Hierarchické radiační metody

KSVI MFF UK Praha

e-mail: Josef.Pelikan@mff.cuni.cz

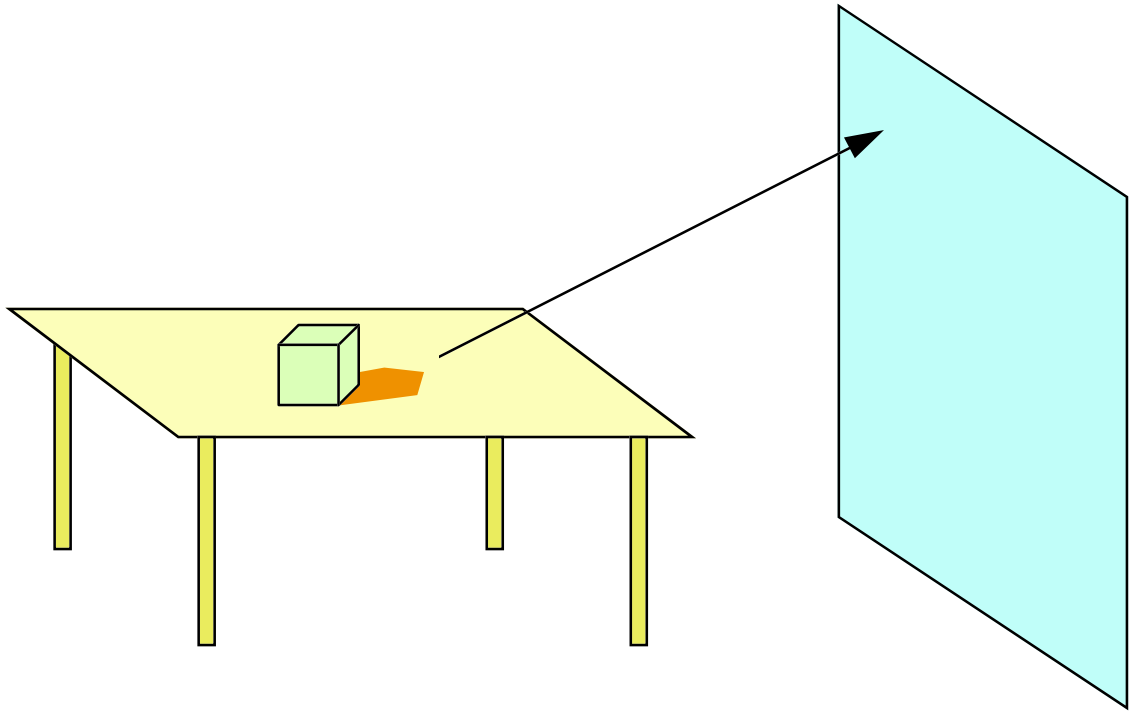
WWW: http://cgg.ms.mff.cuni.cz/~pepca/

Urychlení radiační metody

- adaptivní **zjemňování sítě**
  - snaha o zmenšení počtu elementů při zachování kvality aproximace
- **hierarchické** radiační metody
  - zachovávají rozdělení povrchu scény na elementy
  - snaha zmenšit počet použitých konfiguračních faktorů
  - menší matice řešené soustavy

