
Realistic Image Synthesis

- Radiosity -

Philipp Slusallek
Karol Myszkowski
Gurprit Singh

Overview

- **Today**
 - Radiosity Equation
 - Basic Radiosity formulation
 - Form Factor computation
 - Radiosity algorithms
 - Meshing techniques
 - Finite Element Radiosity formulation
 - Rendering Equation revisited: transport mechanisms

Reading Materials

Basic:

- **F.X. Sillion, C. Puech, Radiosity & Global Illumination**
- **Cohen MF., Wallace JR., Radiosity and Realistic Image Synthesis**
- **Foley J.D., van Dam A., Feiner S.K., Hughes J.F., Computer Graphics: Principles and Practice, Chapter 16.13**

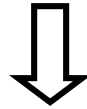
Advanced:

- **Glassner A.S., Principles of Digital Image Synthesis, Volume II, Chapters 17 and 18**

Derivation of the Radiosity Equation

- **Simplification of the rendering equation**

$$L(x, \theta_o, \varphi_o) = L_e(x, \theta_o, \varphi_o) + \int_{\Omega} \rho_{bd}(x, \theta_o, \varphi_o, \theta, \phi) L_i(x, \theta, \varphi) \cos \theta d\omega$$



$$B_i = E_i + \rho_i \sum_{j=1}^N B_j F_{ij}$$

- All surfaces in the scene are Lambertian
- Equation expressed in terms of the radiosity quantities
- Integration domain split into N pieces corresponding to discrete patches in the scene
- Constant radiosity and reflectance assumptions for each patch

Radiosity Equation and Form Factors

- The radiosity equation can be formulated as:

$$B_i = E_i + \rho_i \sum_{j=1}^N B_j \frac{1}{A_i} \int_{x \in P_i} \int_{y \in P_j} \frac{\cos \theta \cos \theta'}{\pi r^2} V(x, y) dx dy$$

$$B_i = E_i + \rho_i \sum_{j=1}^N B_j F_{ij}$$

- where F_{ij} is the form factor:

$$F_{ij} = \frac{1}{A_i} \int_{x \in P_i} \int_{y \in P_j} \frac{\cos \theta \cos \theta'}{\pi r^2} V(x, y) dx dy$$

Form Factors: Basic Properties

$$F_{ij} = \frac{1}{A_i} \int_{x \in P_i} \int_{y \in P_j} \frac{\cos \theta \cos \theta'}{\pi r^2} V(x, y) dx dy$$

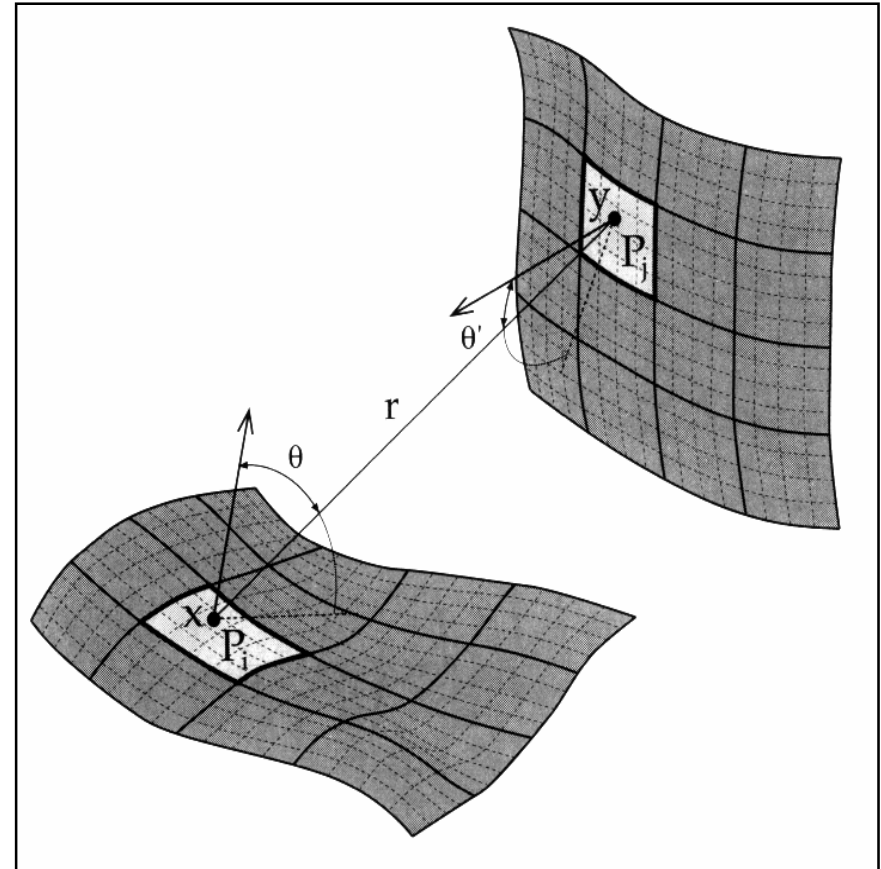
- **The form factor describes the fraction of energy which leaves element i and arrives at element j .**

- it is a dimensionless quantity,
- depends only on the scene geometry,
- the reciprocity law:

$$A_i F_{ij} = A_j F_{ji}$$

- for closed environments:

$$\sum_{j=1}^N F_{ij} = 1$$



- for convex surfaces (polygons):

$$F_{ii} = 0$$

Radiosity Equation

$$B_i = E_i + \rho_i \sum_j B_j F_{ij} \Rightarrow (\mathbf{1} - \rho\mathbf{F})\mathbf{B} = \mathbf{E}$$

$$\begin{pmatrix} 1 - \rho_1 F_{11} & -\rho_1 F_{12} & \cdots & -\rho_1 F_{1N} \\ -\rho_2 F_{21} & 1 - \rho_2 F_{22} & \cdots & -\rho_2 F_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ -\rho_N F_{N1} & -\rho_N F_{N2} & \cdots & 1 - \rho_N F_{NN} \end{pmatrix} \begin{pmatrix} B_1 \\ B_2 \\ \vdots \\ B_N \end{pmatrix} = \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_N \end{pmatrix}$$

- B, E depend on wavelengths (often just RGB is considered)
- $E_i > 0$ only for light sources
- Energy conservation: $\sum_j F_{ij} \leq 1$ and $\rho_i \leq 1$
- ➔ Matrix is diagonally dominant
 - ➔ Ensures convergence for many iterative solvers, e.g., the Gauss-Seidel method

Interpreting the Radiosity Equation

- **More intuitive formulation**

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

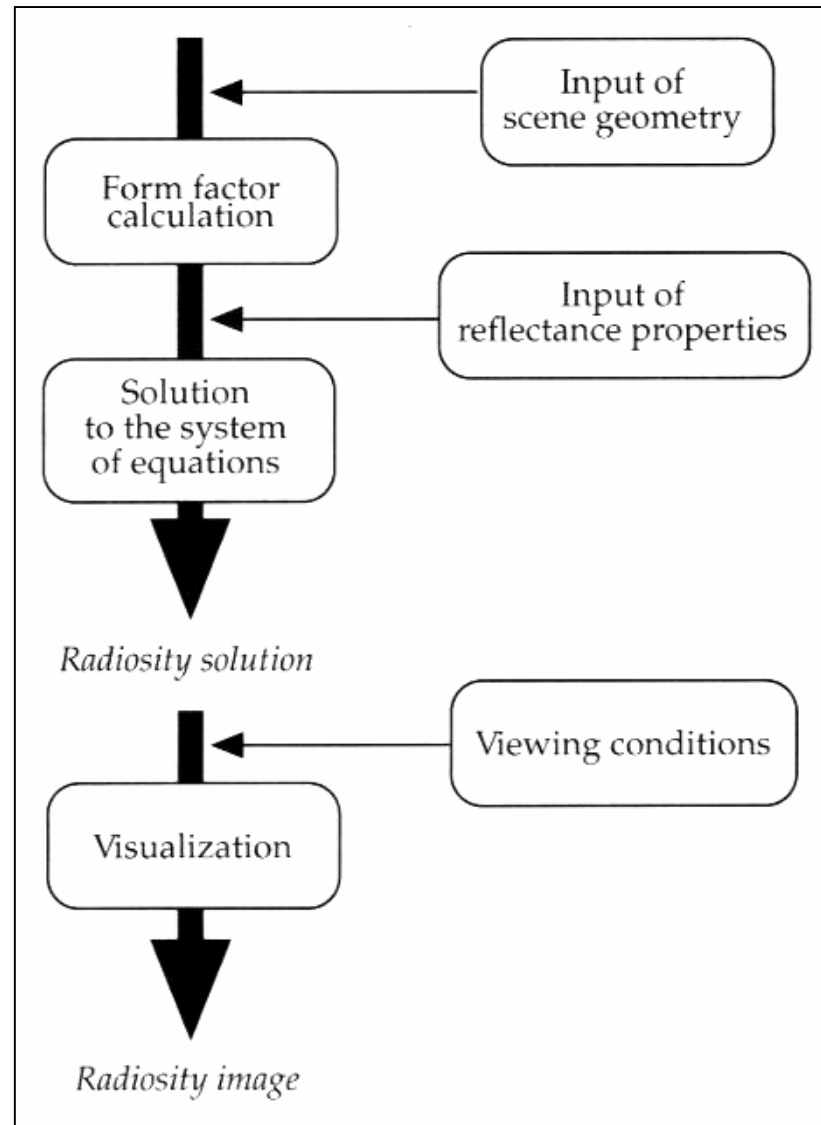
$$B_i A_i = E_i A_i + \rho_i \sum_j B_j A_i F_{ij}$$

since $A_i F_{ij} = A_j F_{ji}$

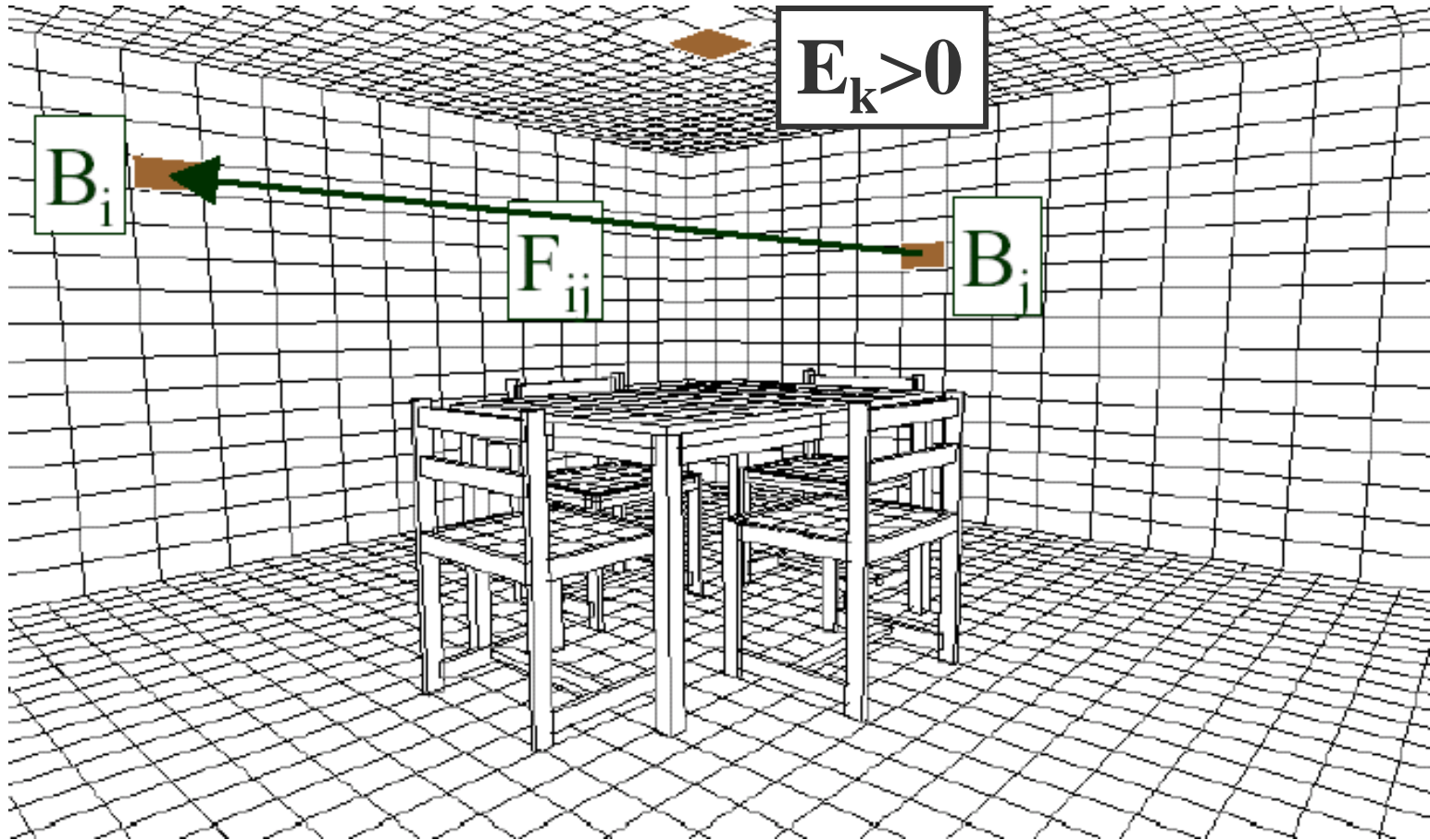
$$B_i A_i = E_i A_i + \rho_i \sum_j B_j A_j F_{ji}$$

- **The power $B_j A_j$ emitted by element j is multiplied by F_{ji} and contributes to the reflected power $B_i A_i$ of element i .**

Radiosity Solution

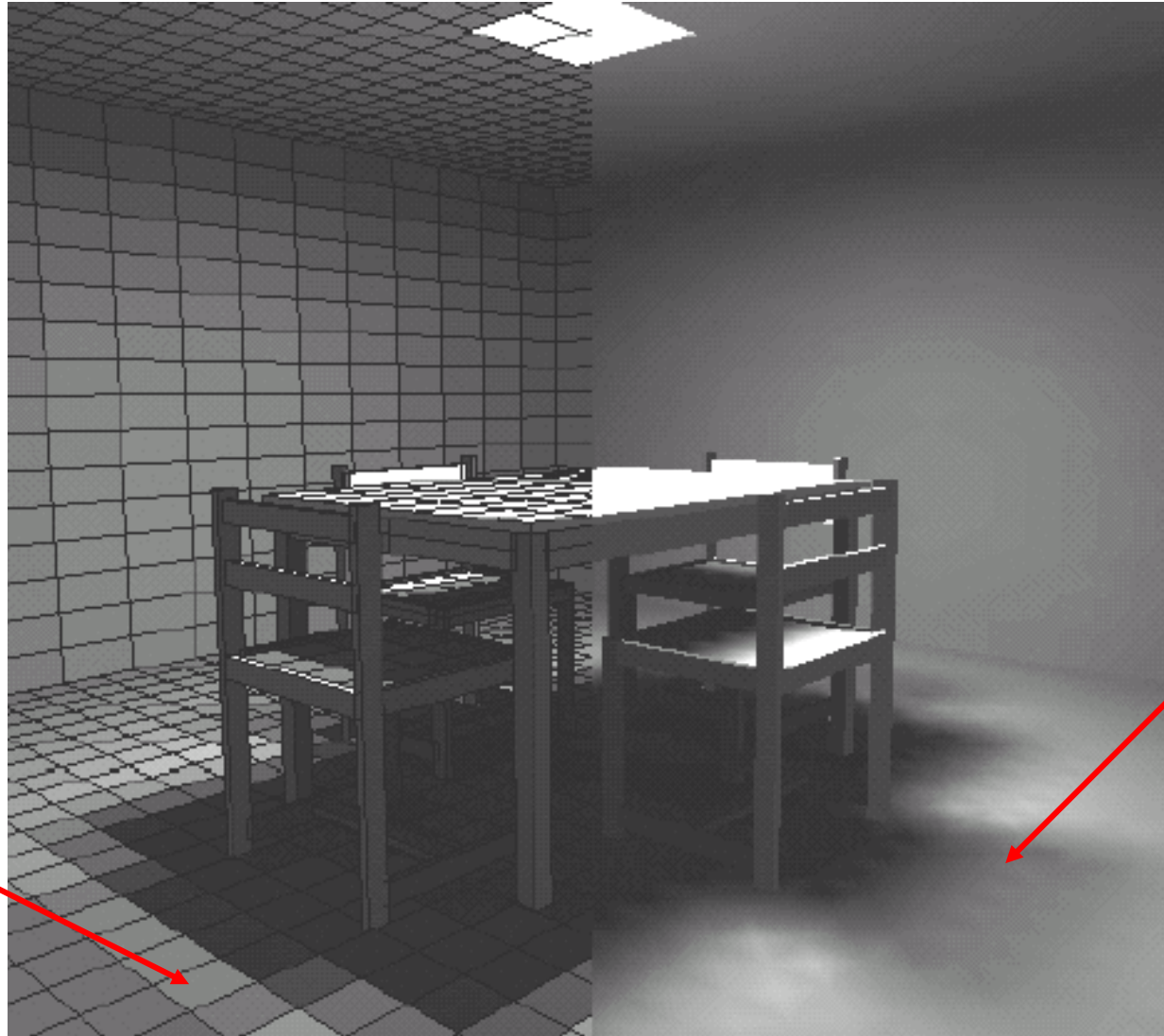


Radiosity Solution: Scene Meshing



Radiosity Solution: Results

Direct results of the radiosity equation solution: the assumption of constant radiosity value for each mesh element



Linearly interpolated radiosity values between mesh vertices

Form Factors

Differential – differential :

