

Lighting



Light types

- Ambient
- Point light
- Spot light
- Area light



Ambient Light

- Uniformly distributed throughout a scene in all directions.
- Fills areas that are not directly exposed to light and lightens shadows.
- A scene purely illuminated by ambient light, all surfaces receive the same amount of light.





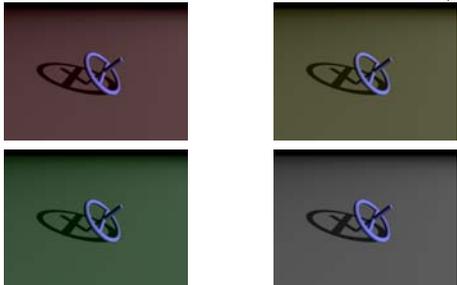
Ambient Light



```
global_settings {  
    ambient_light color  
}
```

This statement will set the default color for the ambient light.

Result



Point Light



- Radiates light outward from a single position and shines evenly in all directions.



Light Fading

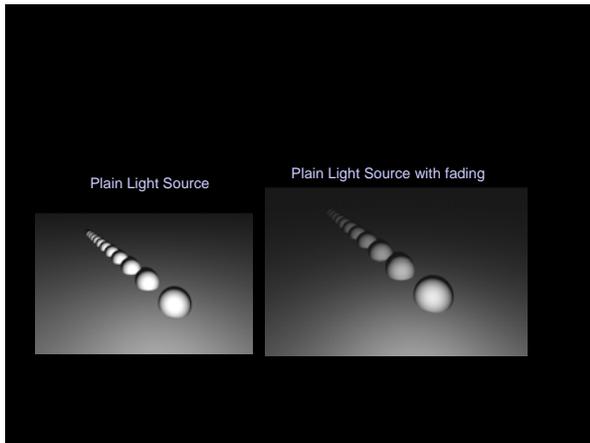


- It is not realistic for the plane to be evenly illuminated off into the distance.
- In real life, light gets scattered as it travels so it diminishes its ability to illuminate objects the farther it gets from its source.
- Objects closer to the light source get more light than the ones farther.

Light Fading



```
light_source {  
  <10,10,-10>  
  color White  
  fade_distance 10  
  fade_power 1  
}
```

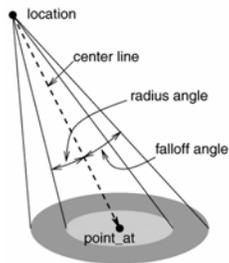


Spot Light



- A spotlight is a point that radiates light only in a specified direction.

```
light_source {  
  <x,y,z> color <colorspec>  
  spotlight  
  point_at <x,y,z>  
  radius <spotlight angle>  
  falloff <falloff angle>  
  [lightness tightness]  
}
```



Spot Light



- This has spotlight with the same radius and falloff angles

```
light_source {  
  <100, 100, -200>  
  color rgb <1, 1, 1>  
  spotlight  
  point_at <0, 5, 0>  
  radius 1.5  
  falloff 1.5  
}
```

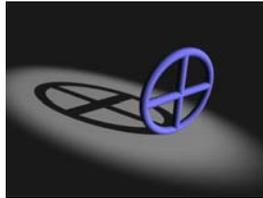


Spot Light



- This has spotlight with different radius and falloff angles

```
light_source {  
  <100, 100, -200>  
  color rgb <1, 1, 1>  
  spotlight  
  point_at <0, 5, 0>  
  radius 1.5  
  falloff 2.5  
}
```





Cylindrical Light



- Constant radius and falloff regardless of distance.
- A cylindrical light source is just like a spotlight, except that the radius and falloff regions are the same no matter how far from the light source our object is. The shape is therefore a cylinder rather than a cone.

Cylindrical Light



- This has spotlight with different radius and falloff angles

```
light_source {  
  <100, 100, -200>  
  color rgb <1, 1, 1>  
  cylinder  
  point_at <0, 5, 0>  
  radius 1.5  
  falloff 8  
}
```

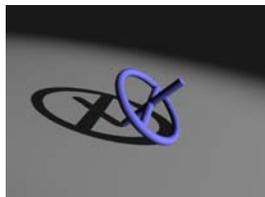


Spot Light



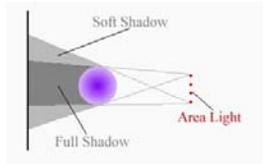
- This has spotlight with different radius and falloff angles

```
light_source {  
  <100, 100, -200>  
  color rgb <1, 1, 1>  
  spotlight  
  point_at <0, 5, 0>  
  radius 1.5  
  falloff 8  
}
```



Area Light

- More realism
- Spreads light intensity over a rectangle.





