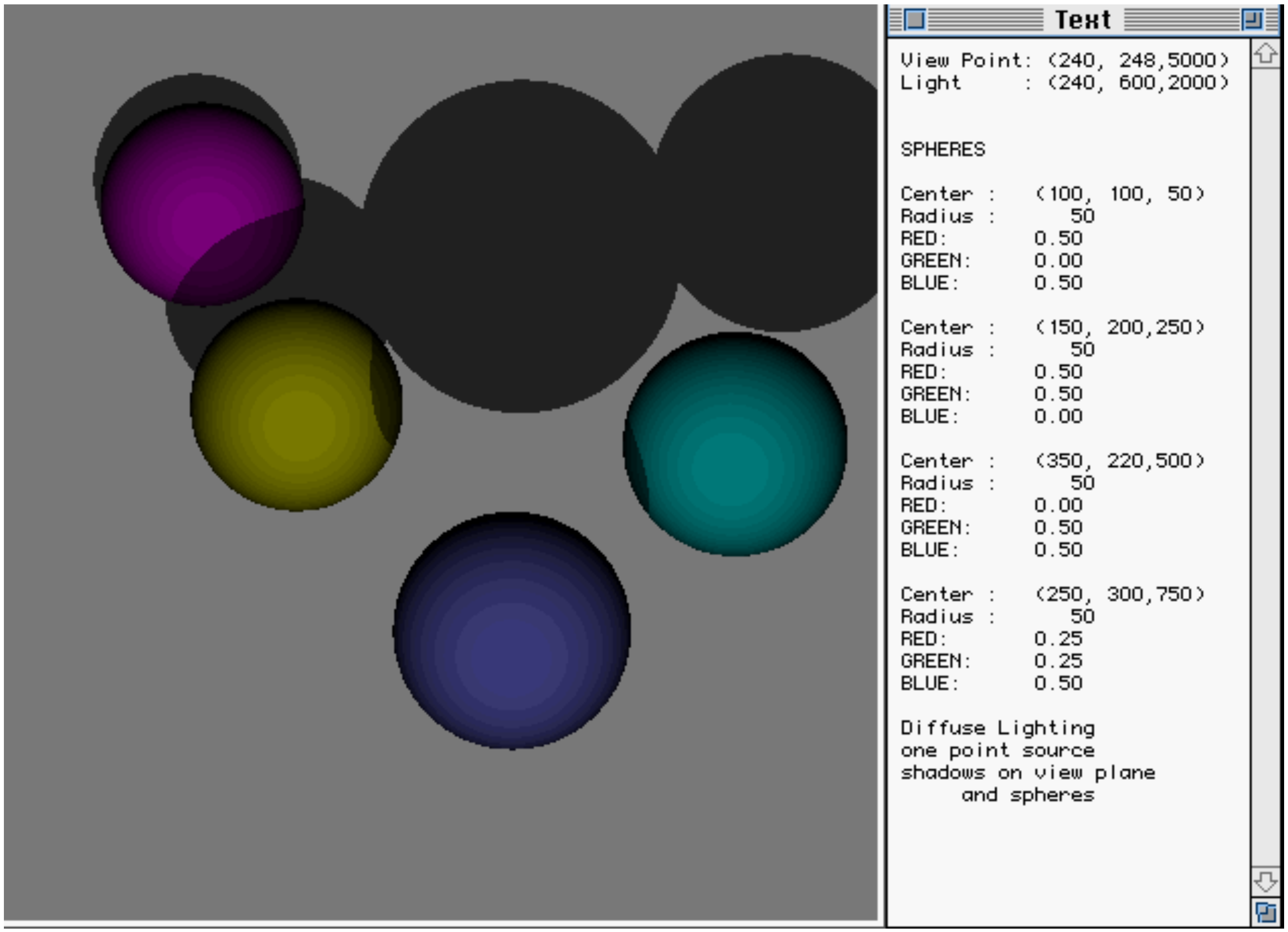
Diffuse Light 2



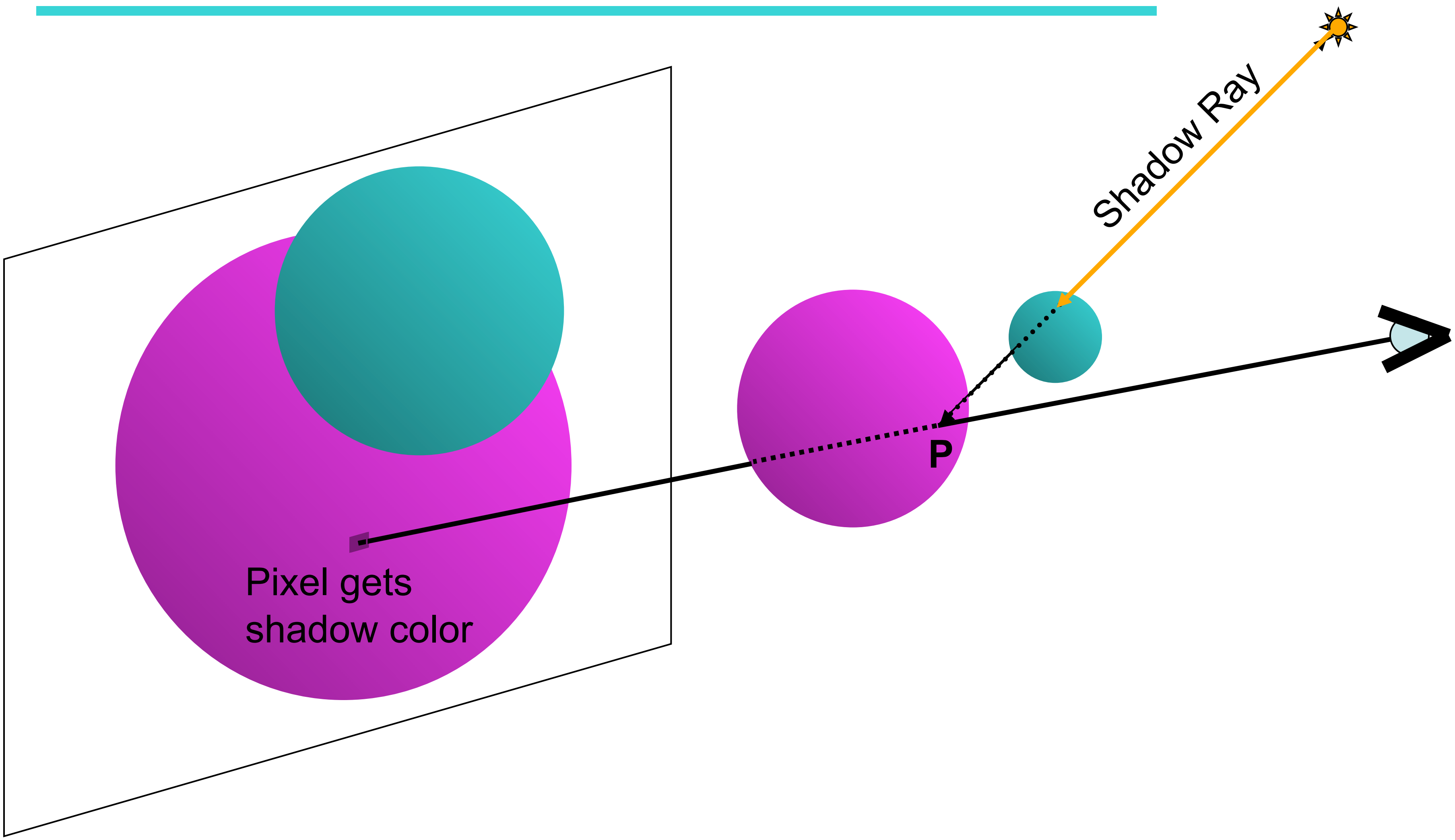
Shadows on Spheres



More Shadows



Finding Shadows



Shadow Color

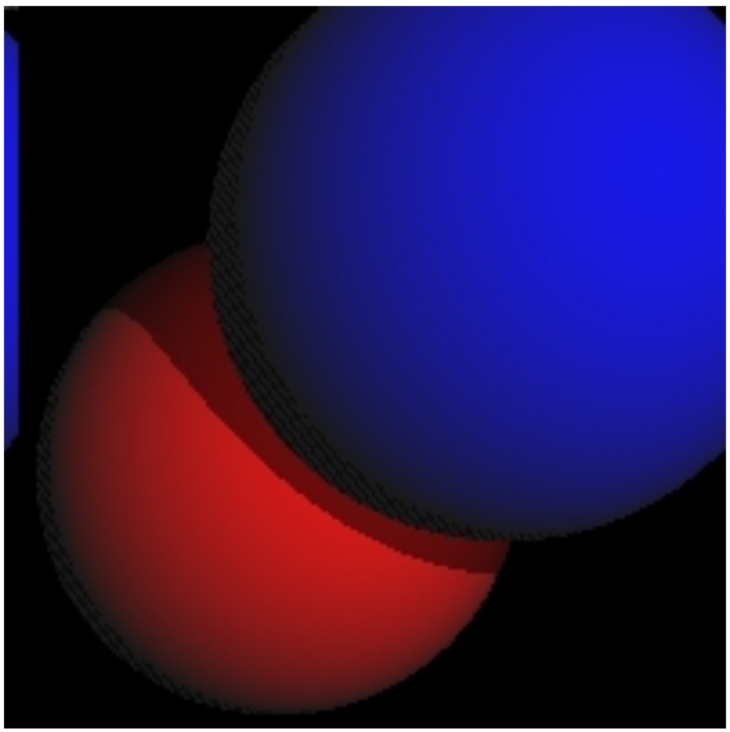
- Given

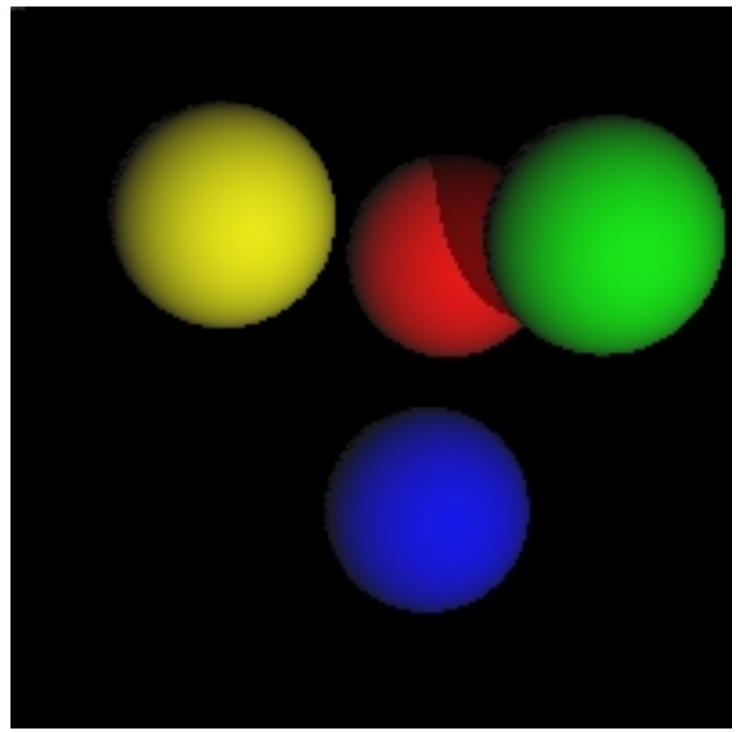
  Ray from P (point on sphere S) to L (light)

  $P = P_0 = (x_0, y_0, z_0)$ and $L = P_1 = (x_1, y_1, z_1)$

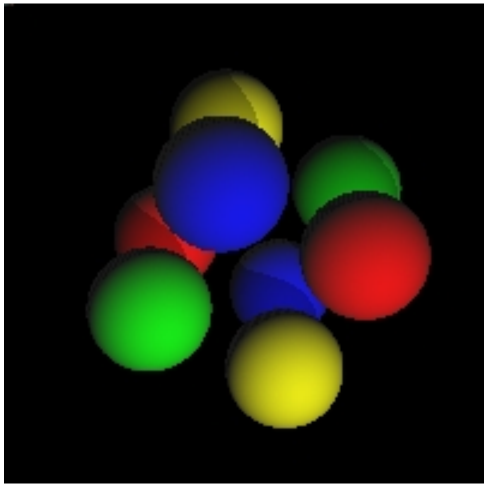
- Find out whether the ray intersects any other object (sphere).
  - If it does, P is in shadow.
  - Use only ambient light for pixel.

