Zvyšovanie realizmu živých objektov

17.5.2005

Boris Zápotocký

Trendy

- sme schopní renderovať realisticky vyzerajúce stroje, budovy, neživú prírodu
- snaha zlepšiť vzhľad (realizmus) živých objektov, scenérií
  - pridanie srsti, vlasov u živočíchov
  - steblá trávy, stromy + pohyb vo vetre

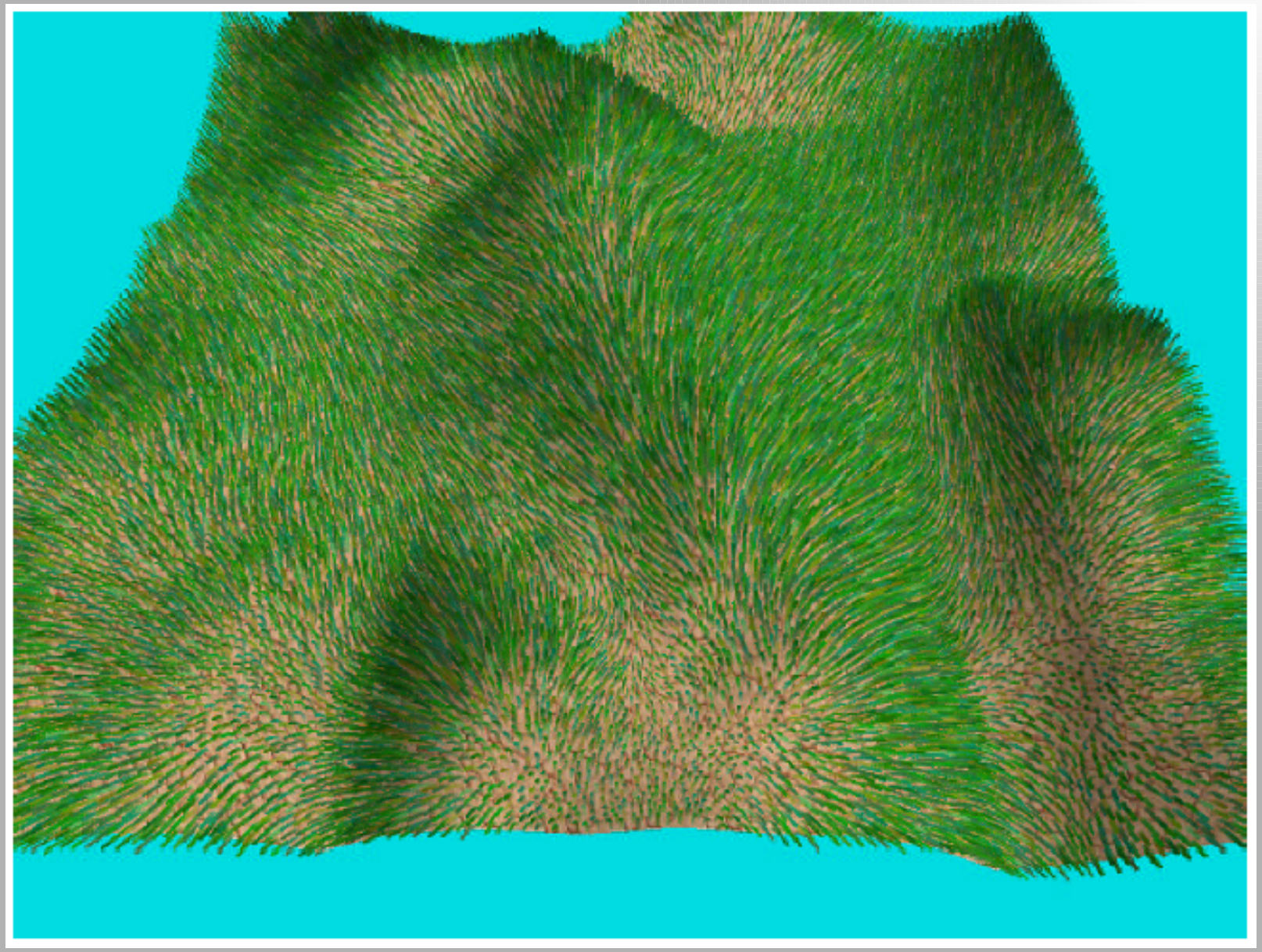
Real-time animovaná tráva

- väčšina real-time aplikácií výrazné zjednodušenie vonkajšieho prostredia, vážne ovplyvňuje realistickosť

- jednoduchá metóda, trávnaté plochy ovplyvnené vetrom (použitie *vertex shaders*)
- posun povrchu podľa lokálneho vektora vetra, zachovanie dĺžky stebiel trávy
- použiteľné aj pre vlasy a srsť

Metóda

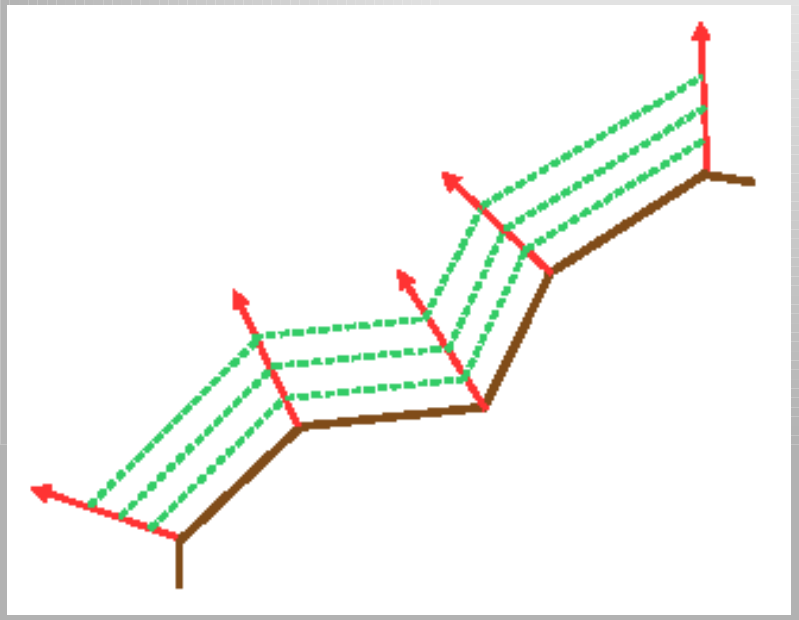
- Tráva zostavená z vrstiev (*transparent*) nad povrchom
- pohyb vrcholov vrstiev real-time (simulované veterné pole)
  - lokálny smer vetra
  - posun podľa globálnej funkcie intenzity
  - smer vetra uložený na úrovni vrcholov (možné úpravy pre jednotlivé vrcholy)
  - vrcholy rozdelené na skupiny podľa hodnoty intenzity (globálne správanie vetra podľa členitosti terénu – terénne vlny, atď.)



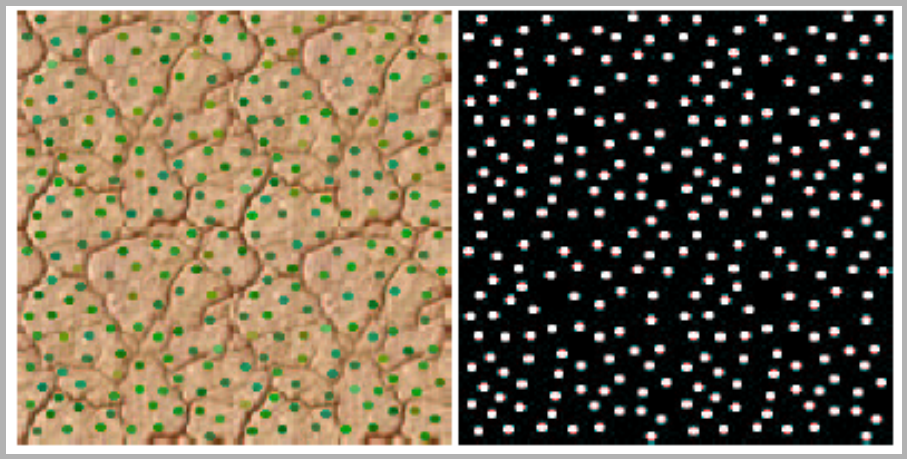
ukážka výsledného efektu

Metóda

- vrstvy = kópie terénu
  (paralelné, zväčšujúca sa vzdialenosť pozdĺž normál)

