Overview

- Global illumination, the rendering equation
- Ray tracing, HB Ch 14-6, pp527-537, pp 541-543

Introduction

Previously we covered *local* illumination models.

- These models consider only light coming *directly* from a light source to a surface.
- The simplicity of these models makes them efficient.

But local illumination models cannot produce some interesting and important effects:

- reflection of light off one object onto another
- soft shadows

These effects require *global* illumination algorithms.

- These algorithms model how light bounces around the entire scene.
- The extra complexity makes these algorithms computationally expensive.
- Nevertheless, the expense is worthwhile in some applications:
  - architecture
  - advertising
  - entertainment

The rendering equation

Kajiya published this description of global illumination in 1986.

The light passing from point x_2 to point x_1 is a sum of:

- the light emitted at x_2 that reaches x_1
- the light emitted by all other points that is reflected by x_2 onto x_1

More formally, the light intensity passing from x_2 to x_1 is:

$$I(x_1, x_2) = g(x_1, x_2)\left(\varepsilon(x_1, x_2) + \int_S \rho(x_1, x_2, x_3)\, I(x_2, x_3)\, dx_3\right)$$

- $g(x_1, x_2)$ is the geometry term, which is 0 when x_2 and x_1 are occluded from each other
- $\varepsilon(x_1, x_2)$ is the light emitted at x_2 towards x_1
- S is the set of all surface points
- $\rho(x_1, x_2, x_3)$ is the fraction of the intensity from another point, x_3, reflected by x_2 to x_1

A solution to I for all pairs of points would produce the correct lighting in an image.

Note that I occurs on both sides, making the solution difficult.

The rendering equation

The most popular classes of global illumination algorithms find *approx* solutions to the rendering equation.

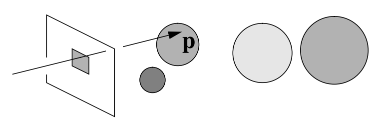
- *Recursive ray-tracing* algorithms:
  - Their approximations handle *specular* reflection more correctly; most complete solution of illumination-reflection model in CG.
  - Their solutions are *view dependent*: the illumination must be recomputed each time the camera moves.
  - Expensive! Still these provide solution to global
  - illumination problem that accounts for: hidden surfaces, shading from direct illumination, shading from global illumination, shadows.
- *Radiosity* algorithms:
  - Their approximations handle *diffuse* reflection only.
  - Their solutions are *view independent*: once computed the illumination is correct for any camera position.

Ray Tracing

1. Definition
2. The illumination model at a surface point
3. Recursive implementation
4. Optimizing
5. Distribured ray tracing

Recursive ray tracing

- Recall the ray-casting algorithm for visible-surface determination.



  - For each pixel, cast a ray from the center of projection (camera position) into the scene.
  - The surface visible at the pixel has the closest intersection with the ray, at point **p**.

These rays are *primary rays*.

- Recursive ray tracing also casts *secondary rays*.
  - Secondary rays help to determine the illumination at the visible surface from other objects.
  - Rene Decarts, 17th century, used ray tracing to explain the shape of the rainbow
  - In CG, both Kay and Whitted published recursive ray tracing idea in 1979-80.
- Secondary rays:
  - reflected
  - trnsmitted
  - shadow

Shadow rays

One secondary ray is a *shadow ray.*

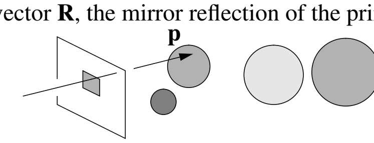
- For each light source, cast a shadow ray from **p** along the vector, **L**, to the light source.
- If the shadow ray does not intersect other objects, then the light source adds illumination to **p**, shading model(e.g. Phong) is used give local illumination
  - local illumination diffuse term
  - local illumination specular term

  $I_{local} = I_{ambient} + I_{difuse} + I_{specular}$
  - In this local illumination model we may include another term accounting for light from a light source seen through transparent object.
- Otherwise, **p** is in a shadow and gets no direct illumination from that light source, but will include only the ambient term.

Reflected Ray

Another secondary ray is a *reflected ray.*

- If **p** is on a *specular* surface, cast a reflected ray along the vector **R**, the mirror reflection of the primary ray.



  - **R** is in the same plane with the primary ray and the normal at the point, and makes the same angle with **N** as the primary ray, but is on the "other side" on the normal.
  - Light coming to the point along **R** would be reflected back along the primary ray by a mirror reflection, and it will be scaled according to global specular (mirror) reflection coefficeint k_{gs},

  $$I_{gspecular} = k_{gr} I_{reflected}$$

  Note that these are calculated recursively

  $I_{reflected} = I(p_r)$

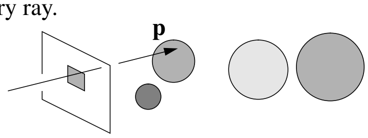
  where p_r is the first hit of the reflected ray from p.

