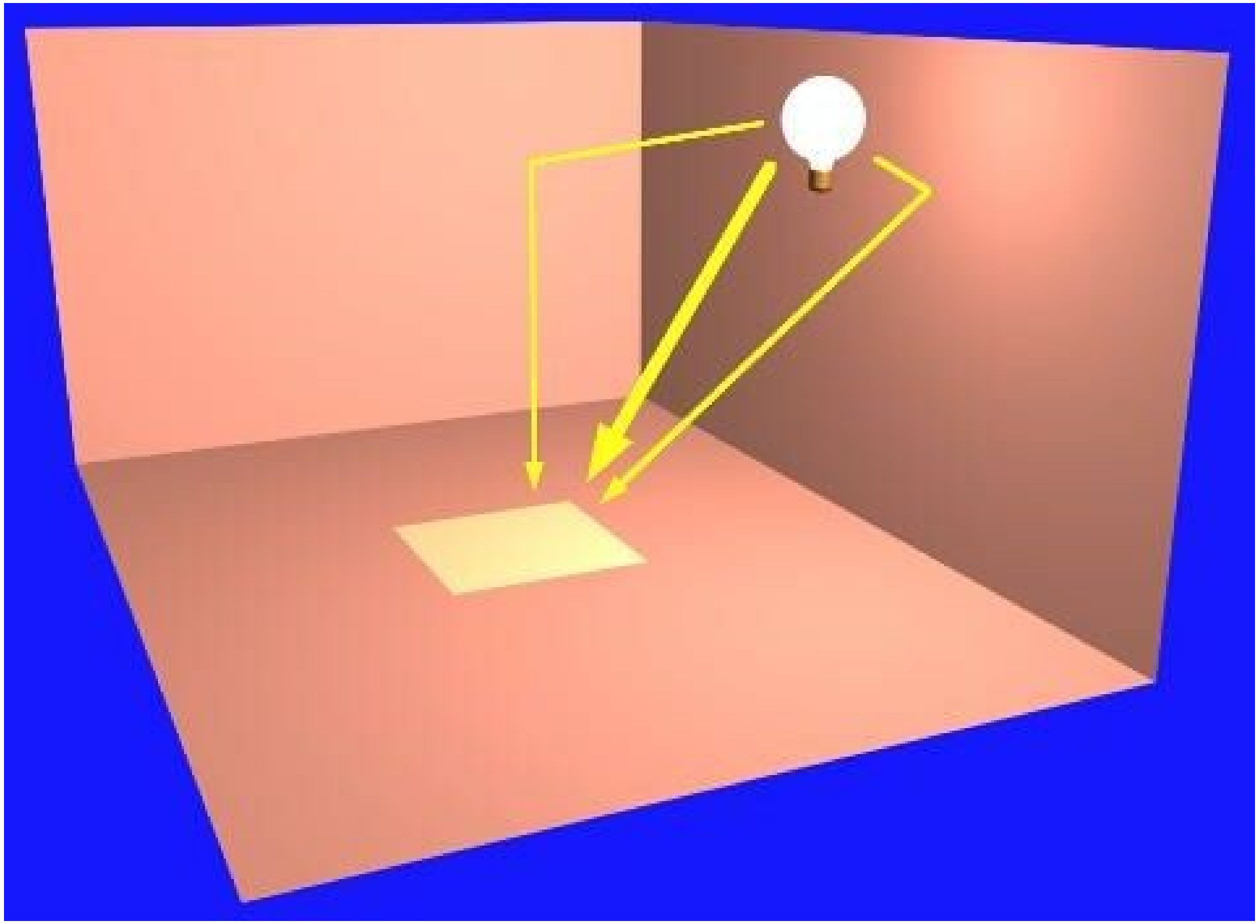
7M836
Animation & Rendering

Global illumination, radiosity

Arjan Kok, Kees Huizing, Huub van de Wetering
h.v.d.wetering@tue.nl

Direct and indirect illumination



Global illumination

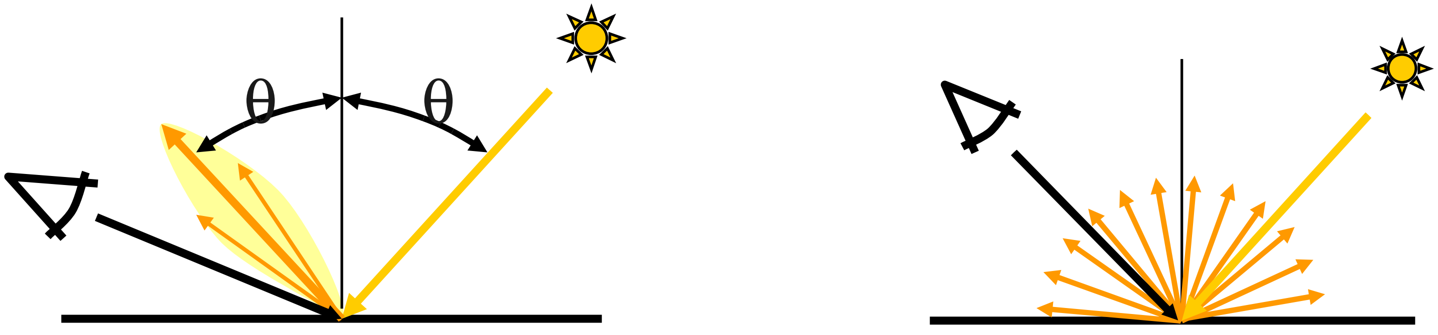
- Light paths
  - Complete solution: L(D|S)*E
  - Ray tracing: L(S*)DS*E

- Still missing in ray tracing:
  - Diffuse interreflection: D*
  - Diffuse/specular interreflection: (D|S)*
  - These terms approximated with ambient light