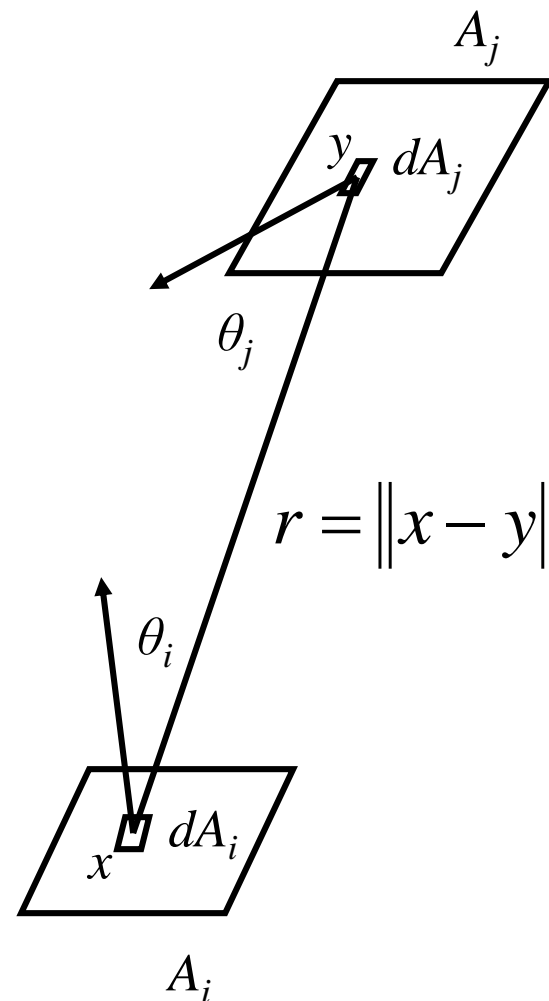
$$F_{dA_i, dA_j} = \frac{\cos \theta_i \cos \theta_j}{\pi r^2} V(x, y) dA_j$$

Differential – finite :

$$F_{dA_i, A_j} = \int_{A_j} \frac{\cos \theta_i \cos \theta_j}{\pi r^2} V(x, y) dA_j$$

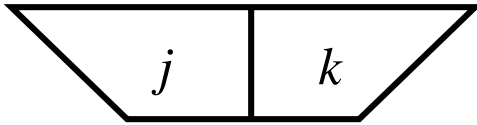
Finite – finite :

$$F_{A_i, A_j} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos \theta_i \cos \theta_j}{\pi r^2} V(x, y) dA_j dA_i$$

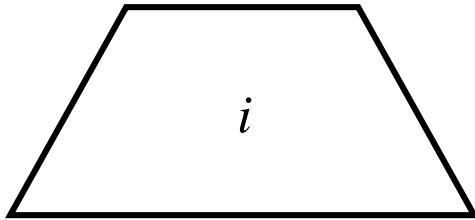


Form Factor Algebra

- Useful to determine the full form factor through decomposing elements into simpler shapes or sub-elements



Additivity property: $F_{i,(j+k)} = F_{i,j} + F_{i,k}$

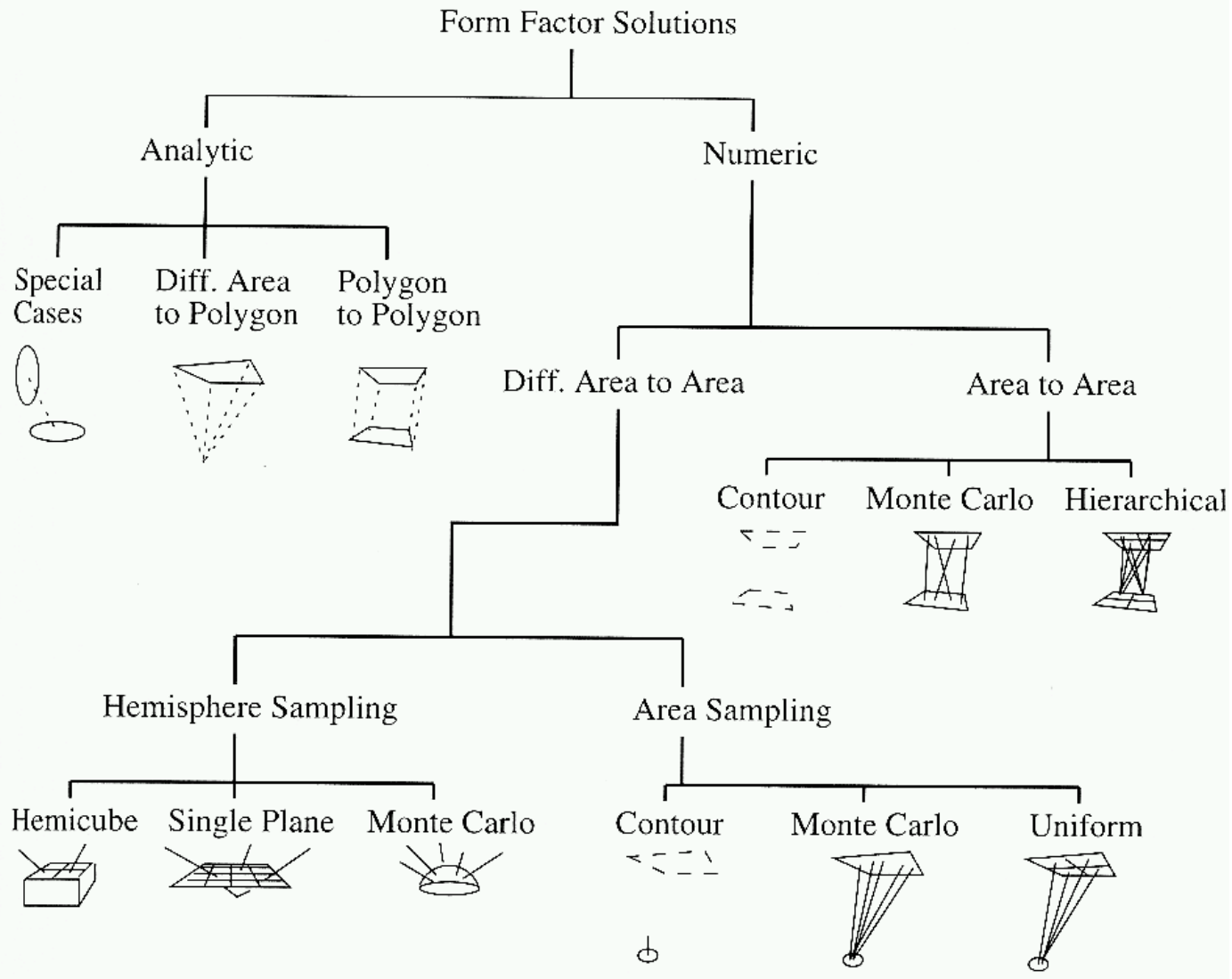


Weighted average : $F_{(j+k),i} = \frac{A_j F_{i,j} + A_k F_{i,k}}{A_j + A_k}$

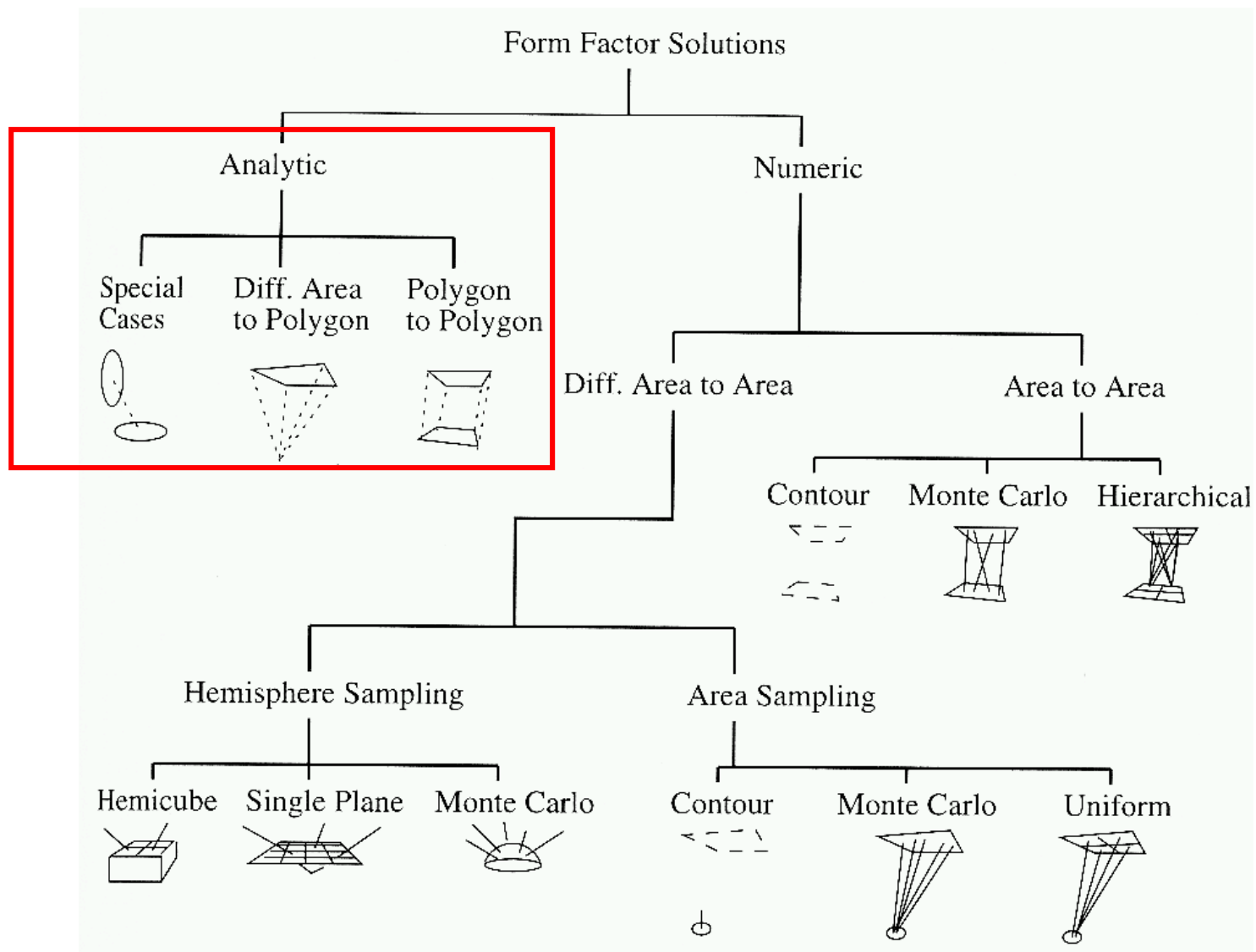
- Example of the additivity property application:

$$F_{dA_i, A_j} = \int_{A_j} \frac{\cos \theta_i \cos \theta_j}{\pi r^2} V(x, y) dA_j \quad \Rightarrow \quad F_{dA_i, A_j} \approx \sum_{k=1}^m \frac{\cos \theta_i^k \cos \theta_j^k}{\pi (r^k)^2} V(x, y^k) \Delta A_j^k$$

Classification of Form Factor Solutions



Classification of Form Factor Solutions



Analytic Form Factors

- **Possible only for simple shapes**

- Already Lambert (1760) found a closed form expression for the form factor between a differential element and a polygon

- **Example: point-to-disc**

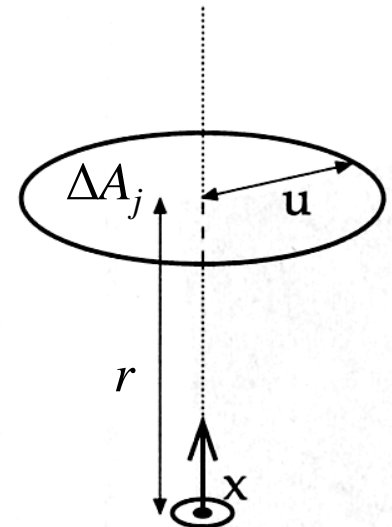
- point x is must be located on the line perpendicular to the disc and passing through it center

$$F_{dx, A_j} = \frac{\Delta A_j}{\pi r^2 + \Delta A_j}$$

- Useful approximation for an oriented disc (no singularity for $r \rightarrow 0$)

$$F_{dx, A_j} = \frac{\Delta A_j \cos \theta_i \cos \theta_j}{\pi r^2 + \Delta A_j}$$

- For another useful formula refer to the Cohen&Wallace book Chapter 4.6

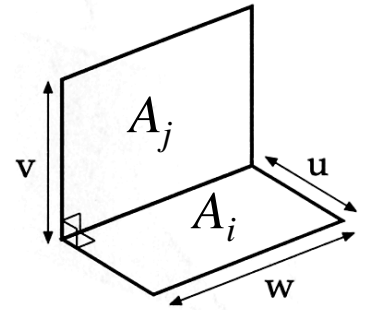


Analytic Form Factors

- **Rectangle-to-rectangle**

$$X = \frac{u}{w} \quad Y = \frac{v}{w}$$

$$F_{A_i A_j} = \frac{1}{\pi X} \left\{ X \tan^{-1} \left(\frac{1}{X} \right) + Y \tan^{-1} \left(\frac{1}{Y} \right) - \sqrt{X^2 + Y^2} \tan^{-1} \left(\frac{1}{\sqrt{X^2 + Y^2}} \right) \right\}$$
$$+ \frac{1}{4\pi X} \left\{ \ln \left[\frac{(1 + X^2)(1 + Y^2)}{1 + X^2 + Y^2} \right] + X^2 \ln \left[\frac{X^2(1 + X^2 + Y^2)}{(1 + X^2)(X^2 + Y^2)} \right] + Y^2 \ln \left[\frac{Y^2(1 + X^2 + Y^2)}{(1 + Y^2)(X^2 + Y^2)} \right] \right\}$$



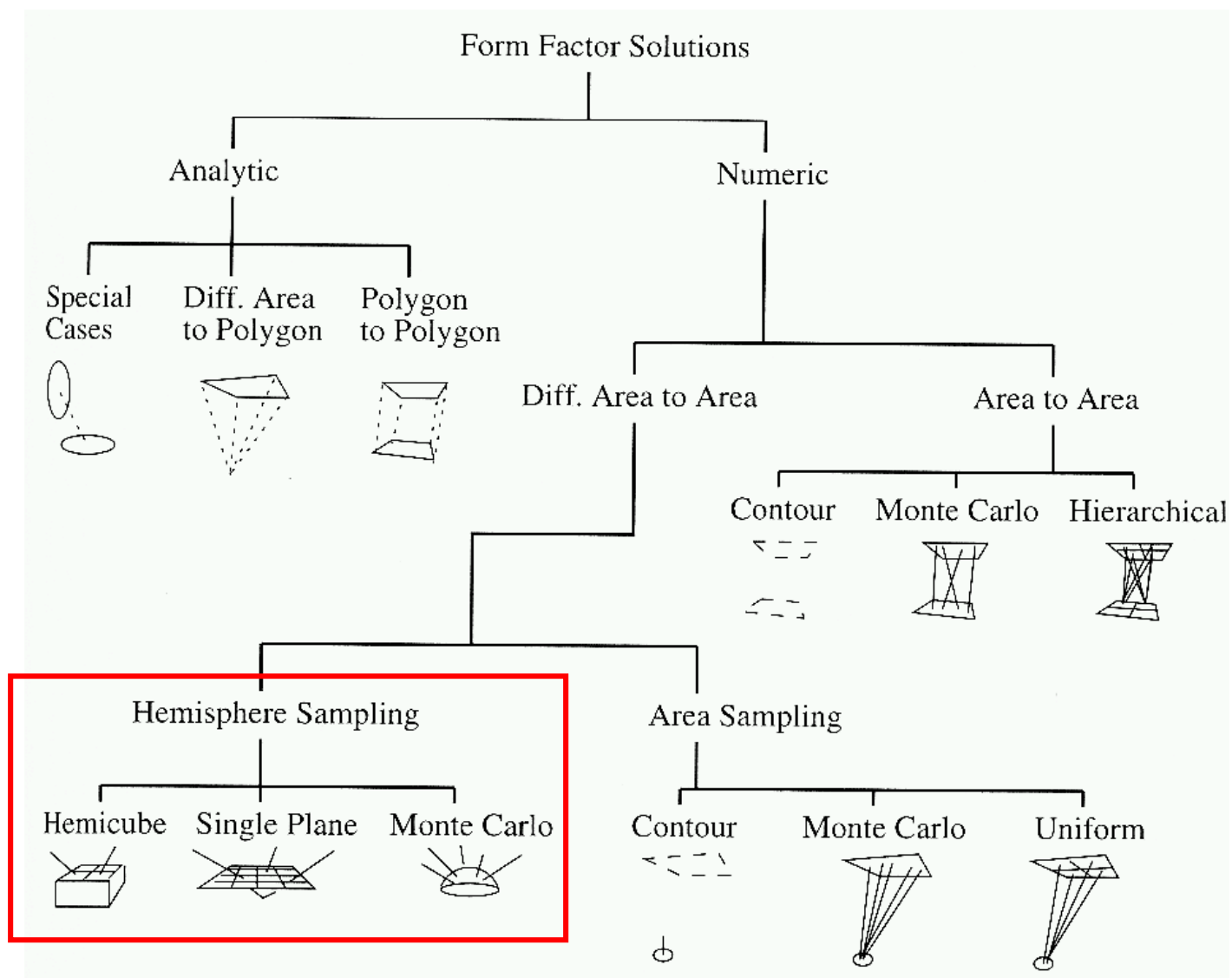
- **General polygon-to-polygon**

- a closed form solution of the form factor between two general polygons was developed just in 1993 by Schroder and Hanrahan (Siggraph)
- very complex formulae based on the dilogarithm function

Visibility for Analytic Form Factors

- **When it is known that no occluders between two interacting surfaces are located ($V(x,y)=1$) then the analytic form factor calculation offer the best accuracy at reasonable cost for planar surfaces.**
- **For the partial occlusion case, the analytical formula usually cannot be derived.**
- **Computing visibility $V(x,y)$ in the form factor integral is like solving a hidden surface problem from the point of view of each patch in the scene:**
 - Usually the most costly part of the radiosity computation
- **Basic methods of $V(x,y)$ estimation:**
 - Ray tracing: easy to implement, but costly. Just shoot k rays between two patches x and y , and count the number of rays l which reach the target patch, then $V(x,y)=l/k$.
 - Shaft-culling can be used to speedup the visibility computation.