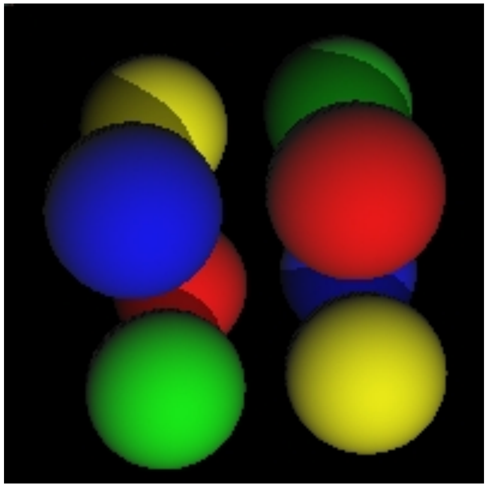
Shape of Shadows

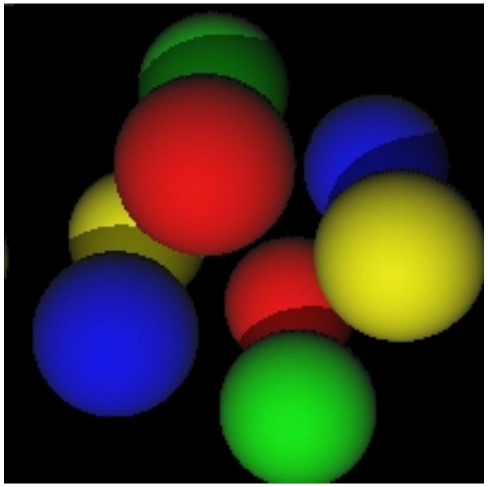


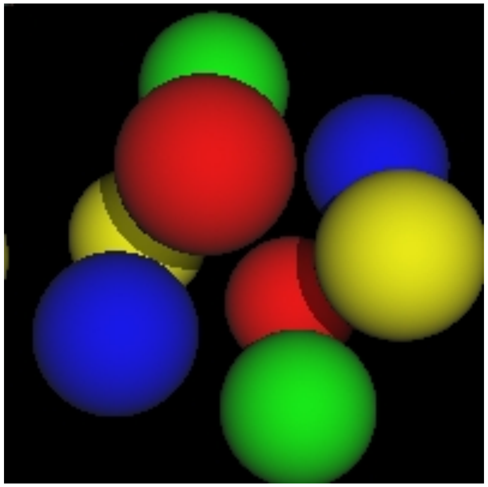


Different Views

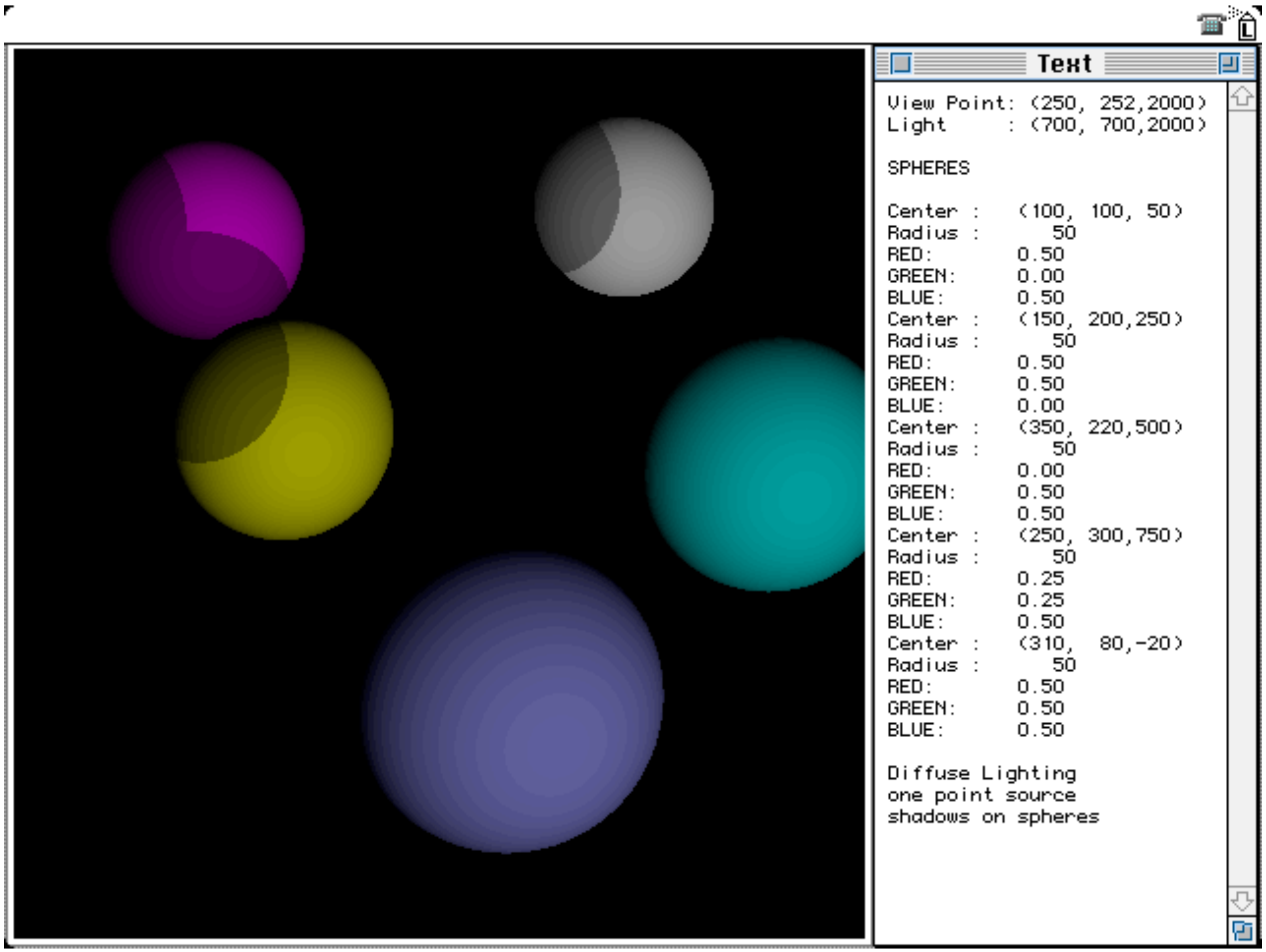




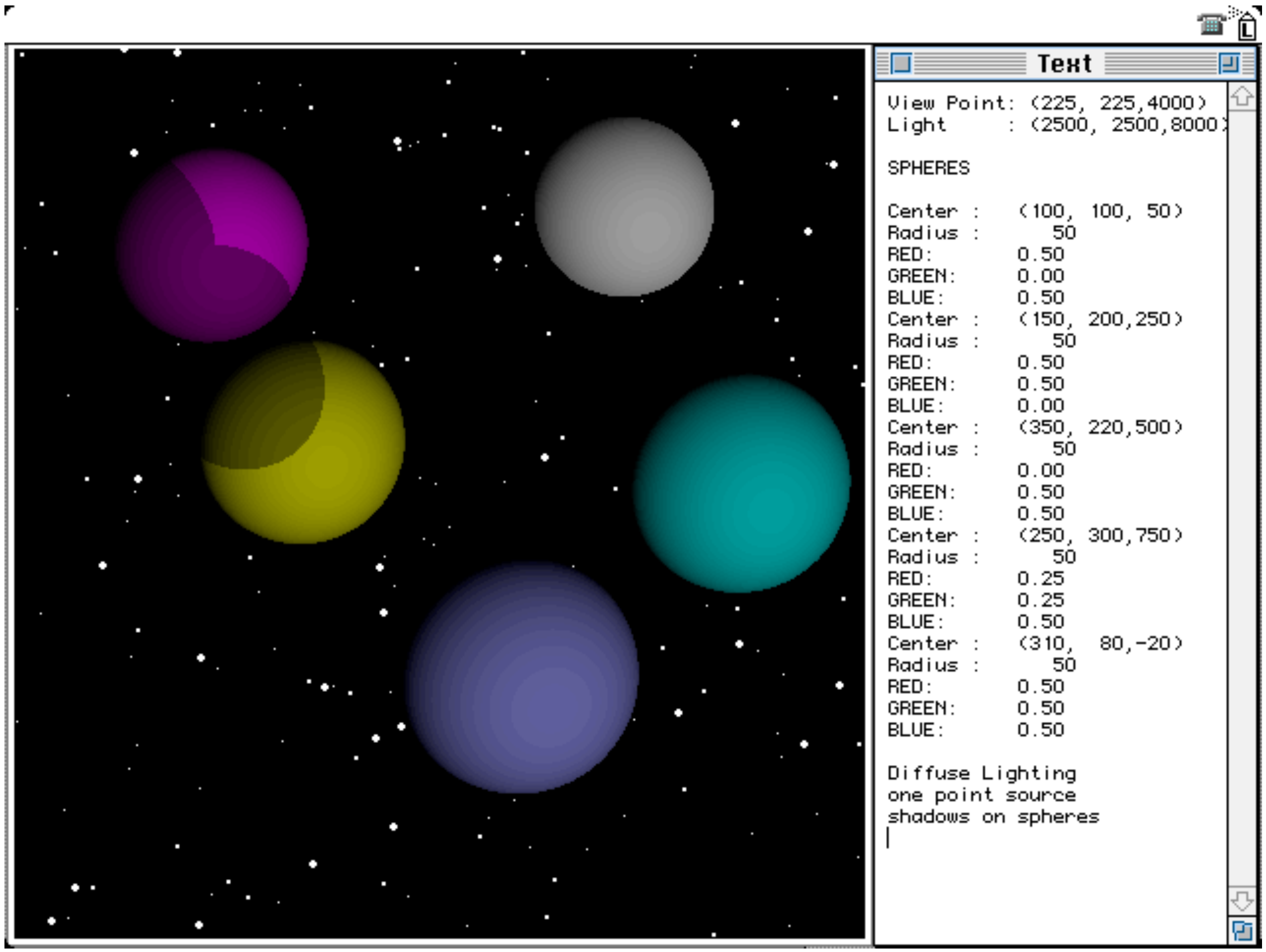




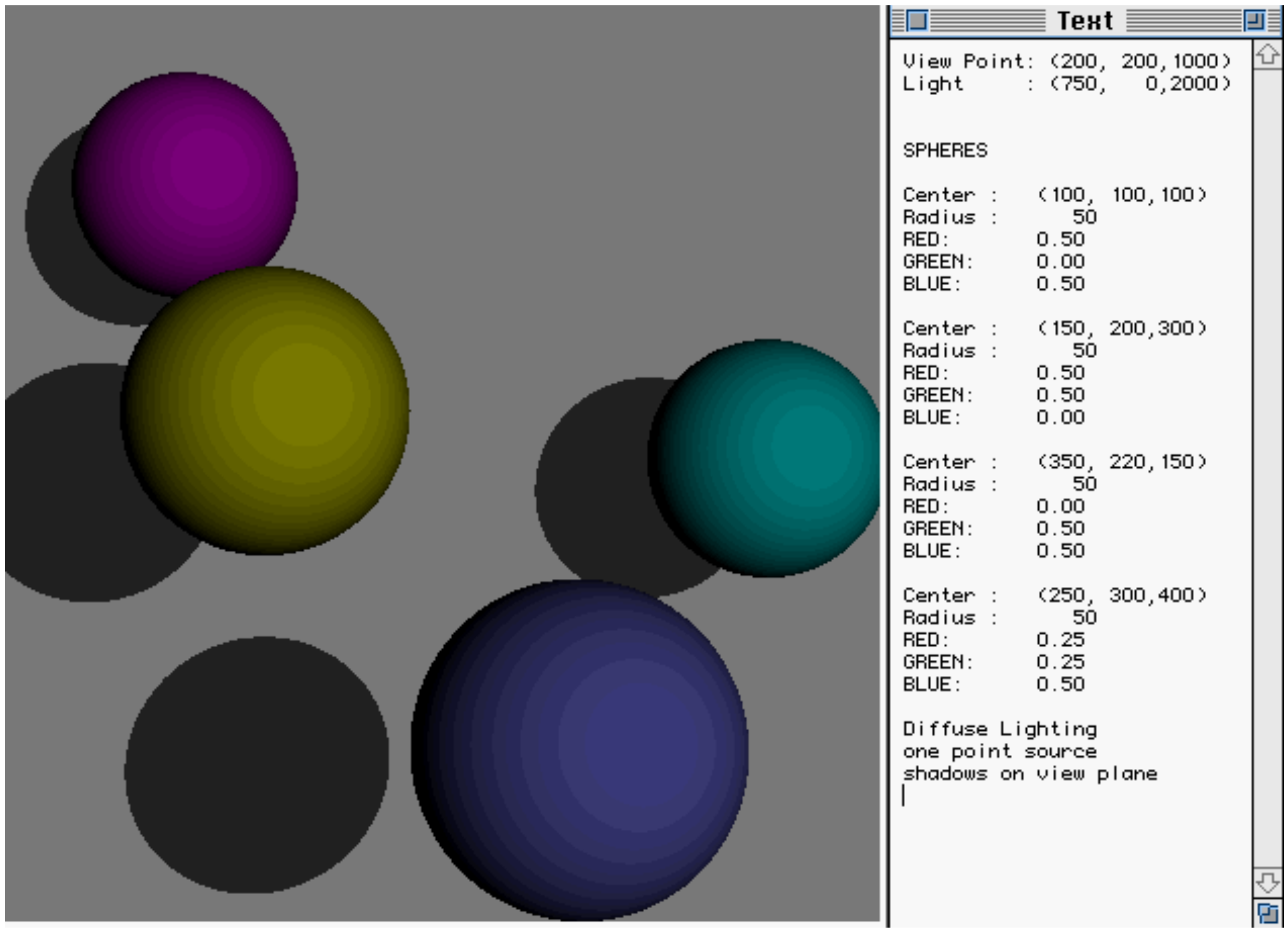
Planets


Text
View Point: (250, 252,2000)
Light : (700, 700,2000)
SPHERES
Center : (100, 100, 50)
Radius : 50
RED: 0.50
GREEN: 0.00
BLUE: 0.50
Center : (150, 200,250)
Radius : 50
RED: 0.50
GREEN: 0.50
BLUE: 0.00
Center : (350, 220,500)
Radius : 50
RED: 0.00
GREEN: 0.50
BLUE: 0.50
Center : (250, 300,750)
Radius : 50
RED: 0.25
GREEN: 0.25
BLUE: 0.50
Center : (310, 80,-20)
Radius : 50
RED: 0.50
GREEN: 0.50
BLUE: 0.50
Diffuse Lighting
one point source
shadows on spheres

