EECS 487

Non-photorealistic Rendering

Lee Markosian

December 11, 2006

Whether to use photorealism depends on the purpose of the image:

- Documentation
- Illustration
- Story-telling
- Expression

Whether to use photorealism depends on the purpose of the image:

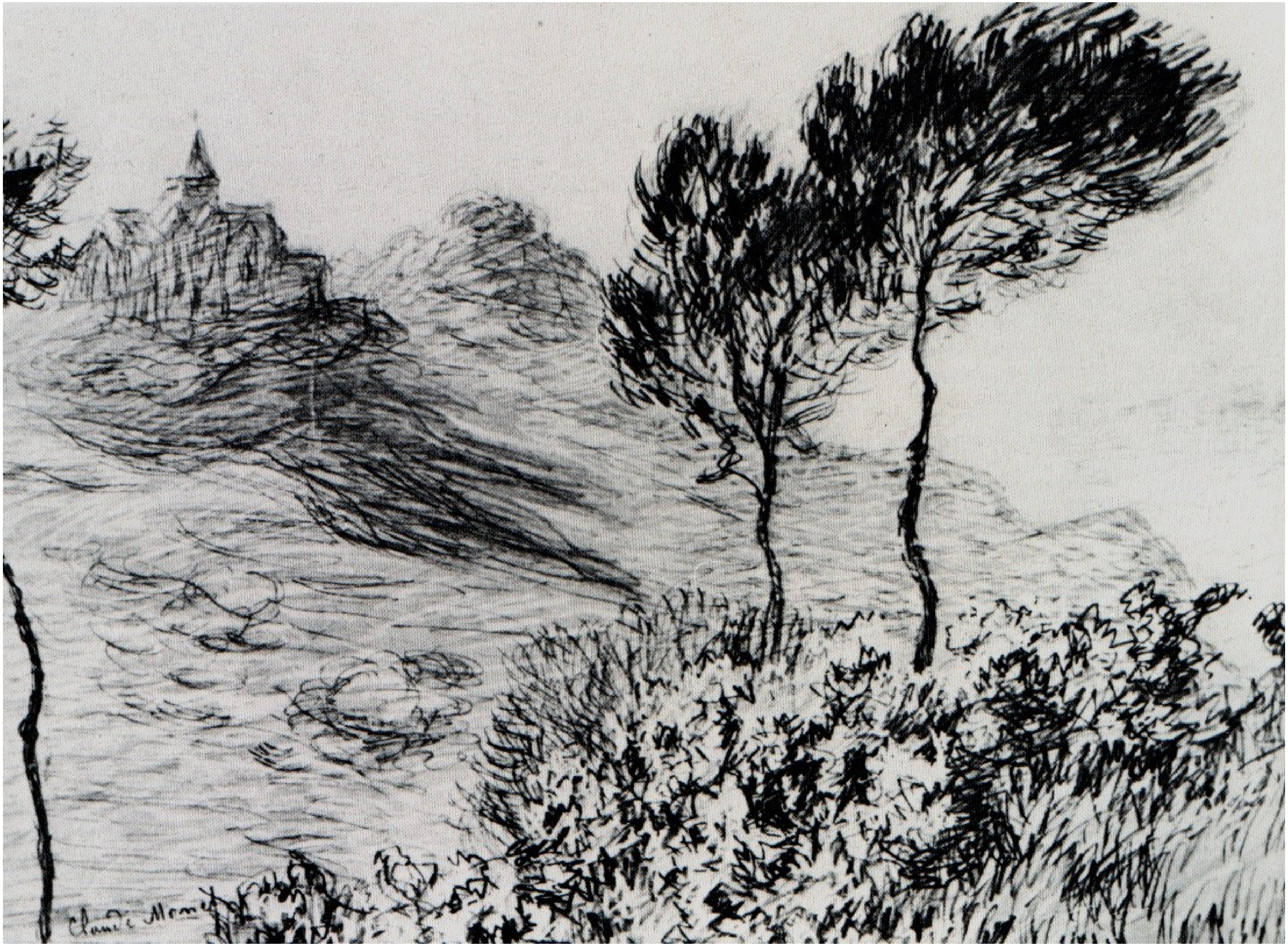
- Training/simulation (yes)
- Documentation (yes)
- Illustration (not usually)
- Story-telling (sometimes)
- Expression (sometimes)