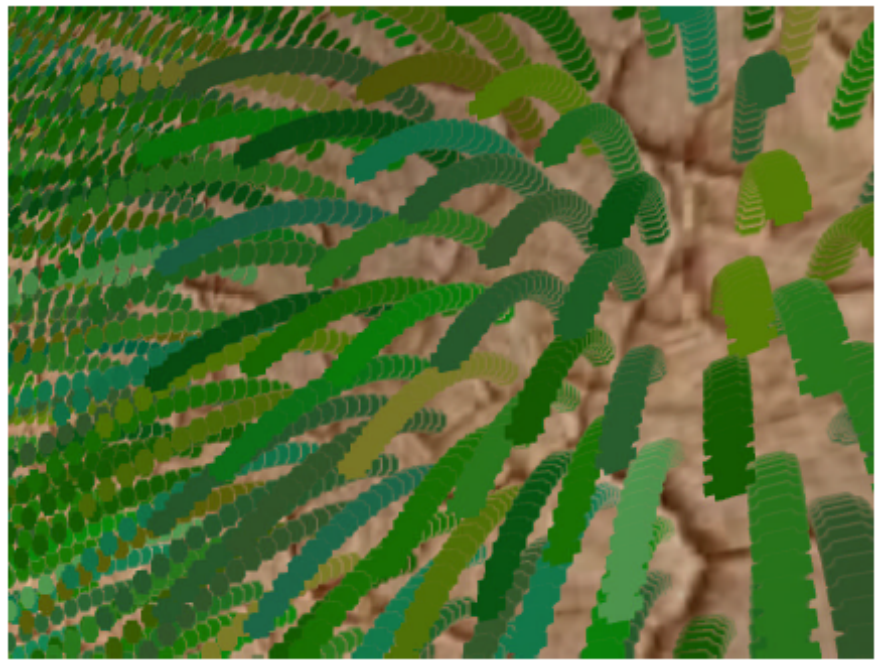


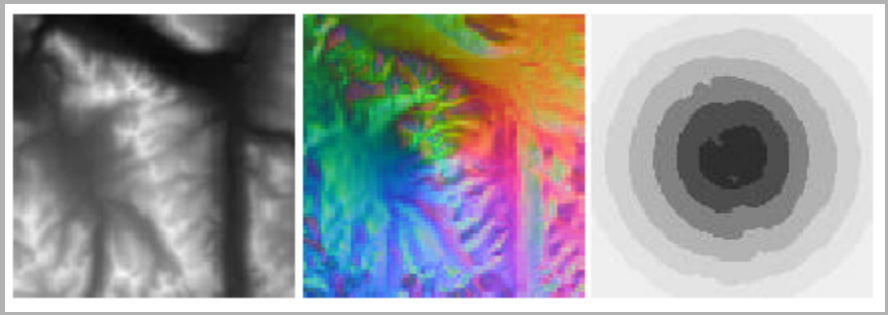
- tráva z jedinej textúry
  (α-kanál, hodnota reprezentuje výšku trávy, 0 zem, 1-255 tráva v rôznych vrstvách)



Metóda

- vrstvy renderované zdola nahor (α-test)
- *wind vector* – heuristika, simulácia...
  - *len príspevok kolmý na normálu povrchu*
  - *dá sa obísť za cenu zníženia výkonu*



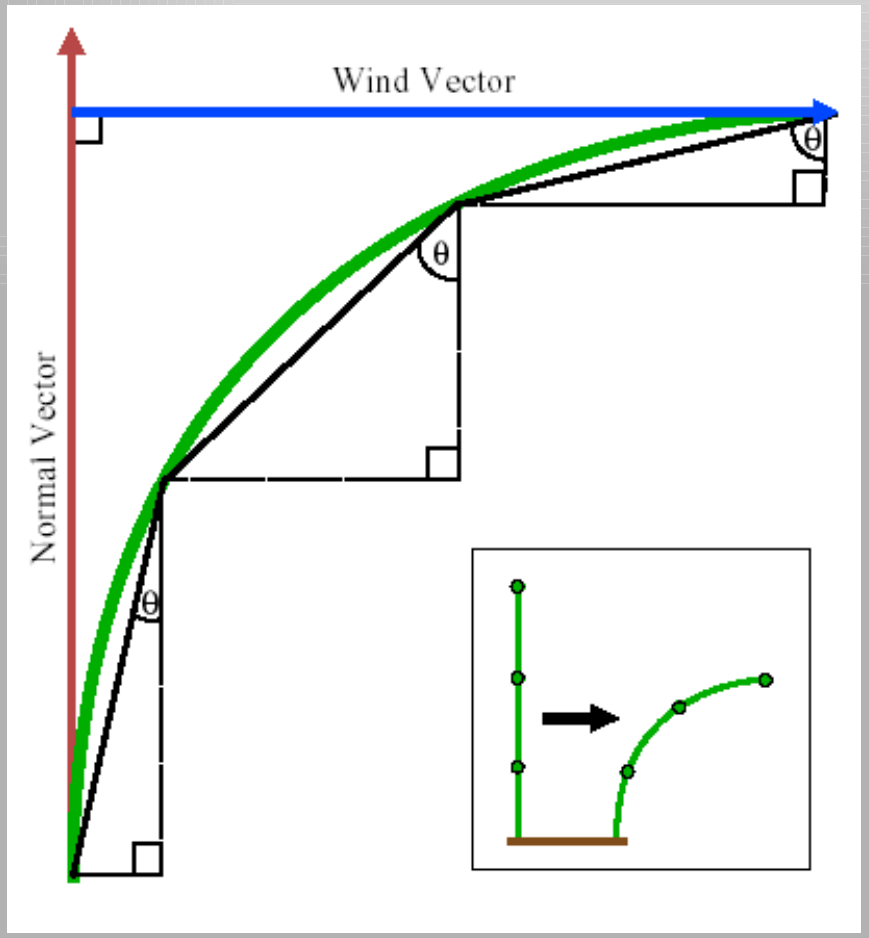


výšková mapa, *wind vector, frame*

Metóda

- posun vrcholov pozdĺž normály a *wind vector* v každom frame

- dĺžka posunu zachováva vzdialenosť medzi vrstvami (zabraňuje zmene výšky trávy pri animácii)



Vzorce

- pohyb pozdĺž normály:

$$\sum_{i=0}^{N} \cos\left(\frac{i \cdot I \cdot \pi \cdot S}{2 \cdot N}\right)$$

- pohyb pozdĺž *wind vector*:

$$\sum_{i=0}^{N} \sin\left(\frac{i \cdot I \cdot \pi \cdot S}{2 \cdot N}\right)$$

i *vrstva,* **I** *intenzita vetra vo vrchole,* **S** *pokojová vzdialenosť medzi vrstvami,* **N** *počet vrstiev*

Metóda

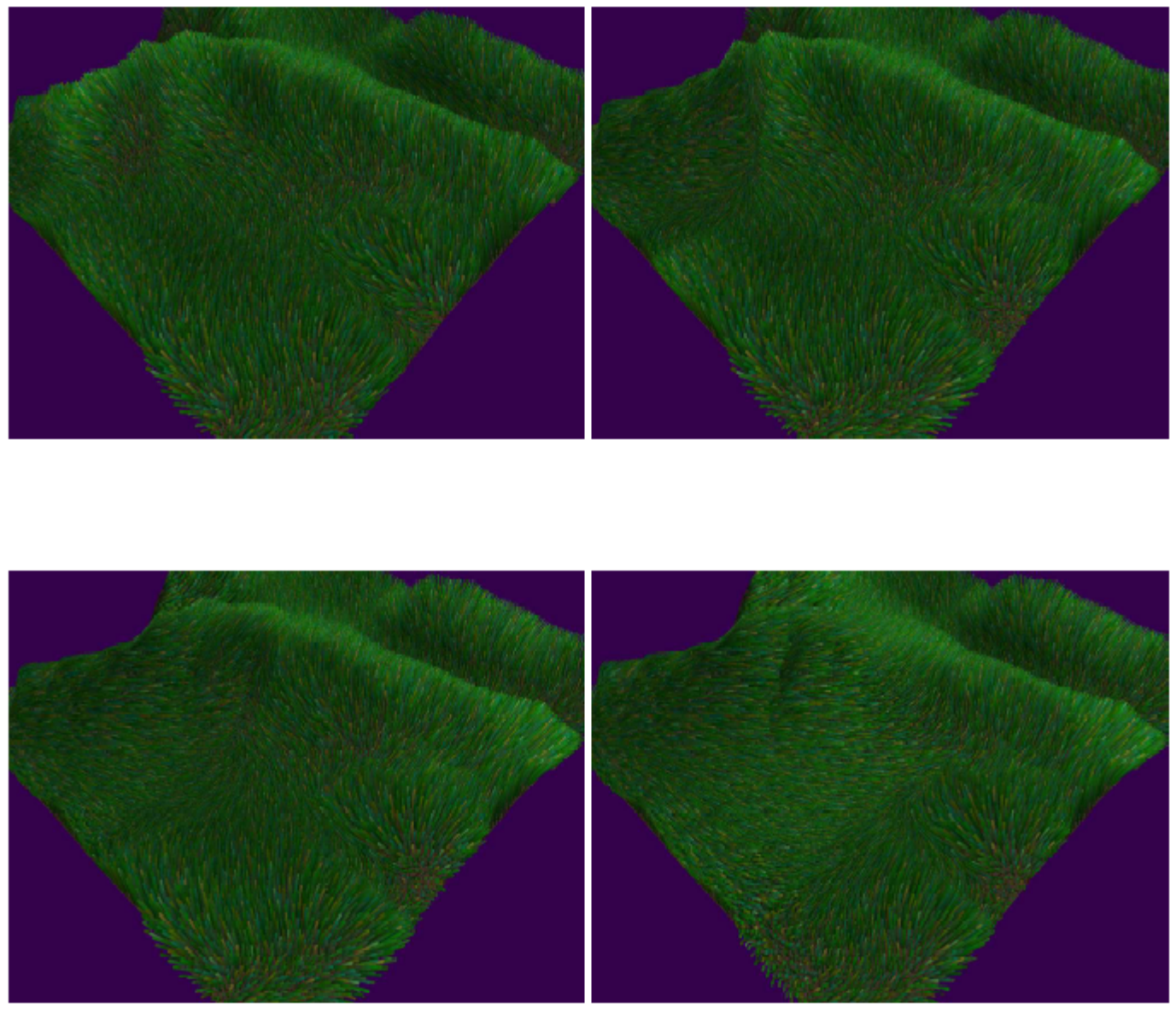
- steblo trávy je zložené zo svojich segmentov
- intenzita vetra - ľubovoľná fcia (najlepšie periodická, spojitá s hodnotami medzi –1.0 a 1.0)
- frame – podľa vzdialenosti od zdroja vetra, efekt skôr na steblá bližšie
  - Zvyšuje realizmus, vlnenie trávy...