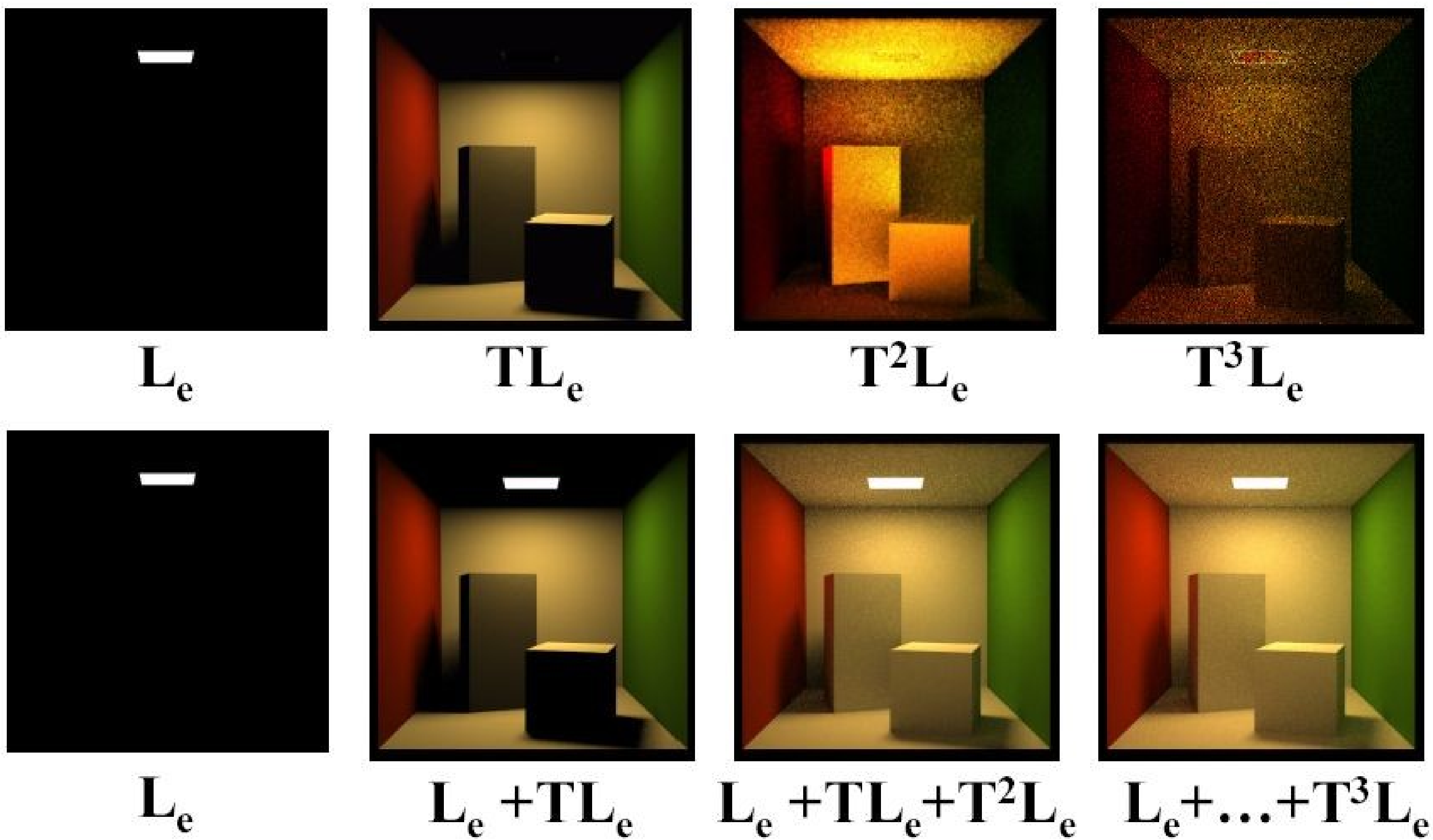
Global illumination

Diffuse interreflection

- Why specular reflection in in ray tracing and why diffuse interreflection not?



Indirect diffuse illumination



Radiosity method

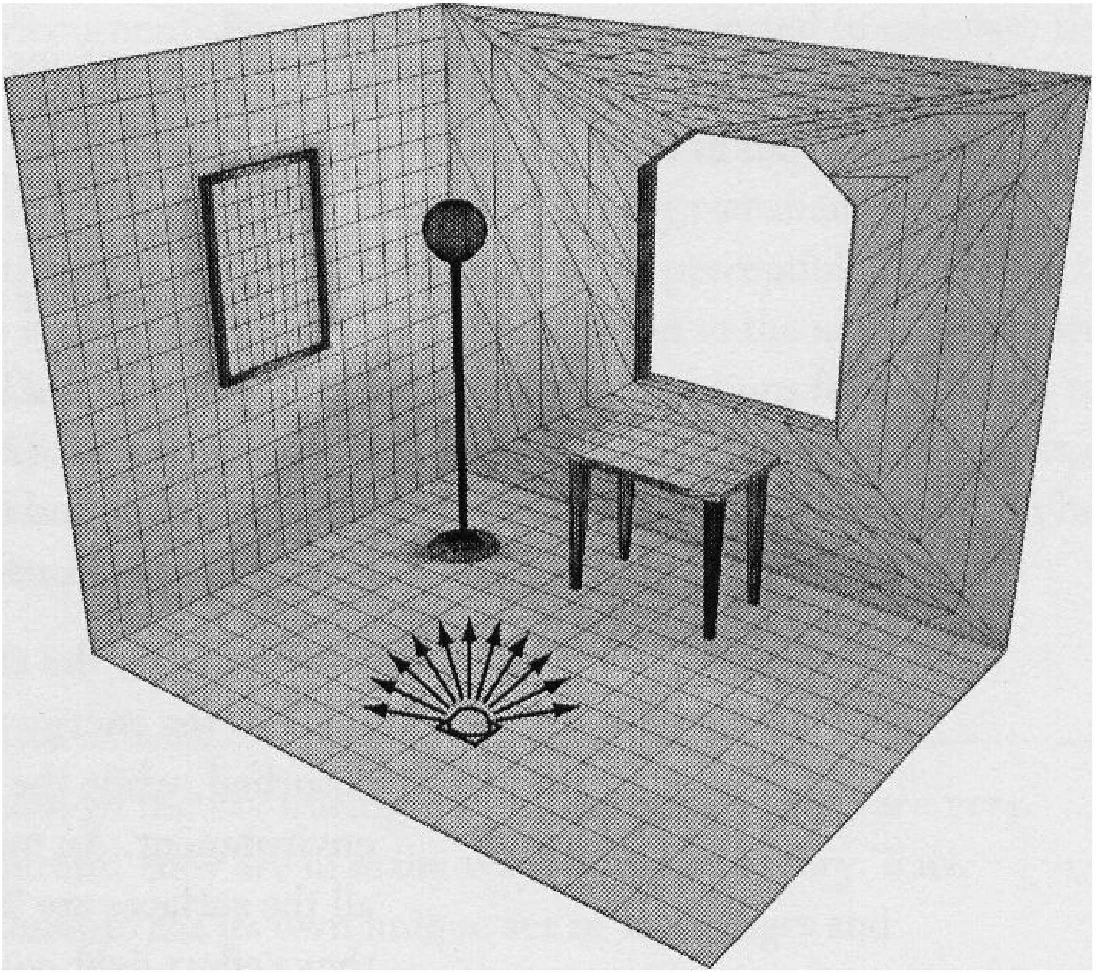
- Radiosity method computes *diffuse interreflection* of objects
  - Diffuse interreflection is no longer approximated by ambient light

Radiosity method

Basic idea

- Subdivide all polygons in elements for which one energy value (~ color) must be computed
- Treat every element in scene as source of light
- Compute for each element the total amount of radiated energy

Radiosity



Radiosity effects

- Soft shadows
- Color bleeding

Radiosity example

Radiosity

- Radiosity (B) is energy per unit area that leaves surface

  so

- Radiosity is sum of energy
  - Emitted by surface (light sources)
  - Reflected by surface

Radiosity equation

$$B_i A_i = E_i A_i + \rho_i \sum_j B_j A_j F_{ji}$$

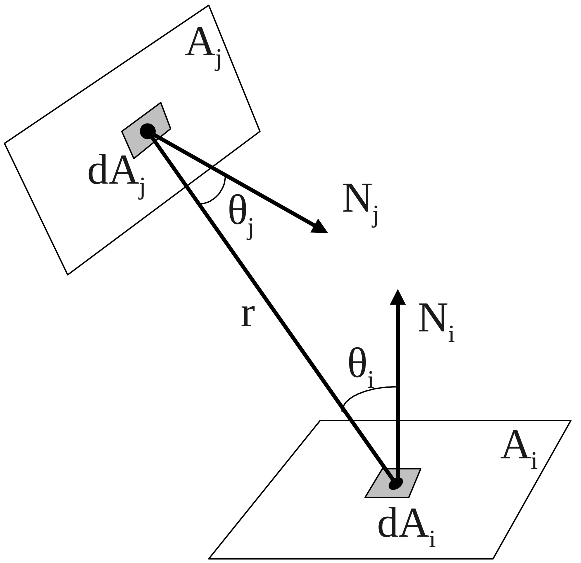
- B_i = Radiosity of element i
- E_i = Emission of element i
- ρ_i = Reflection coefficient (diffuse) of element i ($k_d C_d$)
- F_{ji} = Form-factor between elements j en i (= fraction of energy leaving element j that reaches element i)
- $F_{ij} A_i = F_{ji} A_j$

$$B_i = E_i + \rho_i \sum_j B_j F_{ij}$$

Form factor

- Form-factor F_{ij} is fraction of total energy of patch i that reaches patch j

$$F_{ij} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos\theta_i \cos\theta_j}{\pi r^2} V_{ij} dA_j dA_i$$