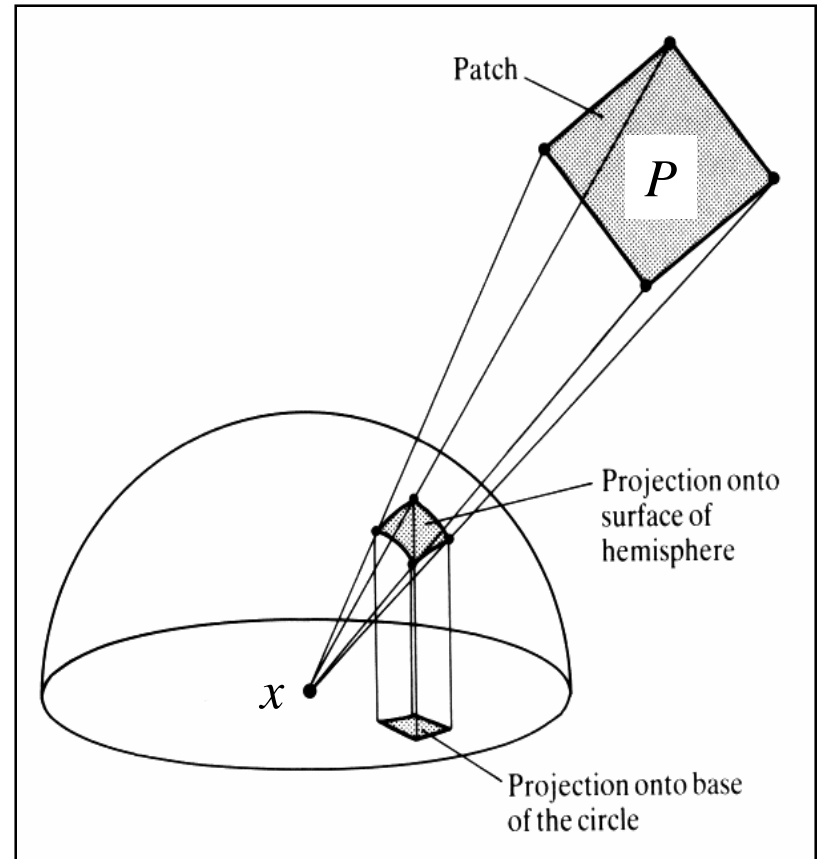
Classification of Form Factor Solutions



Nusselt Analog

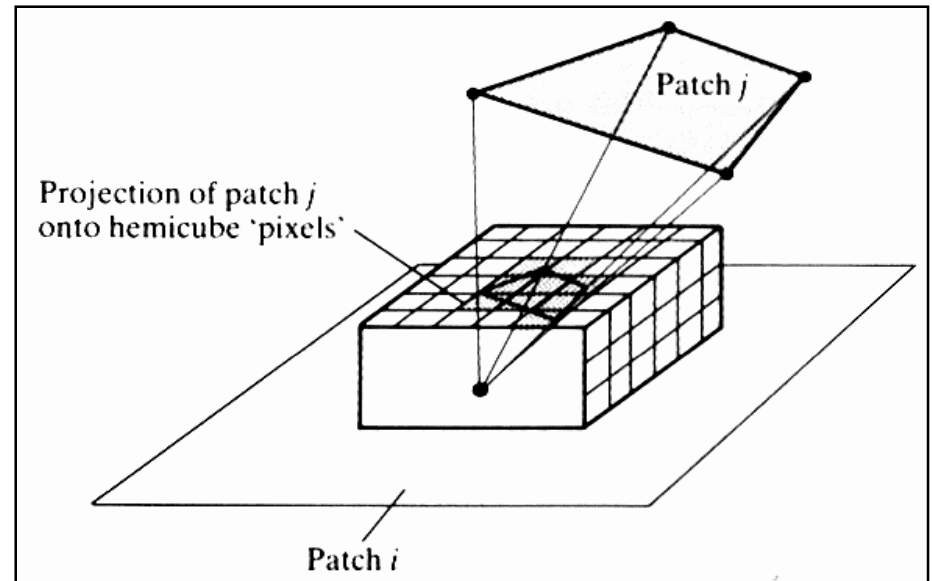
- **Form factor between a point x and a patch P :**
 - 1 Project patch P onto the unit hemisphere centered at x and then orthogonally down onto the base circle.
 - 2 The form factor is, then, the area projected on the base of the hemisphere divided by the area of the base (π).

$$F_{x,P} = \frac{1}{\pi} \int_{\Omega} \cos \theta d\omega$$



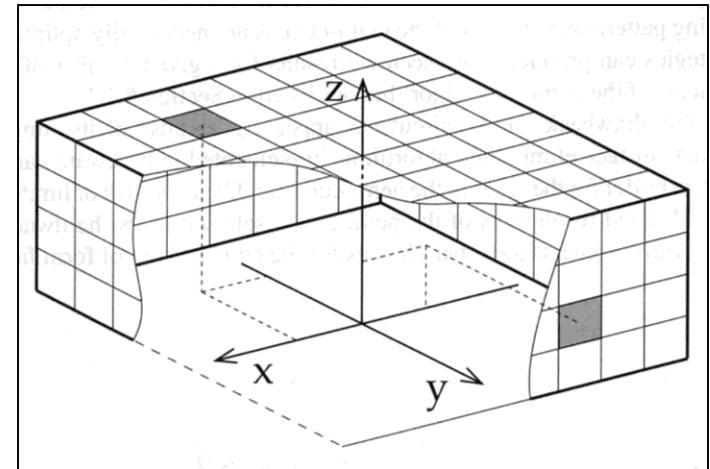
Hemicube (1)

- **Based on the Nusselt analog**
 - Instead of hemisphere use “hemicube”
- **Subdivide the hemicube faces into small, usually square (“pixel”) areas ΔA**
- **Precompute analytically delta-form factors ΔF for those areas**



$$\Delta F_{top-face} = \frac{\Delta A}{\pi(1 + x^2 + y^2)^2}$$

$$\Delta F_{side-face} = \frac{z\Delta A}{\pi(1 + z^2 + y^2)^2}$$



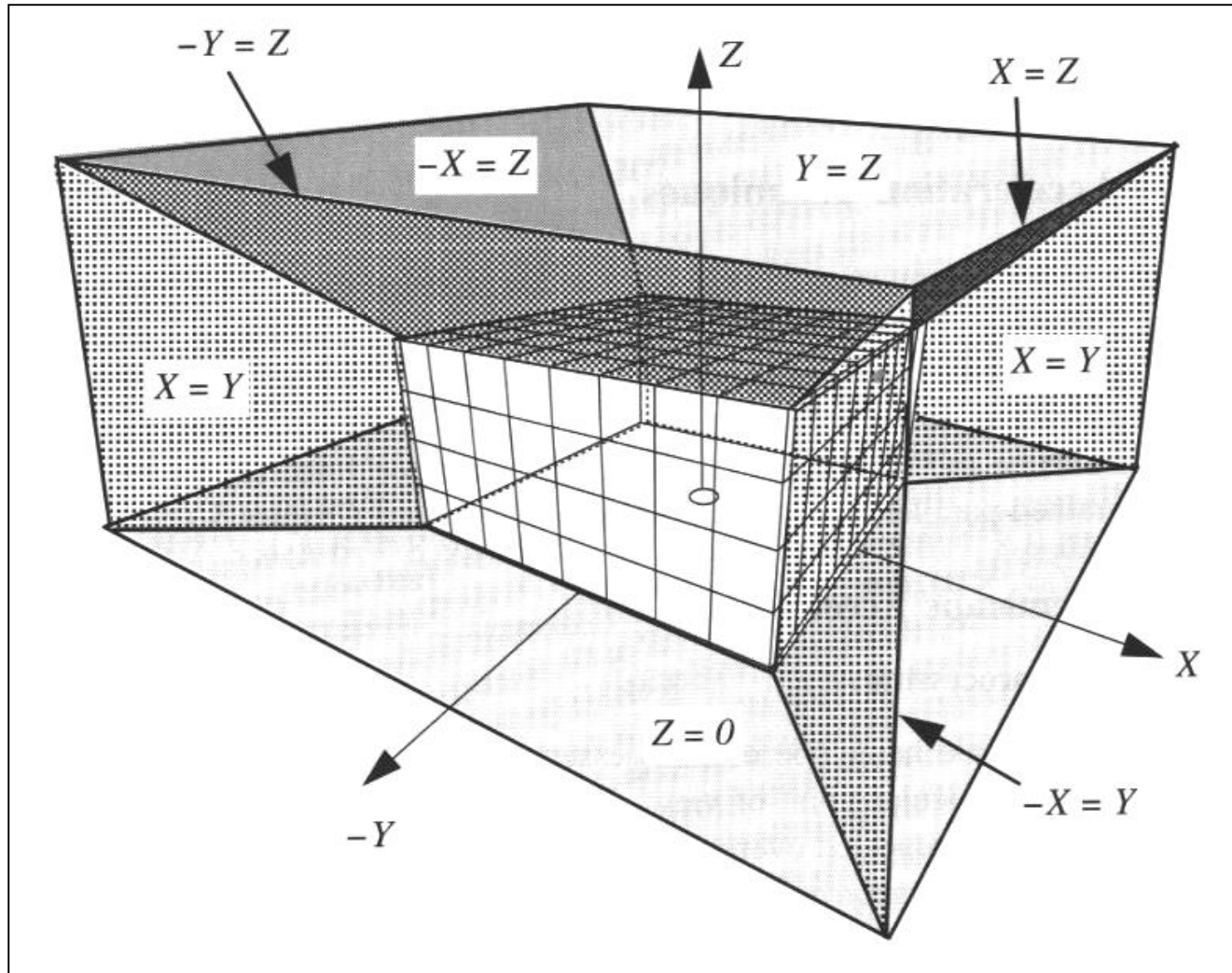
Hemicube (2)

- **Encode using unique colors C_j each patch j in the scene (the item buffer algorithm)**
- **Project (scan convert) the whole scene onto the hemicube faces**
 - Graphics hardware can be used for this purpose
- **For each uniquely color encoded patch j sum the delta form factors ΔF_q for areas ΔA_q which represent the patch projection onto the hemicube**

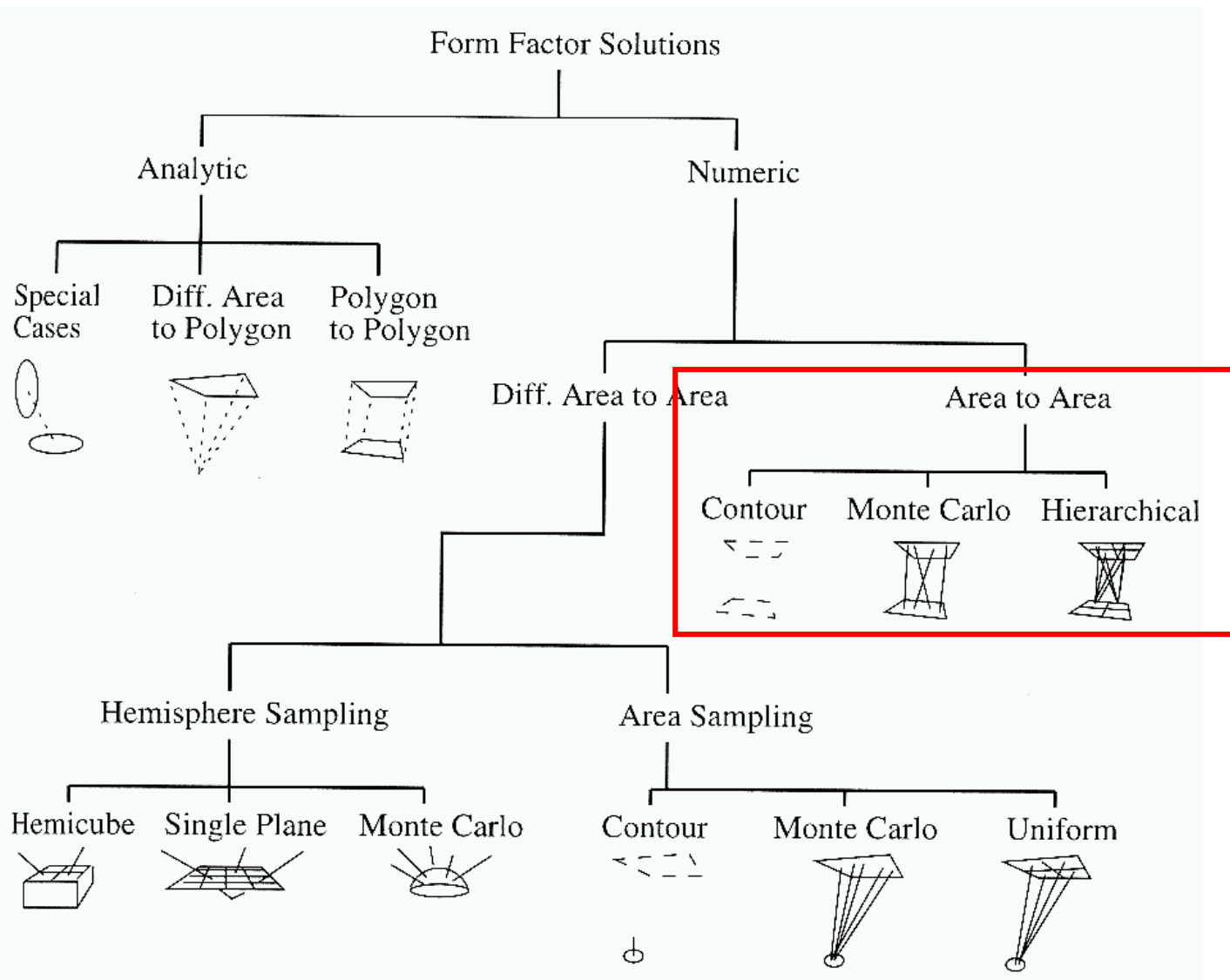
$$F_{x,P_j} = \sum_{q \in C_j} \Delta F_q$$

- **The speed and accuracy of the hemicube method for the form factor computation can be affected by changing the size and number of discrete areas on the faces of the hemicube.**

Hemicube Positioning and Clipping



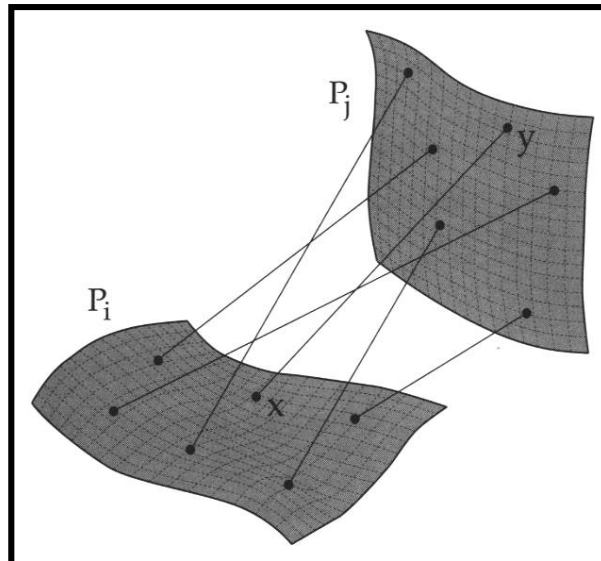
Classification of Form Factor Solutions



Monte Carlo Integration

- Form factor calculation between two patches P_i and P_j requires solution of double integral over surfaces of the patches.
- A random sample in this 4-D space is a pair of uniformly distributed random points from each patch surface.
- The probability density is the constant $1/(A_i A_j)$. If we use one or K samples, we get the following estimates:

$$F_{ij} \approx A_j \frac{\cos \theta \cos \theta'}{\pi r^2} V(x, y) \quad F_{ij} = \frac{1}{K} \sum_{k=1}^K A_j \frac{\cos \theta_k \cos \theta'_k}{\pi r_k^2} V(x_k, y_k) = \frac{1}{K} \sum_{k=1}^K F_{ij}^k$$



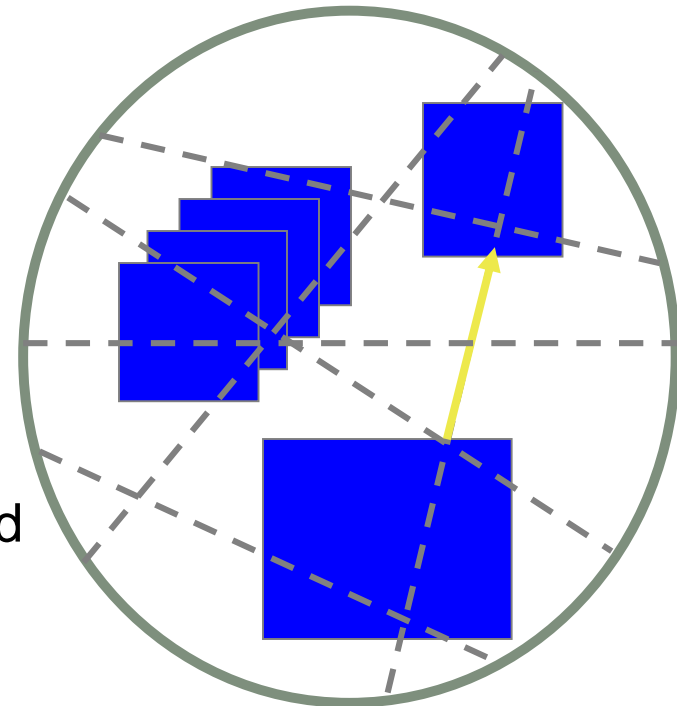
Line Densities

- Shoot the global lines randomly distributed in space
- The form factor between a pair of patches A_i and A_j can be estimated as:

$$F_{A_i, A_j} = \frac{N(A_i, A_j)}{N(A_i)}$$

where

- $N(A_i, A_j)$ is the number of lines crossing both patches without any occluder located between A_i and A_j
- $N(A_i)$ is the number of lines crossing A_i



Radiosity Solution Techniques

- **Classical Radiosity**

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

$$\begin{Bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{Bmatrix} = \begin{Bmatrix} B_1^e \\ B_2^e \\ \vdots \\ B_n^e \end{Bmatrix} + \underbrace{\begin{Bmatrix} \rho_1 & 0 & \cdots & 0 \\ 0 & \rho_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \rho_n \end{Bmatrix}}_{\mathbf{I}} \begin{Bmatrix} F_{11} & F_{12} & \cdots & F_{1n} \\ F_{21} & F_{22} & \cdots & F_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ F_{n1} & F_{n2} & \cdots & F_{nn} \end{Bmatrix} \begin{Bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{Bmatrix}$$

Radiosity Solution Techniques

- **Matrix formulation**

$$\begin{pmatrix} 1 - \rho_1 F_{11} & -\rho_1 F_{12} & \cdots & -\rho_1 F_{1N} \\ -\rho_2 F_{21} & 1 - \rho_2 F_{22} & \cdots & -\rho_2 F_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ -\rho_N F_{N1} & -\rho_N F_{N2} & \cdots & 1 - \rho_N F_{NN} \end{pmatrix} \begin{pmatrix} B_1 \\ B_2 \\ \vdots \\ B_N \end{pmatrix} = \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_N \end{pmatrix}$$

$$B = B_e + \mathbf{T}B$$

$$B = [\mathbf{I} - \mathbf{T}]^{-1} B_e$$

Matrix Properties

- **Size:** $n \times n$
 - Where n is the number of elements (more general the number of basis functions)
- Sparsity: $T_{ij} = 0$ iff: $F_{ij} = 0$ or $\rho_i = 0$
 - Depends mostly on the visibility between elements
 - For two separate rooms the matrix will be block diagonal \rightarrow two subproblems \rightarrow more efficient
- Symmetry
 - Not symmetric in the form shown in the previous slide
 - Can be made symmetric due to the reciprocity property of form factors ($F_{ij}A_i = F_{ji}A_j$)
- **Matrix is diagonally dominant**
 - Ensures convergence for many iterative solvers, e.g., the Gauss-Seidel method
$$\sum_{j=1, j \neq i}^n |T_{ij}| \leq |T_{ii}|, \forall i$$