Výsledky

- Zostava: Pentium 1.7GHz, 512MB RAM, nVIDIA GeForce3 32MB
- Testy: 32bit colour, 16bit Z-buffer

		Resolution	
Shells	Polygons	640 x 480	1024 x 768
0	512	293.5	130.8
8	4608	47.0	22.8
16	8704	25.6	12.4
32	16896	13.4	6.5
64	33280	6.9	3.3

www.cs.ubc.ca/labs/imager/th/pdf/Bakay2003.pdf



Real-time animovaná tráva 2

- detailné polygónové spracovanie príliš náročné
- potrebujeme jednoduchú alternatívu
  - množstvo stebiel, ale málo polygónov
  - dostatočné hustý vzhľad z rôznych uhlov pohľadu
  - realistická animácia

Príprava

- jednoduchý polygón, namapovaná textúra s niekoľkými steblami trávy (*α* a *color layer* – viac kombinácií pre lepší efekt*)*
- pásy polygónov nevhodné
- polygóny sa pretínajú (konfigurácia hviezda, trojuholník)

- normály vrcholov paralelné s vertikálnymi hranami => správne osvetlenie
- zrušený *back-face culling*
- veľa objektov, blízko pri sebe

  => efekt prírodnej lúky

Animácia

- pohyb horných vrcholov polygónov
- výpočty založené na *sin* a *cos* fciách

- tri spôsoby animácie:
  - animácia zväzkov objektov
  - animácia jednotlivých vrcholov
  - animácia jednotlivých objektov

Animácia zväzkov objektov

- Posun vrcholov uniformne pre celú skupinu
  - pre dobrý vzhľad dôležitá vhodná veľkosť skupiny
  - každý zväzok vlastný parameter posunu

Výhody: komplexné výpočty na CPU, nie sú deformácie textúr (konštantná vzdialenosť vrcholov polyg.)

Nevýhody: veľa *draw* volaní na vykreslenie trávnatej plochy, rozlíšiteľné zväzky (rovnaký pohyb)

Animácia jednotlivých vrcholov

- Všetky výpočty na GPU, pohyb vrcholu na základe polohy

  Výhody: málo *draw* volaní, možnosť šírenia efektných vĺn, nie sú rozlíšiteľné zväzky

  Nevýhody: deformácia textúr (hrubnutie, stenčovanie stebiel), nedostatočný lokálny chaos, obmedzená komplexnosť výpočtov

Animácia jednotlivých objektov

- pohyb vrcholov na základe posunu stredu objektu
- pridanie stredu objektu do štruktúry

Výhody: málo *draw* volaní, nie sú deformácie, lokálna pestrosť zlepšuje prirodzený vzhľad

Nevýhody: dáta naviac pre každý vrchol, obmedzená komplexnosť výpočtov

GPU Gems, Chapter 7

Animovanie vlasov *real-time*

- *fyzikálna simulácia pohybu vlasov (hair-strip)*
  - vplyvy: gravitácia, kolízie, trenie...
  - človek 100 000 – 150 000 vlasov, rôzna farba, variabilná dĺžka, vlnitosť, hrúbka...

    => zjednodušené fyzikálne pravidlá a detekcia kolízií

  - aproximácia pruhmi vlasov (strips, clusters)
  - viac strips = vyššia realistickosť = vyššia výpočetná zložitosť

Princíp modelu

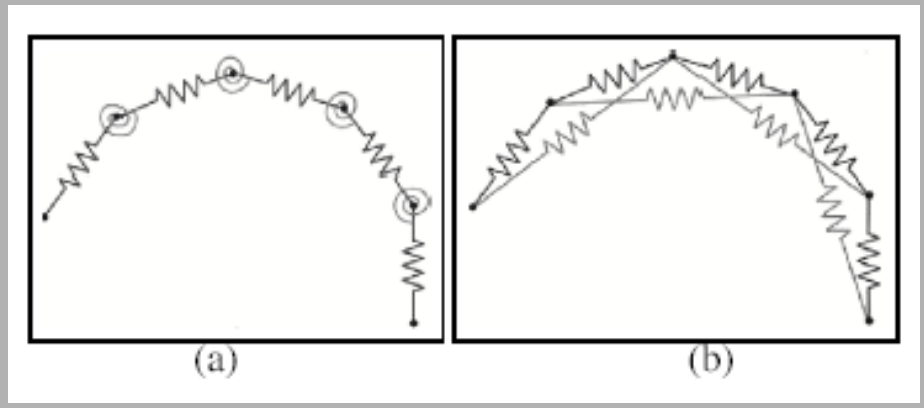
- Čo zahŕňa renderovanie vlasov:
  - spracovanie farby, textúry, tiene, lesk, transparentnosť, anti-aliasing

Popisovaný model:

- textúra s α-kanálom mapovaná na strips (pre efekt rôzne ukončených vlasov, rozstrapkanie...)
- zamerané na dynamiku, žiadna vlnitosť a iné štýly...
- „systém pružín“:

  každý strip sústava bodov

  (ovplyvňujú tuhosť aj elasticitu celého stripu)



Dynamika

- efekt ohybovej tuhosti a kolízie s hlavou na príklade:

- sily vplývajúce na pohyb stripu:

  sily medzi pružinami, gravitácia, spätný ráz kolízií, trenie...

  - *Komplexná hair-to-hair interakcia príliš náročná, implementovaná interakcia strip-telo*

Implementácia

- simulácia (každý frame):
  - inicializácia, riešenie silových pôsobení, update
  - každý strip separátne (pôsobenie jeho častíc medzi sebou)
  - častice inicializované do stabilných stavov
  - každá nová sila pridáva podmienku obmedzujúcu pohyb celej masy (pružina je ďalšia sila pôsobiaca medzi dvomi masami, ktoré spája...)
  - externé sily pridané nakoniec, určujú výsledný smer pohybu vlasov
  - pri update sa zmena smeru a rýchlosti pohybu častíc premietne do novej polohy

Vlastnosti štruktúr

- *masa:* pozícia, sila pôsobiaca v daný okamih
- *pružina:* fyzikálne vlastnosti pružiny, vnútorné trenie, dĺžka v stabilnom stave
- ďalšie konštanty: trenie vzduch, zem, spätný ráz zeme, absorbcia zeme, gravitačné zrýchlenie

- hlava = guľa, zem = jednoduchá rovina
- *nepočítajú sa kolízie medzi telom a modelom*

Implementácia

- *Eulerova metóda* riešenia pohybových rovníc (nie vždy presná, ľahká implementácia)
- Implementácia v C++ a OpenGL

- ukážka pseudokódu:

Algoritmus pre riešenie simulácie vlasov

```
procedure SolveHairSimulation {

  for i from 0 to numHairStrands do {
    // Initialize the masses for this frame
    for j from 0 to hair(i)->numParticles do
        hair(i)->particle(j)->force := 0;

    // Solve spring forces and apply forces
    // on the masses that this spring connects
    for j from 0 to hair(i)->numSprings do {
      hair(i)->spring(j)->force := 0;
      // Solve spring equation by taking the
      // spring's internal friction into account
      solve(hair(i)->spring(j));

      // Apply the spring force to the masses
      // in opposite directions
      applyForce(hair(i)->spring(j)->particle1,
                 hair(i)->spring(j)->force);
      applyForce(hair(i)->spring(j)->particle2,
                 -1 * hair(i)->spring(j)->force);
    }
```

Algoritmus pre riešenie simulácie vlasov

```
// Apply other forces
for j from 0 to hair(i)->numParticles do {
  // Apply gravitation
  applyForce(hair(i)->particle(j),
             gravitation_force);
  // Apply air friction
  applyForce(hair(i)->particle(j),
             air_friction_force);
  // Collide with the head
  if hair(i)->particle(j) is colliding
     with the head then do
     applyForce(hair(i)->particle(j),
                head_repulsion_force);
```

