Watkinsův algoritmus řádkového rozkladu

CGG MFF UK Praha

pepca@cgg.mff.cuni.cz

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

Watkinsův algoritmus

- **nepotřebuje výstupní buffer**
  - rastrový výstup generuje po jednotlivých řádkách
- **vyplňuje plochy**
  - lze i stínovat
- **žádné pixely se nekreslí zbytečně**
  - nepřekresluje se
  - výhodné zejména pro stínování
- vychází z 2D algoritmu - **vyplňování n-úhelníka v rovině**

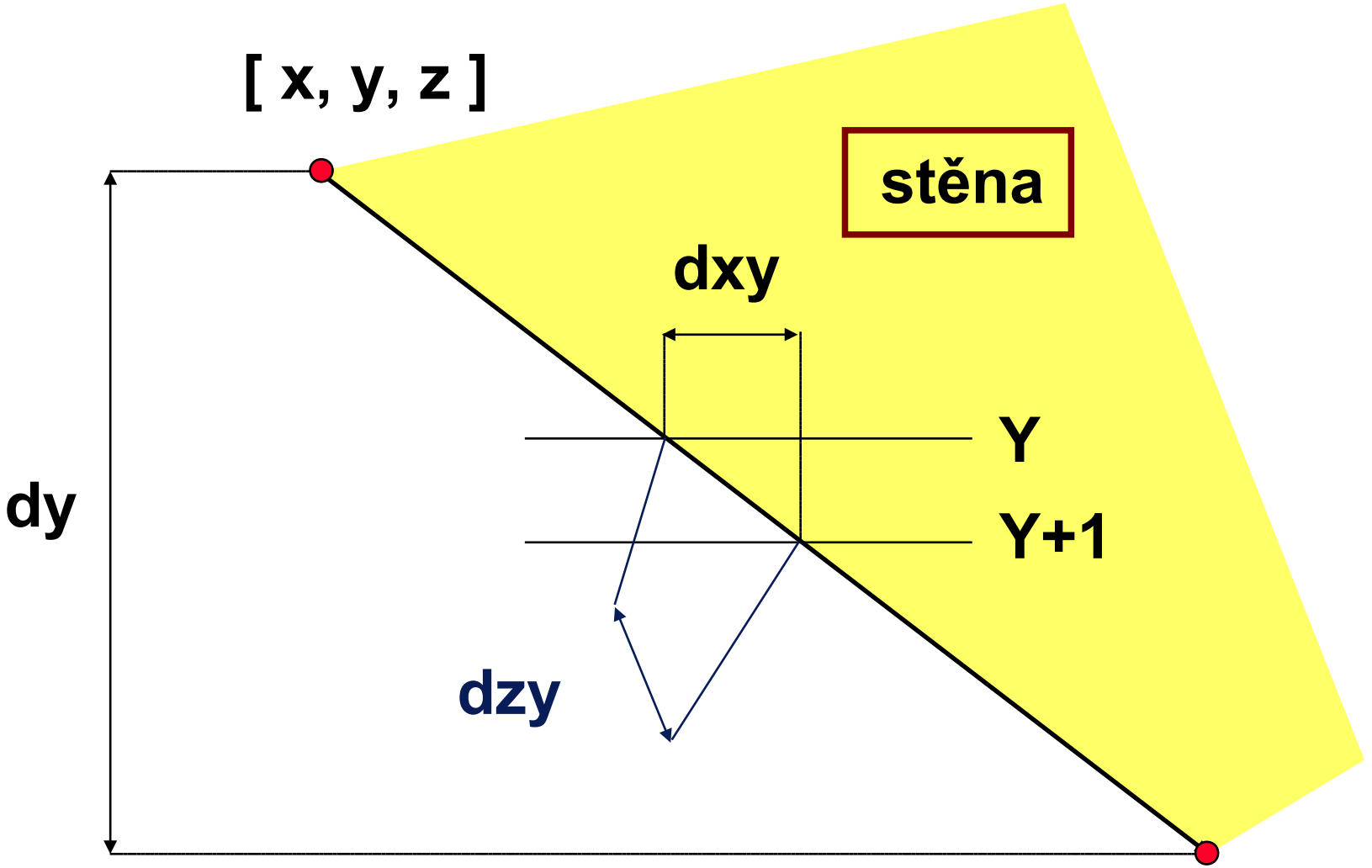
Předpoklady

- scéna je složena z **rovinných plošek**
- každá ploška je zadána posloupností svých vrcholů (3D souřadnice, **z** je hloubka)
- ✔ **zjednodušení**: plošky smějí mít společné body pouze na obvodu (nesmějí se prosekávat)

1. předzpracování

- ze scény **odstraníme odvrácené plošky**
  – podle normálového vektoru
- přivrácené plošky rozložíme na **jednotlivé hrany**
- odstraníme **vodorovné hrany**
- pro ostatní hrany vytvoříme tzv. **pracovní záznamy**