- V_{ij} = 1 if dA_j visible from dA_i
  0 if not visible

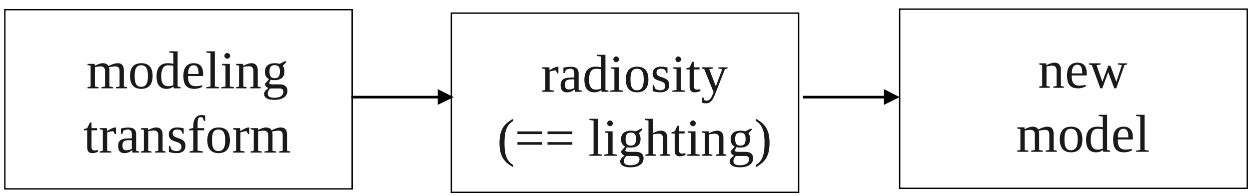
Radiosity solution

- Solve for all elements the radiosity equation:

$$B_i = E_i + \rho_i \sum_j B_j F_{ij}$$

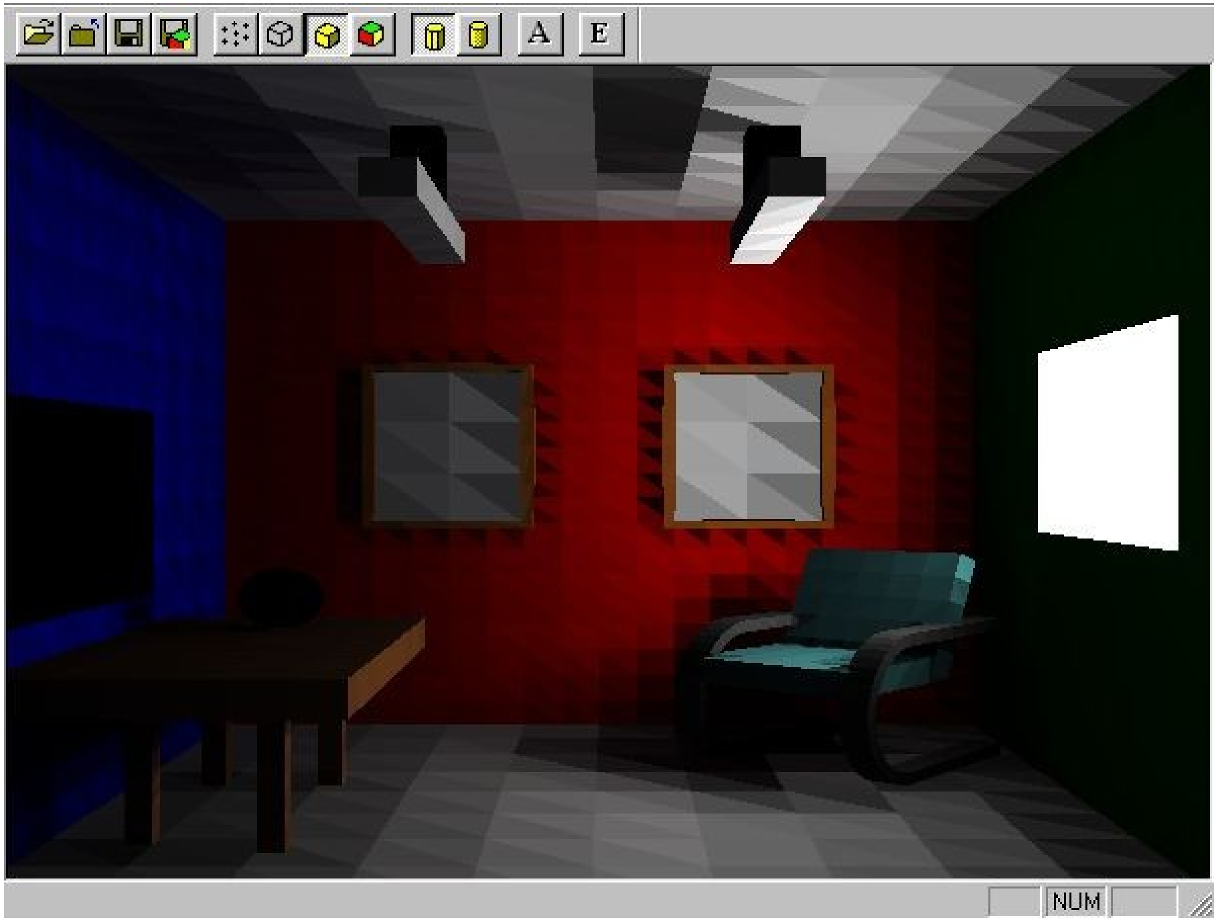
Radiosity

- Result of radiosity: radiosity value (~ diffuse intensity) for each element

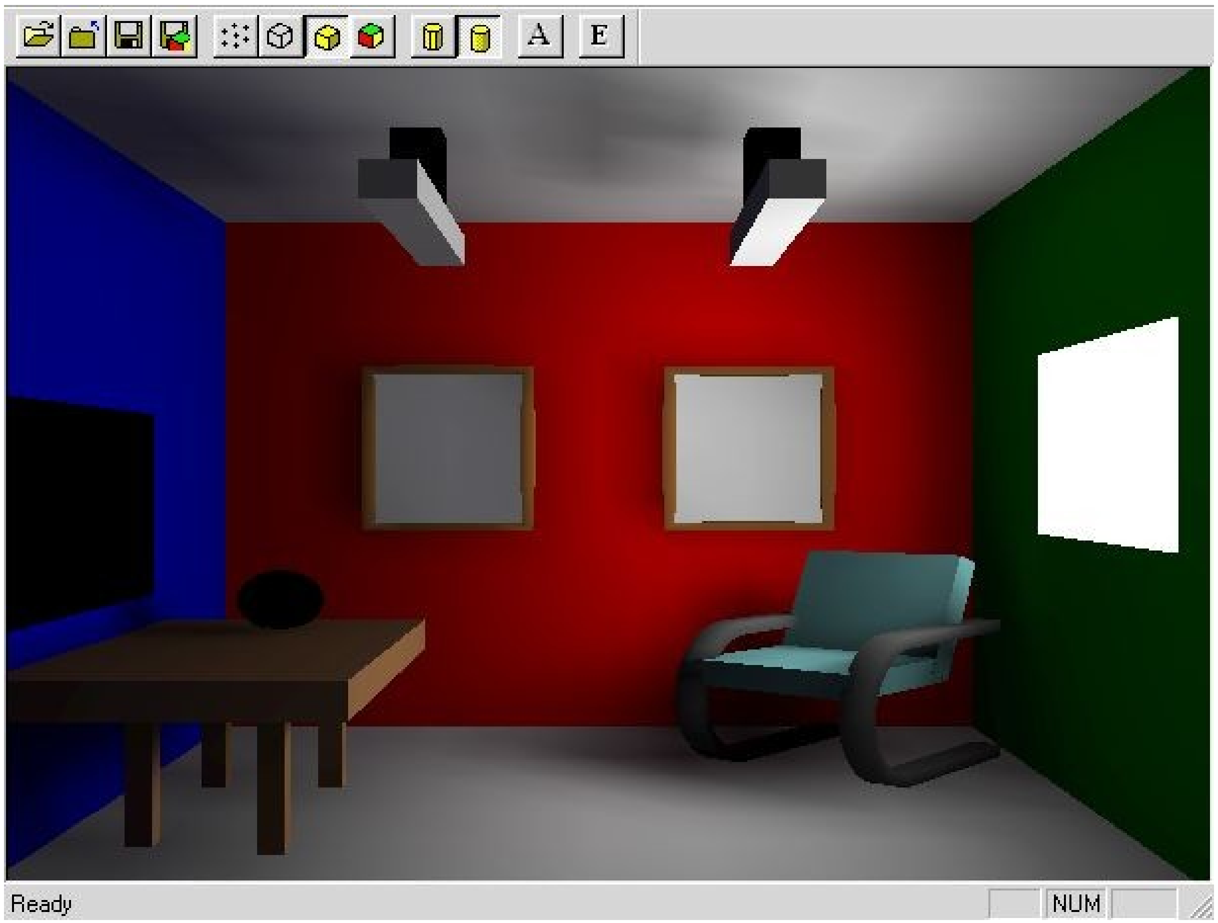


- Radiosity is viewpoint independent
  - One solution can be used to generate several images

Radiosity result – flat



Radiosity result – Gouraud



Radiosity solution

- Solve for all elements the radiosity equation:

$$B_i = E_i + \rho_i \sum_j B_j F_{ij}$$

- Methods:
  - Full-matrix radiosity
  - Progressive refinement radiosity

Matrix radiosity

$$\begin{bmatrix} 1-\rho_1 F_{1,1} & . & . & . & -\rho_1 F_{1,n} \\ -\rho_2 F_{2,1} & 1-\rho F_{2,2} & . & . & -\rho_2 F_{2,n} \\ . & . & . & . & . \\ . & . & . & . & . \\ -\rho_n F_{n,1} & . & . & . & 1-\rho_n F_{n,n} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ . \\ . \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ . \\ . \\ E_n \end{bmatrix}$$

Matrix radiosity

- Disadvantages of matrix method
  - High time and memory cost: (O(n^2))
    - n = number of elements
  - All form-factors have to be computed
    - Even for areas in scene where there is almost no energy exchange

Progressive radiosity

- $B_j = E_j + \rho_j \sum_i B_i F_{ji}$
- Contribution B_i to B_j : $\rho_j B_i F_{ji}$
- Select an element i
- Shoot all energy from element i, and compute contribution to radiosities of all other elements j
- Initially all radiosities of all elements are 0, except radiosities of light sources