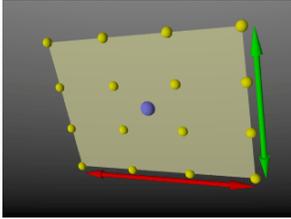
Area Light

```
area_light <side-1>, <side-2>,  
          len-1, len-2
```

- *<side-1>* and *<side-2>* describe orientation, length of rectangle. Should be perpendicular.
- *<len-1>* and *<len-2>* are the number of lights along the corresponding dimensions of the light



Area Light

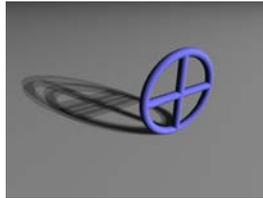


Area Light



This has a 24x24 plane (YZ)
with 4 lights

```
light_source {  
  <100, 100, -200> color  
  White  
  area_light  
    <0, 24, 0>,  
    <0, 0, 24>,  
    2, 2  
}
```

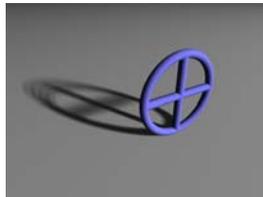


Area Light



This has a 24x24 plane (YZ)
with 16 lights

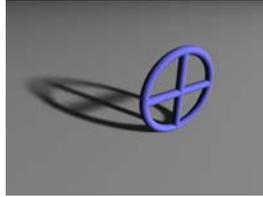
```
light_source {  
  <100, 100, -200> color  
  White  
  area_light  
    <0, 24, 0>,  
    <0, 0, 24>,  
    4, 4  
}
```



Area Light

This has a 24x24 plane (YZ)
with 64 lights

```
light_source {  
  <100, 100, -200> color  
  white  
  area_light  
    <0, 24, 0>,  
    <0, 0, 24>,  
    8, 8  
}
```



Natural?

- Banding
- NFIN



Area Light

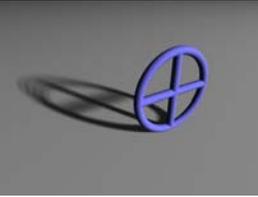
- Jitter
 - Moves individual point sources in the light by a small random amount.
 - Breaks up bands of intensity.



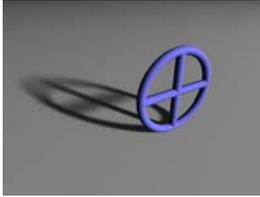
Area Light



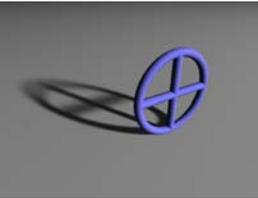
Without Jitter



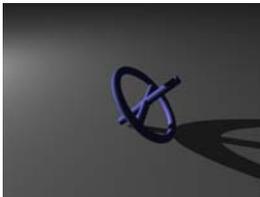
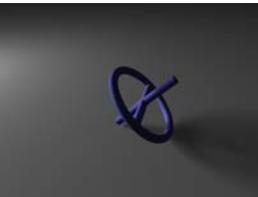
With Jitter



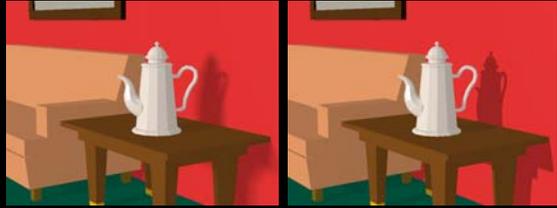
Area Light vs Point Light



Area Light vs Point Light



Area Light vs Point Light



Other things to look at



- Parallel lights
- looks_like
- projected_through



Color in lights



- Adds drama, atmosphere
- Avoid white on white
- Use for testing lights
- Examples from TERA / Toy Story