Pracovní záznam pro hranu



Pracovní záznam pro hranu

```
x  : real;      { x horního koncového bodu, později
                  souřadnice průsečíku s aktuální řádkou }
y  : integer;   { y horního koncového bodu }
z  : real;      { z horního koncového bodu }
dy : integer;   { výška hrany v pixelech: |y2-y| }
dxy : real;     { změna x při posunutí na následující
                  řádku (směrnice pro x): (x2-x)/dy }
dzy : real;     { změna z při posunutí na následující
                  řádku (směrnice pro z): (z2-z)/dy }
st : ^stena;    { odkaz na stěnu, do které patří }
```

Záznam pro stěnu

```
dzx : real;
```
{ změna ***z*** při posunutí na následující pixel (směrnice pro ***z***): ***dz/dx*** }

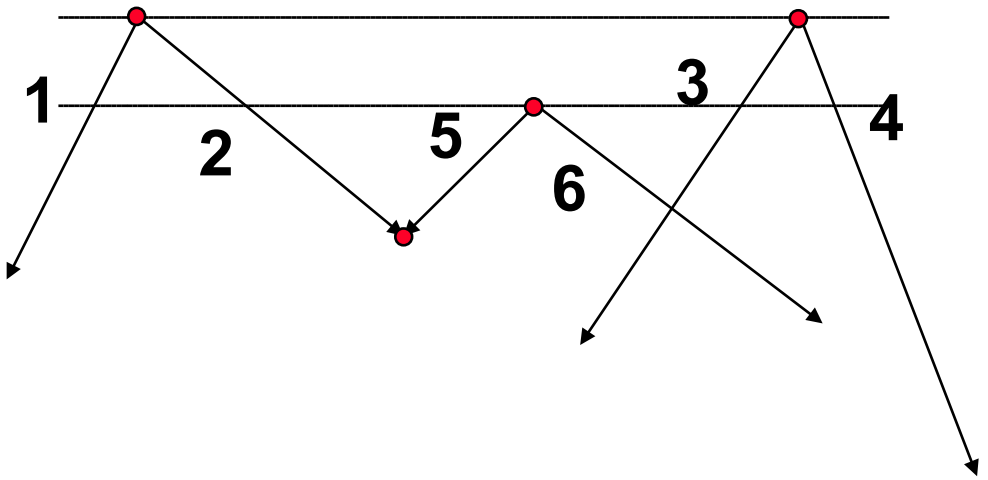
```
barva : barvy;
```
{ barva stěny }

```
...;
```
{ další pomocné údaje pro výpočet viditelnosti }

2. inicializace seznamu *S*

Všechny předzpracované hrany setřídíme do **vstupního seznamu *S*** podle kritérií:

1. vzestupně podle ***y***
2. vzestupně podle ***x***
3. vzestupně podle ***dxy***

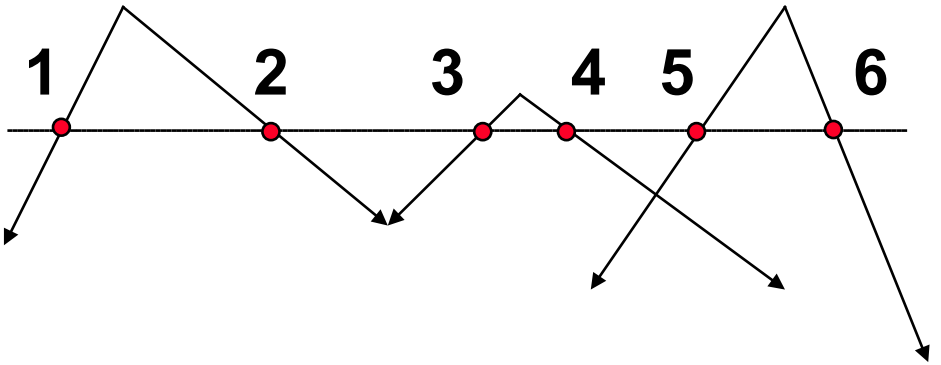


3. inicializace seznamu *A*

Aktuální seznam *A* bude obsahovat všechny hrany, které protínají aktuální řádku. Seznam budeme udržovat setříděný:

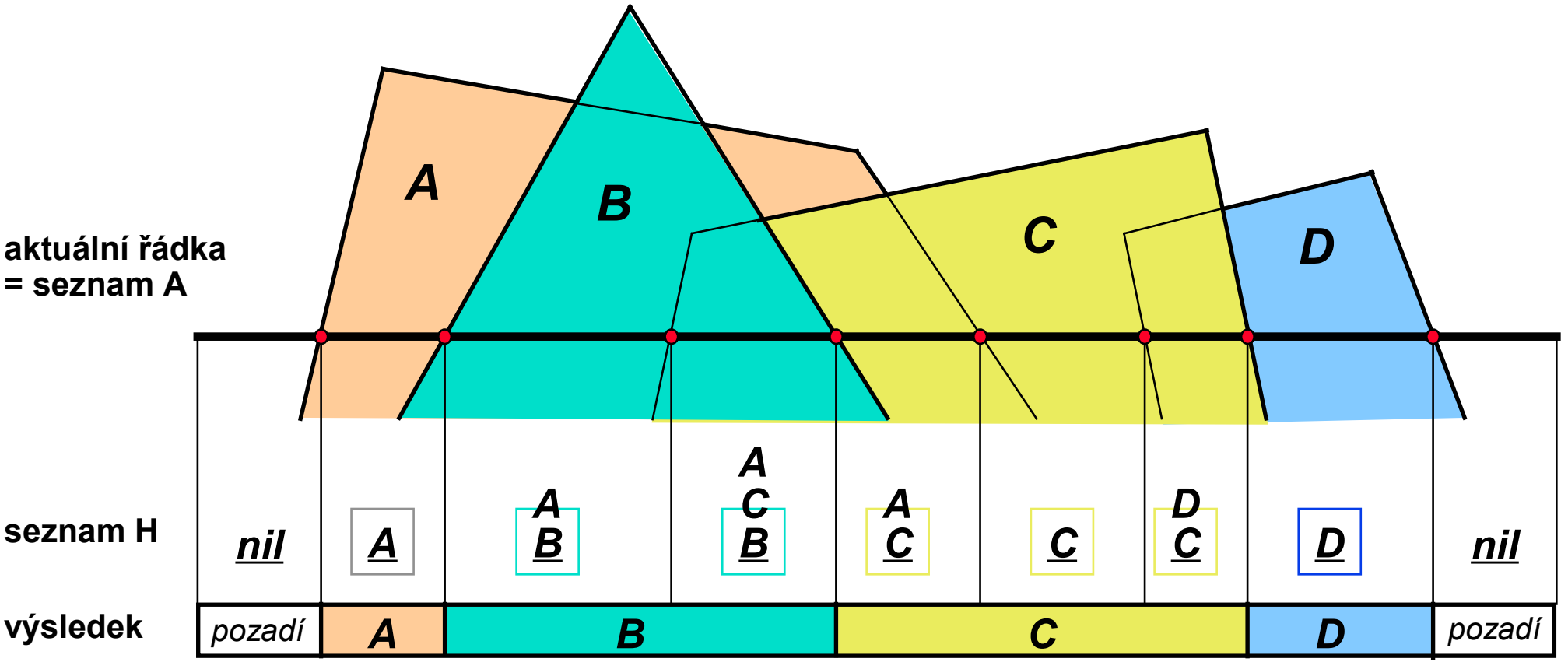
❷ vzestupně podle ***x***

❸ vzestupně podle ***dxy***



Na začátku zařadíme do *A* **počáteční úsek seznamu *S*** - hrany se shodným (minimálním) ***y***

4. výpočet viditelnosti na řádce



4. výpočet viditelnosti na řádce

Je třeba projít **aktuální seznam *A*** a určit viditel-nost. K tomu se dá použít **pomocný seznam stěn *H*** seřazených podle hloubky:

① procházíme ***A*** a na stěnu každé hrany se podíváme do ***H***

② je-li stěna v seznamu ***H***, odstraníme ji a naopak

Při zatřiďování stěn do ***H*** používáme ***z***, ***dzy***

➡ **první prvek *H*** určuje viditelnou stěnu, pokud je ***H*** prázdný, kreslíme **barvu pozadí**

5. přechod na další řádku

Aktualizace seznamu *A*:

```
dy := dy - 1;
 if dy=0 then
x := x + dxy;
z := z + dzy;
```

„odstraň hranu ze seznamu *A*"

- kontrola setřídění *A*
- zatřídění nových hran z ***S*** do ***A*** (počáteční úsek ***S***)

6. podmínka ukončení cyklu

- jestliže je alespoň jeden ze seznamů ***A***, ***S*** **neprázdný** a ještě jsme nedokreslili celou obrazovku, výpočet pokračuje krokem **4**

- jinak algoritmus **končí**
  - případný nedokreslený zbytek obrazovky vybarvíme barvou pozadí

Průnik dvou ploch

- v seznamu ***H*** si vymění dvě plochy své pořadí během procházení aktuální řádky
- do ***A*** přidáme umělou **pomocnou hranu**
- v určování viditelnosti se vrátíme na jejího **předchůdce**

Konec

Další informace:

- **J. Foley, A. van Dam, S. Feiner, J. Hughes**: ***Computer Graphics, Principles and Practice***, 680-686