Matrix Properties

- **Spectral radius**

- Indicates the speed of iterative methods convergence
 - More iterations needed for surfaces with higher reflectance
- If \mathbf{T} has a norm less than one, then $\mathbf{K} = [\mathbf{I} - \mathbf{T}]$ is invertible and the corresponding Neumann series of successive multiplications of \mathbf{T} will converge to the inverse

$$\text{If } \|\mathbf{T}\| < 1 \quad \text{then} \quad \mathbf{K}^{-1} = [\mathbf{I} - \mathbf{T}]^{-1} = \sum_{a=0}^{\infty} \mathbf{T}^a$$

- **Condition**

- Measures the sensitivity of solution to small perturbations in the input
- Radiosity matrices are usually well conditioned, so most solution methods can be used

Radiosity Solution Techniques

- **Matrix Inversion:** $B = [\mathbf{I} - \mathbf{T}]^{-1} B_e$ **Cost= $O(n^3)$**
- **Jakobi Iteration (mimics the Neumann series)**

$$B^0 = B^e$$

$$B^{k+1} = B^e + \mathbf{T}B^k$$

- **Meaning of terms B^k : Light after max. k reflections**

$$B^0 = B^e$$

$$B^1 = B^e + \mathbf{T}B^e$$

$$B^2 = B^e + \mathbf{T}B^e + \mathbf{T}^2B^e$$

- **Good solution after only few iterations**
 - Exponential decay of T^k
 - Cost = $O(n^2)$
- **Self correcting (numerically stable) solution**
 - Convergence to a fix-point even after perturbations

Radiosity Solution Techniques

- **Jacobi Iteration**

until converged

forall i

$$\hat{B}_i^{k+1} = E_i + \rho_i \sum_j B_j^k F_{ij}$$

forall i

$$B_i^{k+1} = \hat{B}_i^{k+1}$$

- **Gauss-Seidel Iteration (aka Gathering)**

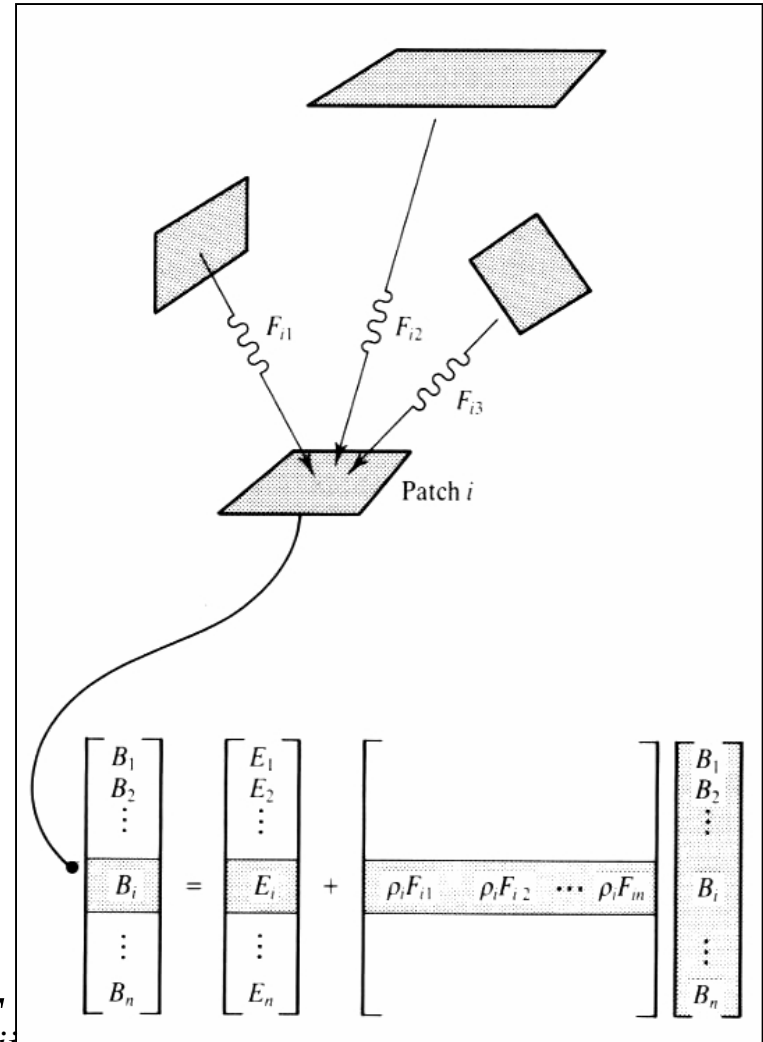
- Faster convergence and simpler implementation

- Immediately uses B_i terms computed in the previous steps

until converged

forall i

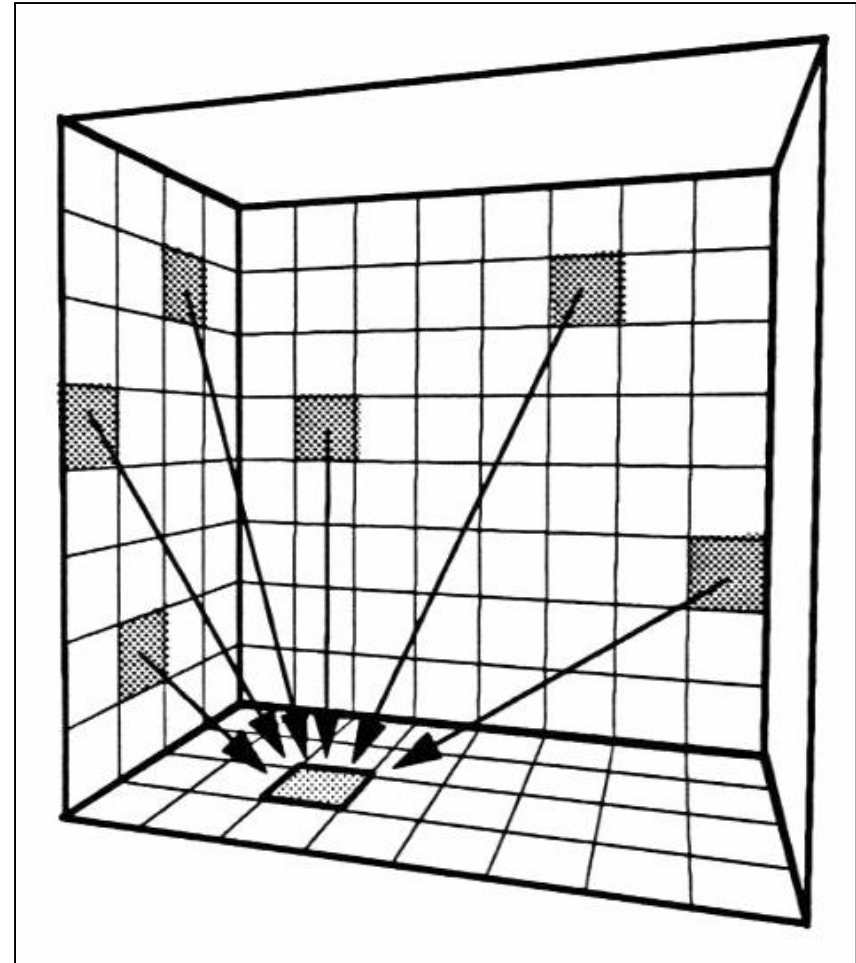
$$B_i^{k+1} = E_i + \rho_i \sum_{j=1}^{i-1} B_j^{k+1} F_{ij} + \rho_i \sum_{j=i+1}^n B_j^k F_{ij}$$



Radiosity Solution Techniques

Gathering:

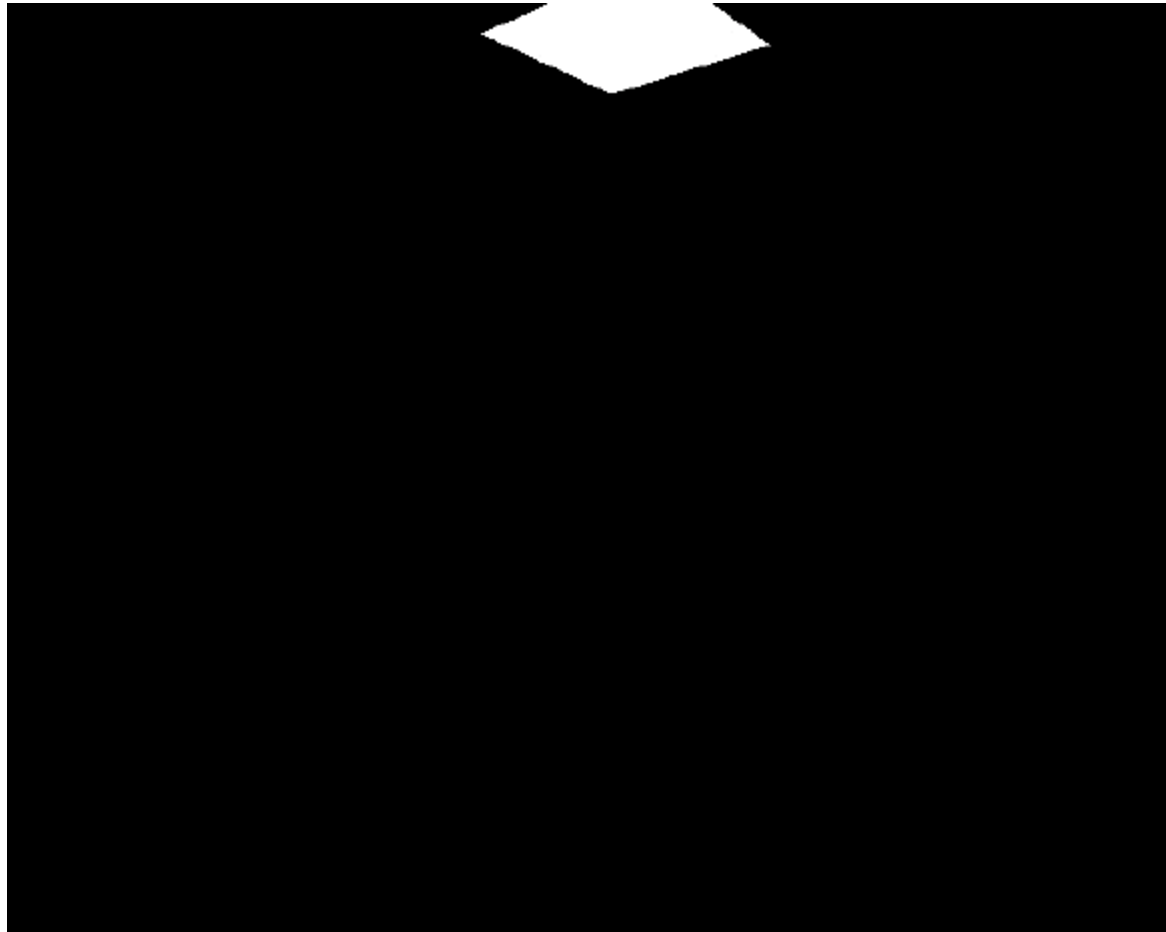
- A Gauss-Seidel step in which the radiosity of a single patch i is updated by *gathering* radiosities from all other patches.
- One Gauss-Seidel step is a full matrix vector multiply!!
 - $O(n^2)$
- To update radiosity B_i of all patches all form factors are needed F_{ij}
 - Hence the method is also called the full matrix radiosity



$$B_i^{k+1} = E_i + \rho_i \sum_{j=1}^{i-1} B_j^{k+1} F_{ij} + \rho_i \sum_{j=i+1}^n B_j^k F_{ij}$$

Radiosity Solution Techniques

- **Jacobi Iteration 0**



Radiosity Solution Techniques

- Jacobi Iteration 1



Radiosity Solution Techniques

- Jacobi Iteration 2



Radiosity Solution Techniques

- Jacobi Iteration 3

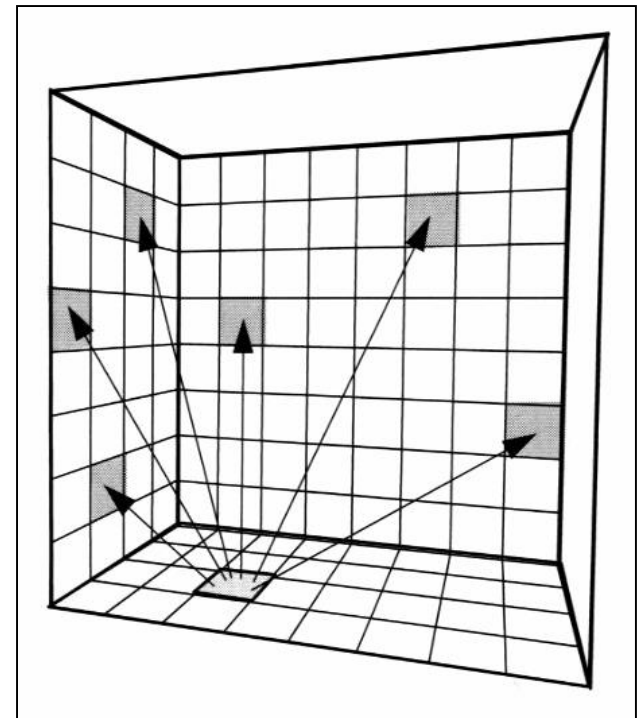


Radiosity Solution Techniques

- **Southwell Iteration (aka Shooting)**

- Patches keep radiosity B_i and unshot radiosity ΔB_i
- Each iteration shoots radiosity from patch with max. unshot power $\Delta B_i A_i$ to all others
- Update radiosity B_j and unshot radiosity ΔB_j of each of the receiving patches

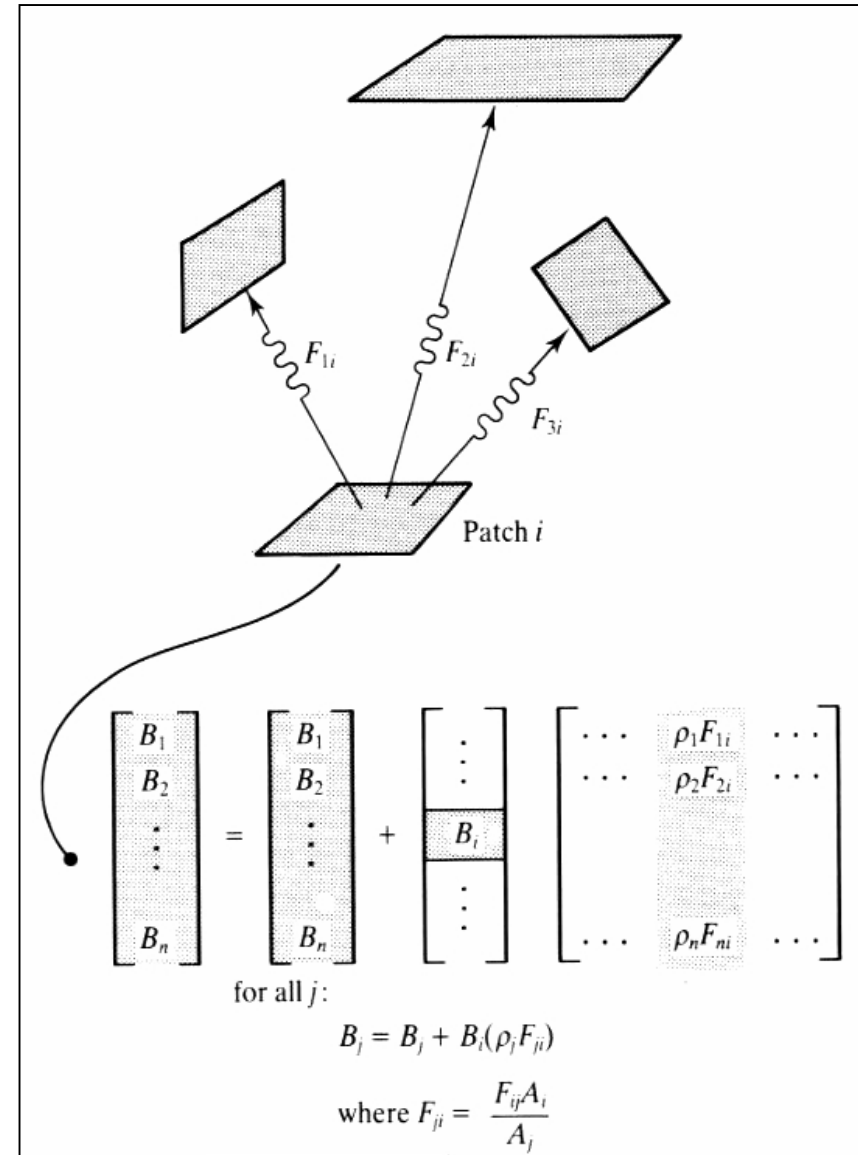
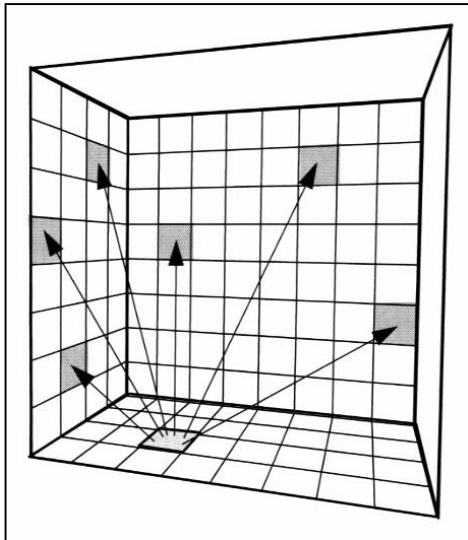
```
1  for ( all  $i$  ) {
2     $B_i = E_i$  ;
3     $\Delta B_i = E_i$  ;
4  }
5  while ( not converged ) {
6    pick  $i$ , such that  $\Delta B_i * A_i$  is largest ;
7    for ( every element  $j$  ) {
8       $\Delta rad = \Delta B_i * \rho_j F_{ji}$  ;
9       $\Delta B_j = \Delta B_j + \Delta rad$  ;
10      $B_j = B_j + \Delta rad$  ;
11   }
12    $\Delta B_i = 0$  ;
13   display the image using  $B_i$  as the intensity of element  $i$  ;
14 }
```