Radiosity

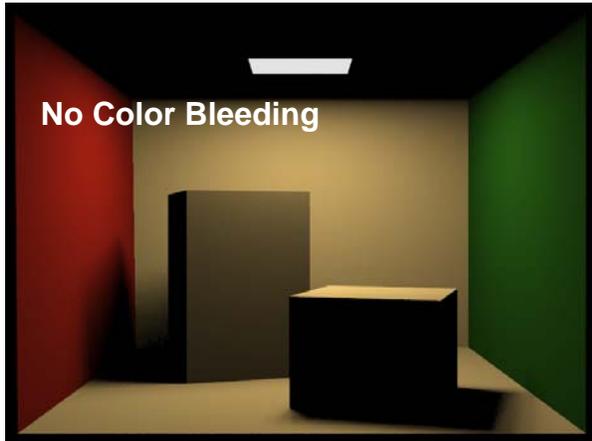


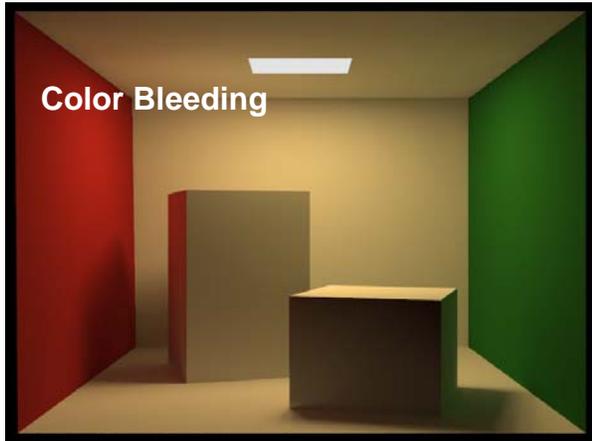
- More accurate model of reflected light
- Replaces ambient component
- From engineering: thermal transfer

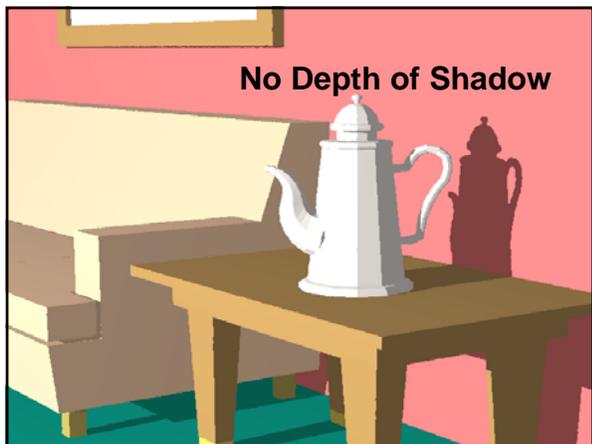
Visual Cues

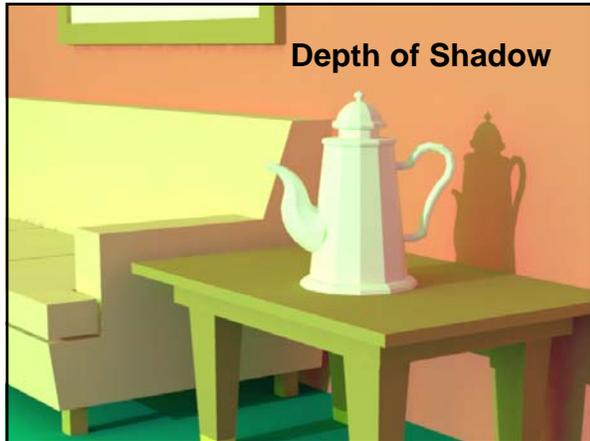


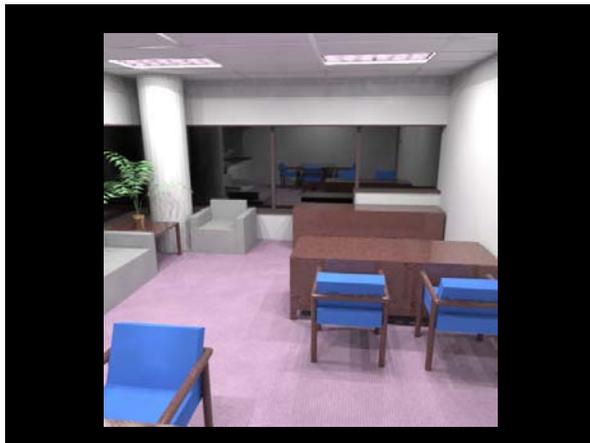
- Color bleeding
- Variation in depth of shadow

