Computer Science 112
Art & Science of Computer Graphics
The College of Saint Rose
Fall 2015

Topic Notes: Lighting

We have seen how to add light bulbs to our scenes, now we look at that subject more carefully.

Lights

All of the lights have have used so far as of the `Light` class. They act like a light bulb hanging in space, shining equally in all directions from a center point.

A `Light` is defined by its `pos`ition, `color`, and `intensity`. If we create a `Light` as:

```
myBulb = Light()
```

- By default, the `Light`'s position is the origin. We can set it with the `pos` message:

  ```
  myBulb.pos((100, 100, -100))
  ```

  This puts the light at (100, 100, -100) when added to the scene.

- A `Light` can have a color specified:

  ```
  myBulb.color(red)
  ```

  This will make our `Light` produce red light instead of the default white.

- The light's intensity ranges from 0 (no light produced) to 1 (a very bright bulb), and is set with the `intensity` message:

  ```
  myBulb.intensity(0.4)
  ```

  By default, the `intensity` is 0.5.

We can also tell a `Light` not to cause shadows (if that's what we want for some reason):

```
myBulb.shadows(False)
```

If we later wanted to turn shadows back on:

```
myBulb.shadows(True)
```

We have used Ambrosia's built-in `bulb`, which is defined internally by Mead as follows:

```
bulb = Light().color(white)
```

Since the default color is white, Ambrosia could just as well define `bulb` as:

```
bulb = Light()
```

The bulb takes on the default position at the origin and `intensity` of 0.5.

We have also seen that we can use this default bulb and send it the `translate` message to move it to the desired point in our scene:

```
scene.add(bulb, translate(100, 100, -100))
```

This creates an Ambrosia `bulb` at the origin then translates it to (100,100,-100).

We can now take a look at Ambrosia's default `scene` definition:

```
scene = Group().add(bulb,translate(0,300,-300))
```

When we redefine the `scene` as an (initially) empty group:

```
scene = Group()
```

we no longer have the default bulb at (0,300,-300).

If we do this, however, we must then tell the default `camera` that it should use our new `scene` object instead of the original with:

```
camera.subject(scene)
```

Note that we could set the `camera`'s `subject` to any object if we ever want to generate an image with just that object.

Alternately, if we want to remove the default `bulb` (and anything else we've added) from the default `scene`, we can issue the statement:

```
scene.clear()
```

Style tip:

Keep the lighting of your model separate from the objects you place in the scene. One way to do this is to create a `Group` (perhaps named `lighting`) that defines all of your light sources. You can then easily add this to your scene.

Spotlights

We can also define and add *directed point sources of light*, called `Spotlights`, to our models.

In addition to the position, intensity, and color attributes, we can define other characteristics of a `Spotlight`. Suppose we have a `Spotlight` created with

```
spot = Spotlight()
```

- Center of interest: much like the center of interest we have seen on the `Camera` object, this is where the point source of light is aimed:

  ```
  spot.COI([100, 100, 0])
  ```

  This will aim our `Spotlight` at (100,100,0). By default, a `Spotlight` is aimed at the origin.

- *Radius*: the half-angle of the main (fully illuminated) part of the cone of light. This will define the size of the brightest part of the "spot" produced by the light when it strikes an object.

  ```
  spot.radius(15)
  ```

  The `radius` can be a value between 0 and 90. The default `radius` is 10 degrees.

- *Falloff*: an angle that describes how wide of a fading area is produced between the main part of the cone of light and the unilluminated area.

  ```
  spot.falloff(10)
  ```

  The default `falloff` is 5 degrees.

On the Wiki: Spotlights