Qualities of hand-drawn images

- Many details left out
- Some details emphasized
- Stylization / abstraction
  - used to evoke complex things
- Recognizable individual style


Dr. Seuss

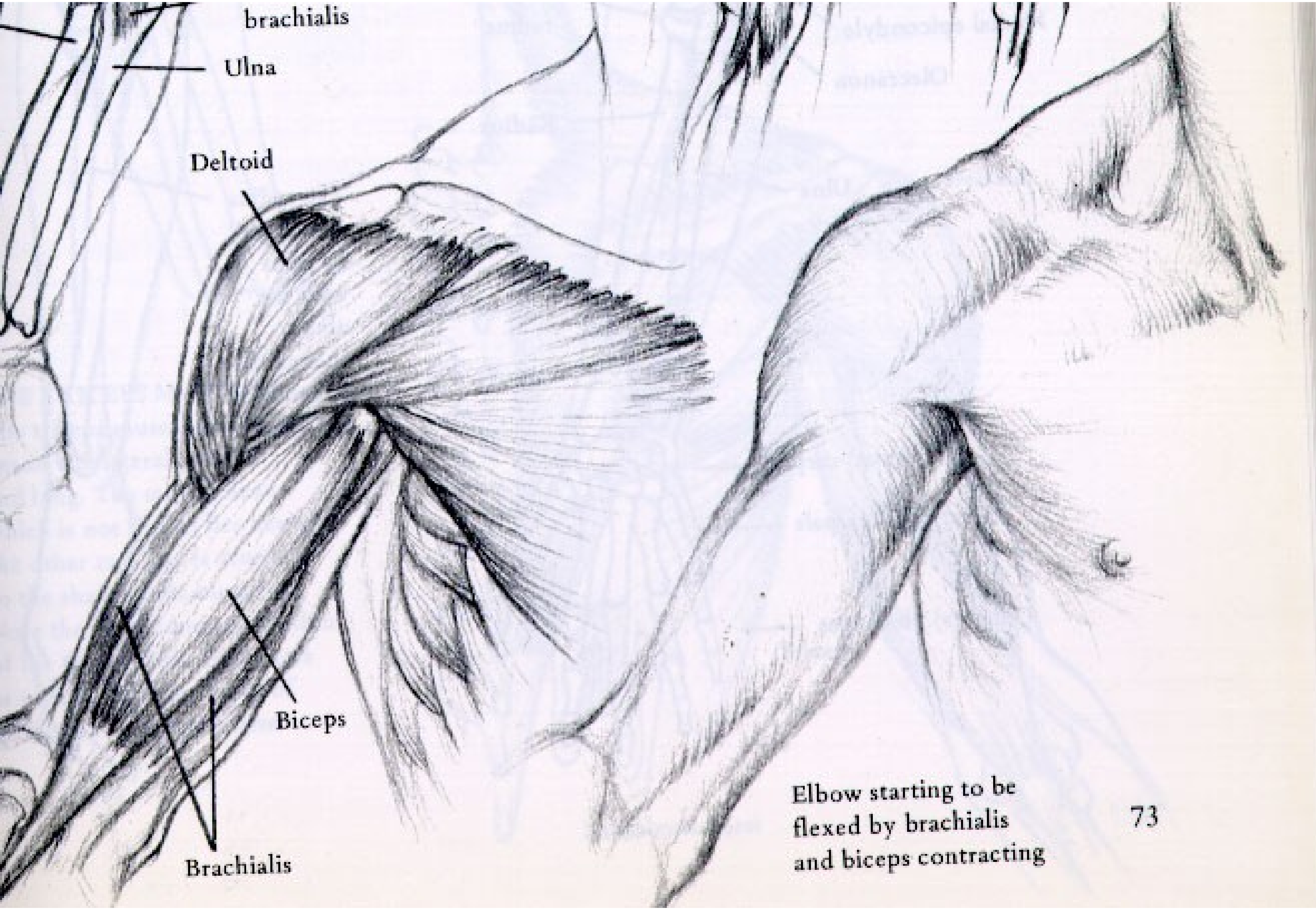
Realistic modeling and rendering of plant ecosystems, SIGGRAPH 1998