Vzdálená interakce

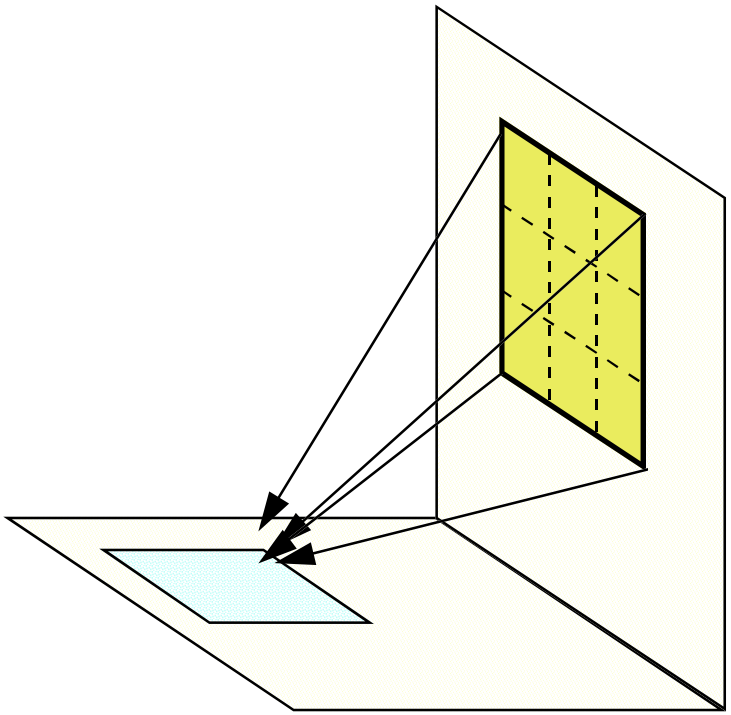
- dopadající energie je málo závislá na drobných detailech zářiče
  - ale závisí na blízkém okolí přijímače (stíny)



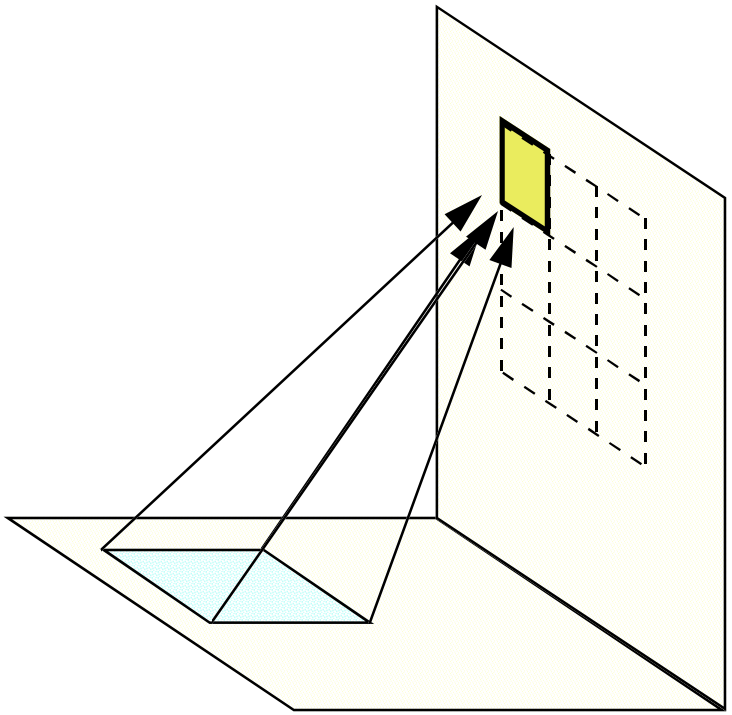
Hierarchické metody

- pro účely **vyzařování** se sousední elementy sdružují do skupin = **ploch**
  - vyzařuje se průměrná radiosita skupiny elementů
- při **přijímání** energie se respektuje původní **jemné dělení** povrchu scény
  - korektní výpočet stínů, linií nespojitosti, ..
- nepotřebuji počítat matici konfiguračních faktorů **velikosti N × N** !