Radiosity Solution Techniques

Shooting: the radiosity of all patches is updated for each iteration:

- Evaluate a “column” of form factors, that is, the form factors from this patch to every other patch in the environment.
- One step: Dot product $\rightarrow O(n)$
- The solution is progressively refined: **progressive radiosity**



Ambient Term in Progressive Radiosity

- **Solution proceeds from dark to light as unshot power goes to zero**
- **Unshot radiosity can be approximated by an ambient term and added to the radiosity solution only for image display**

$$B_i^{display} = B_i + \rho_i B_{ambient}$$

- **Unshot power U approximated by its average**
 - Updated for each image display

$$U = \frac{\sum_{i=0}^n \Delta B_i A_i}{\sum_{i=0}^n A_i}$$

- **Can be used for overrelaxation → not discussed here**

Ambient Term in Progressive Radiosity

- **Average reflectance** $\rho_{avg} = \frac{\sum_{i=0}^n \rho_i A_i}{\sum_{i=0}^n A_i}$

- **Total reflection R (taking into account interreflections)**

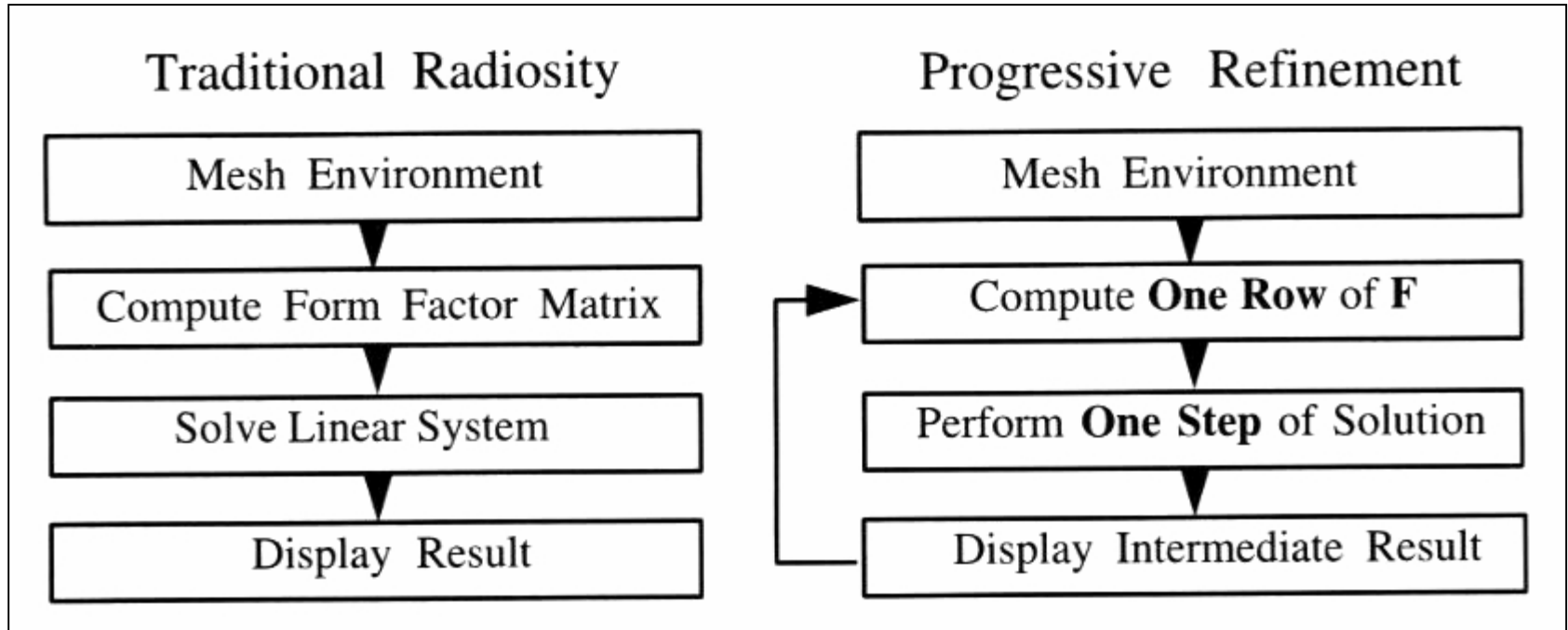
$$R = 1 + \rho_{avg} + \rho_{avg}^2 + \rho_{avg}^3 + \rho_{avg}^4 + \dots = \frac{1}{1 - \rho_{avg}}$$

- **Ambient radiosity approximation:**

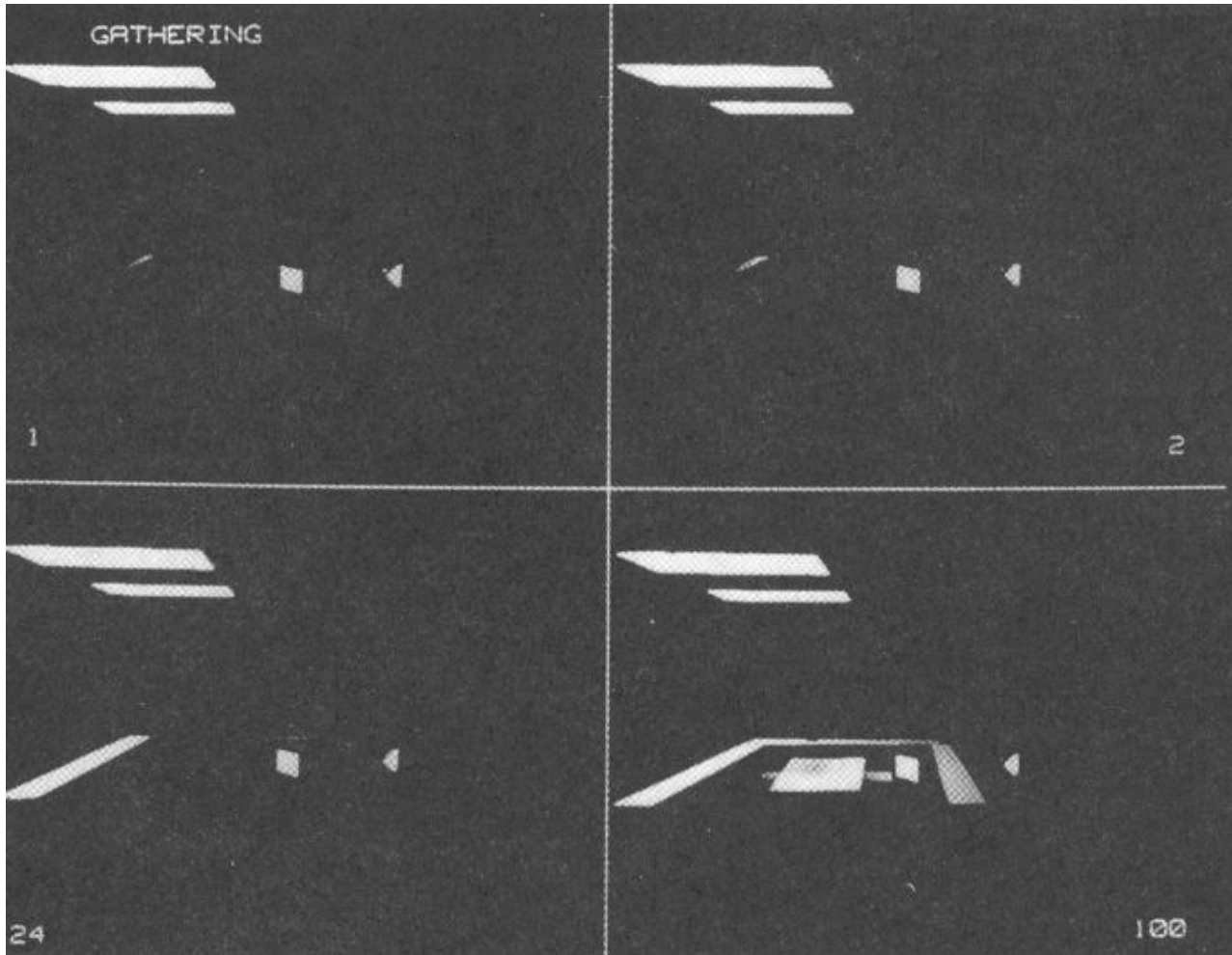
$$B_{ambient} = RU$$

- $B_{ambient}$ **goes to zero as the radiosity solution converges**

Traditional Radiosity vs. Progressive Refinement

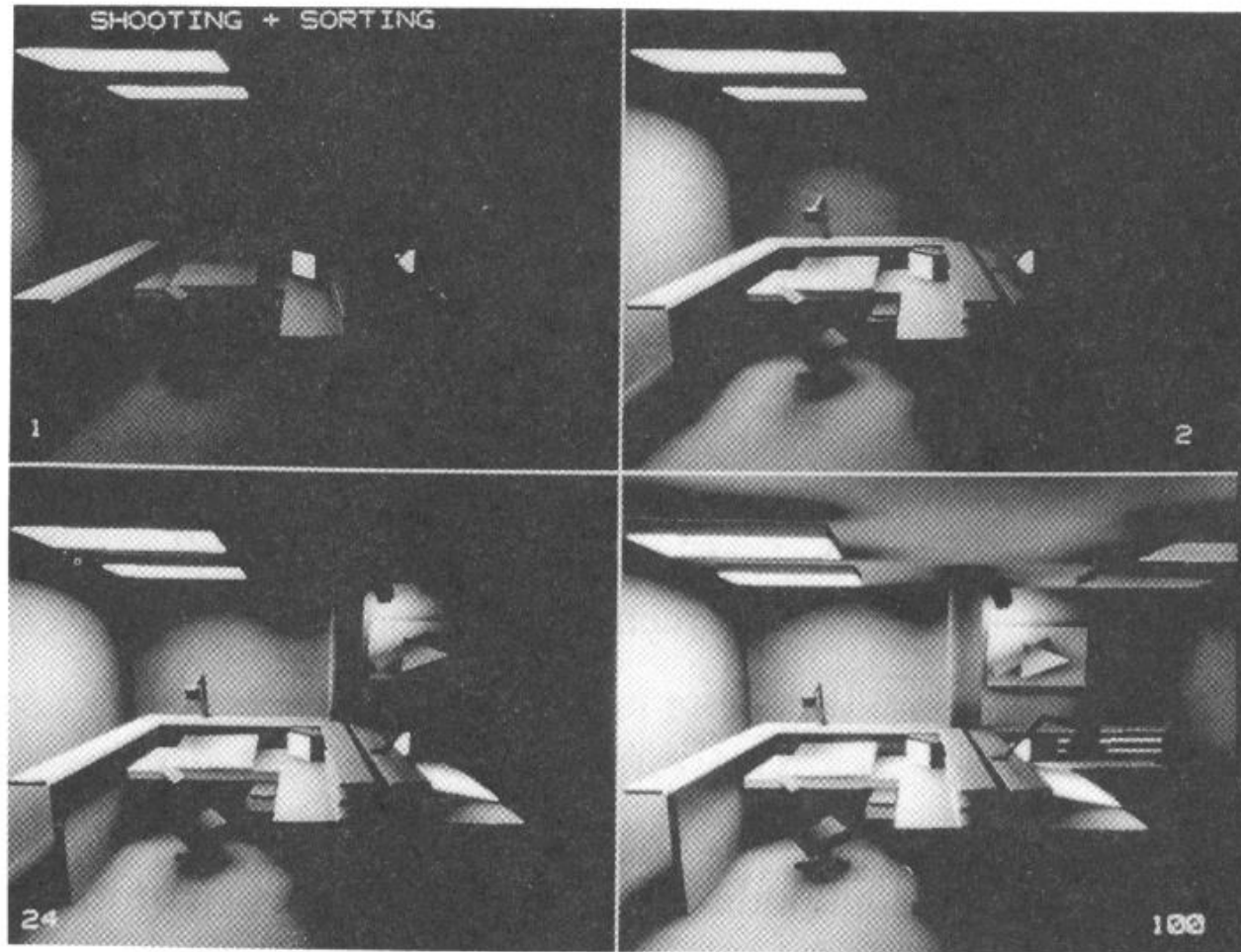


Results: Gauss-Seidel Iteration



After 1, 2, 24, and 100 Gathering-steps

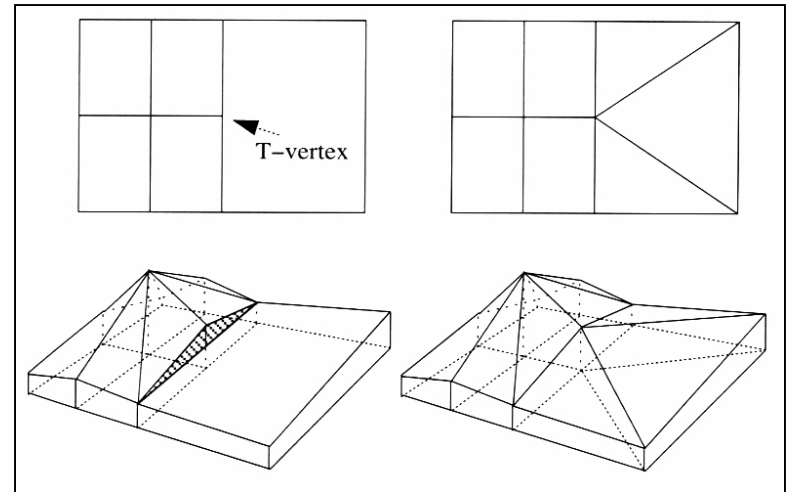
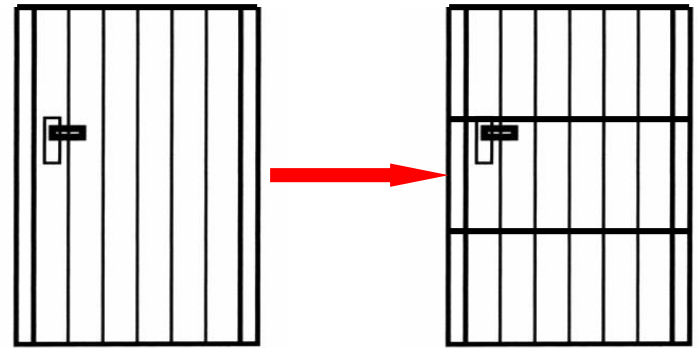
Results: Progressive Radiosity



After 1, 2, 24, and 100 Shooting-steps

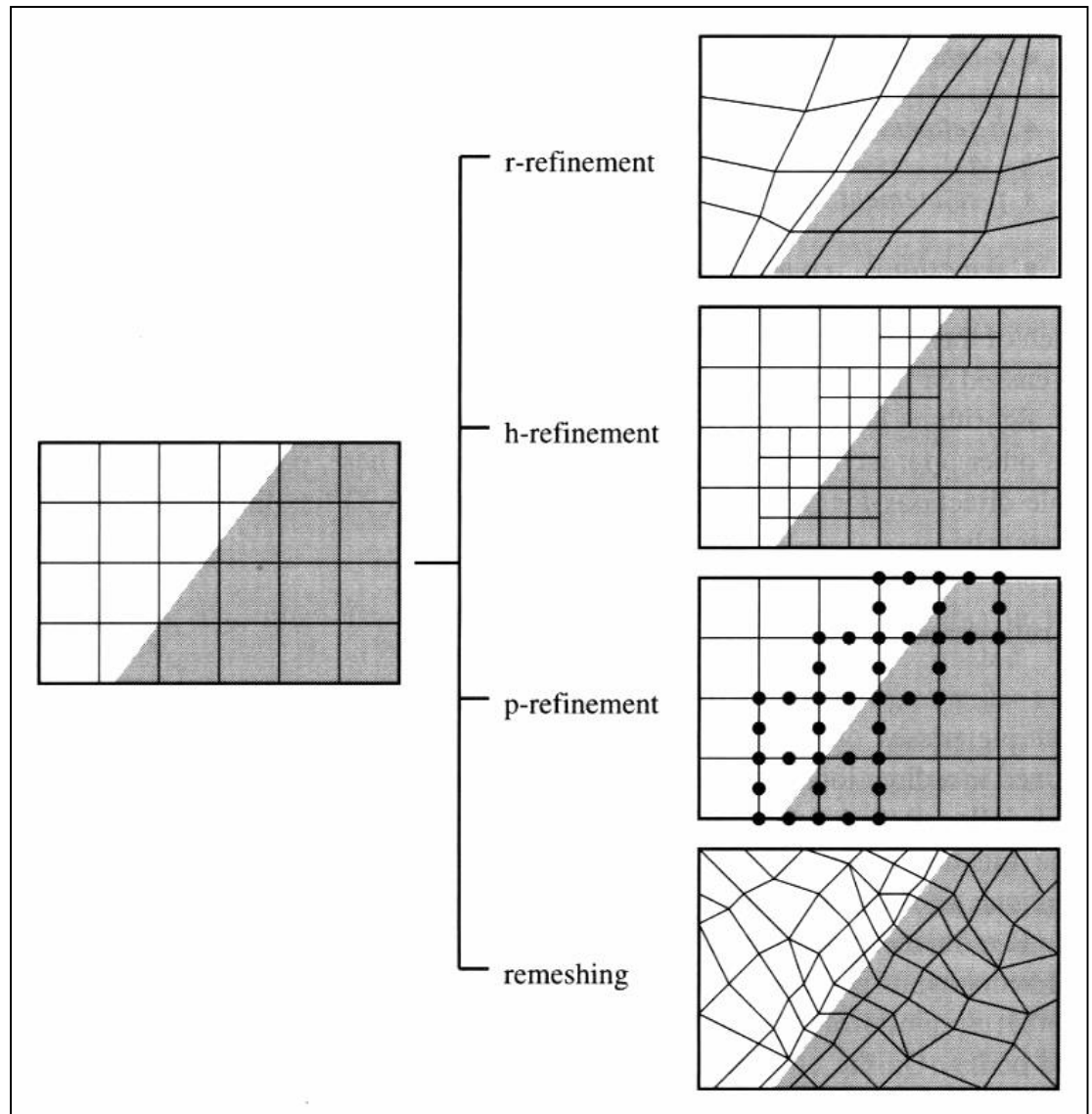
Meshing: Domain Subdivision

- **Mesh density**
- **Mesh element characteristics**
 - Triangles, rectangles, or both
 - Aspect ratio
 - A better aspect ratio of mesh elements improves the accuracy of form factors and quality of shading
 - Mesh grading
 - Smooth transition between regions in which lighting function changes quickly
 - Element order and continuity
 - Polynomial order
 - Continuity order along element borders
 - Mesh conformance
 - At least C^0 continuity along element borders