Progressive radiosity

for all patches i
    $radiosity_i$:= emission value to radiosity
    $unshot\_radiosity_i$:= emission value

do

    determine patch i with most unshot energy
    for all other patches j **do**
        contribution := $\rho_j B_i F_{ji}$
        $radiosity_j$:= $radiosity_j$ + contribution
        $unshot\_radiosity_j$:= $unshot\_radiosity_j$ + contribution
    $unshot\_radiosity_i$:= 0
until convergence reached

Progressive radiosity

Initialization

B_j, $\Delta B_j = E_j$ for all light source elements

B_j, $\Delta B_j = 0$ for all other elements

do

find element i with most "unshot" energy $\Delta B_i A_i$;

for all other elements j do

compute form-factor F_{ij}

$\Delta Rad = \rho_j * \Delta B_i * F_{ij} * (A_i/A_j)$

$\Delta B_j = \Delta B_j + \Delta Rad$

$B_j = B_j + \Delta Rad$;

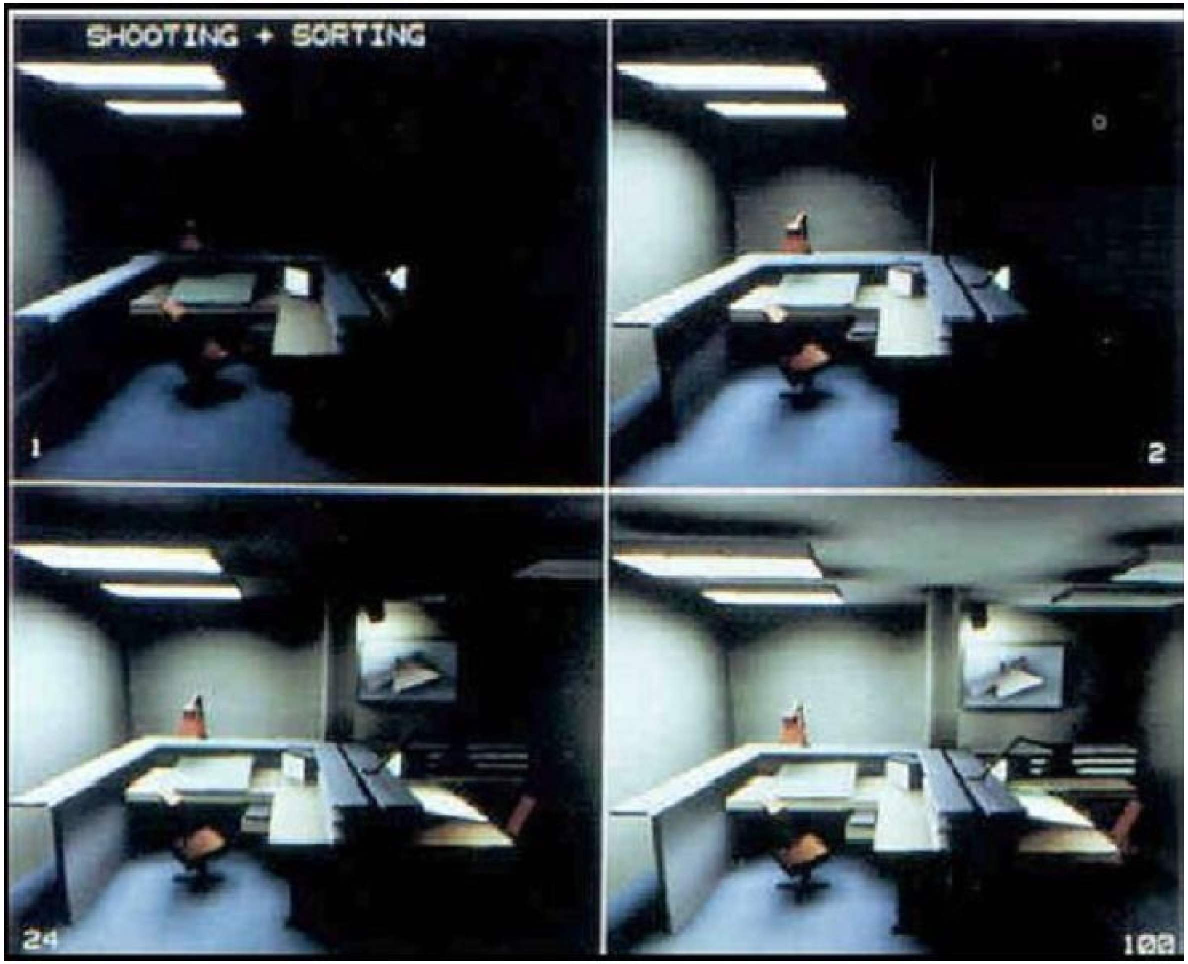
$\Delta B_i = 0$;

until convergence reached

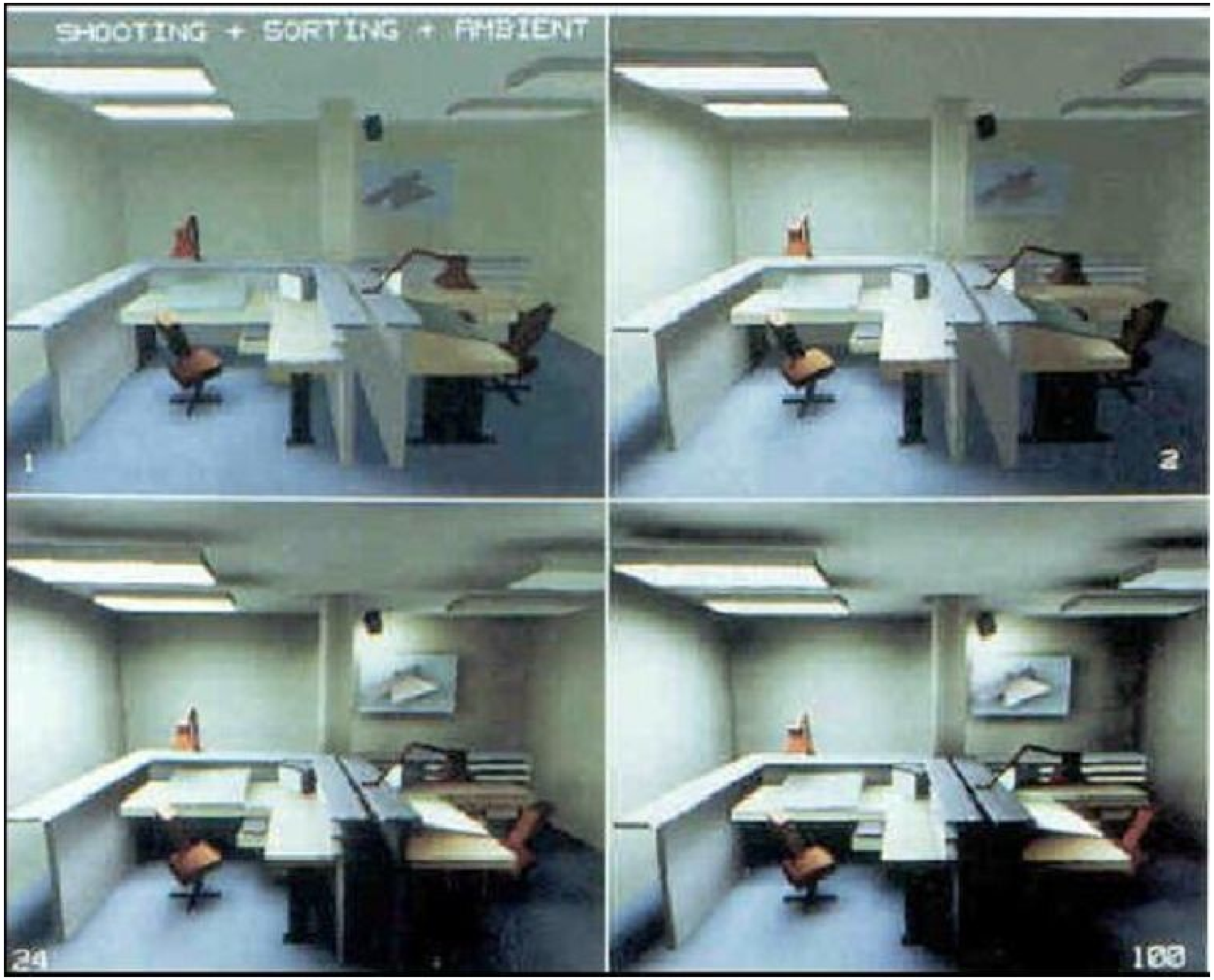
Progressive radiosity

- Advantages
  - Less memory consumption
  - Ability to inspect process
  - Ability to stop before process completely converged