Algoritmus pre riešenie simulácie vlasov

```
        // Collide with the ground
        if hair(i)->particle(j) is colliding
           with the ground then do
        {
          // Absorb some energy from the hair
          // by applying an absorption force
          applyForce(hair(i)->particle(j),
                     ground_absorption_force);
          // Apply the ground friction
          applyForce(hair(i)->particle(j),
                     ground_friction_force);
          // Apply the ground repulsion
          applyForce(hair(i)->particle(j),
                     ground_repulsion_force);
        }
      }
    }
}
```

Riešenie pohybových rovníc častíc

```
procedure SimulateHair(dt) {

  // Solve the simulation
  SolveHairSimulation();

  for i from 0 to numHairStrands do {
    for j from 0 to hair(i)->numParticles do {
      // Calculate new velocity of each particle
      hair(i)->particle(j)->velocity +=
        dt * (hair(i)->particle(j)->force /
              hair(i)->particle(j)->mass);
      // Calculate new position of each particle
      hair(i)->particle(j)->position
        += dt * hair(i)->particle(j)->velocity;
    }
  }
}
```

Výsledky:

Zostava: Pentium IV (HT) 2.80GHz, NVIDIA GeForce FX 5600, 512MB RAM

NHS	*PES*	*NSP*	*SES*	*NSS*	*FPS*
100	10	1000	17	1700	62.50
100	15	1500	27	2700	62.50
100	20	2000	37	3700	62.50
100	25	2500	47	4700	31.50
200	5	1000	7	1400	62.50
200	10	2000	17	3400	62.50
200	12	2400	21	4200	32.50
200	14	2800	25	5000	10.60
300	6	1800	9	2700	62.50
300	7	2100	11	3300	45.20
300	8	2400	13	3900	32.00
300	10	3000	17	5100	6.41
400	3	1200	3	1200	62.50
400	5	2000	7	2800	45.20
400	6	2400	9	3600	21.25
400	7	2800	11	4400	8.00