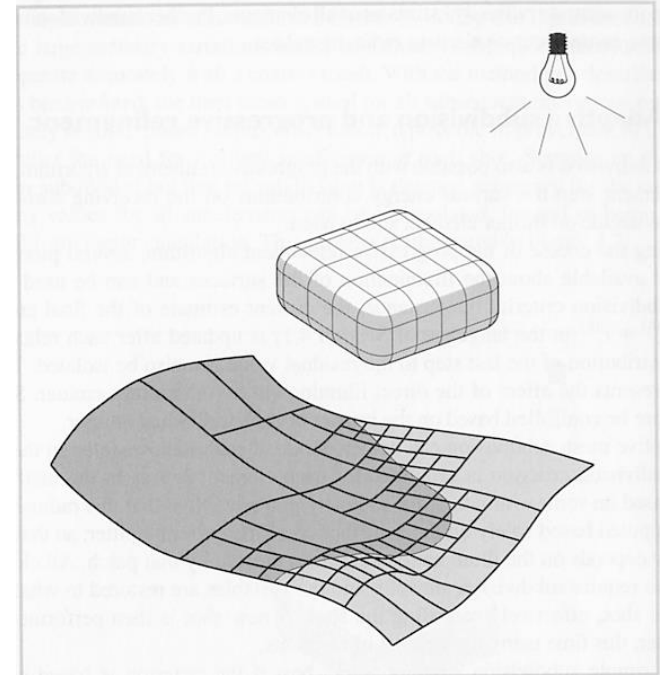
Automatic Meshing Techniques

- **A priori meshing**
 - the right choice of the mesh element sizes is critical
- **A posteriori meshing**
 - **r-refinement:** reposition nodes,
 - **h-refinement:** subdivide existing elements,
 - **p-refinement:** increase polynomial order of existing elements,
 - **remeshing:** replace existing mesh with new mesh.
 - Can be interleaved with computation

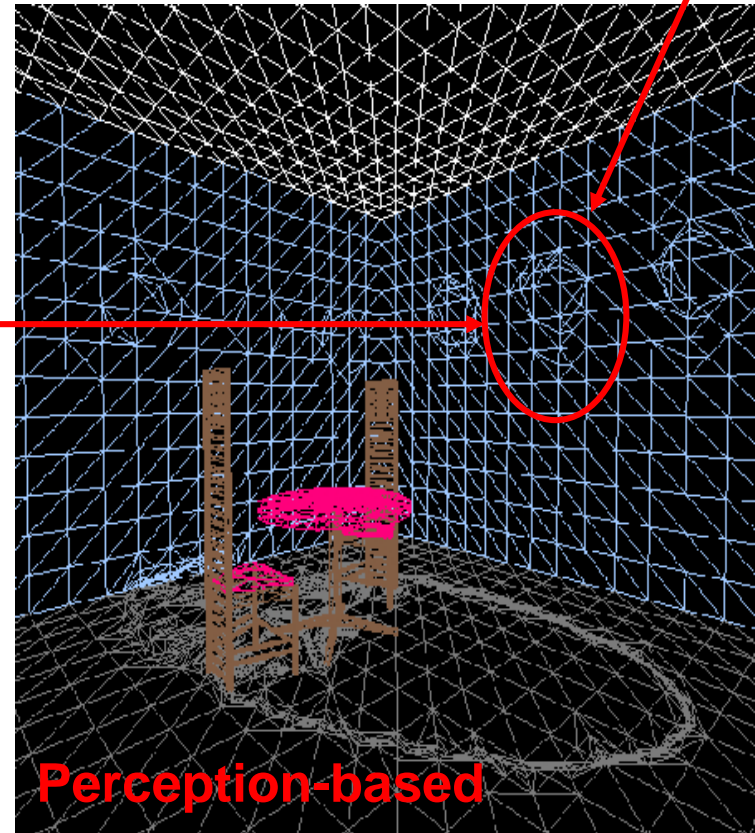
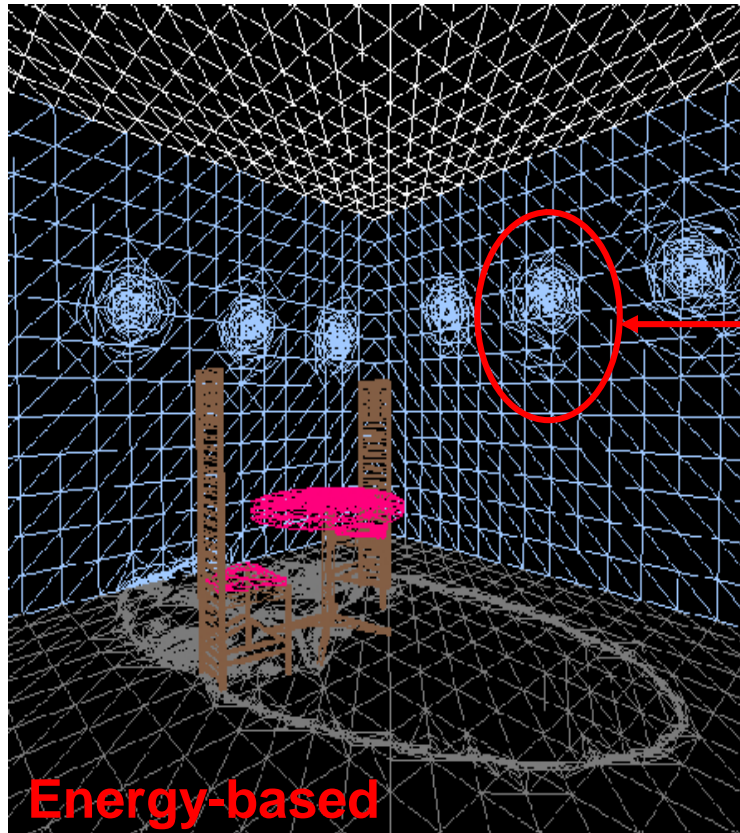


Adaptive Mesh Subdivision

- **Mesh should adapt to local complexity of the lighting distribution**
- **Gathering iteration:**
 - Estimate the radiosity gradient
 - Subdivide elements with gradients over some threshold value and gather for each
 - Recursively keep subdividing until excessive gradients are removed or mesh granularity limit is reached
 - Do the next iteration of Gauss-Seidel
- **Gradient measurement**
 - Compare radiosity between neighboring elements or their vertices (usually computed as area weighted average)
 - Sample radiosity along mesh edges (usually for primary lights only)
 - Consider perception to select visually meaningful gradient threshold values (Weber law, contrast sensitivity, visual masking)
- **Issue: Gradient may be smoothed out by other contributions later**

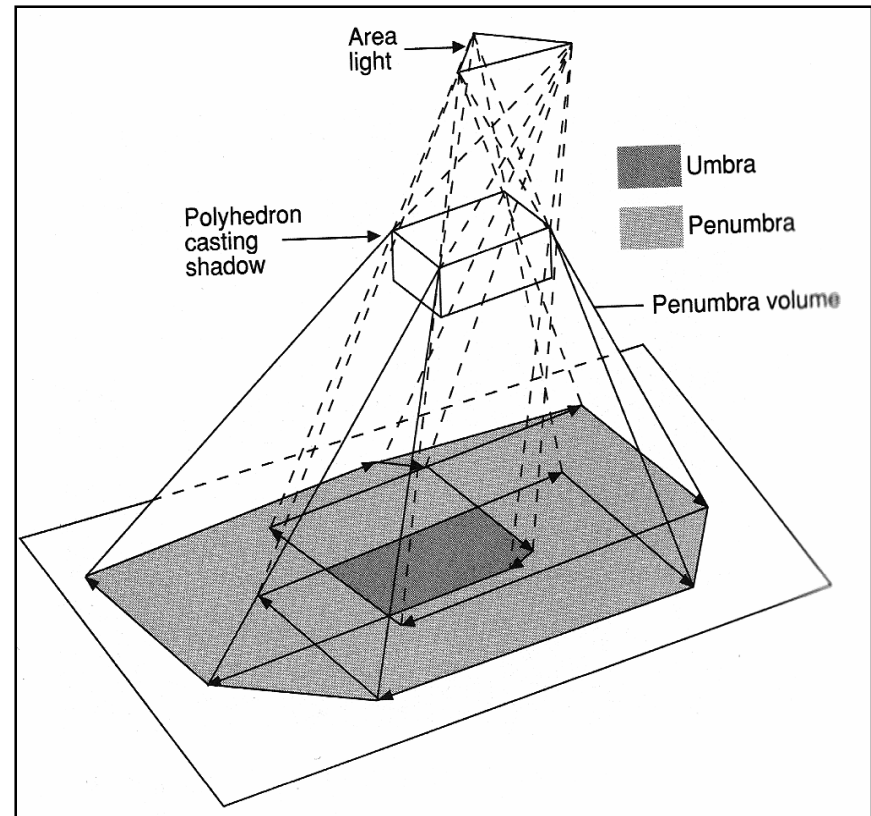
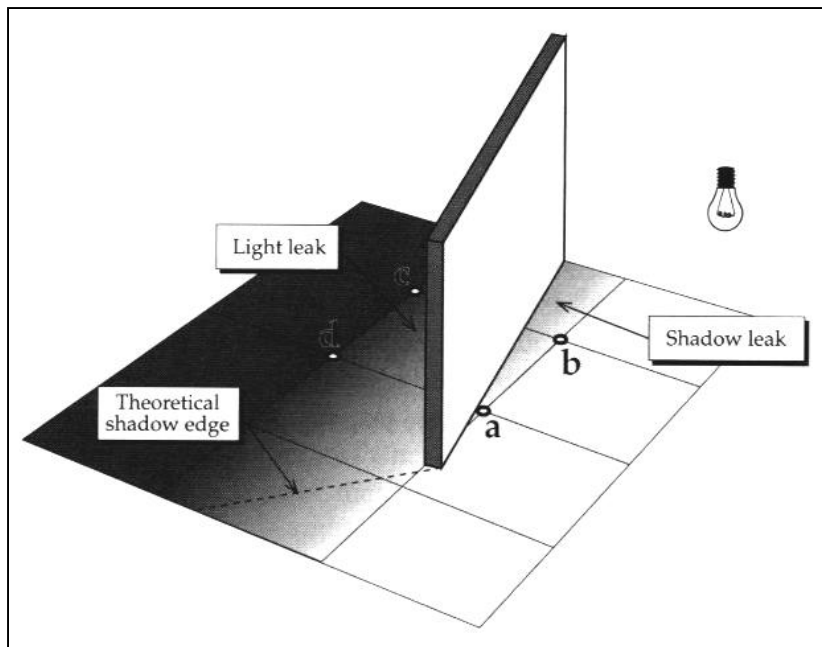


Adaptive Mesh Subdivision



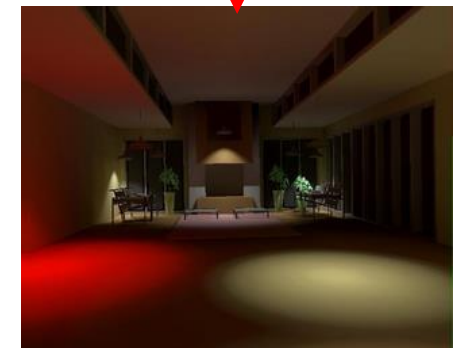
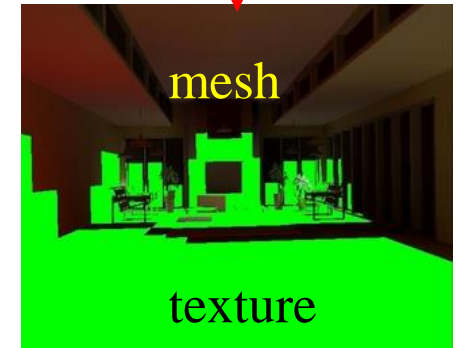
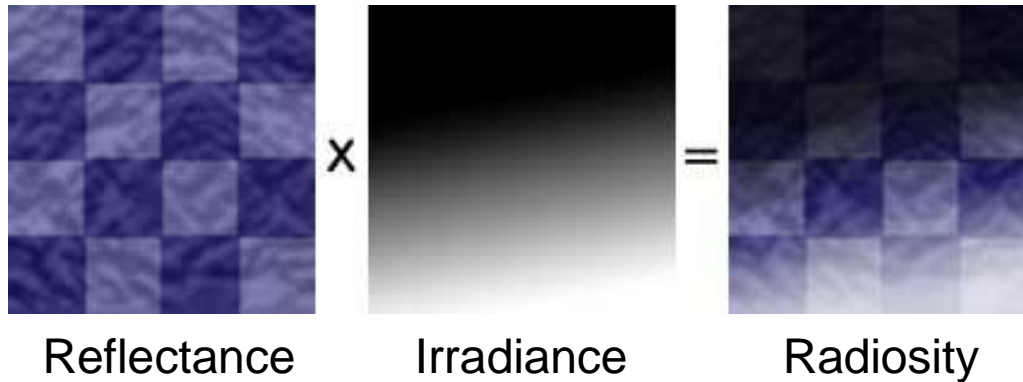
Problems with FE Lighting

- Light and shadow leaks
- Adaptive mesh subdivision can reduce those effects
- Discontinuity meshing
 - Mesh edges generated along predicted discontinuities of the radiosity function



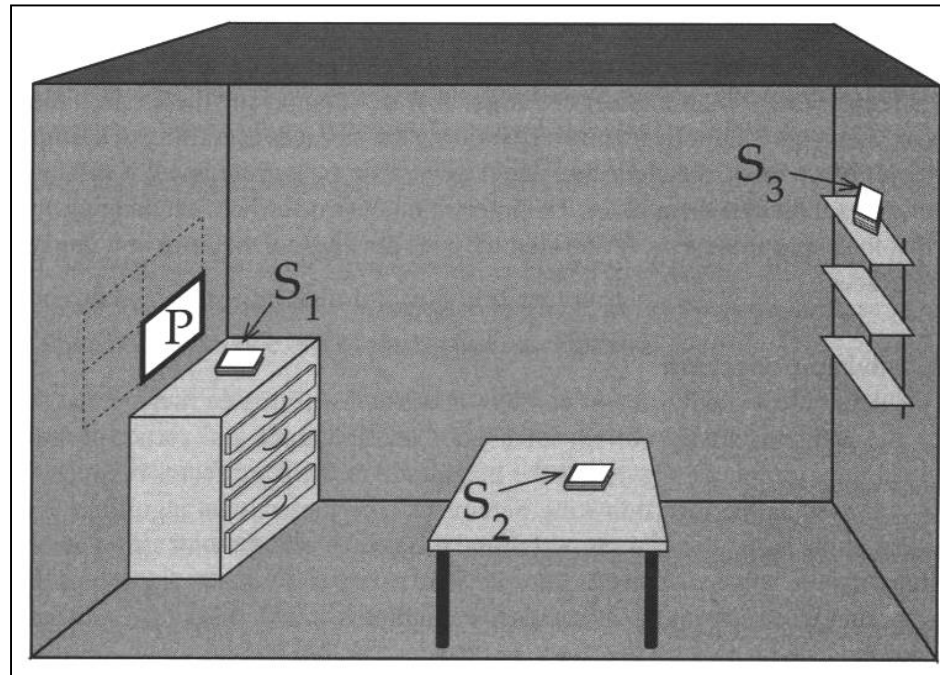
Mesh vs. Textures

- Replacing mesh by textures in the scene regions populated by many small mesh elements – postprocessing
- **Light maps (i.e. in Quake)**
 - Storing pre-calculated illumination
 - Often very low resolution
 - Multiplication of illumination with the base texture



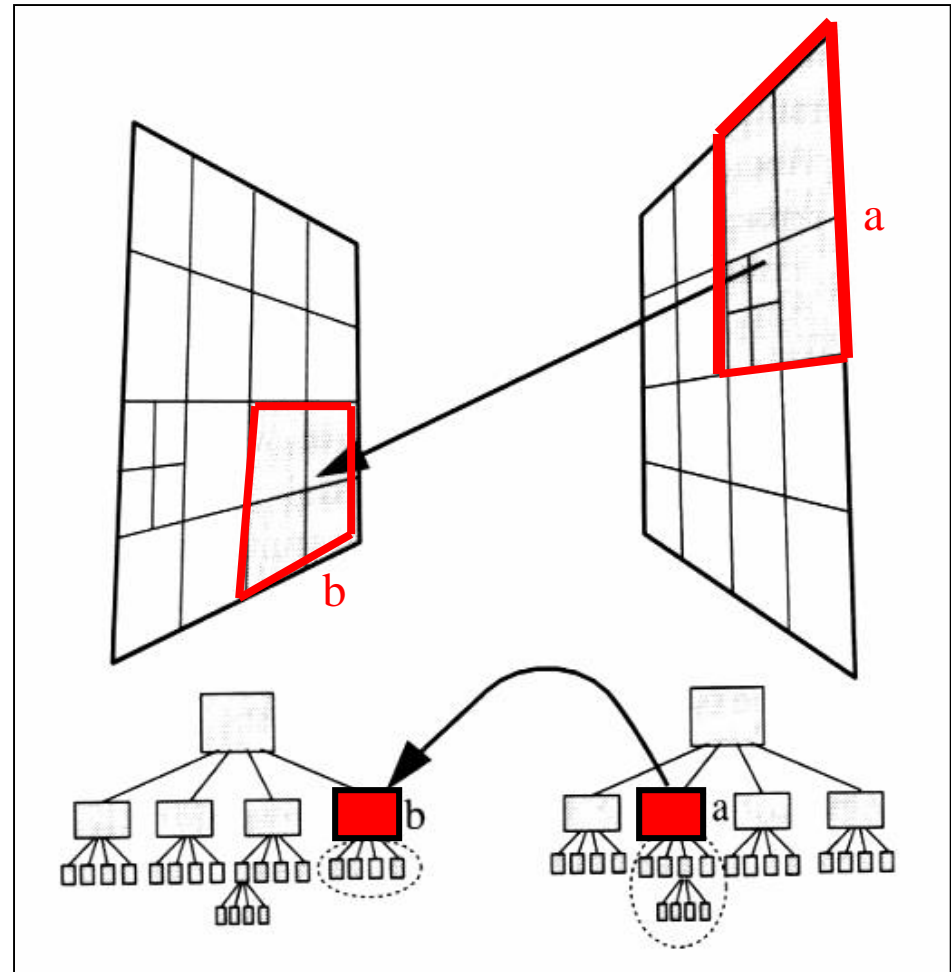
Hierarchical Energy Exchange

- P and S_1 are very close to each other,
 - the surfaces should be subdivided into smaller patches when significant energy exchange between them is considered.
- P and S_2 are more distant
 - the surfaces subdivision is not necessary.
- P and S_3 are very far to each other
 - P can be clustered (grouped into a single entity) with neighboring surfaces to speedup calculations.