Distribuce energie



vyzařování



příjem

Dvoustupňová hierarchie

- **Cohen** a spol.: 1986
- pro **vyzařování** se elementy sdružují do ploch ("patches")
  - plocha vyzařuje průměrnou radiositu svých elementů (vážený průměr podle $\mathbf{A_q}$)
- **přijímaná** radiosita se přepočítává na jednotlivé elementy plochy
- **adaptivní dělení** ploch a elementů v průběhu výpočtu

Postup výpočtu

1. prvotní rozdělení **M** ploch na **N elementů** (**M<<N**)
   - plochy budou sloužit jako zdroje, elementy jako přijímače
2. výpočet **konfiguračních faktorů** z každého elementu do každé plochy
   - matice ($\mathbf{F_{qj}}$) velikosti $\mathbf{M \times N}$

Postup výpočtu

❸ výpočet **konfiguračních faktorů** mezi jednotlivými **plochami**

– matice (F_{ij}) velikosti $M \times M$

$$F_{ij} = \sum_{q \in i} F_{qj} \frac{A_q}{A_i}$$

❹ řešení soustavy rovnic pro **radiosity ploch**

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

Postup výpočtu

5. výpočet **radiosit jednotlivých elementů**

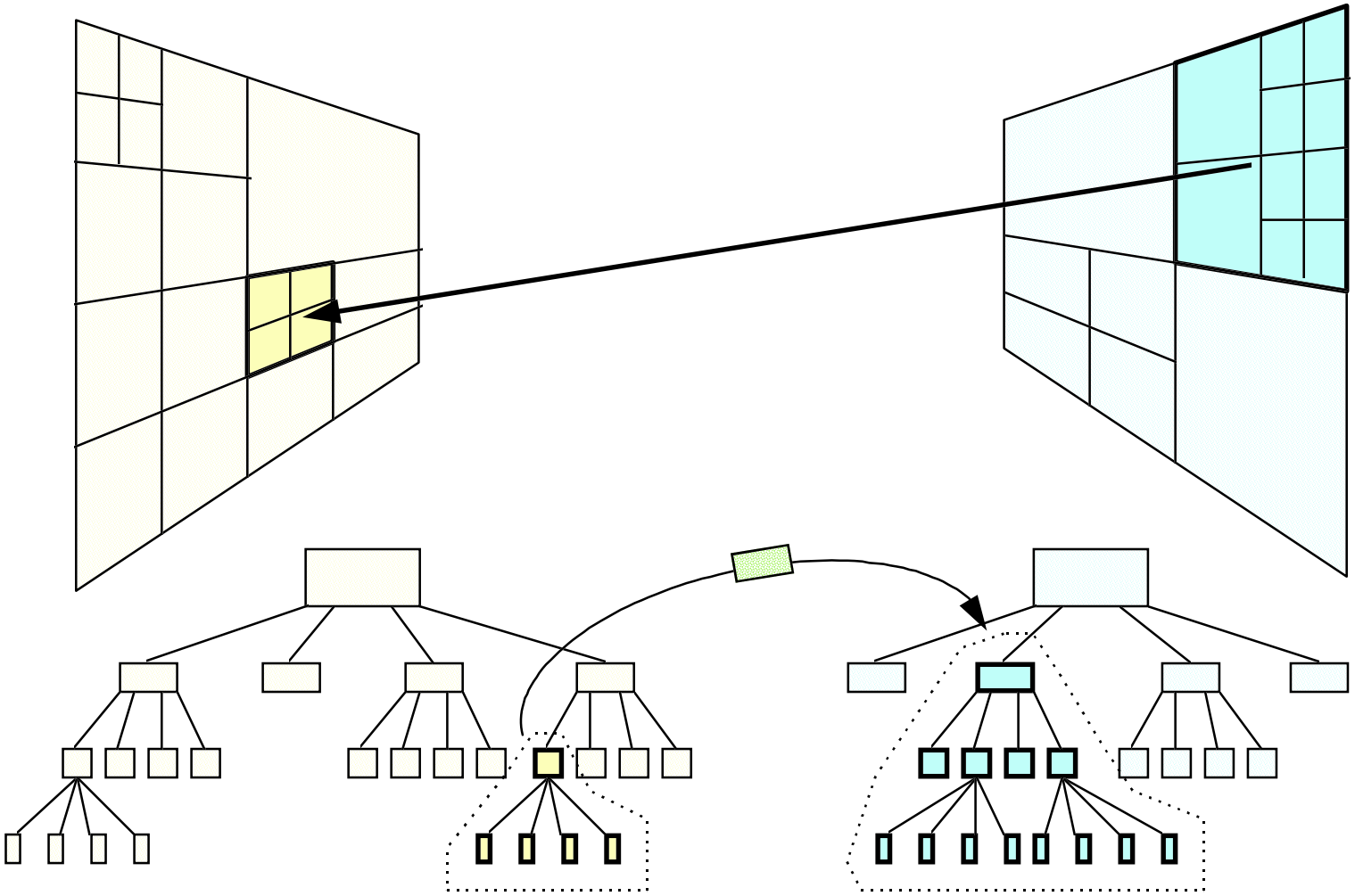
$$B_q = E_q + \rho_q \cdot \sum_{j=1}^{M} B_j F_{qj}$$

