Computer Animation: Art, Science and Criticism

Tom Ellman

Harry Roseman

Lecture 12
Ambient Light

- Emits two types of light:
  - “Directional” light, coming from a single point
    - Contributes to diffuse shading.
    - Depending on direction between surface normal and light direction.
  - “Omnidirectional” light, starting from nowhere in particular and going in no particular direction.
    - Contributes to ambient shading.
    - Independent of direction of surface normal.
- Proportions of each type of light can be varied.
- Does not contribute to specular shading.
Ambient Light

Directional

Omnidirectional
Orthographic view of sphere with Blinn shading lit by ambient light. AmbientShade = 0.45 (some directional, but mostly omnidirectional). How can we see the light itself? (demo-12-ambient-blinn-sphere.mb)
Orthographic view of sphere with Blinn shading lit by ambient light. 
AmbientShade = 0.0 (none directional, all omni-directional):

demo-12-ambient-blinn-sphere.mb
Orthographic view of sphere with Blinn shading lit by ambient light. AmbientShade = 1.0 (all directional, none omnidirectional):

demo-12-ambient-blinn-sphere.mb
Point Light

• Emits light in all directions from a specified point in space.
• Typically used to simulate light from a source confined to a small region of space.
• E.g., A small incandescent light bulb.
• Can contribute to diffuse and/or specular shading.
Point Light
Orthographic view of sphere with Blinn shading lit by point light.

demo-12-point-blinn-sphere.mb
Directional Light

- Parallel rays of light traveling in a specified direction.
- Similar to a point light placed very far away from the scene.
- Location of light object in scene space has no influence on shading.
- Useful for modeling sunlight.
Directional Light
Orthographic view of sphere with Blinn shading lit by directional light.

demo-12-directional-blinn-sphere.mb
Spot Light

• A point light emitting rays only within a specified cone.

• Cone angle attribute specifies the shape of the cone.

• Penumbra attribute specifies how fast brightness decays outside the cone.
Spot Light

θ = Cone Angle

ϕ = Penumbra Angle

No Light

Penumbra

Light Cone

Penumbra

Penumbra

No Light
Plane with Lambert shading lit by spot and directional lights. Cone Angle = 40 degrees; Penumbra = 3.5 degrees:

demo-12-spot-plane.mb
Area Light

• Defined by a rectangular plane and a normal direction.

• Light is emitted by all points on the plane.

• Equal amounts of light are emitted in all directions up to 90 degrees from the normal.
Area Light

Light Plane:

Normal Direction
Plane with Lambert shading lit by area and directional lights. How can we see the light itself?

demo-12-area-plane.mb
Attributes of Non-Ambient Lights

• Emits Diffuse: Whether or not the light contributes to diffuse shading.
• Emits Specular: Whether or not the light contributes to specular shading.
• Decay Rate: How the intensity of the light varies with distance:
  – No Decay: Intensity is constant.
  – Linear: Intensity $\propto 1/$Distance.
  – Quadratic: Intensity $\propto 1/$Distance$^2$.
  – Cubic: Intensity $\propto 1/$Distance$^3$. 
Light Reveals Modeling

- Shading varies across a diffusely lit surface.
- Shading depends on angle between surface normal and light direction.
- (Lambert’s cosine law.)
- Shading indicates surface orientation at each point on the surface.
Lambert’s Cosine Law

\[ I_D = K_D \cdot I_{LD} \cdot \cos \theta \]
Vase with lambert material: demo-12-modeled-vase.mb
Mapping Light Attributes

• Many light attributes can be mapped.
• A mapped attribute potentially takes a different value for each surface point:
  – Based on the point’s (u,v) coordinates.
  – Based on the point’s (x,y,z) coordinates.
• E.g., Mapping the intensity or color of a light creates a “throw pattern”.
Wave pattern converted into a 3D map via a perspective projection and applied to the intensity attribute of a point light:

demo-12-intensity-mapped-vase.mb
Wave pattern converted into a 3D map via a perspective projection and applied to the intensity attribute of a point light:

demo-12-intensity-mapped-vase.mb
Three Point Lighting

• Key Light:
  – Brightest light on main subject.
  – Reveals location of main light source.

• Fill Light:
  – Dimly illuminates areas of main subject unlit by key light.
  – Often represents an indirect light source.

• Back Light:
  – Illuminates main subject from behind.
  – Separates subject from background.
Illumination by key light only:
demo-12-three-point-lighting.mb
Illumination by fill light only:

demo-12-three-point-lighting.mb
Illumination by key light and fill light.
demo-12-three-point-lighting.mb
Indirect Lighting

• A surface illuminated with diffuse light can illuminate other surfaces.
• E.g., A room is filled by light reflected from its white walls.
• Standard illumination models don’t account for such “indirect lighting”.
• Radiosity and Photon mapping are able to handle indirect lighting – at great computational expense.
• Area lights can be used to fake indirect lighting.
Red area lights are hidden in the red cube in an effort to fake the indirect lighting of the floor and gray cube by light reflected from the red cube: demo-12-fake-indirect-light.mb
Linking Lights and Objects