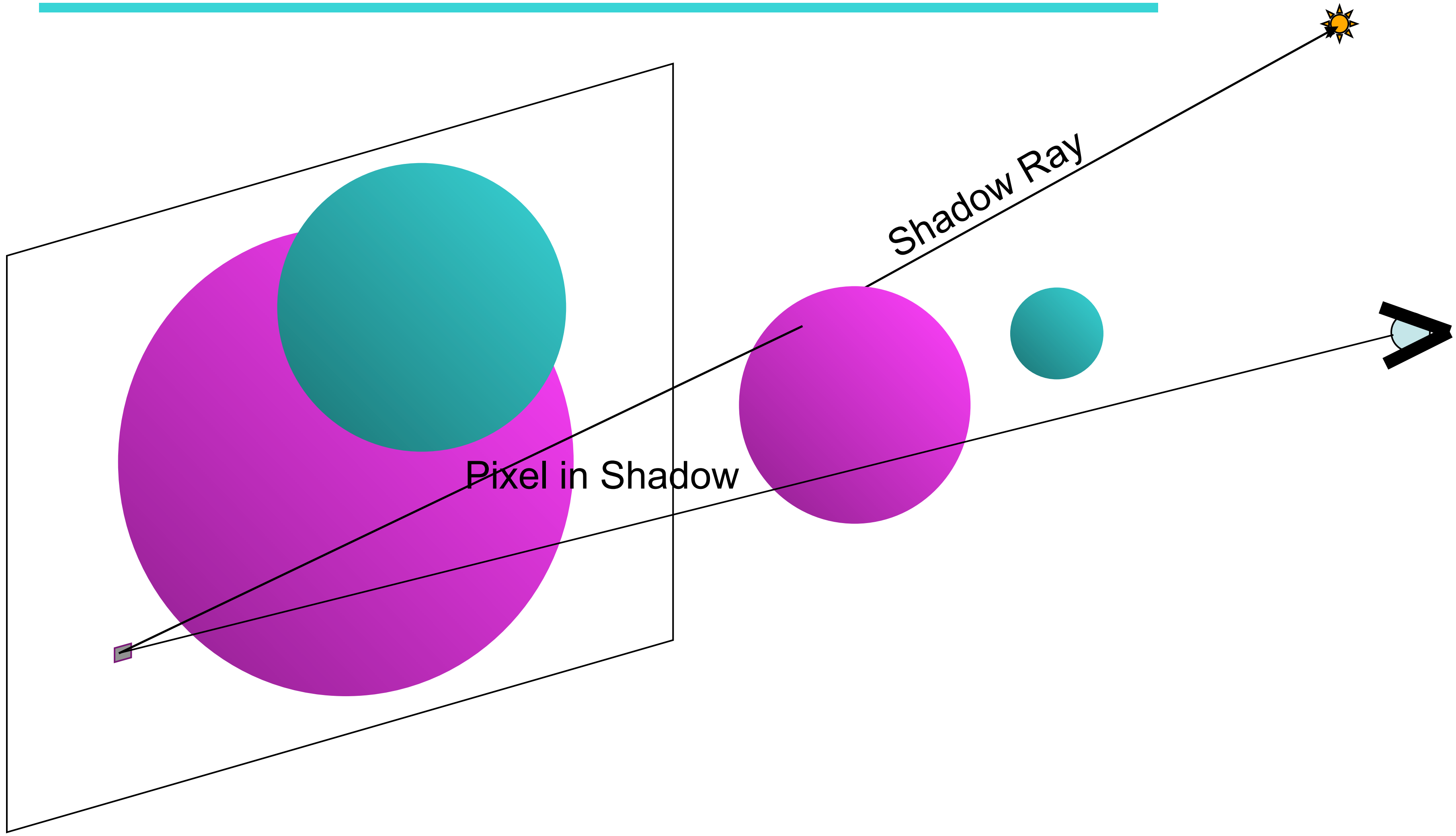
Starry Skies



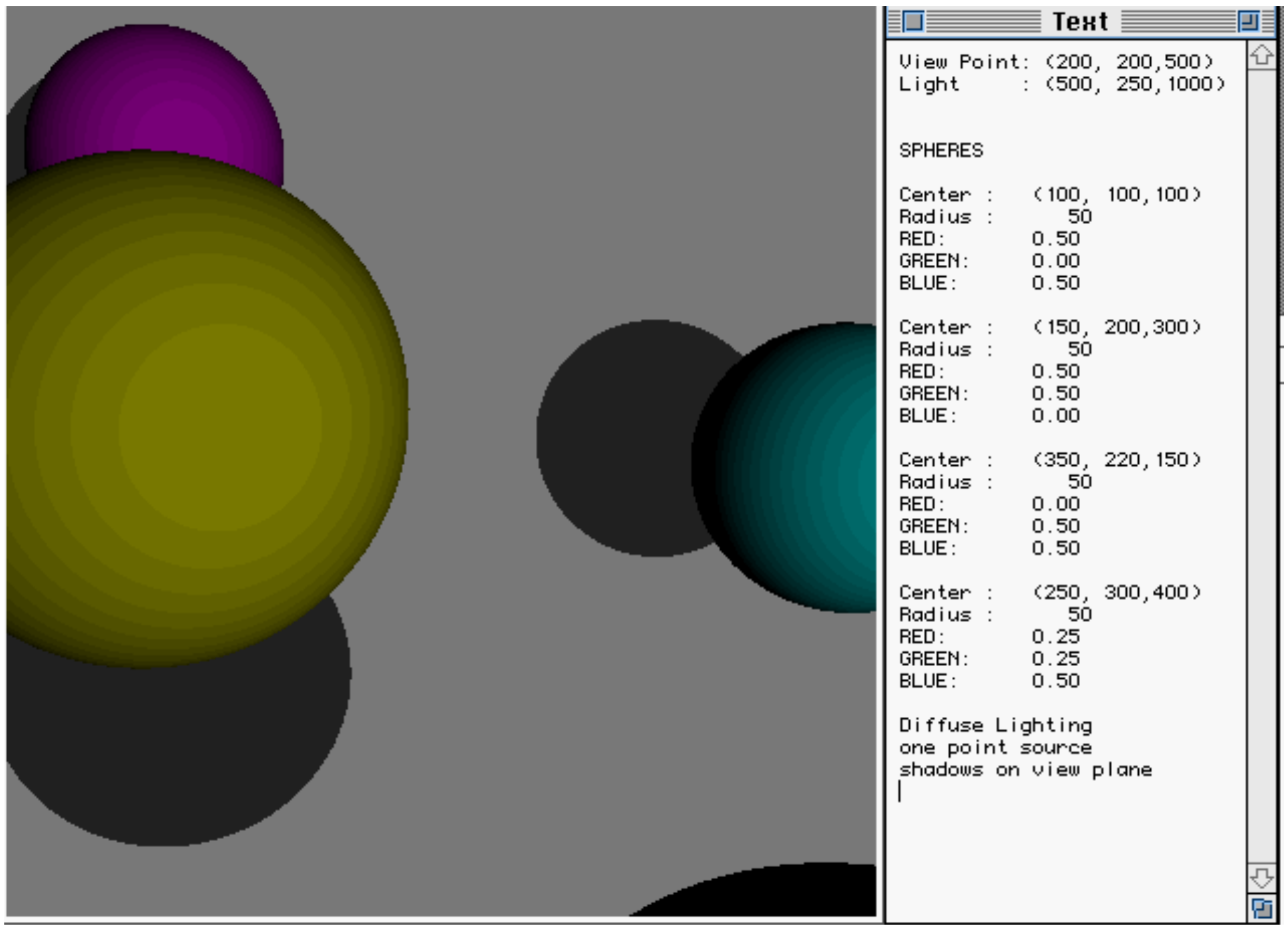
Shadows on the Plane



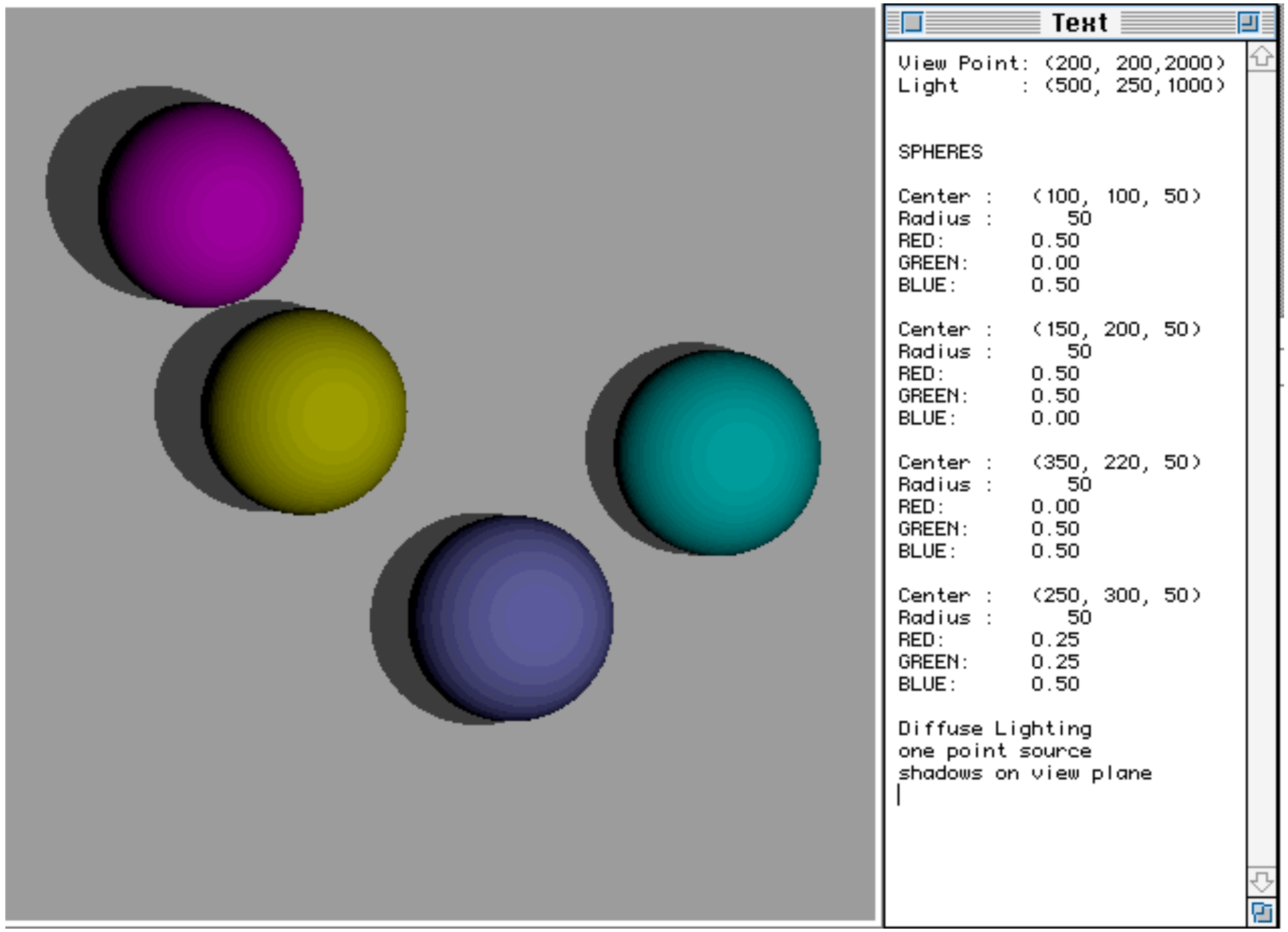
Finding Shadows on the Back Plane



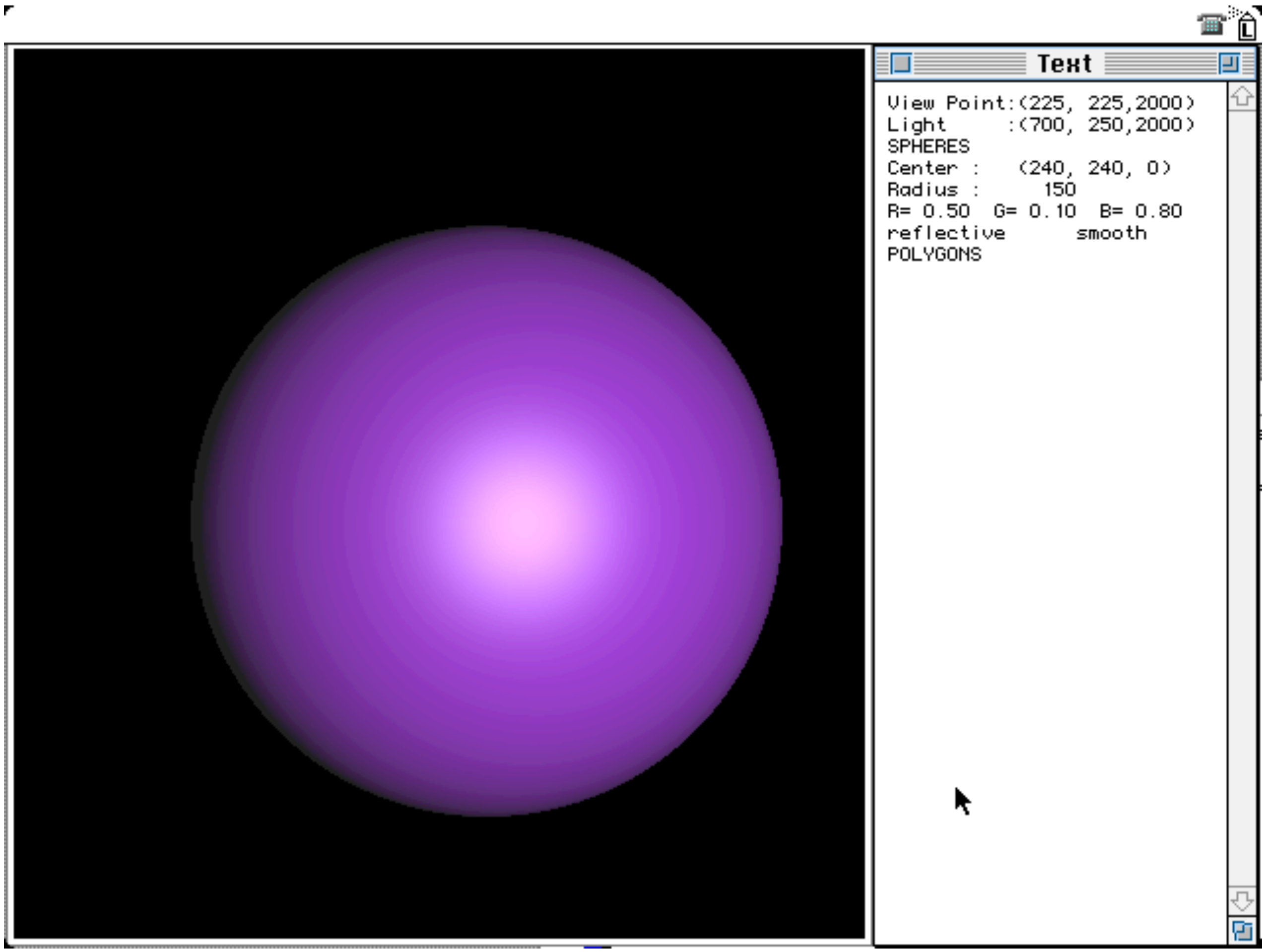
Close up

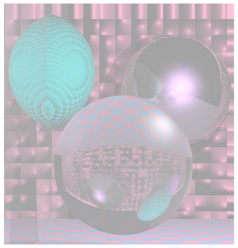

Text
View Point: (200, 200,500)
Light : (500, 250,1000)
SPHERES
Center : (100, 100,100)
Radius : 50
RED: 0.50
GREEN: 0.00
BLUE: 0.50
Center : (150, 200,300)
Radius : 50
RED: 0.50
GREEN: 0.50
BLUE: 0.00
Center : (350, 220,150)
Radius : 50
RED: 0.00
GREEN: 0.50
BLUE: 0.50
Center : (250, 300,400)
Radius : 50
RED: 0.25
GREEN: 0.25
BLUE: 0.50
Diffuse Lighting
one point source
shadows on view plane

On the Table


Text
View Point: (200, 200,2000)
Light : (500, 250,1000)
SPHERES
Center : (100, 100, 50)
Radius : 50
RED: 0.50
GREEN: 0.00
BLUE: 0.50
Center : (150, 200, 50)
Radius : 50
RED: 0.50
GREEN: 0.50
BLUE: 0.00
Center : (350, 220, 50)
Radius : 50
RED: 0.00
GREEN: 0.50
BLUE: 0.50
Center : (250, 300, 50)
Radius : 50
RED: 0.25
GREEN: 0.25
BLUE: 0.50
Diffuse Lighting
one point source
shadows on view plane