Monet

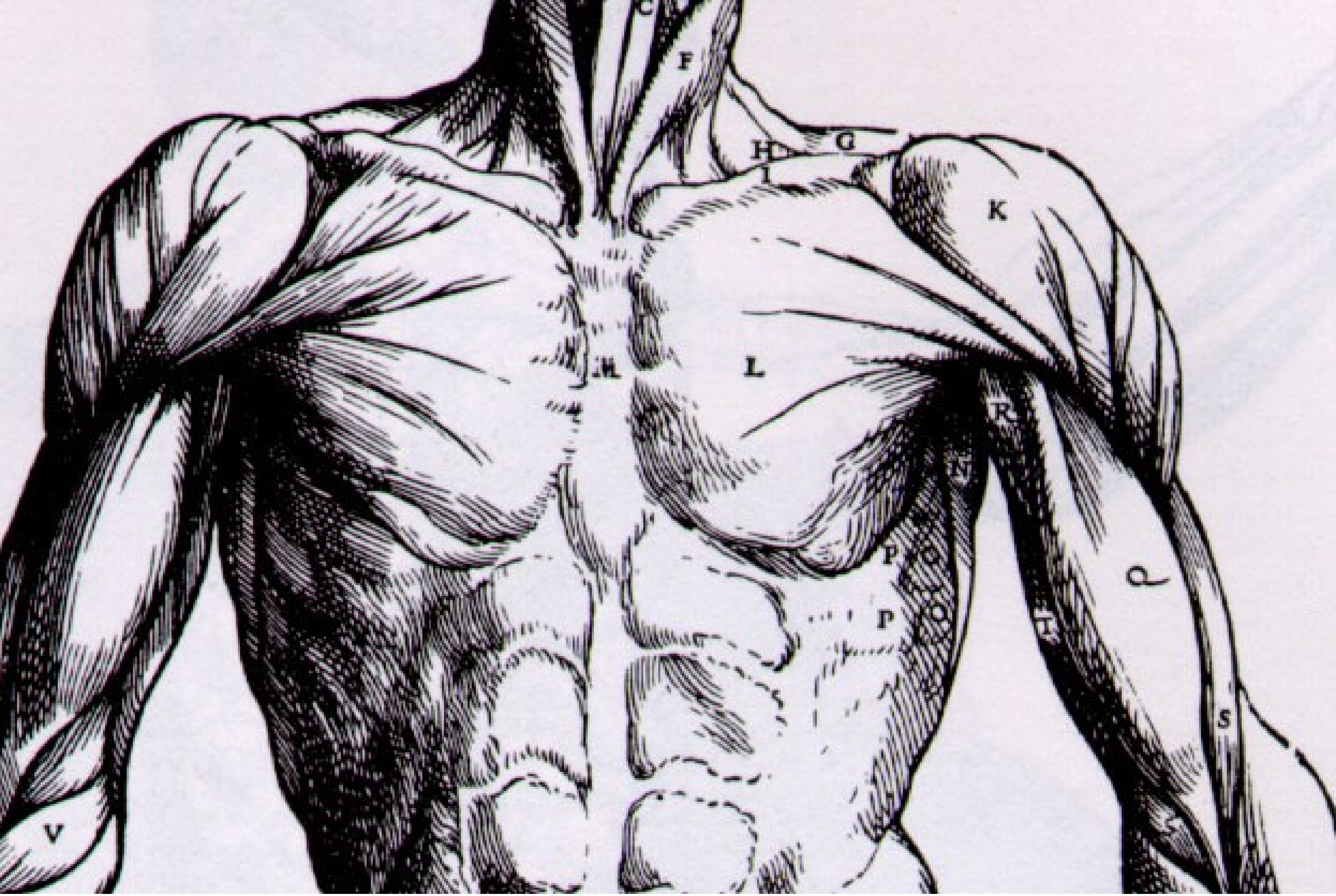
Uri Shulevitz

Charles Burns