6. adaptivní **dělení elementů** s velkým gradientem osvětlení
   - rozdělím-li element **q** do několika menších, v matici (F_{qj}) se pouze příslušná řádka nahradí několika novými
   - řešení stačí jen lokálně přepočítat

Vícestupňová hierarchie

- plochy se dělí na **čtvrtiny** (quadtree)
  - vhodné pro trojúhelníky i čtyřúhelníky
- struktura dělení je reprezentována **stromem**
  - listy = elementy, vnitřní uzly obsahují průměrné hodnoty celého podstromu
- **konfigurační faktory** se neukládají do matice
  - energii si mohou předávat libovolné podstromy v hierarchii
  - konfig. faktor ukládám k odkazu mezi dvěma uzly

Předávání energie



Uzel stromu (“Node”)

```
struct Node {
  float     Bg;          // gathering radiosity
  float     Bs;          // shooting radiosity
  float     E;           // emission
  float     A;           // patch area
  float     ro;          // reflectivity
  struct Node *(N[4]);   // list of children
  struct Link *L;        // list of links
  };
```

Bg ... přijatá radiosita, která nebyla dosud vystřelena

Bs ... celková vyzařovaná radiosita

L ... spojový seznam interakcí (odkazů), obsahuje odkazy na uzly, které na mne září

Přenos energie (spoj, “Link”)

```
struct Link {
   struct Node *p;        // shooting node
   struct Node *q;        // gathering node
   float        Fqp;      // form factor
   struct Link *L;        // next link
   };
```

Přenos energie z plochy **p** na plochu **q**:

p ... emitující plocha

q ... přijímač (u něj je uložen tento seznam)

Fqp ... konfigurační faktor z plochy **q** na plochu **p**

Schéma algoritmu

1. prvotní rozdělení scény na **M velkých ploch**
2. každou dvojici spojím záznamem pro **přenos energie**
   - vyhovuje-li vzájemná konfigurace obou ploch, spojím je přímo; jinak je hierarchicky dělím
3. iterační algoritmus **přenášející energii** přes vybudované spoje (dokud soustava nezkonverguje)

Hlavní rutina, inicializace

```
SolveSHR() {      // solve simple hierarch. radiosity
  InitBs();
  BuildLinks();   // create initial link structure
  SolveHR();      // call the system solver once
}

InitBs() {        // initialize shooting radiosity
  foreach node n
    n.Bs = n.E;   // set shooting radios. to emission
}

BuildLinks() {    // build initial set of links
  foreach node a
    foreach node b
      Refine(a,b); // links each pair of nodes
}
```

Vytvoření spoje

```
Refine(a,b) {     // establish links between a and b
  if ( Oracle(a,b) )
    Link(a,b);    // linking these nodes is fine
  else {
    node = Divide(a,b);  // pick and subdivide 1 node
    if ( node == a )     // a was subdivided
      foreach child r of a
        Refine(r,b);     // descendants of a
    else
    if ( node == b )     // b was subdivided
      foreach child r of b
        Refine(a,r);     // descendants of b
    else
      Link(a,b);         // not subdividable at all
    }
}
```

Orákul (“Oracle”)