Progressive radiosity



Progressive radiosity + ambient



Radiosity

- Result: 1 radiosity value per element
  so 1 color per element

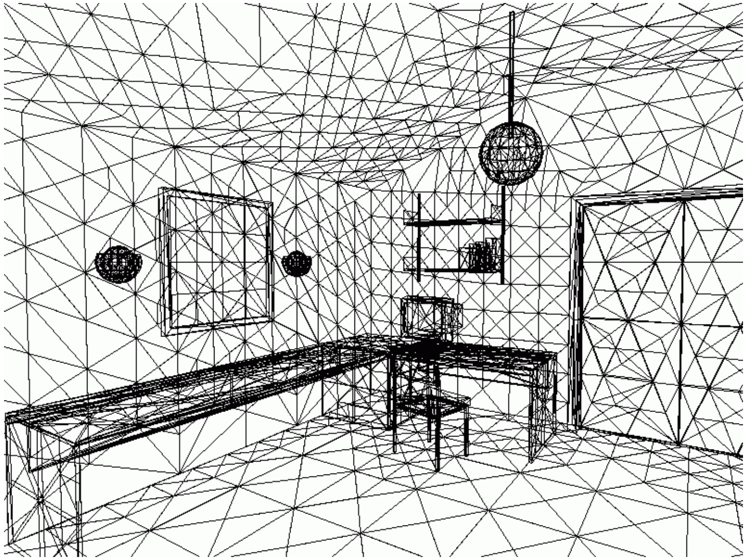
- Meshing:
  - Partition surfaces in scene into small elements

- Meshing conditions
  - Good representation of intensity changes
  - No more elements than necessary

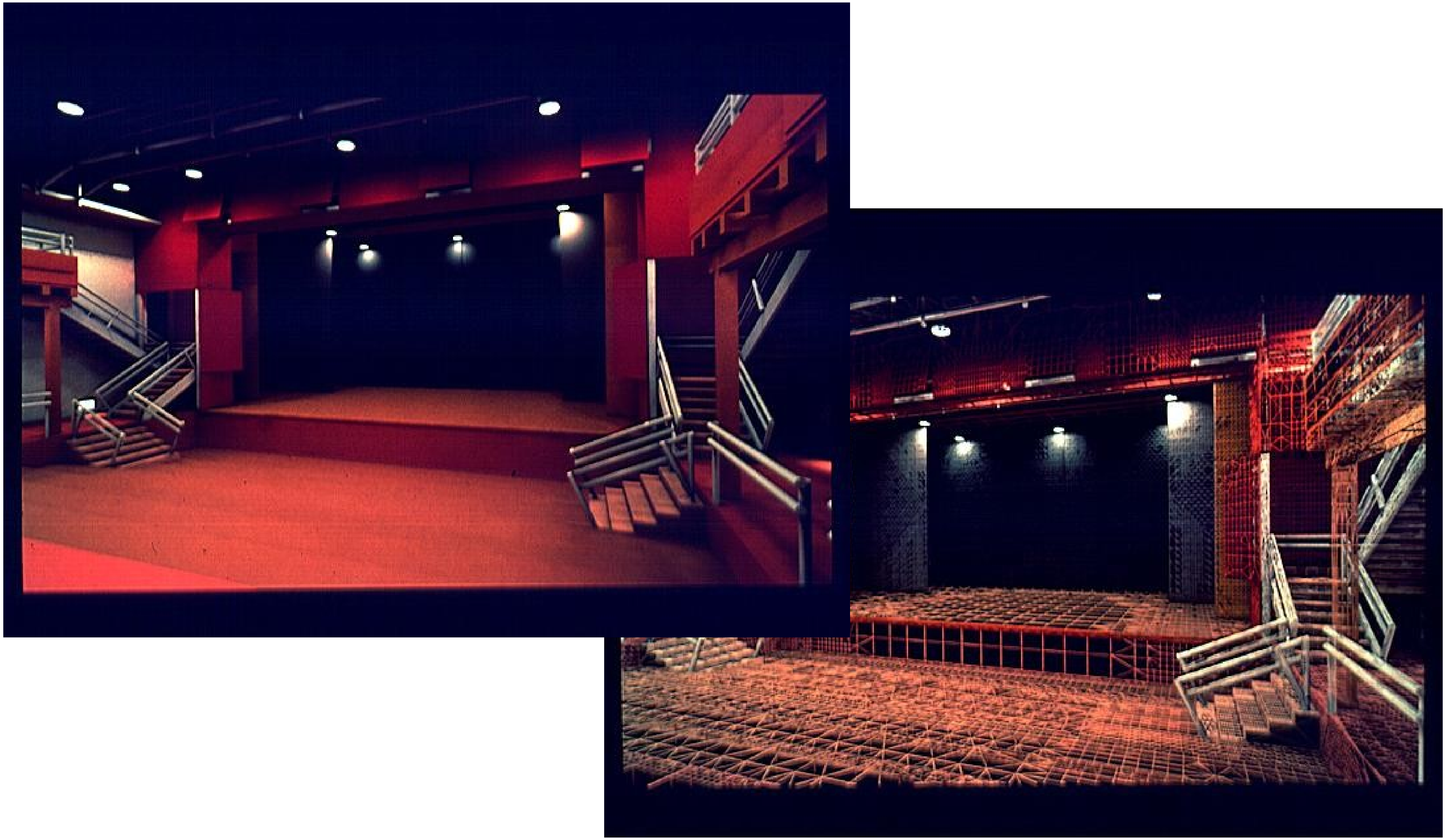
Meshing

- Uniform meshing
- Adaptive meshing
  - Make (uniform) start mesh and modify mesh (more elements) where large intensity differences found
- Discontinuity meshing
  - Determine before radiosity computations where large intensity changes will occur. Mesh finer along intensity transitions