  Not ethat k_s controls specular highlight in local model.
- Add the illumination coming back along **R** to **p**, computed *recursively,* to the local illumination term.
  - Treat the *reflected* ray like a *primary* ray for .

Transmitted ray

Another secondary ray is a *transmitted (refracted) ray.*

- If **p** is on a *transparent* surface, cast a transmitted ray along the vector **T**, which is a refraction of the the primary ray.



  - Light coming to the point along **T** would be transmitted back along the primary ray , and it will be scaled according to coefficeint k_{gt},

  $$I_{gtransmitted} = k_{gt} I_{transmitted}$$

  Note that these are calculated recursively

  $I_{transmitted} = I(p_t)$

  where p_t is the first hit of the transmitted ray from p.
- Add the illumination coming back along T to **p**, computed *recursively,* to the local illumination term and the global specular term at p

  $$I(p) = I_{local}(p) + k_{gr} I(p_r) + k_{gt} I(p_t).$$

Local terms are those calculated at each surface intersection owing to direct illumination and ambient light.

  - Treat the transmitted ray like a *primary* ray for **p**....

Comments

- Ray tracing as a hybrid model: direct specular rays are distributed empirically, reflected and transmitted rays are traced back (in one diection). Sharp images!
- Four mechanisms of light transport between surfaces:
  - diffuse to diffuse surface
  - specular to diffuse (ray tracing partially models this in the local illumination term)
  - diffuse to specular
  - specular to specular (ray tracing models this but specular reflectance is followed in a single direction)
- Three contributions to the intensity of an intersection point encounterd during ray tracing(from local, from reflected contribution, from transmitted contribution)
- The recursion base cases
  - If a ray hits no object, only background color.
  - If a ray hits a nonspecular opaque surface, use the local diffuse illumination only at that surface
  - If the recursion has gone "too deep," stop it , no more contributions

Simple recurcive ray tracing procedure

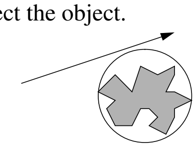
```
Ray_Trace(start,direction,depth,color){
 if (depth>maxdepth) pcolor=black
 else {
  find closest intersection p (if any)
  if (no intersection) color=bck_grnd
  else {
   1. lcl_color=calc local contribution
   2. find reflected ray R
   3. Ray_Trace(p,R,depth+1,relf_color)
   4. find transmitter ray T
   5. Ray_Trace(p,T,depth+1,trsm_color)
   6. AccumulateColor(color,
         lcl_color, refl_color,
         trsm_color, k_gr, k_gt)
  }
 }
}
```


Optimizations: Recursive ray tracing

1. Bounding volumes

The numbers of rays can be huge, intersections can consume 95% , optimizations on intersection calculations are essential.

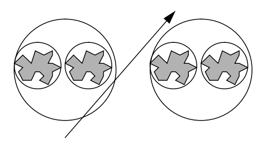
- Before testing a complicated object, test a simple bounding volume
- If a ray does not intersect the bounding volume, it cannot intersect the object.



Hierarchies of bounding volumes:

- Enclose groups of bounding volumes with larger bounding volumes. Continue this idea to define a hierarchy.
- If a ray does not intersect one volume in the hierarchy, it cannot intersect that volume's children.

Adaptive selection volume types (scene dependent...)



2. Adaptive depth control

- Scenes with more specular and transparent surface need bigger depth
- Opaque objects (depth 1, the ray tracer equivalent to local shading and hidden surface removal)
- Hall and Greenberg(1983): adaptive depth control which depends on the material of the objects being traced and interacting with the rays.
  - *Percentage of the scene that contains highly transparent and specular surfaces is , in general , small, so inefficient to trace all till max depth*
  - While ray tracing , accumulate (multiply) various coefficients that attenuate the intensity of the light (reflected or reansmitted). When the product gets below certain *accumulation-weight-parameter* stop the recurrsion
  - adaptive depth control prevents a ray that initialy hits almost opaque surface from inducing a transmited ray that is then traced...
  - Avg depth traced, shown empirically for typical scenes between 1 and 2
- *First-hit speed up:* hybrid approach, modified hidden surface removal alg (Z-buffer) more efficient for first hit. Later hits use ray tracing with adaptive depth
- For scenes of reasonable complexity adaptive depth control, bounding volumes, first-hit speed up lead to reducing time in half compared to naive ray tracer.

3. Spatial coherence, space subdivision methods.

- Note that individual rays are traced independently. Naive ray tracing does exhaustive search (possibly with some optimization on bounding volumes, etc.)
- Spatial coherence idea: divide space occupied by scene into regions, and test occupancy of regions through which ray is traveling. If there is occupancy, then objects in reagion are tested for intersections

Ray tracing results