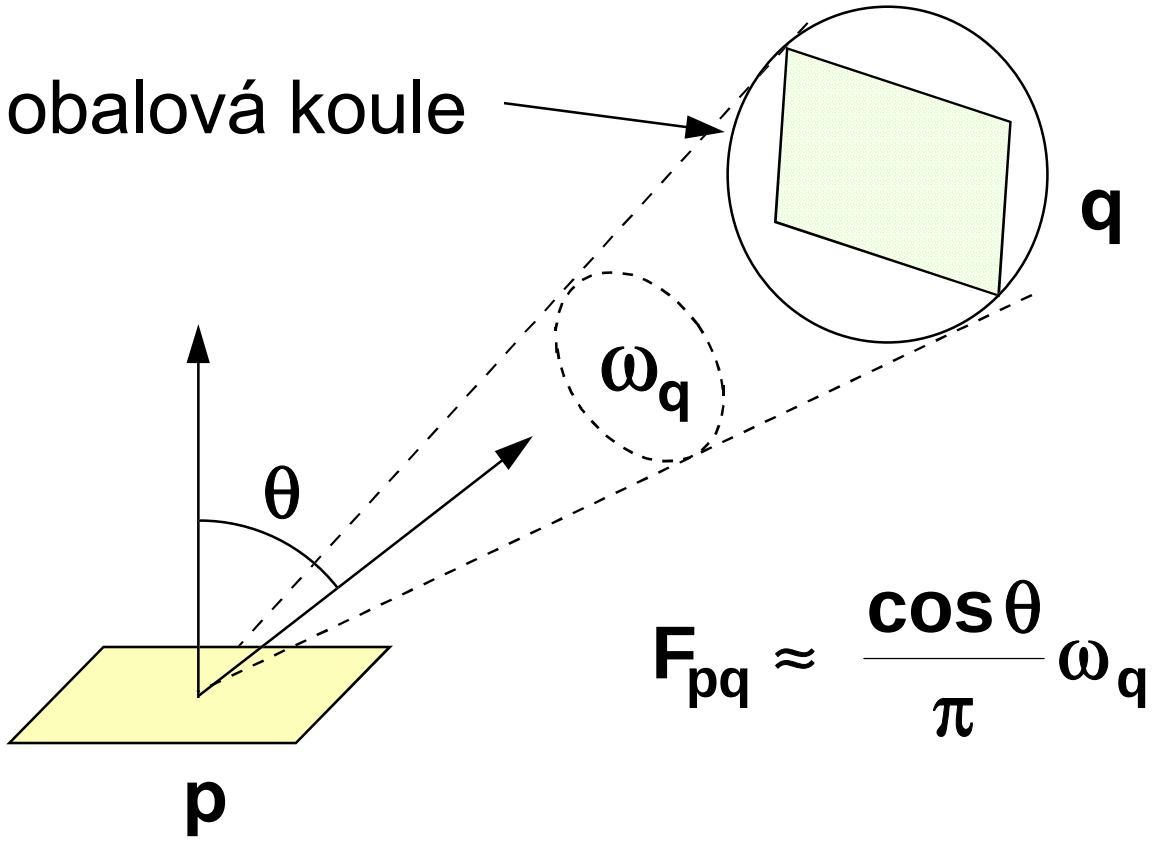
- musí rozhodnout, zda propojením dvou ploch nevznikne velká aproximační chyba
  - odhad variace konfiguračního faktoru bez jeho výpočtu

➡ kritérium **plochy**

➡ odhad velikosti **konfiguračního faktoru**

  - za předpokladu plné viditelnosti

Odhad konfiguračního faktoru



$$F_{pq} \approx \frac{\cos\theta}{\pi}\omega_q$$

Orákul, iterační řešení

```
BOOL Oracle(a,b) {   // OK to link a and b
  if ( a.A < Amin && b.A < Amin ) return TRUE;
    // they are small enough to be OK
  return ( EstimateFormFactor(a,b) < Fmin );
    // is the form factor small enough?
}

SolveHR() {          // balance energy using HR links
  while not converged {
    foreach root node r
      GatherRad(r);      // gather up incoming energy
    foreach root node r
      PushPullRad(r,r.Bg); // give energy to children
    }
}
```

Sbírání energie

```
GatherRad(n) {      // get radiosity into this node
  n.Bg = 0.0;
  foreach link L into n
    n.Bg += n.ro * L.Fqp * L.p.Bs;
      // L.p is shooter patch
  foreach child r of n
    GatherRad(r);  // accumulate for each child
}
```

Sbírá radiositu ze všech spojů do proměnné **Bg**.