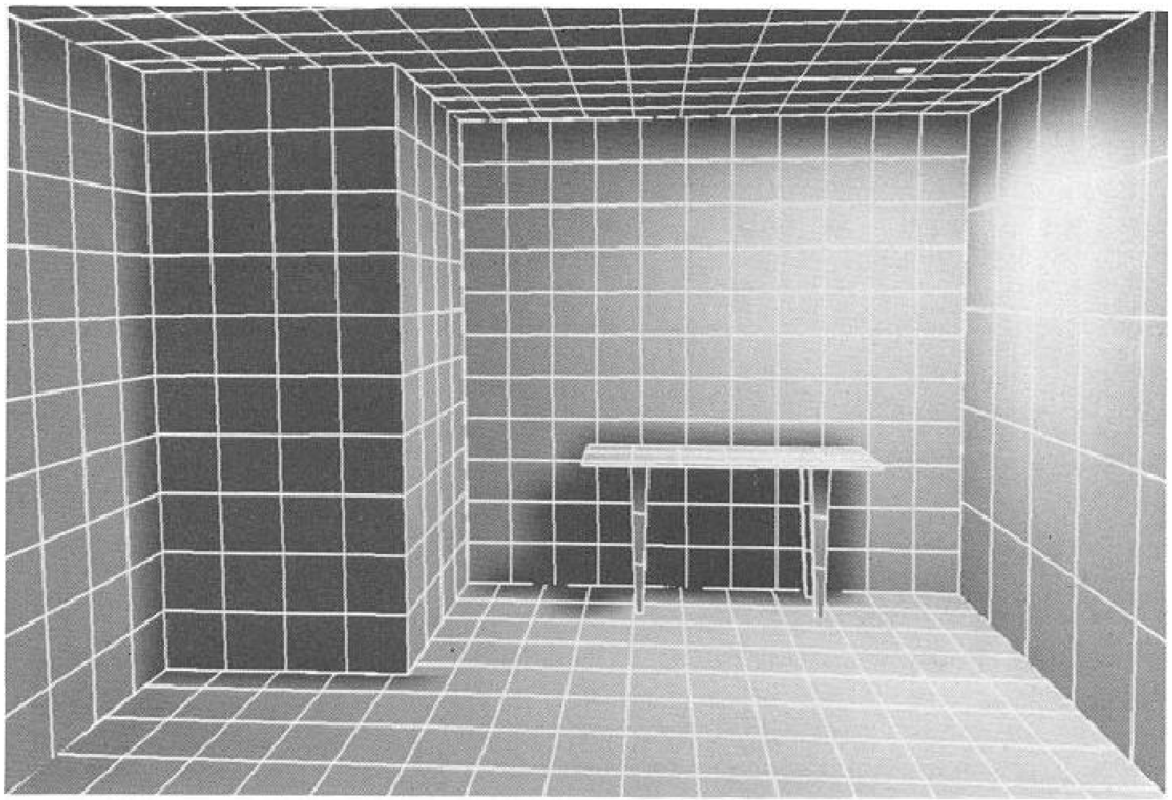
Meshing



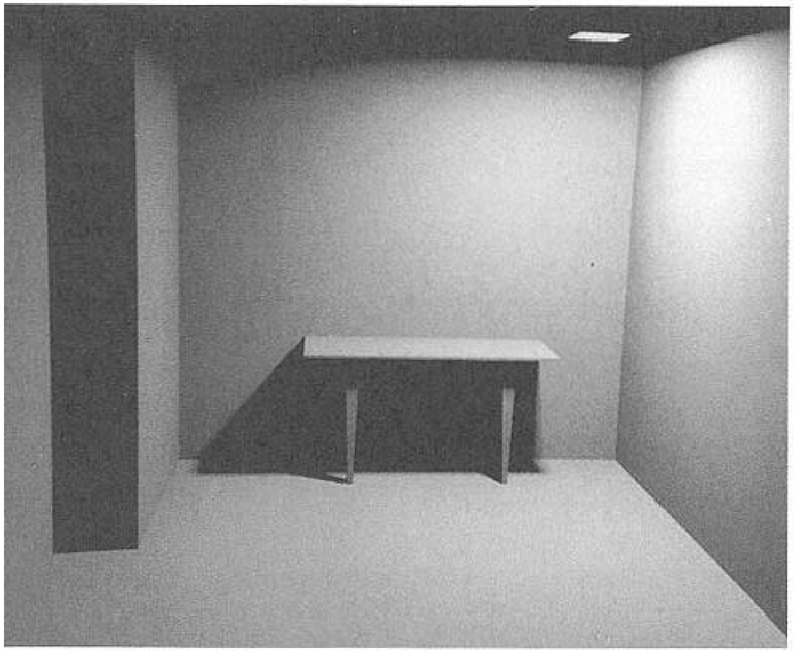
Meshing

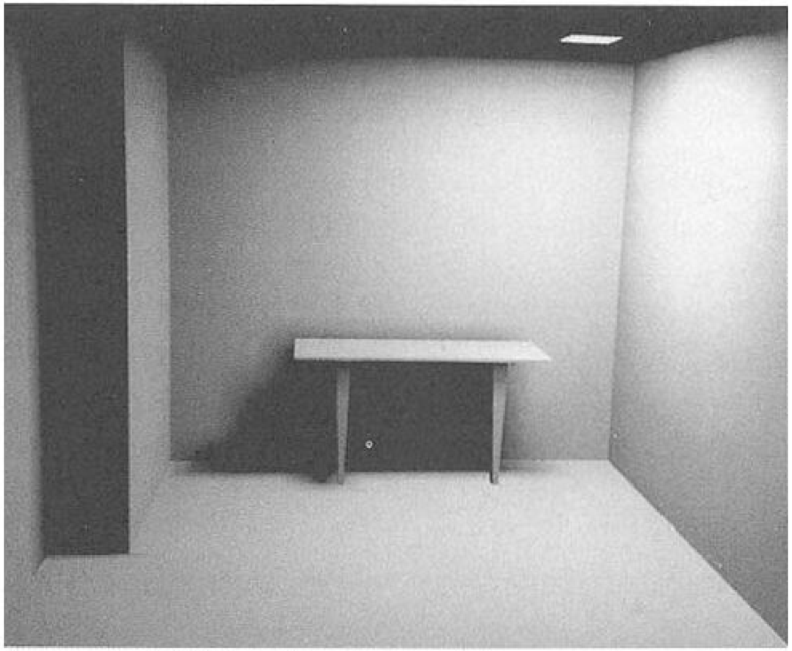


Uniform mesh

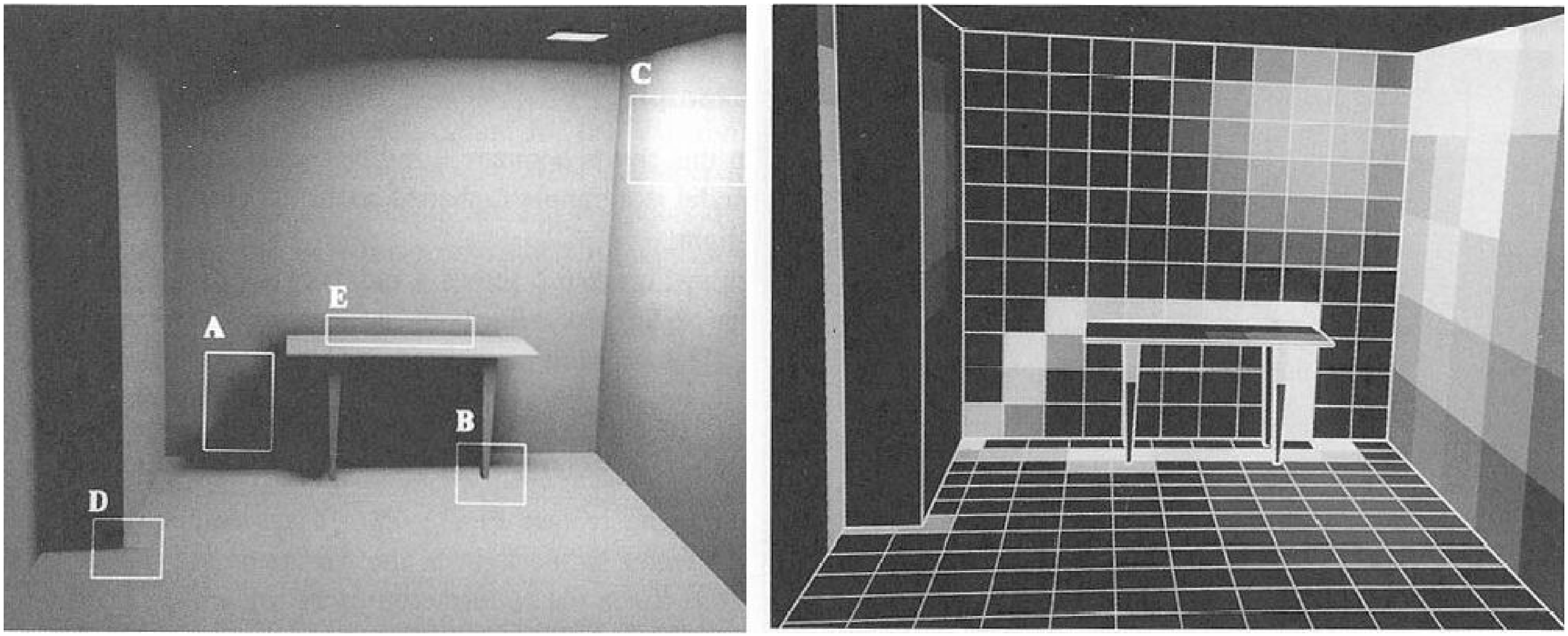


Reference picture & uniform mesh