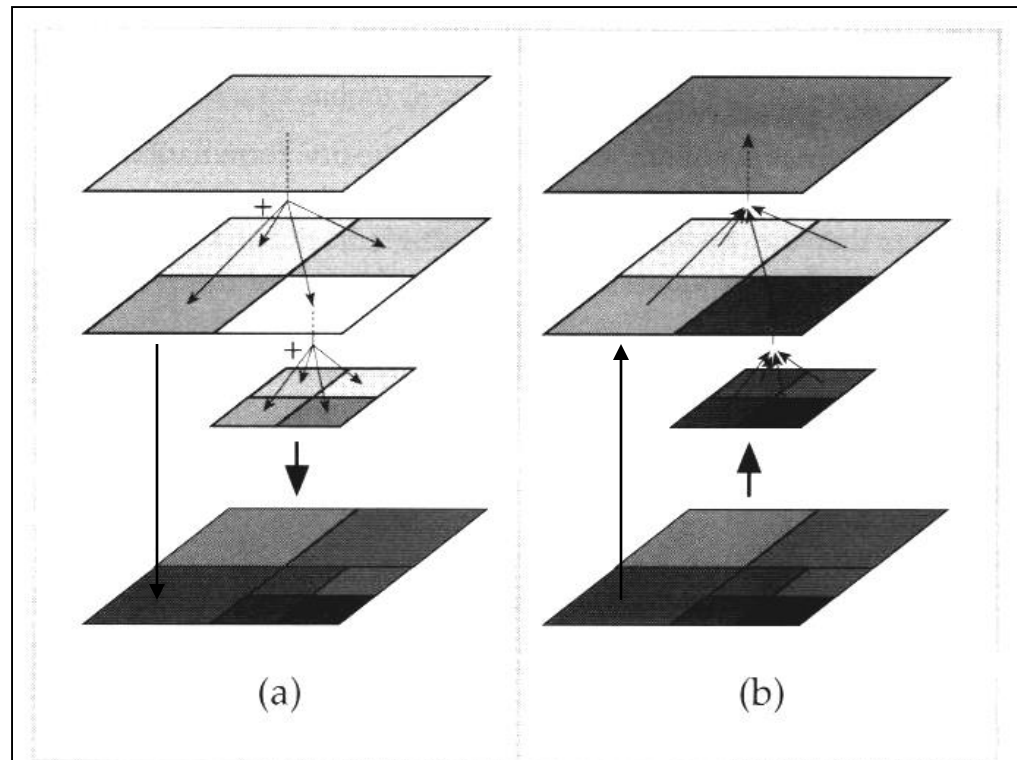
Hierarchical Radiosity: Constructing Hierarchy

- Estimate the form factor between the two surfaces using a simple formula.
- If the estimated form factor is below the threshold, or if the surfaces are too small to be subdivided, record their interaction at the current level.
- Otherwise, subdivide one of the surfaces and recurse.
- Create links between patches at the recorded levels of hierarchy.



Hierarchical Radiosity: Solution

- **Propagate energy along all links**
 - Different contributions at all levels
- **Create consistent multi-level representation:**
 - **Push** radiosity down to the leaves.
 - **Pull** radiosity values up the hierarchy.
- **Refine links**
- **Repeat propagation, push-pull, and link refining steps until the solution convergence**



Combination of radiosity exchanges at all levels of the hierarchy. Note that darker patches here indicate greater radiosity values. (a) Radiosity values are pushed to the leaves and added along the way. (b) Radiosity values are pulled up the tree and averaged at each level. As a result each level has a correct representation of the radiosity received at levels both higher and lower in the tree. For example, the lower-right corner of the second level has seen its radiosity increase to reflect the energy received by its parent and by its children.

Progressive Radiosity vs. Hierarchical Radiosity

- **Progressive Radiosity (PR)**

- + Produces quickly approximated, but meaningful images
- + Requires little memory for calculations
- Slow convergence in complex environments
 - Not practical for complex scenes

- **Hierarchical Radiosity (HR)**

- + Less sensitive to the scene complexity comparing to PR
- Relatively much time is required to produce the first image
 - Initial linking phase between m input polygons should be completed
 - Clustering can shorten this waiting time
- $O(n)$ links are created per element (\sim fixed subdivision of FF \rightarrow of solid angle)
- Links might require huge memory
 - But “getting rid of links”: Do not store links, recreate and cache them
 - Create links based on error (\sim power transported) \rightarrow # decreases exponentially
- Quality of the final solution:
 - The quality of shading might be poor because of differences between neighboring mesh elements
 - Final gathering at rendering stage helps – but adds huge cost

Three Radiosity Algorithms

- **Full matrix radiosity (Gathering)**
- **Progressive radiosity (Shooting)**
- **Hierarchical radiosity (either)**

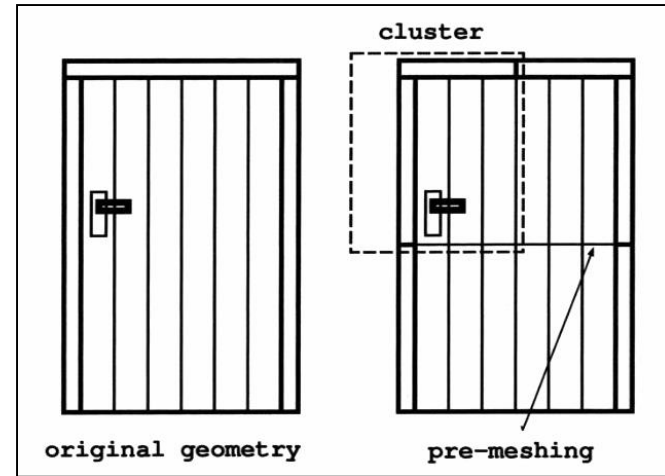
Computational complexity and required memory:

Algorithm	Computational complexity	Memory
Matrix (Gauss-Seidel)	$O(n^2)$	n^2
Progressive Radiosity	$O(n^2)$	n
Hierarchical Radiosity	$O(m^2 + n)$	$\sim(m^2 + n)$

where, m and n are numbers of input polygons and mesh elements (after meshing), respectively.

Clustering

- **What to do if initial mesh is too fine (m)**
- **Hierarchical grouping of mesh into *cluster***
 - Usually performed within the hierarchical radiosity framework
 - Reduces m in the hierarchical radiosity formulation and thus improves the computational complexity $O(m^2+n) \rightarrow O(n)$
- **Initial scene pre-meshing and then clustering offers some advantages:**
 - Spatial coherence (locality) of resulting clusters increases the number of possible high level energy transfers between clusters within the imposed error bound.
 - Clustering relies less on the input geometry, and depends rather on a local position of surfaces in a scene.
 - A better aspect ratio of mesh elements improves the accuracy of form factors and quality of shading.
 - Many issues: cluster criteria, local power redistribution, etc.



General Basis Functions

- **Idea**

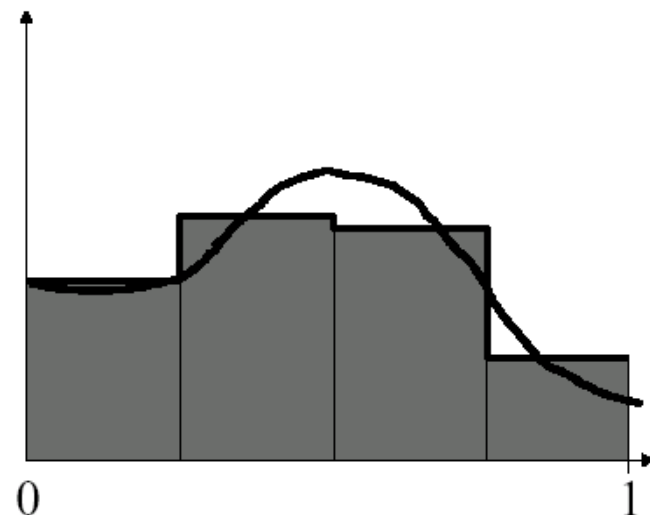
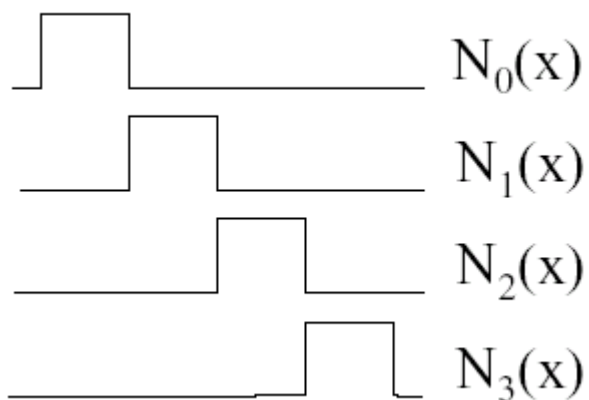
- Generalize representation from classical radiosity (constant basis)
- Choose finite function basis
 - Linear combinations of basis functions
- Must generalize concept of form factors

- **Solution**

- General: The dimension of radiosity function space is infinite
 - Radiosity values must be computed at infinitely many points to fully describe the true radiosity function
- Approximation: Choose radiosity representation as a linear combination of basis functions $N_i(x)$
 - $B(x) \approx \sum b_i N_i(x)$
 - $N_i(x)$ span a finite dimensional functions space
 - Approximate the infinite-dimensional functions space L_2
 - Must find the “closest” representation in this space
- All math is performed in this finite function space

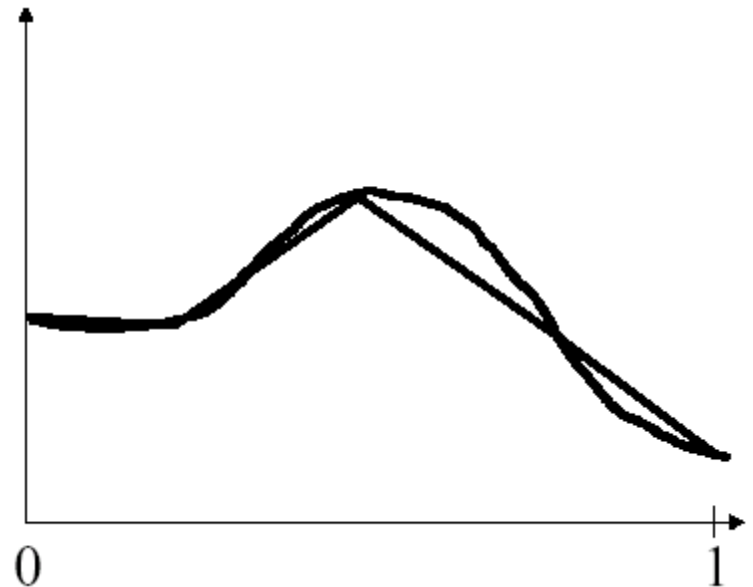
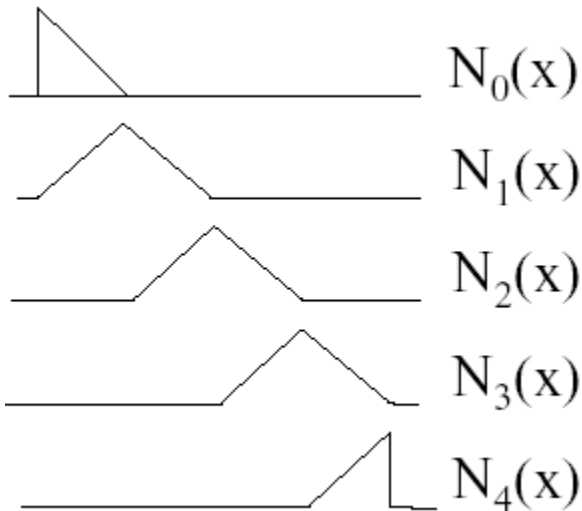
General Basis Functions

- **Box Basis**
 - Piecewise constant basis function



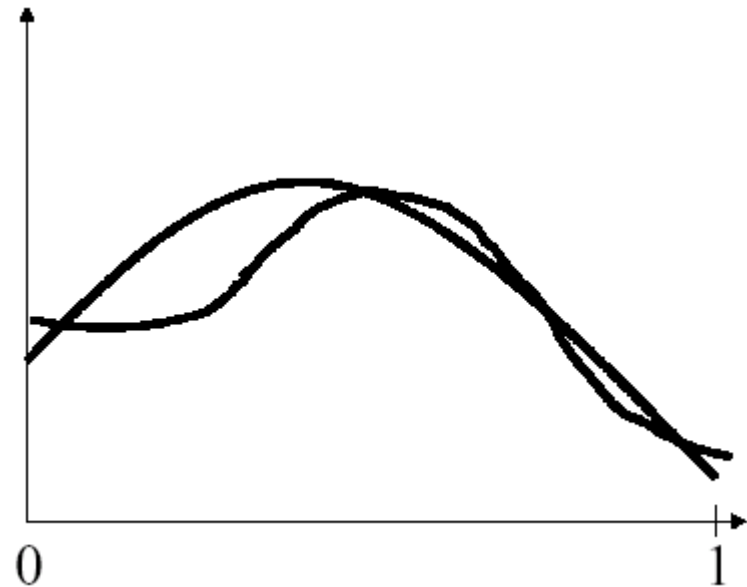
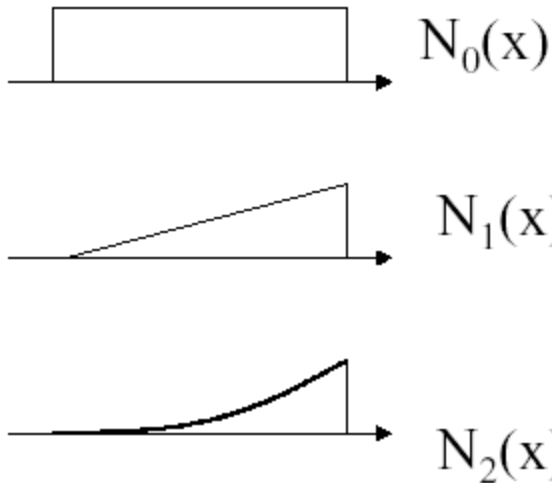
General Basis Functions

- **Linear Basis Functions**
 - Hat Functions



General Basis Functions

- **Polynomial Basis Functions**
 - Different representations
 - Monom, Lagrange, splines, ...

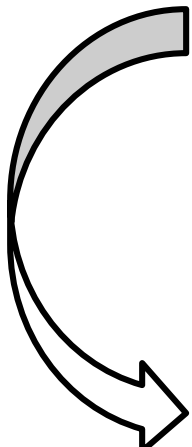


General Basis Functions

- **Projection of functions**

- Dot product for functions $\langle f(x), g(x) \rangle = \int f(x)g(x)dx$

- Projection of functions into a subspace


$$f(x) = \sum_i f_i N_i(x)$$

Orthogonal Basis Function

$$\int N_i(x)N_j(x)dx = \begin{cases} \text{non zero} & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

$$\langle f(x), N_i(x) \rangle = f_i \langle N_i(x), N_i(x) \rangle \rightarrow f_i = \frac{\int f(x)N_i(x)dx}{\int N_i^2(x)dx}$$

Orthonormal Basis Function

$$\int N_i(x)N_j(x)dx = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

$$\langle f(x), N_i(x) \rangle = f_i \rightarrow f_i = \int f(x)N_i(x)dx$$

Finite Element Radiosity Formulation

$$B(x) = E(x) + \rho_d(x) \int_{y \in \mathcal{S}} B(y) \frac{\cos \theta \cos \theta'}{\pi r^2} V(x, y) dy$$

$$\text{using } B(y) = \sum_j b_j N_j(y)$$

$$B(x) = E(x) + \rho_d(x) \sum_j b_j \int_{y \in \mathcal{S}} N_j(y) \frac{\cos \theta \cos \theta'}{\pi r^2} V(x, y) dy$$

$$\text{using } b_i = \frac{\int B(x) N_i(x) dx}{\int N_i^2(x) dx}$$

$$b_i = e_i + \rho_i \sum_j b_j \frac{1}{\int_{x \in \mathcal{S}} N_i^2(x) dx} \int_{x \in \mathcal{S}} N_i(x) \int_{y \in \mathcal{S}} N_j(y) \frac{\cos \theta \cos \theta'}{\pi r^2} V(x, y) dy dx$$

Finite Element Radiosity Formulation

- **Generalized Form Factors**

$$b_i = e_i + \rho_i \sum_j b_j F_{i,j}$$

$$F_{i,j} = \frac{1}{\int_{x \in S} N_i^2(x) dx} \int_{x \in S} N_i(x) \int_{y \in S} N_j(y) \frac{\cos \theta \cos \theta'}{\pi r^2} V(x, y) dy dx$$

- These formulas hold for any orthogonal basis function
 - Must evaluate a double integral
 - Higher order basis functions have $(n+1)$ coefficients (linear)
 - $(n+1)^2$ coefficients for a patch (area)
 - Needs $(n+1)^4$ coefficients for representing form factor
 - One for each pair of 2D basis coefficients
 - Can get very expensive

Finite Element Radiosity Formulation

- **Constant radiosity approximation is just a special case**

$$b_i = e_i + \rho_i \sum_j b_j F_{i,j}$$

$$F_{i,j} = \frac{1}{\int_{x \in S} N_i^2(x) dx} \int_{x \in S} N_i(x) \int_{y \in S} N_j(y) \frac{\cos \theta \cos \theta'}{\pi r^2} V(x, y) dy dx$$

using piecewise - constant basis function per patch A_i

$$N_i(x) = \begin{cases} 1 & x \in A_i \\ 0 & x \notin A_i \end{cases} \rightarrow \int_{x \in S} N_i^2(x) dx = \int_{x \in A_i} 1 dx = A_i$$

$$F_{i,j} = \frac{1}{A_i} \int_{x \in A_i} \int_{y \in A_j} \frac{\cos \theta \cos \theta'}{\pi r^2} V(x, y) dy dx$$