NHS =

počet strips

PES =

častíc v každom stripe

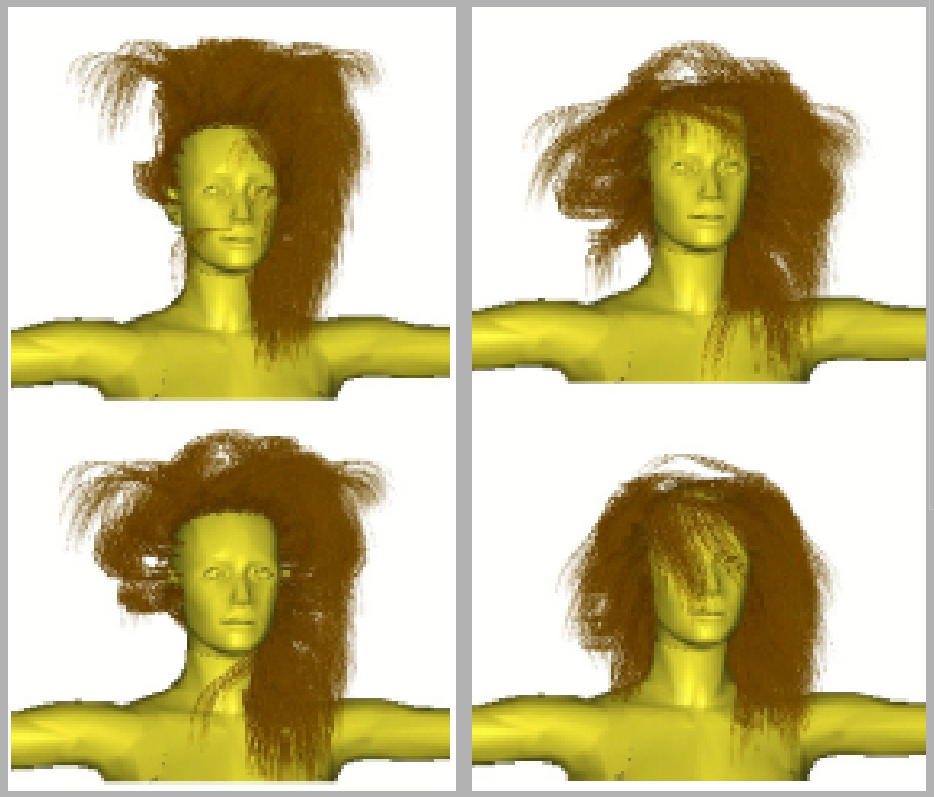
SES =

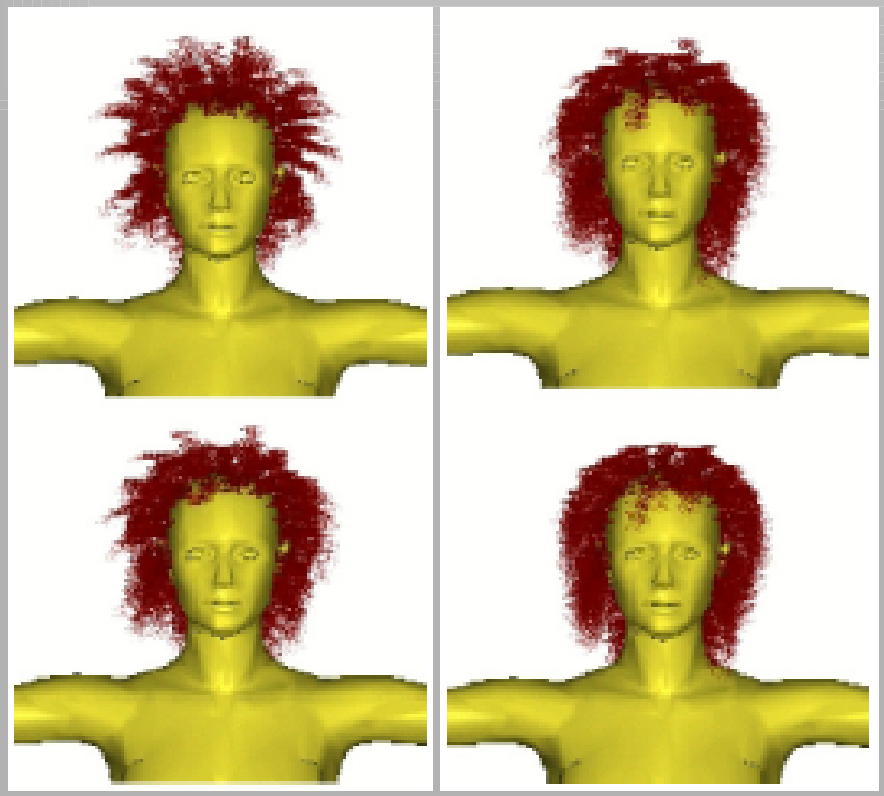
pružín v každom stripe

FPS =

frames/s

wscg.zcu.cz/wscg2005/Papers_2005/Short/A37-full.pdf

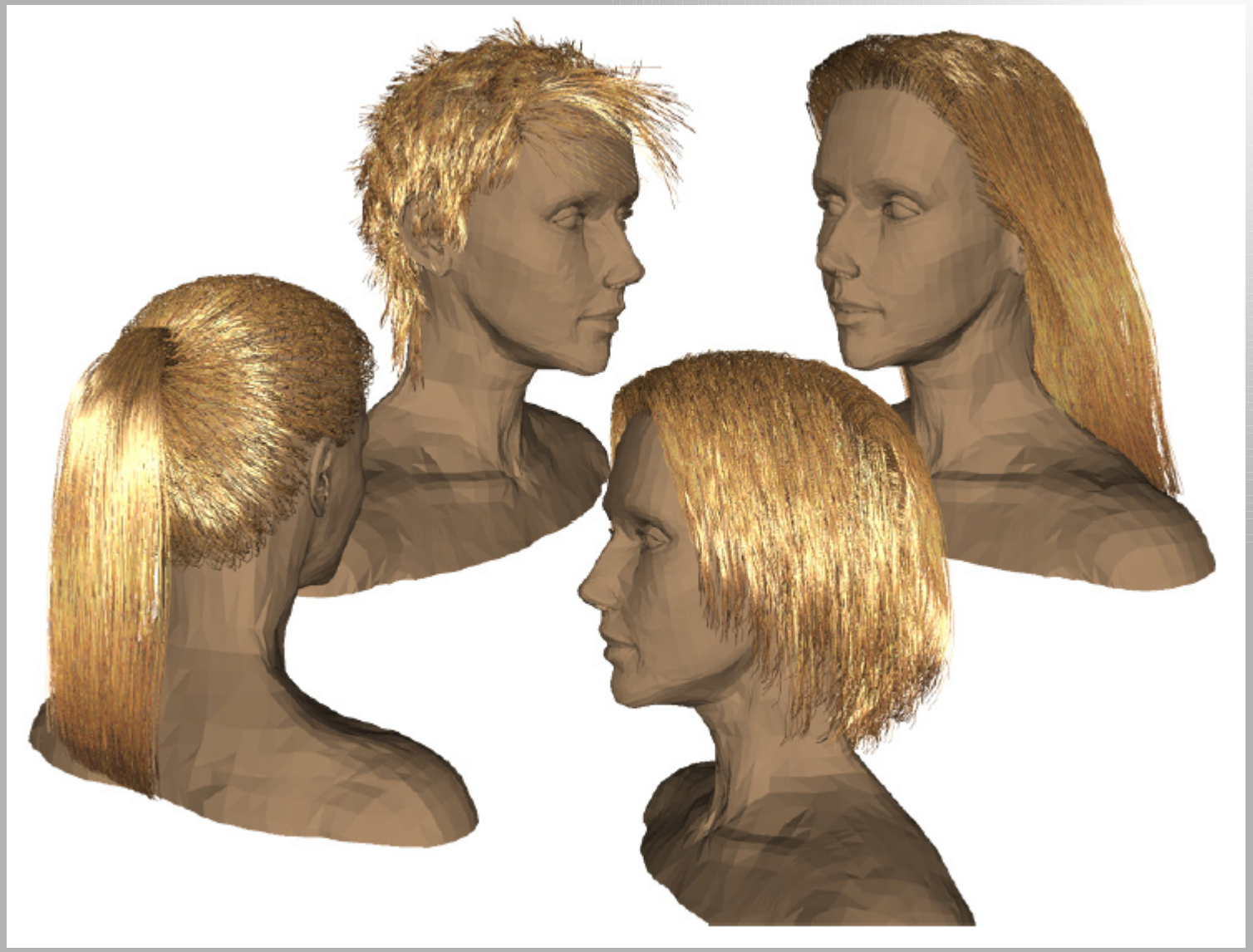




ukážka pohybu vlasov
pri dopade človeka
po výskoku

Animovanie komplexných „hairstyles“

- iný prístup: objemové dáta
- volume free-form deformácia
- model založený na mechanickej deformácii mriežky (systém častíc) – numerické metódy
- animácia vlasov pomocou kvadratickej B-spline interpolácie
- kolízie s povrchom tela
- Level of Detail optimalizácia



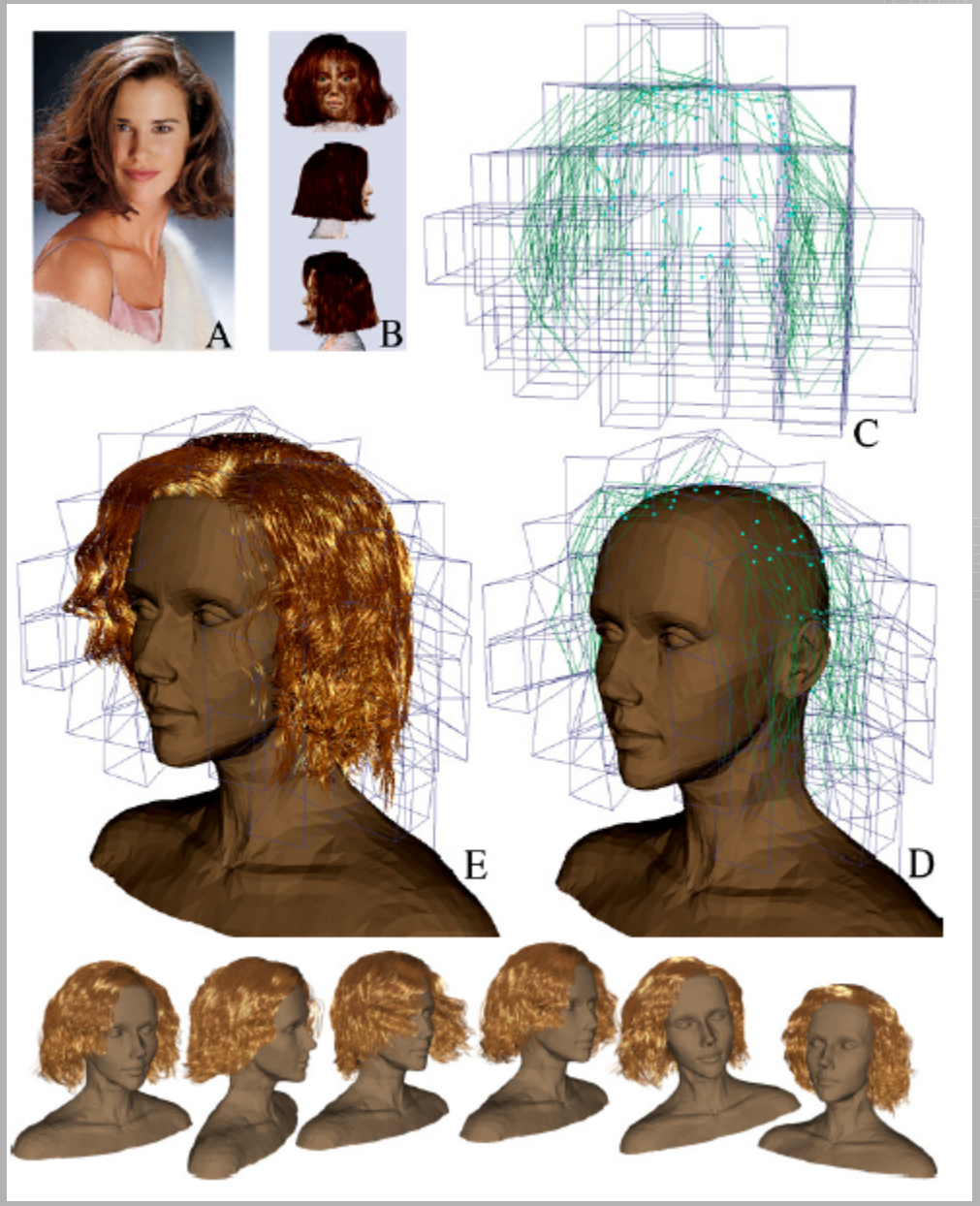
Systém je schopný (real-time) animovať účesy veľkej komplexnosti

Metóda

- princíp:
  - pri predspracovaní definovaná *mriežka*
  - *mechanický model* (prepojenie mriežky a mechanických vlastností vlasov) – založený na systéme častíc vo vrcholoch mriežky
  - možnosť stanoviť vhodný kompromis presnosť <->výpočtová rýchlosť
  - kolízie nad mriežkou
  - výpočty nad mriežkou, renderované parametre interpoláciou

Metóda

- výhody
  - jednoduché interpolácie nad kubickou mriežkou
  - ľahká lokalizácia bodov mriežky
  - malá závislosť na vláknovom charaktere vlasov
  - voľnosť v designe účesov
  - ľahká škálovatelnosť + nezávislosť zložitosti mechanického modelu (mriežka) a zložitosti samotného účesu



Proces animácie:

A vzor

B model účesu

C mriežka a zjednodušený mechanický model

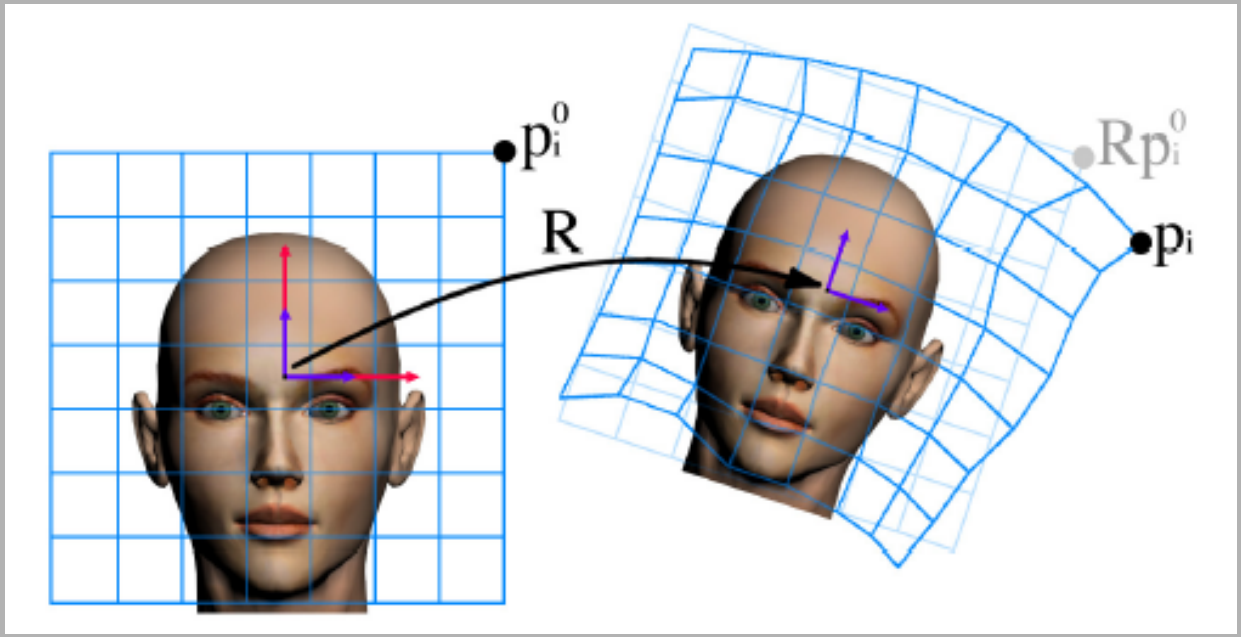
D mechanická deformácia mriežky

E vypočítanie polohy vlasov pomocou interpolácie

+real-time animácia a rendering

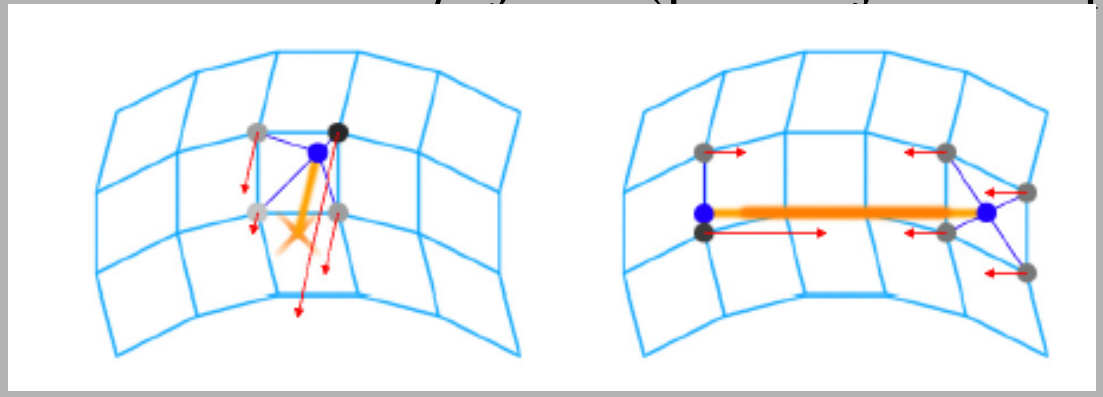
Mriežka

- Pohyb mriežky bez deformácie (rigid-motion)
- Deformácia okolo rovnovážnych bodov (súčasná poloha tuhej (rigid) mriežky



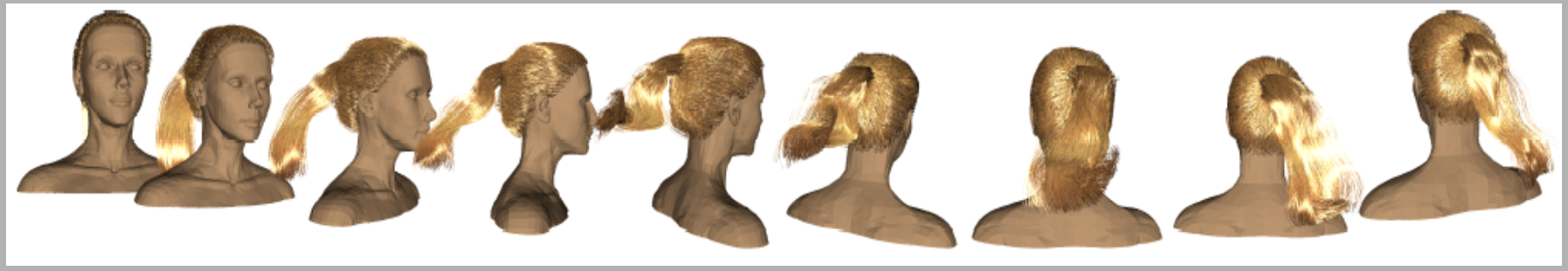
Model

- mechanický model interakcie (vážená suma uzlov mriežky)
  - váhové vektory (definujú miesta pôsobenia síl v mriežke) – *spevňovače mriežky*
    - *lineárne mriežkové pružiny* (spojenie dvoch bodov, prenos energie na príslušné vrcholy mriežky)
    - *lineárne mriežkové napojenia* (priťahujú bod k polohe)



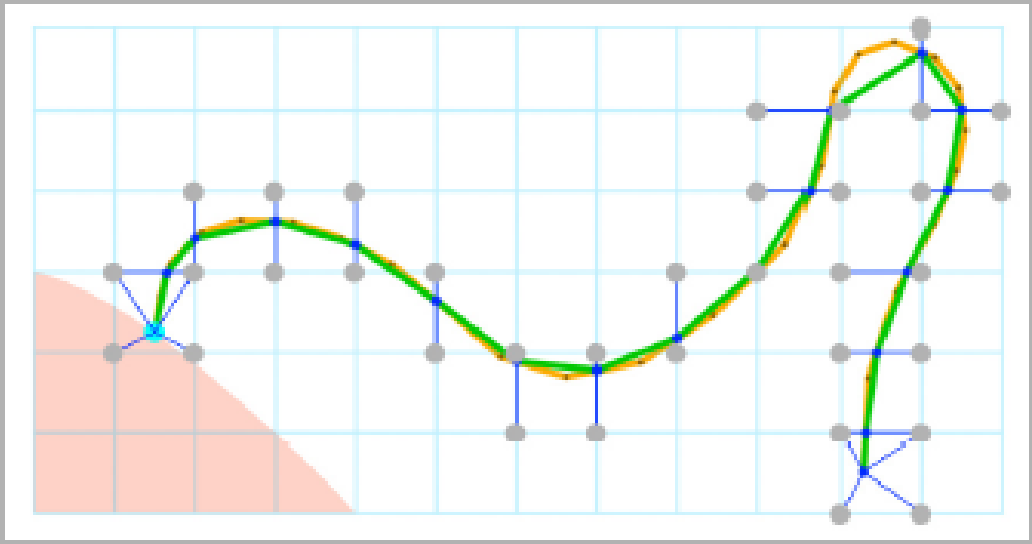
Budovanie vlasových mechanizmov

- mechanický model (zjednodušenie fyzikálnych vlastností vlasov, pripojenie k lebke)
- éter (definuje pokojovú polohu mriežky, externé sily: gravitácia, aerodynamické efekty)
- kolízie (zabraňujú prenikaniu vlasov do tela)



Mechanický model

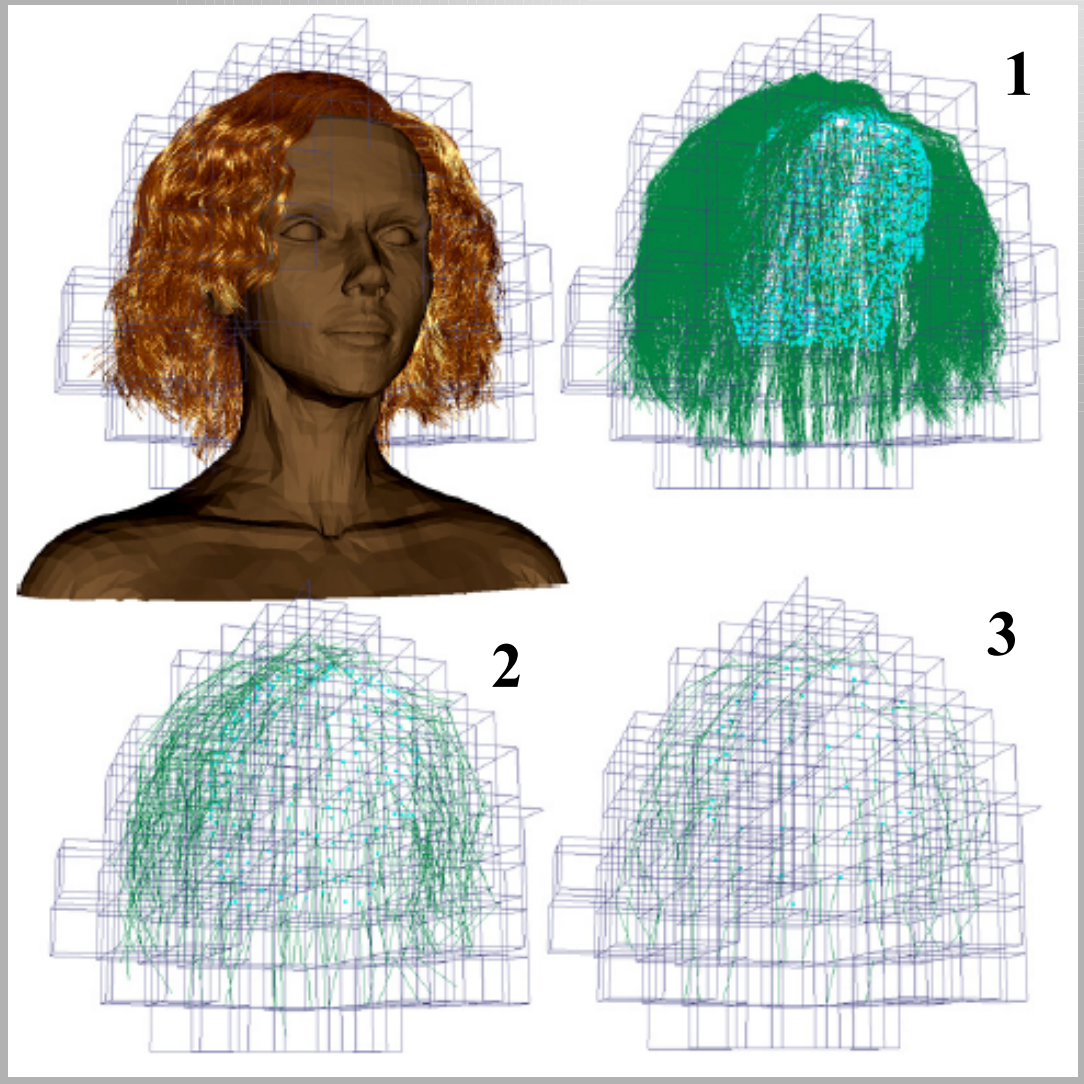
- Pružiny – zabezpečujú elasticitu segmentov prameňa vlasov (parametre podľa druhu modelovaných vlasov)
- Napojenia – pripojenie vlasov k hlave



Mechanický model

- zjednodušenie:
  spájanie a rušenie pružín na základe výpočtu mechanickej chyby...