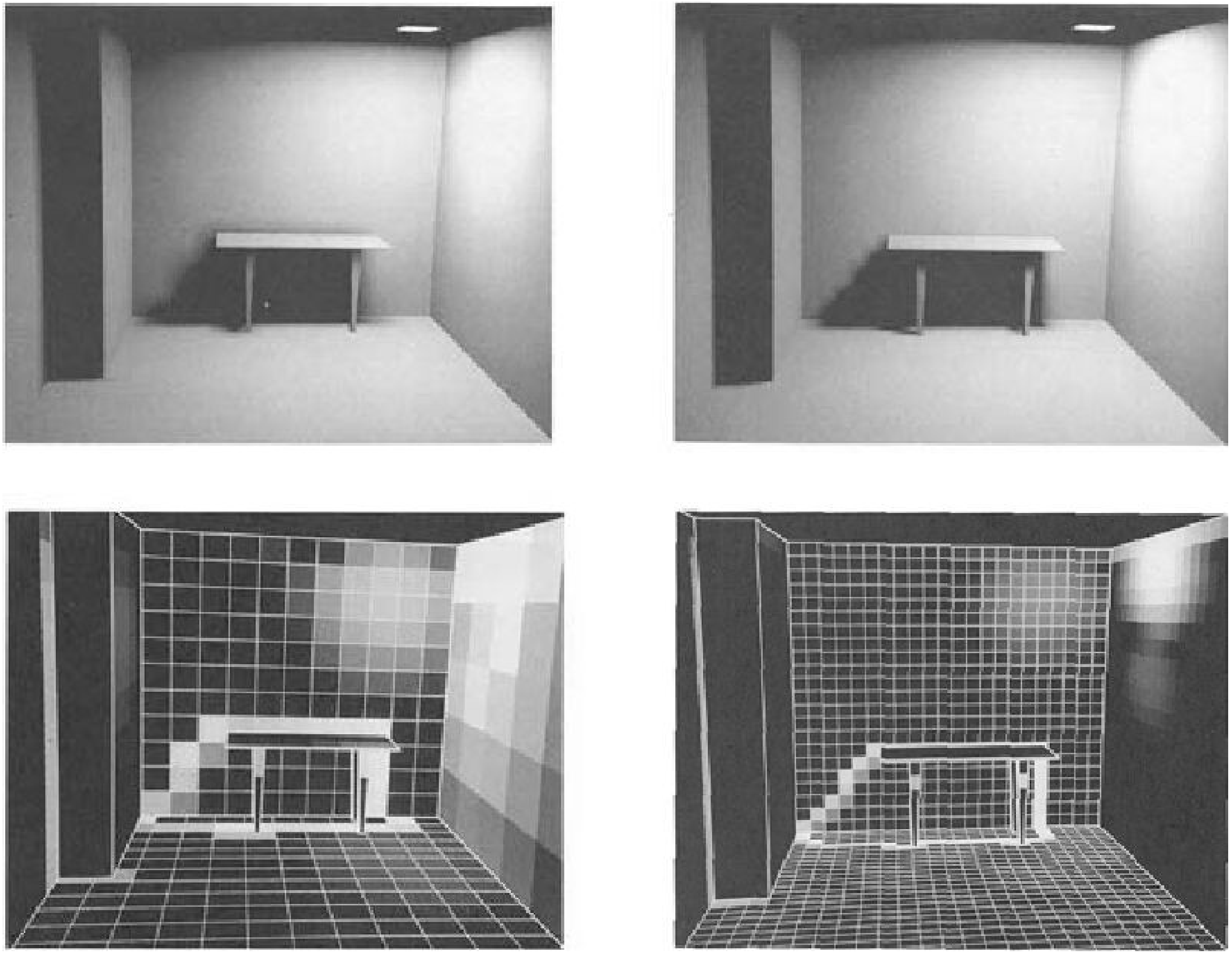


Meshing problems

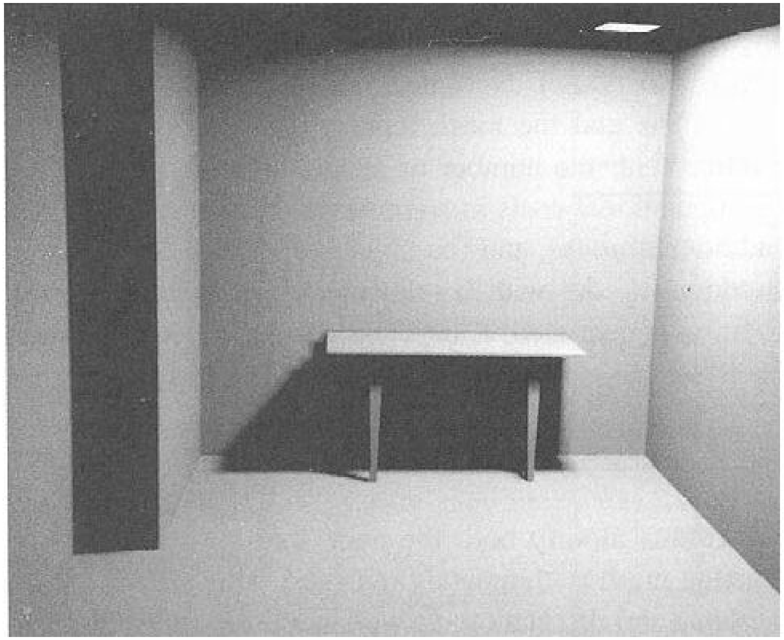


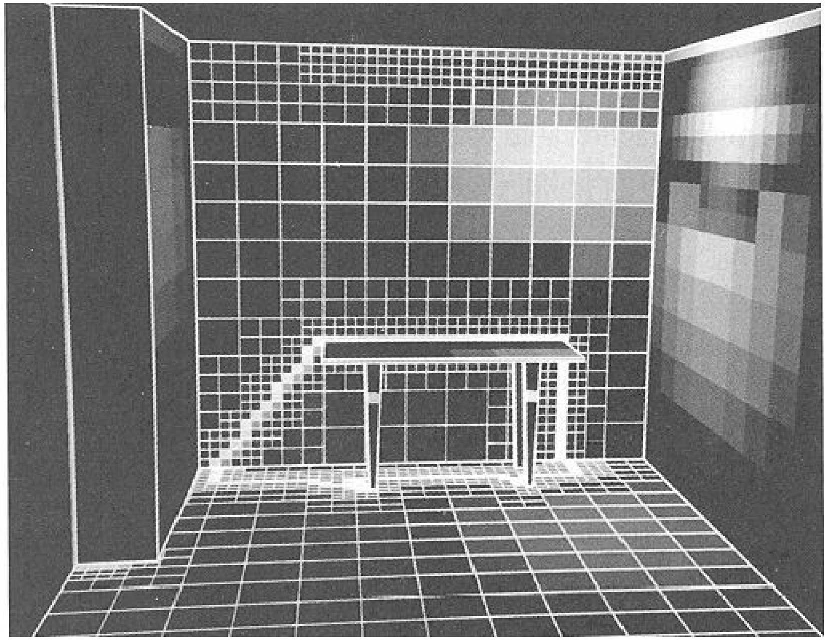
A. Blocky shadows
B. Missing features
C. Mach bands
D. Inappropriate shading discontinuities
E. Unresolved discontinuities