Phong Highlight





Phong Lighting Model

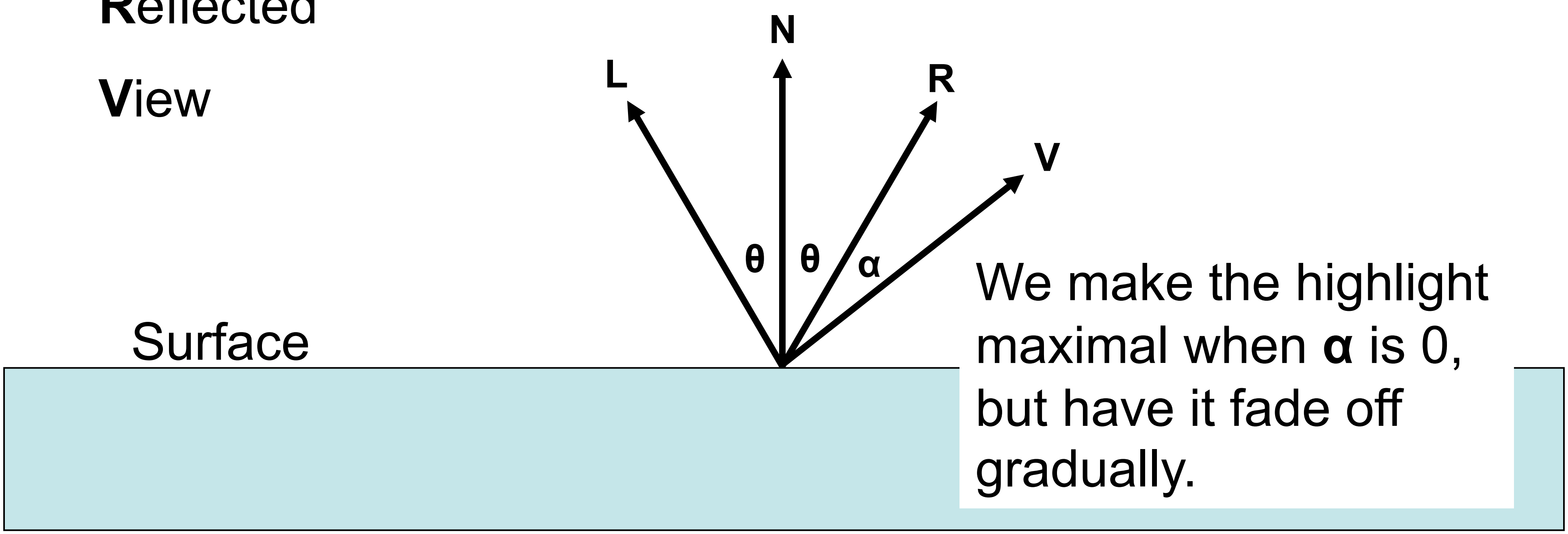
Light

Normal

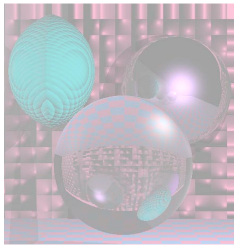
Reflected

View

The viewer only sees the light when **α** is 0.



We make the highlight maximal when **α** is 0, but have it fade off gradually.

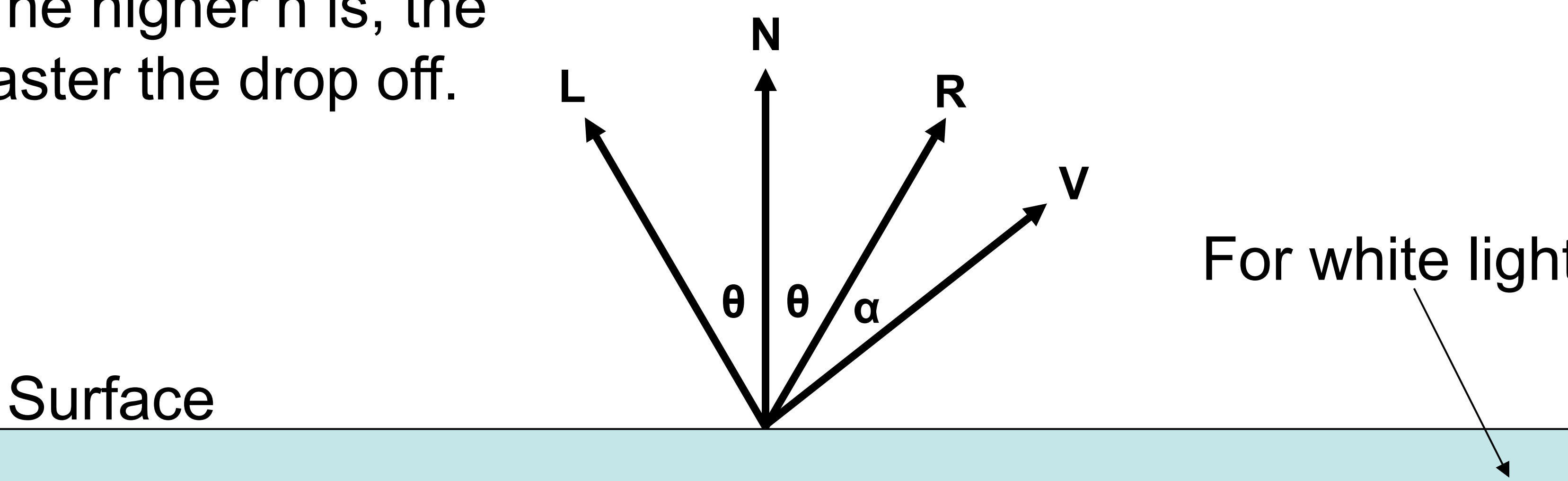


Phong Lighting Model

$\cos(\theta) = \mathbf{R} \bullet \mathbf{V}$

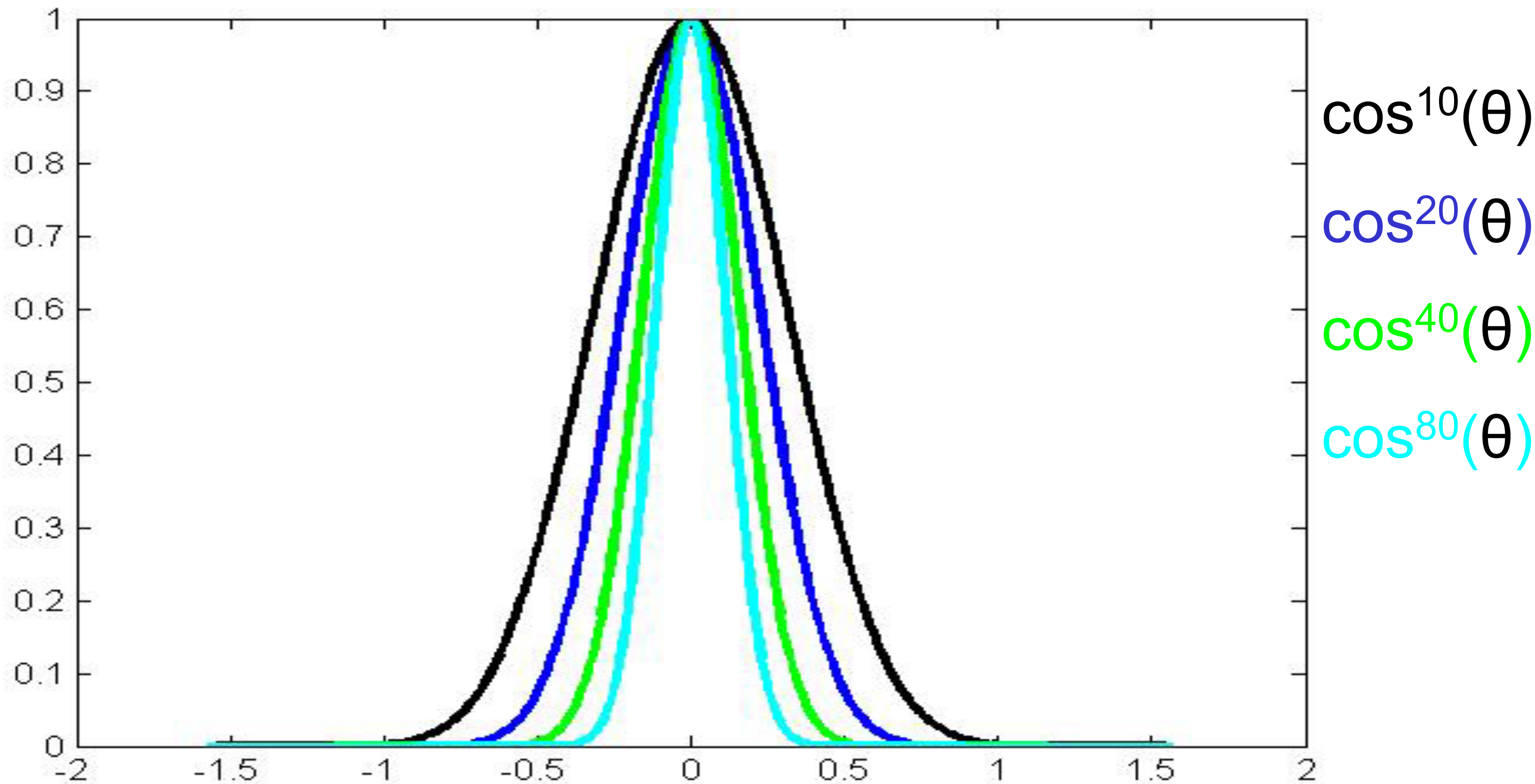
We use $\cos^n(\theta)$.