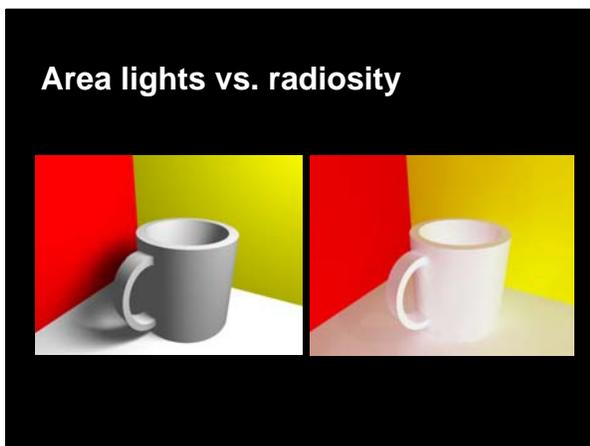








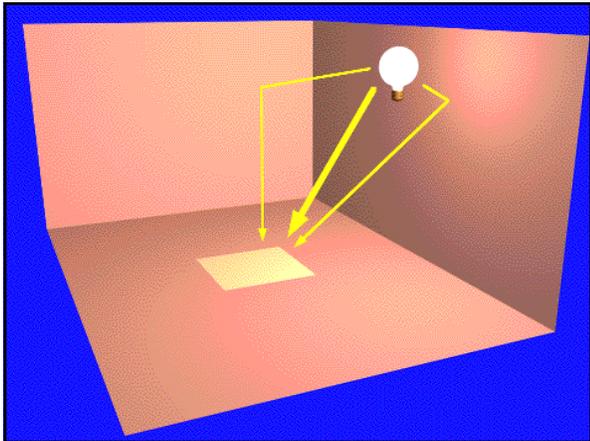


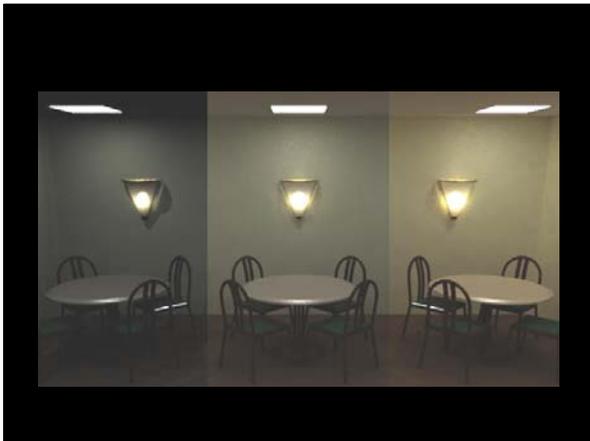


Radiosity algorithm



- Gather all of the light coming to a point on a surface
- Calculate color
- Send out this color as reflection







Finish statement



- diffuse .75 ambient 0

Radiosity in POV-Ray



```
global_settings { radiosity { [RADIOSITY_ITEMS...] }  
}  
RADIOSITY_ITEMS:  
brightness Float | count Integer | distance_maximum  
Float |  
error_bound Float | gray_threshold Float |  
low_error_factor Float |  
minimum_reuse Float | nearest_count Integer |  
recursion_limit Integer
```

```
global_settings {
  radiosity {
    pretrace_start 0.08
    pretrace_end 0.04
    count 35

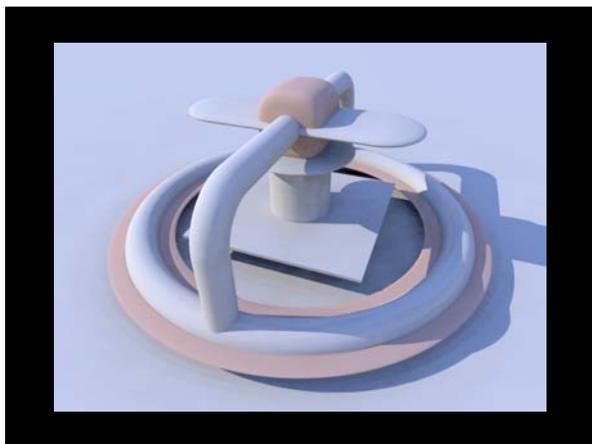
    nearest_count 5
    error_bound 1.8
    recursion_limit 3

    low_error_factor 0.5
    gray_threshold 0.0
    minimum_reuse 0.015
    brightness 1

    adc_bailout 0.01/2
  }
}
```

```
#declare RAD = off;
global_settings {
  #if(RAD)
    radiosity { ... }
  #end
}
```





In POV-Ray



- See section 6.11.11 in the help file for an explanation on each one of these terms
- Enclose your scene and camera within an object or objects so you can get color bleeding