  => *nižšia výpočtová náročnosť*



p – pružina, n – napojenie

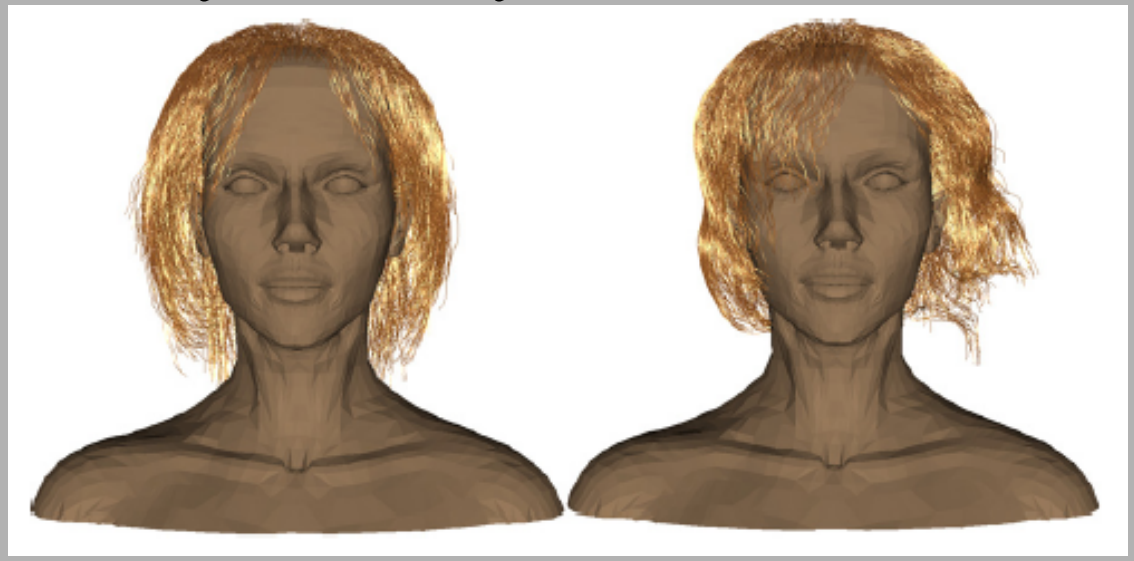
1 *14200p, 3600n*

2 *800p, 200n*

3 *200p, 50n*

Éter

- k zabráneniu nevhodnej deformácie a oscilácie blokov na okrajoch (každý bod mriežky ešte ukotvený k svojej stabilnej polohe)
- nastavenie parametrov umožňuje upraviť celkovú tuhosť účesu (simulácia použitia gélu...)
- externé sily – aerodynamický efekt (ťah na uzol z okolia), vietor

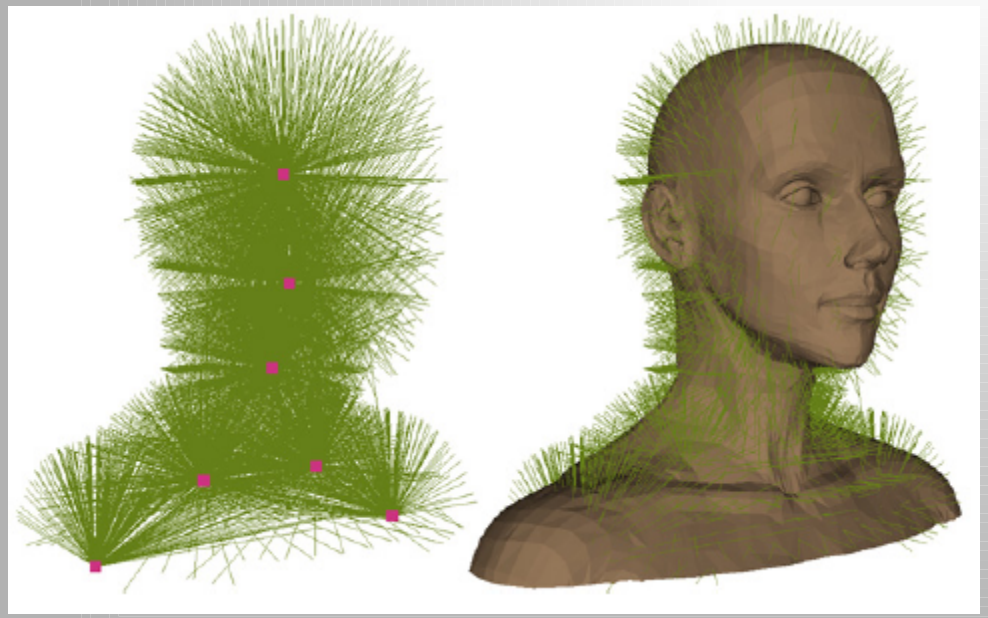


Kolízie

- kolízia jednotlivých vlasov OK, deformácia mriežky zachováva relatívnu pozície medzi prameňmi vlasov
- problém: detekcia a reakcia na kolíziu medzi vlasmi a telom
- *Metaball* (polynóm 6. stupňa), aproximácia povrchu tela – gule s rôznym polomerom + odpudivá sila
- Vysoká presnosť vyžaduje mnoho *metaballs*

  => riešením je pre každý uzol mriežky vlastná reprezentácia najbližšieho okolia pomocou pár *metaballs* s vhodnou presnosťou
- Rádius zvolený, aby sa uzol mriežky nemohol dostať bližšie k telu ako na vzdialenosť „hrúbky vlasu“

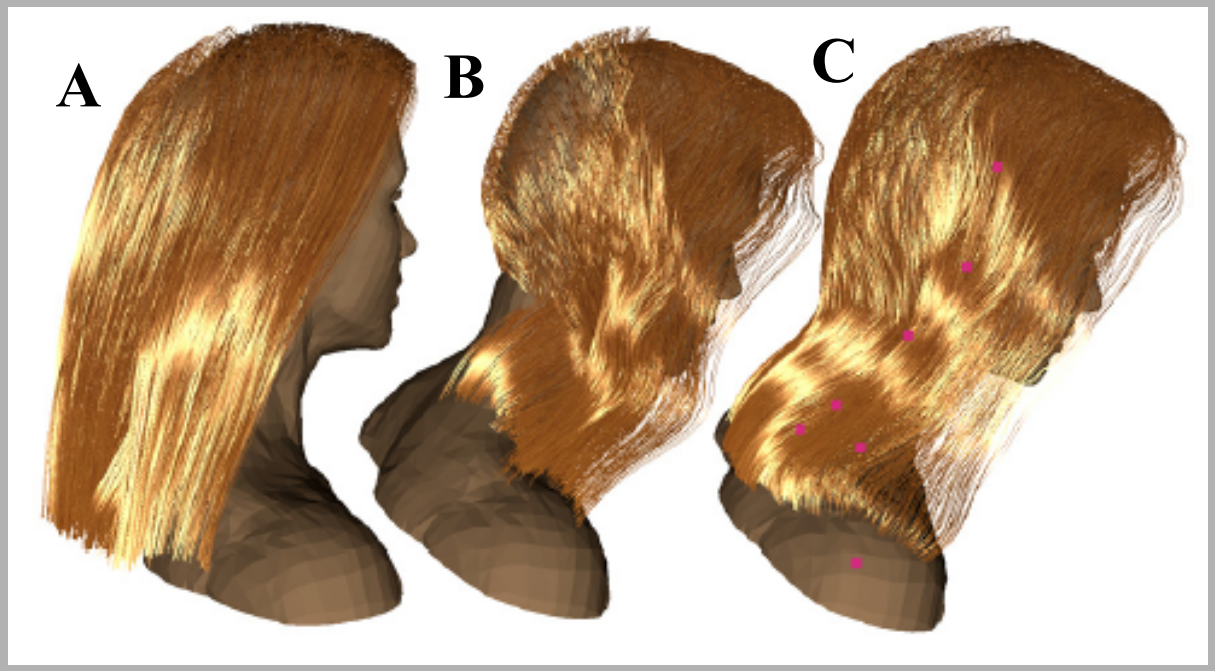
- Ukážka aproximácie pomocou *metaballs*



A účes

B kolízie bez *metaballs*

C kolízie s použitím metaballs



FFD mriežky

- snaha o kompromis medzi spojitosťou a efektívnosťou výpočtu interpolácie

  => kvadratické B-spline krivky (v 3D je teda bod lineárnou kombináciou 27 najbližších uzlov mriežky)
- dobré výsledky aj s lineárnou interpoláciou

  (3x rýchlejšie, akceptovateľný vzhľad ak nie sú príliš veľké deformácie)
- každý vlas predpočítaný vektor váh všetkých uzlov mriežky, pri renderingu interpolácia len ako vážená suma polôh uzlov mriežky

Výsledky

- Implementácia v C++, floating point matematika
- zostava: Pentium4 3GHz, nVIDIA Quadro4 980 XGL, 512MB RAM, Windows2000
- Čas potrebný na mechanický výpočet 1 frame

Mech.Mod --- Lat.Size	0 Att. 0 Spr. 0 Meta.	0 Att. 0 Spr. 7 Meta.	50 Att. 200 Spr. 7 Meta.	100 Att. 400 Spr. 7 Meta.	200 Att. 800 Spr. 7 Meta.
10x10x10	1.1 ms	5.2 ms	10.1 ms	13.8 ms	21.4 ms
5x5x5	0.1 ms	0.2 ms	2.1 ms	3.8 ms	7.6 ms

+ 2,5ms B-spline FFD; 1,0ms linear FFD (10000 uzlov)

http://www.miralab.unige.ch/

A to je koniec...