The higher n is, the faster the drop off.



$$Cp = ka\ (SR, SG, SB) + kd\ \mathbf{N} \bullet \mathbf{L}\ (SR, SG, SB) + ks\ (\mathbf{R} \bullet \mathbf{V})^n (1, 1, 1)$$

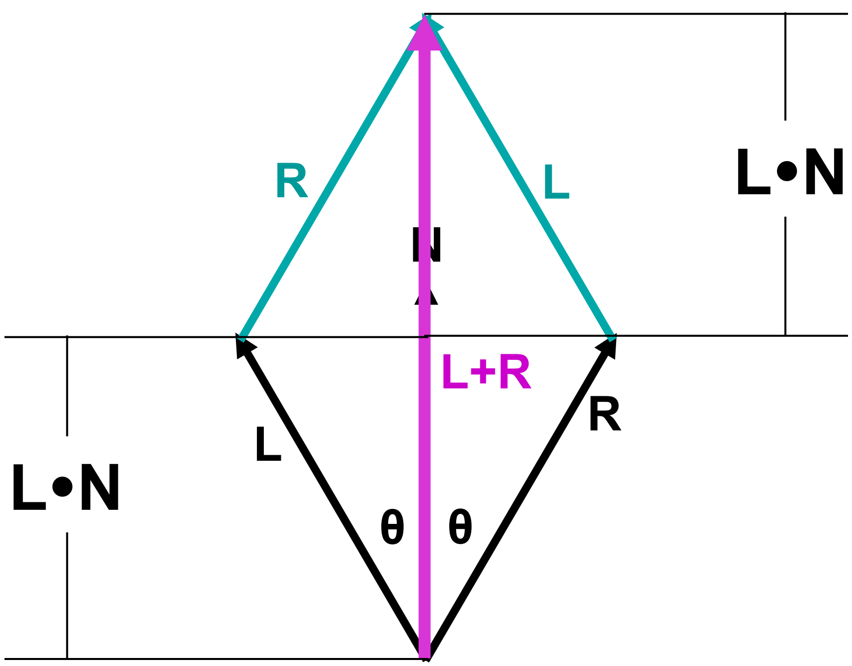
Powers of cos(θ)

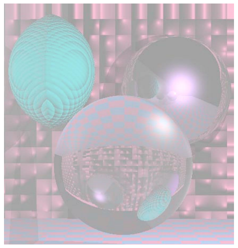


Computing **R**

L + **R** = (2 **L•N**) **N**

R = (2 **L•N**) **N** - **L**





The Halfway Vector

$$\mathbf{H} = \frac{\mathbf{L} + \mathbf{V}}{|\mathbf{L} + \mathbf{V}|}$$

Use **H•N**

instead of **R•V**.

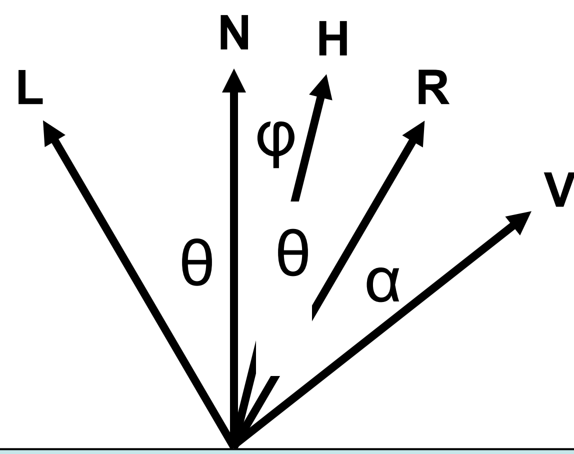
H is less expensive to compute than **R**.

From the picture

$\theta + \varphi = \theta - \varphi + \alpha$

So $\varphi = \alpha/2$.

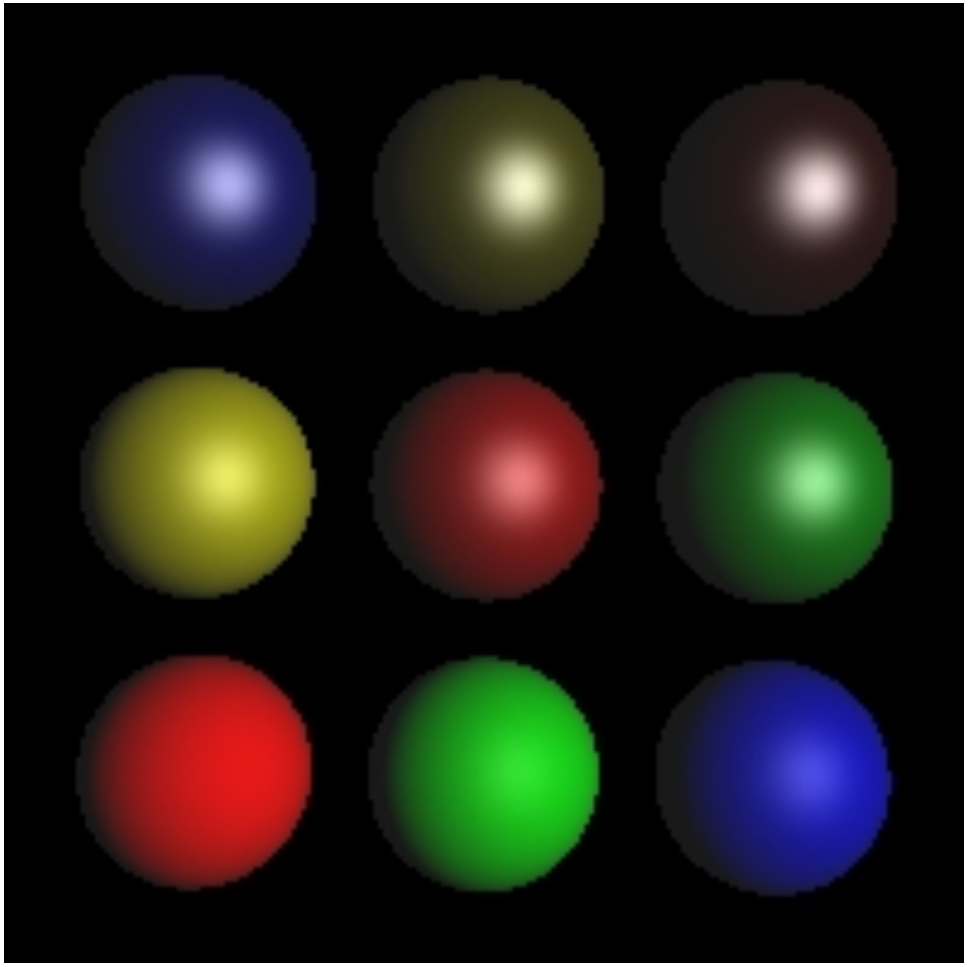
This is not generally true. Why?



Surface

Cp= ka (SR, SG, SB) + kd **N•L** (SR, SG, SB) + ks $(\mathbf{H} \bullet \mathbf{N})^n$ (1, 1, 1)

Varied Phong Highlights



Varying Reflectivity