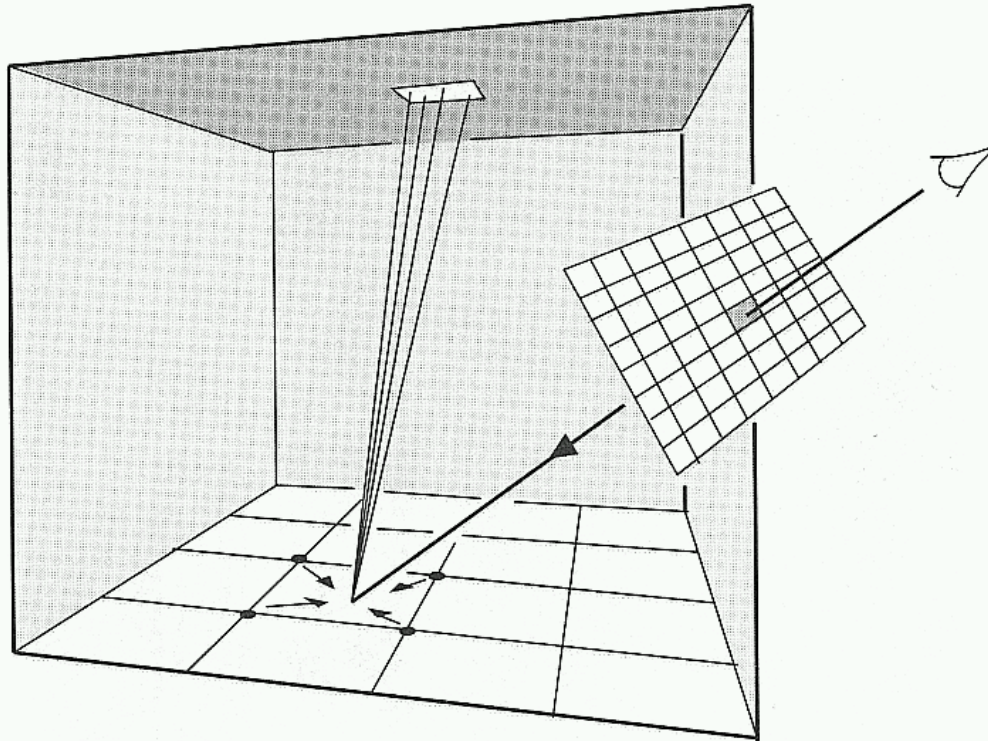
Finite Element Techniques

- **General Problems**

- Representation directly tied to surface representation
 - E.g.: One surface subdivided into small pieces
 - Limited/no change in illumination across patch
 - Representation of illumination may need to be subdivided
- Bias
 - Uses smooth basis function
 - Neighboring points forced to have similar illumination
 - Cannot represent sudden change in illumination
 - Except at patch boundaries
 - Discontinuity Meshing tries to place edges adaptively
 - Many geometric problems, complex implementation
- Little support for generalized BRDFs
 - Required basis functions in space and direction
 - High code complexity, high memory requirements

Reconstruction

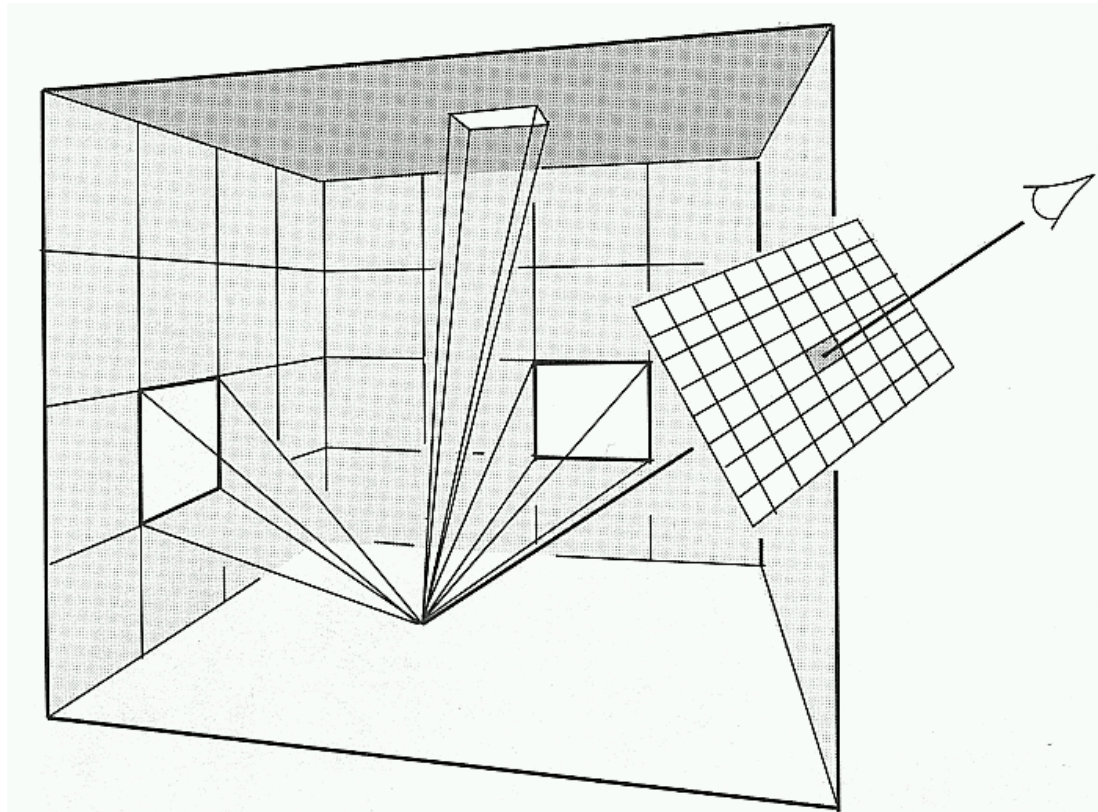
- **Interpolation over adjacent values**
 - Calculate average radiosity value from adjacent surfaces at the vertex
 - Interpolate values between vertices
 - Graphics hardware



Reconstruction

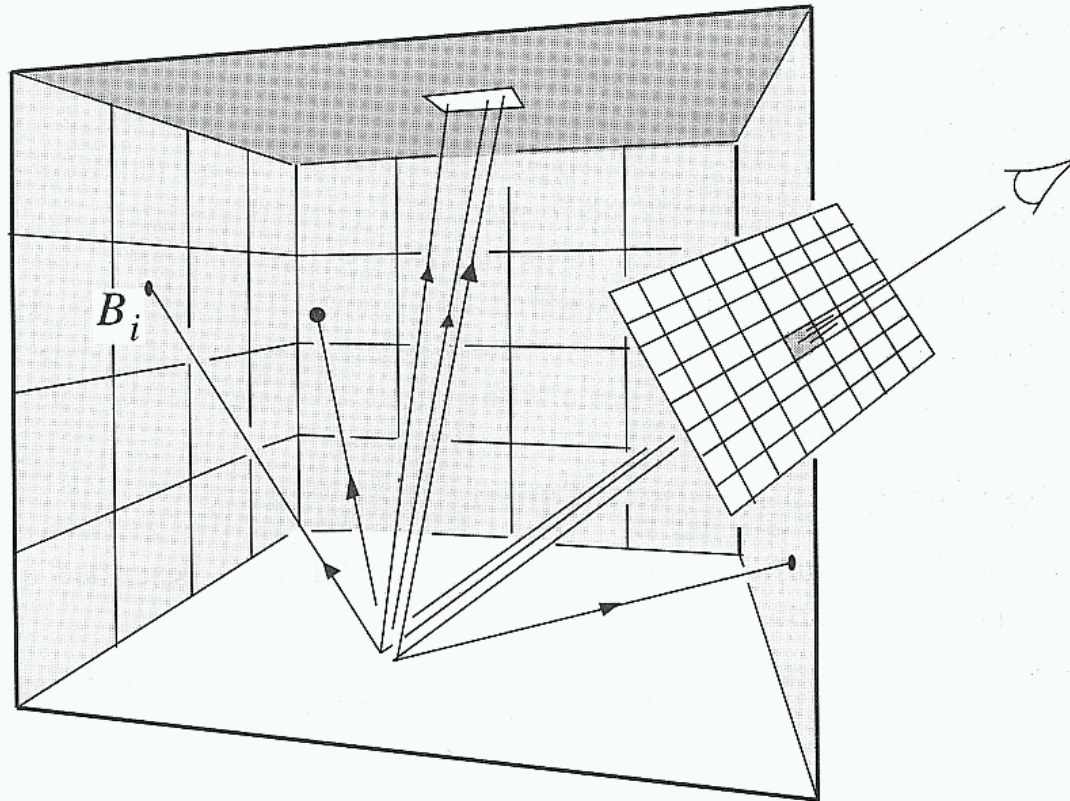
- **Final-Gather**

- Calculate irradiance through collecting illumination of the surfaces
- Requires form factor between finite surface and infinitesimal point

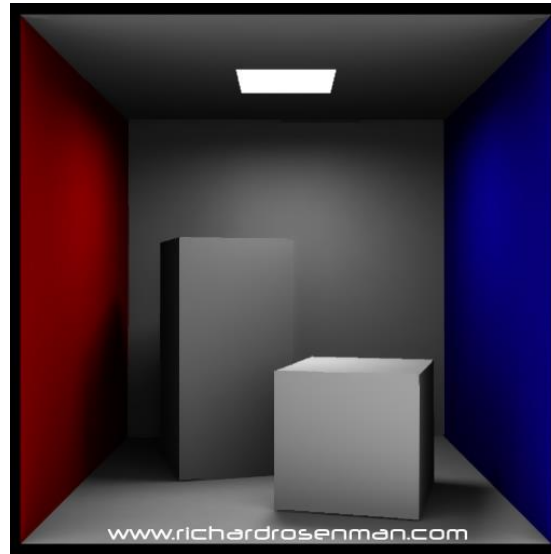
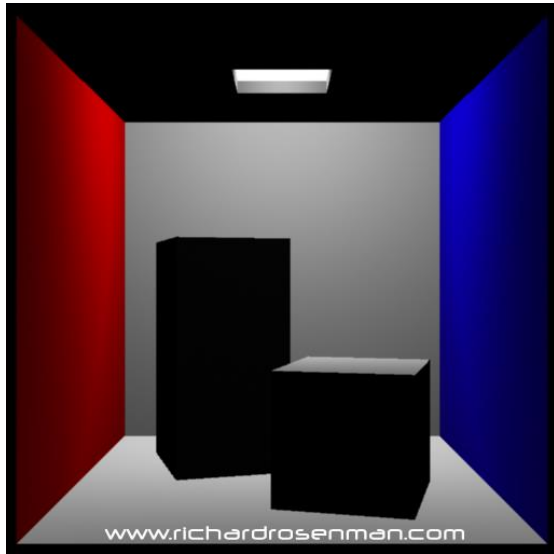


Reconstruction

- **Stochastic integration**
 - Collecting illumination from randomly distributed rays



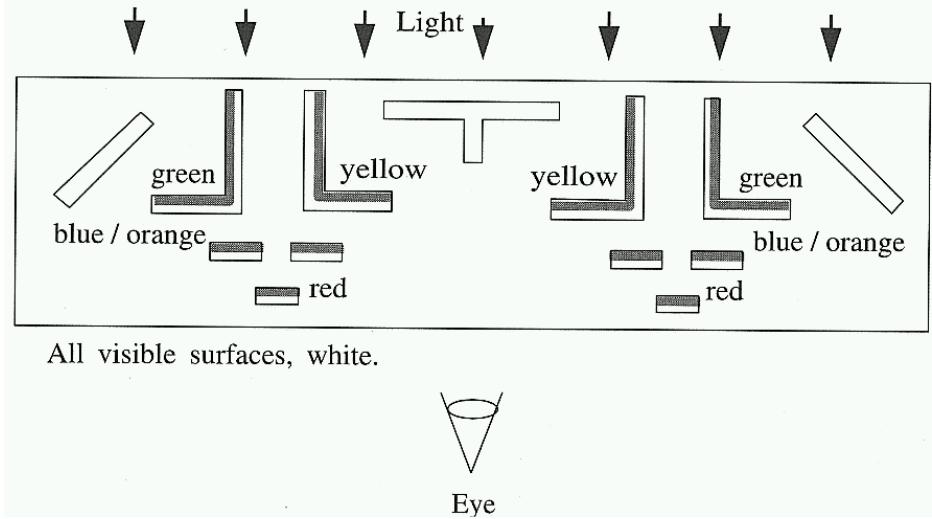
Radiosity Results



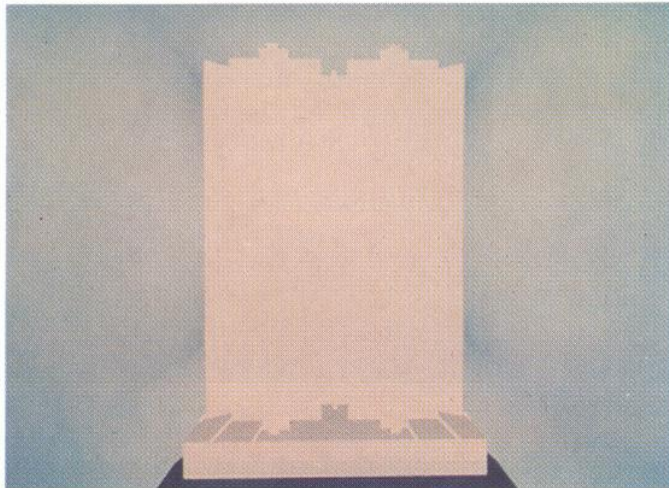
Courtesy Richard Rosenman

- **Only direct illumination (ray tracing)**
 - No shadows (costly for area light sources), unlit surfaces
- **One iteration**
 - Indirect illumination (common approximation in many games and movies)
 - Blurry shadows
- **Multiple iterations (close to equilibrium)**
 - Color bleeding
 - Overall reduced contrast, softer appearance

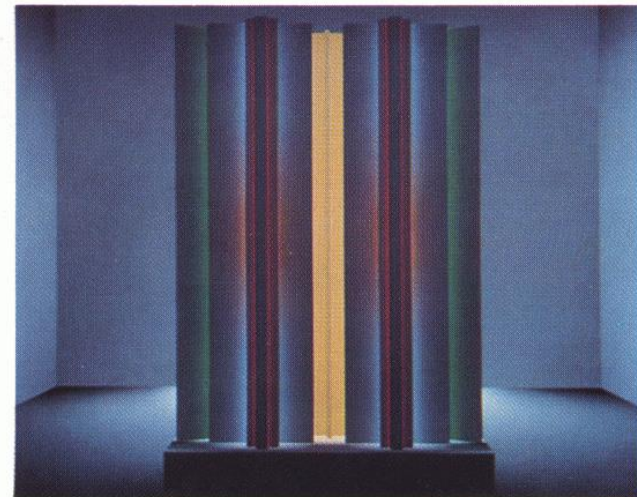
Global Effects



Ray
tracing



Radiosity



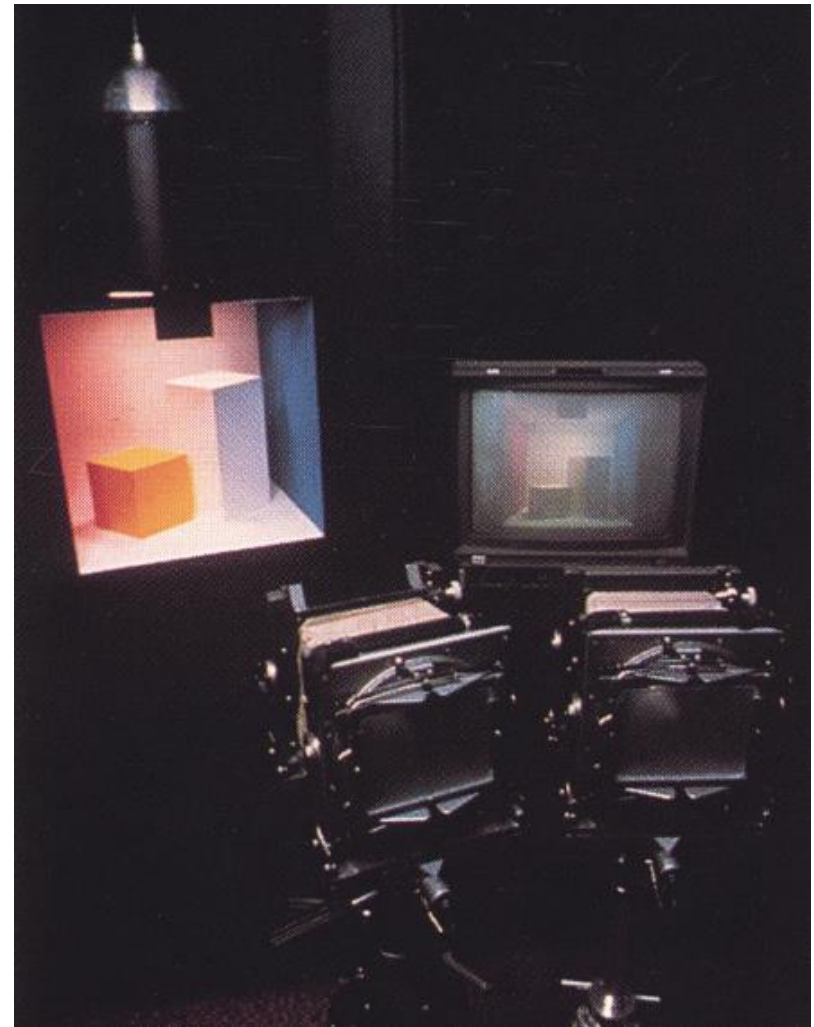
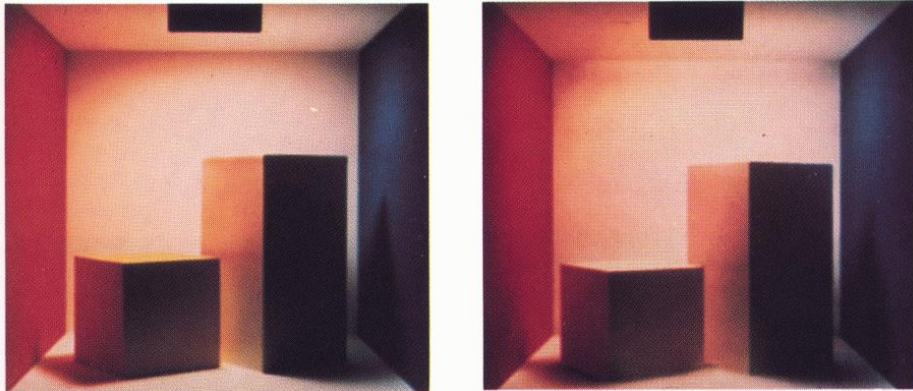
John Ferren, Construction in Wood, A Daylight Experiment, © Cornell

Radiosity Results



© Cornell

Comparison with the Reality



© Cornell