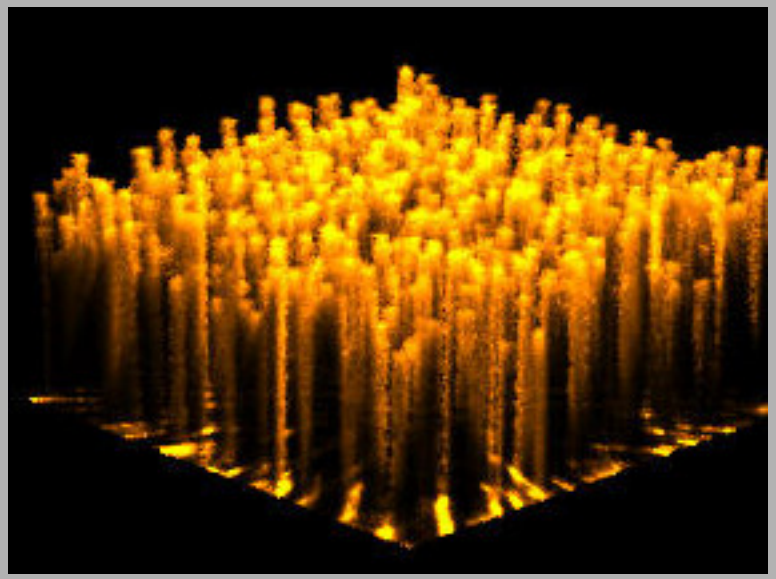
Ďakujem za pozornosť

Renderovanie realistickej srsti živočíchov

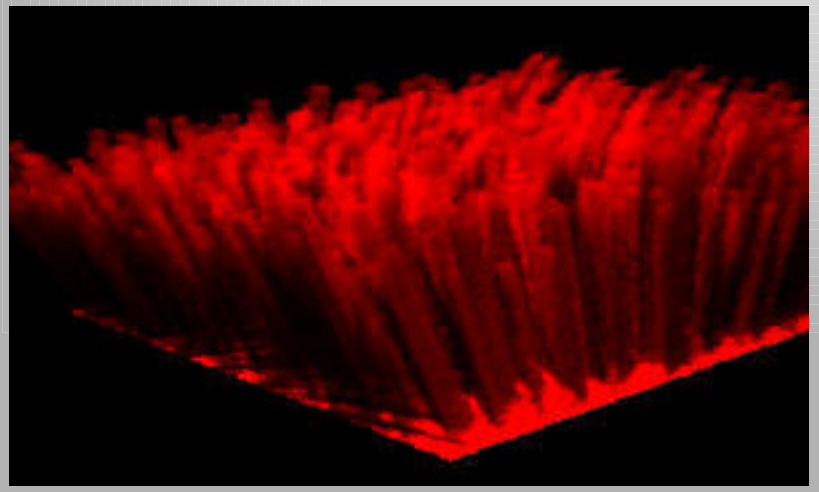
- *dôležitý presvedčivý vzhľad*
- *Tradičný geometrický (polygonálny) prístup príliš náročný*

- Kajiya a Kay: 3D textúry (texels) nad polygonálnym povrchom
  - 3D pole hustoty, dotyčníc, farby
  - MonteCarlo rozptyl, prepúšťanie svetla cez texel (zvyšuje realistickosť)
  - niekoľko ukážok:

Ukážky texelov nad jedným polygónom

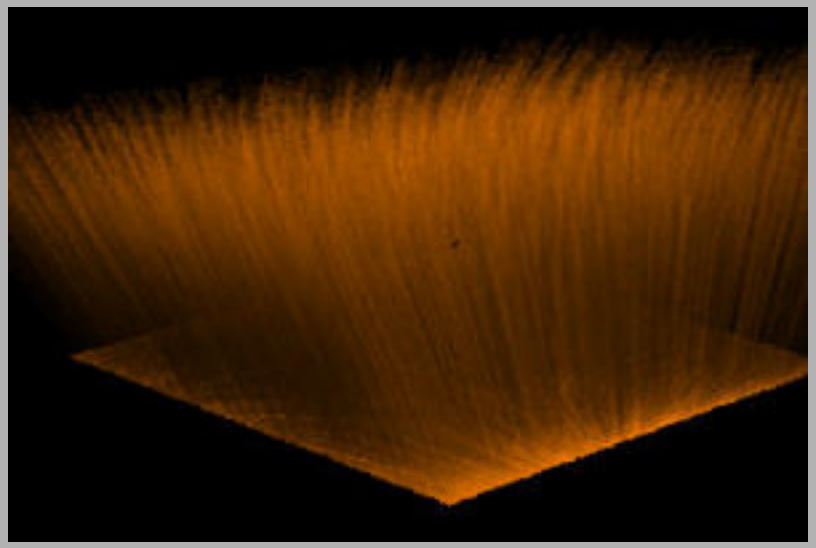


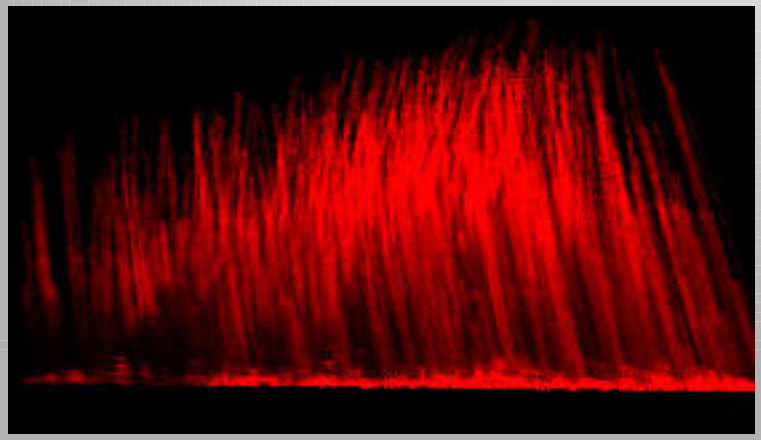
– smer chlpov podľa normál riadiacich vrcholov
– s vypočítanými tieňmi



– perturbácia normál zvyšuje presvedčivosť

Ukážky texelov nad jedným polygónom

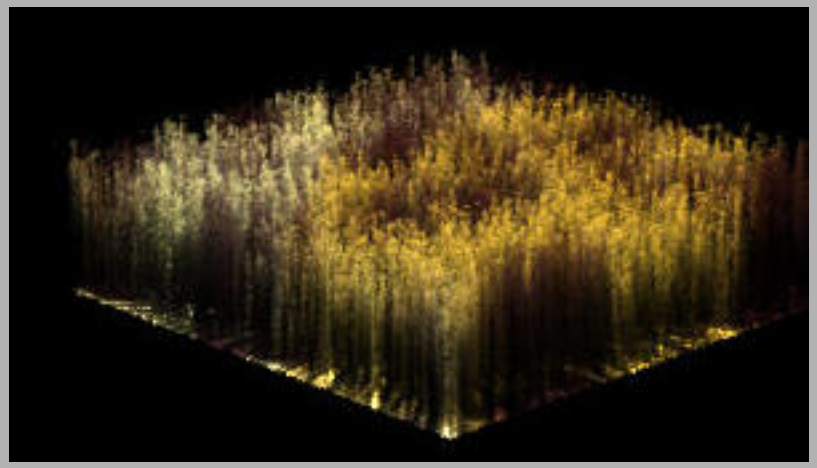




– zmena hustoty, optickej hĺbky, zvýšenie hustoty rastru

– prekrytie dvoch texelov (umožňuje simulovať srsť variabilných dĺžok a hustôt)

Ukážky texelov nad jedným polygónom



– pridanie 2D textúry, pre zmenu farby

– ukážka výsledného efektu

http://www.stanford.edu/~turitzin/cs348b/results/