Increase resolution

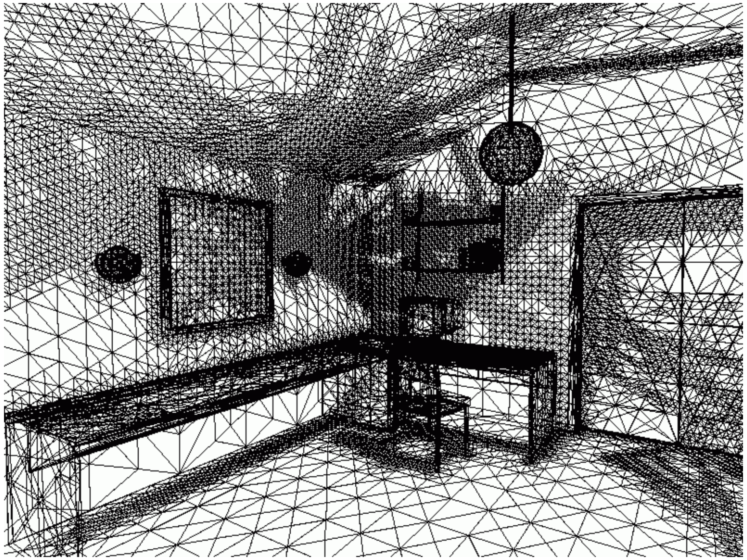


Adaptive meshing

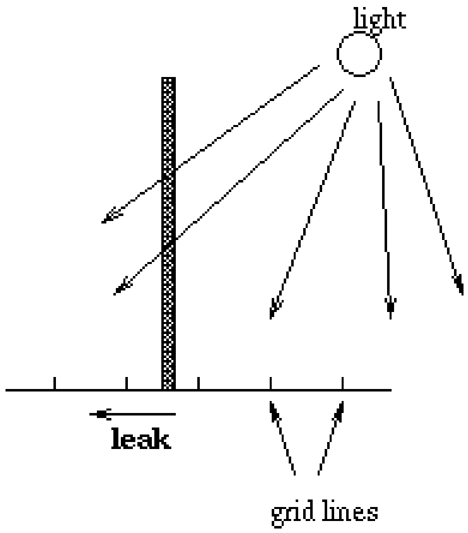


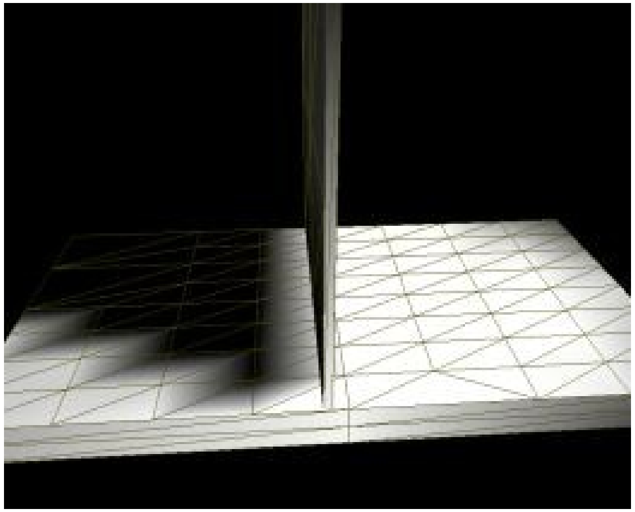


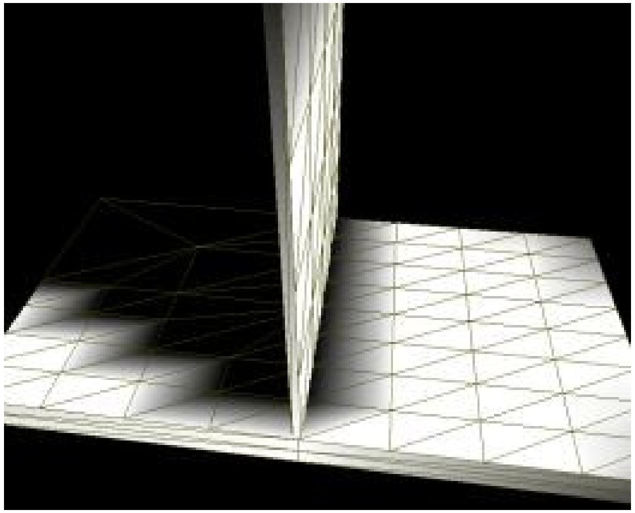
Adaptive meshing



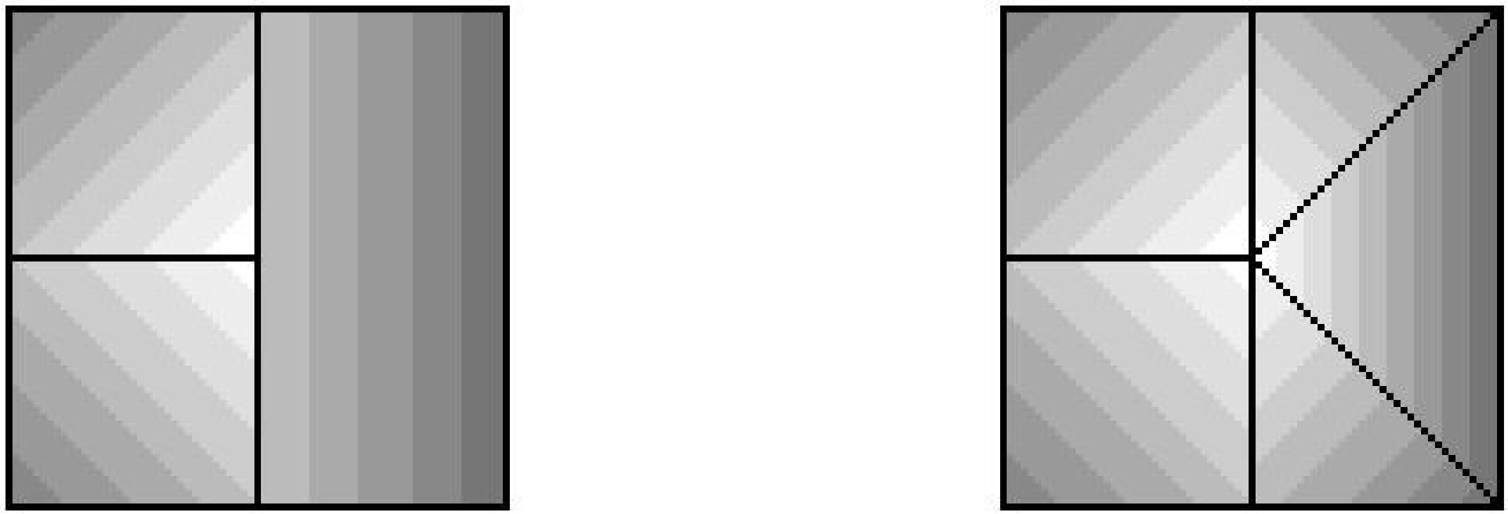
Shadow and light leaks







T-Vertices



Radiosity steps

- Create scene
  - Take care of accurate material and light source definitions
  - During modeling, keep in mind the problems that can occur during application of radiosity
- Make a coarse meshing, compute and inspect results
- Adapt materials and light sources
- Make a “good” meshing
- Compute radiosity results
- Use radiosity results to make one or more images

Radiosity summarized

- Computation of diffuse interreflection
- Result radiosity: intensities for patches
- Viewpoint independent
- Accuracy of results depends on meshing
- High memory and time costs

Povray - radiosity

Mug in a box with 1 red wall,
1 yellow wall, and 4 white walls.

```
light_source {
  < -3.75, 0, 7.>
  color <1,1,1>
}
#include "mug.inc"
object { mug
    . . .
    pigment { color <1,1,1> }
    finish {
      diffuse   0.8
      ambient  0.3
      specular 0.2
    }
}
```

Povray - radiosity

Mug in a box with 1 red wall,
1 yellow wall, and 4 white walls.

```
global_settings {
   radiosity {}
}
light_source {
   < -3.75, 0, 7.>
   color <1,1,1>*0.7
}
#include "mug.inc"
object { mug
     . . .
     pigment { color <1,1,1> }
     finish {
        diffuse    0.8
        ambient  0.0
        specular 0.2
     }
}
```

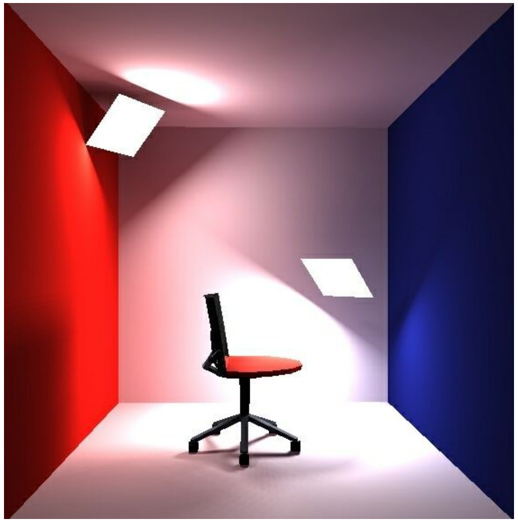
Rendering after radiosity step - 1

- Direct use of radiosity results:
  - Radiosity results replaces diffuse computation in local illumination function
  - Gouraud shading
  - Fine meshing to get nice shadows

Rendering after radiosity step - 2

- Use radiosity results for indirect illumination only
  - Re-compute direct light during rendering
  - Less fine meshing required

Rendering after radiosity step - 3



Rendering after radiosity step - 4

- Use radiosity results to re-compute direct and indirect illumination
  - Use radiosity results as emission values
  - Regard all patches as light sources
  - Coarse meshing suffices

Radiosity

- Viewpoint independent
- Diffuse (inter)reflections
- Color bleeding
- Area light sources
- Soft shadows
- Light paths: D*
  (and (D|S)*)

- High memory usage
- Meshing determines accuracy results
- High computation times

Ray tracing

- Viewpoint dependent
- Real specular reflections
- Transmission + refraction
- Point light sources
- Sharp shadows
- Light paths LDS*E
  (and LS*DS*E)

- Low memory usage

radiosity vs ray tracing