Distribuce energie v hierarchii

```
float PushPullRad(n,Bdown) {
  // node n inherits radiosity Bdown
  if ( n is leaf node )
    Bup = n.E + n.Bg + Bdown;
  else {
    Bup = 0.0;
    foreach child r of n
      Bup += r.A/n.A * PushPullRad(r,n.Bg+Bdown);
    }
  n.Bs = Bup;
  return Bup;
}
```

Propaguje posbíranou radiositu do všech potomků daného uzlu (udržuje konzistenci **Bs** a **Bg**).

Adaptivní algoritmus

- předchozí algoritmus je **statický**
  - hierarchie se konstruuje pouze na začátku, bez znalosti jakéhokoliv mezivýsledku (“F-pravidlo”)
- **dynamický** (adaptivní) **algoritmus** modifikuje hierarchii v průběhu výpočtu
  - zjemňování podle hodnoty radiosity (“BF-pravidla”)
- ➡ snahou je **vyrovnat energie** předávané přes jednotlivé spoje
  - tak mizí rozdíl mezi sbíráním a střílením

Hlavní rutina adaptivního alg.

```
SolveAHR() {      // solve adapt. hierarch. radiosity
  InitBs();
  BuildLinks();  // create initial link structure
  do {
    again = FALSE;
    SolveHR();      // call the system solver
    foreach link L
      if ( RefineLink(L) ) again = TRUE;
        // if any link was refined, re-solve later
    } while ( again );
}
```

Zjemnění spoje

```
BOOL RefineLink(L) {
  // returns TRUE if link was refined
  if ( OracleKeep(L) ) return FALSE;
    // no refinement needed
  node = Divide(L.p,L.q);  // pick and subdivide node
  if ( node == L.p )
    foreach child r of L.p
      Link(r,L.q);      // build new links to L.q
  else
    foreach child r of L.q
        Link(L.p,r);    // build new links from L.p
  DeleteLink(L);        // obsolete link
  return TRUE;
}
```

Orákul pro zjemnění spoje

```
BOOL OracleKeep(L) {    // OK to accept this link?
  if ( L.p.A < Amin && L.q.A < Amin ) return TRUE;
    // patches are small enough to be OK
  return ( L.p.Bs * L.p.A * L.Fpq < Wmin );
    // there is not enough energy to distribute
}
```

- v průběhu výpočtu se mohou mezní hodnoty (**Amin**, **Wmin**) zmenšovat
  - systém pak konverguje rychleji k přesnějším výsledkům
  - “multigridding”

Hierarchické bázické funkce

- **hierarchická reprezentace** průběhu radiosity na povrchu těles
  - podobné principy a vlastnosti jako hierarchická radiační metoda (úspora výpočtu konfig. faktorů)

- příklad: **wavelets**
  - možnost rekonstrukce funkce s různou přesností (v různém rozlišení)
  - hodí se pro reprezentaci funkcí s neomezeným spektrem a nespojitostmi

Konec

Další informace:

- **M. Cohen, J. Wallace**: ***Radiosity and Realistic Image Synthesis***, Academic Press, 167-208
- **A. Glassner**: ***Principles of Digital Image Synthesis***, Morgan Kaufmann, 1995, 937-974
- **Cohen, Greenberg, Immel, Brock**: ***An Efficient Radiosity Approach for Realistic Image Synthesis***, CG&A, vol.6, #3, 26-35