- Defines which lights illuminate which objects.
- Initially a light illuminates all objects.
- In the Attribute Editor, uncheck “Illuminates by Default”.
- Now the light must be explicitly linked to objects.
- Break Links: Select lights and objects and invoke menu item: Lighting/Shading-BreakLightLinks.
- Make Links: Select lights and objects and invoke menu item: Lighting/Shading-MakeLightLinks.
- Relationship Editor:
  - Editable table of light / object links.
  - Light-Centric and Object-Centric modes.
- Links Enhance Control: Designer can control exactly which lights illuminate which objects.
- Links Speed Rendering: If each light is linked only with the objects it needs to illuminate.
Two spheres, each illuminated by its own point light, as indicated in the Relationship Editor snapshots:

demo-12-light-linking.mb
Exercise: Light a Room with a Lamp

• Make a NURB cube to represent a room and remove the front face.
• Make a NURB cylinder without caps to represent a lamp shade.
• Make the lamp shade translucent.
• Use a combination of point and spot lights to make the lamp appear to light the room.
• Make appropriate use of light linking.
• Do not enable shadows for any lights.
Lighting a Room with a Lamp

- Point light $shadePoint$ is linked only to the lamp shade, via transluscence.
- Point light $roomPoint$ is linked only to the room (walls, ceiling, floor) and the base of the lamp.
- Spot light $upperSpot$ is linked only to the room (walls, ceiling, floor).
- Spot light $lowerSpot$ is linked only to the room (walls, ceiling, floor) and the base of the lamp.
- All lights are located inside the lamp shade, where a light bulb would be in reality.
Color Balance in Lighting

• Indoor Scene:
  – Incandescent lamp is white
  – Sunlight entering via window has blue tint.

• Outdoor Scene:
  – Sunlight is white.
  – Incandescent light exiting via window has red tint.
demo-12-outdoor-color-balance.mb
Uses of Shadows

- Defining spatial relationships.
- Alternative angles.
- Suggesting off screen objects.
- Adding contrast.
- Adding to the composition.
- Integrating elements.
Which sphere is bigger? Which sphere is closer?
Shadows provide depth cues.
Shadows reveals alternative angle.
demo-12-profile-shadow.mb
Controlling which Object/Light Combinations Generate Shadows

- **Light Centered Control:**
  - In Attribute Editor check/uncheck “UseDepth Mapped Shadows” or “Use Ray Traced Shadows”.
  - When ray tracing shadows, you must enable ray tracing in the Render Globals window.

- **Object Centered Control:**
  In Attribute Editor check/uncheck CastsShadows under the RenderStats tab.
Depth-Mapped Shadows

• Constructing the Map:
  – Construct an image of the scene as viewed from the light.
  – At each pixel coordinate \((x_L, y_L)\) record the distance ("depth") to the nearest object.

• Using the Map:
  – For each surface point \((x, y, z)\), find the coordinates \((x_L, y_L)\) at which it appears in the depth map.
  – The point is shadowed if the distance from \((x, y, z)\) to the light is greater than the depth at \((x_L, y_L)\).

• Note: The results depend on the resolution of the depth map.
Depth-Mapped Shadows
Ray Traced Shadows

• Given a surface point p being rendered.
• Construct a shadow ray L from p to the light source.
• Determine whether L intersects any objects along the way from p to the light source.
• If so, then the intersected object casts a shadow at point p.
Ray Traced Shadows

Surface Point p

Shadow Ray L

Shadowed light does \textit{not} illuminate p.

Unshadowed light \textit{does} illuminate p.
Properties of Shadows

• Hard v. Soft:
  – Does the shadow have sharp edges?
  – Or does the shadow gradually blend into the illuminated area?

• Focused v. UnFocuseded:
  – Is the shadow equally hard or soft all around its perimeter?
  – Or is the shadow harder near the shadowing object and softer far away from the shadowing object?
Cylinder illuminated by point light with (64 by 64) depth-mapped shadows:

demo-12-point-cylinder-shadow.mb
Cylinder illuminated by point light with (1024 by 1024) depth-mapped shadows:

demo-12-point-cylinder-shadow.mb
Cylinder illuminated by point light with ray-traced shadows.

demo-12-point-cylinder-shadow.mb
Cylinder illuminated by area light with (1024 by 1024) depth-mapped shadows:

demo-12-area-cylinder-shadow.mb
Cylinder illuminated by area light with ray traced shadows.
Use of just one shadow ray results in sampling anomalies.

demo-12-area-cylinder-shadow.mb
Cylinder illuminated by area light with ray traced shadows.
Use of 8 shadow rays avoids sampling anomalies.

demo-12-area-cylinder-shadow.mb
Cylinder illuminated by point light with radius 1.0 with ray traced shadows: Use of 8 shadow rays avoids sampling anomalies: demo-12-point-cylinder-shadow.mb
Transparency and Shadows

• Transparent objects may cast shadows.
• The shade of the shadow will depend on the transparency of the object.
• Transparency may be mapped in Maya.
• E.g., Simulation of stain-glassed windows.
Sphere illuminated by point light through a transparency mapped plane: demo-12-point-light-stained-glass.mb
Connecting file outColor to blinn color.

Connecting file outColor to blinn transparency.

demo-12-point-light-stained-glass.mb

(Used Connection Editor to connect file2.outColor to blinn2.transparency)
Sphere illuminated by area light through a transparency mapped plane: demo-12-area-light-stained-glass.mb
demo-12-light-glow.mb
demo-12-light-glow-lens-flare.mb
demo-12-environment-fog-simple.mb