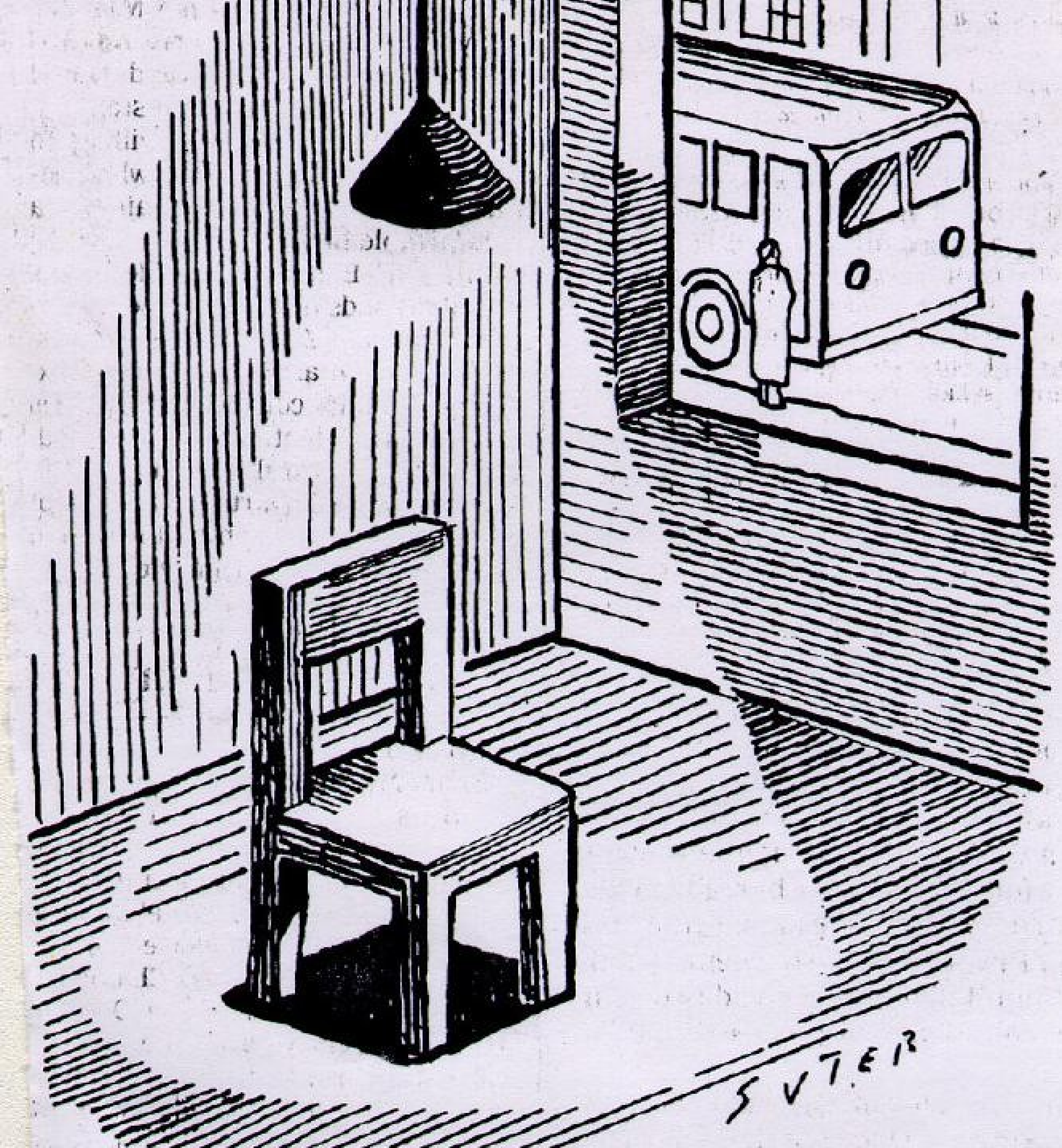


Elbow starting to be flexed by brachialis and biceps contracting

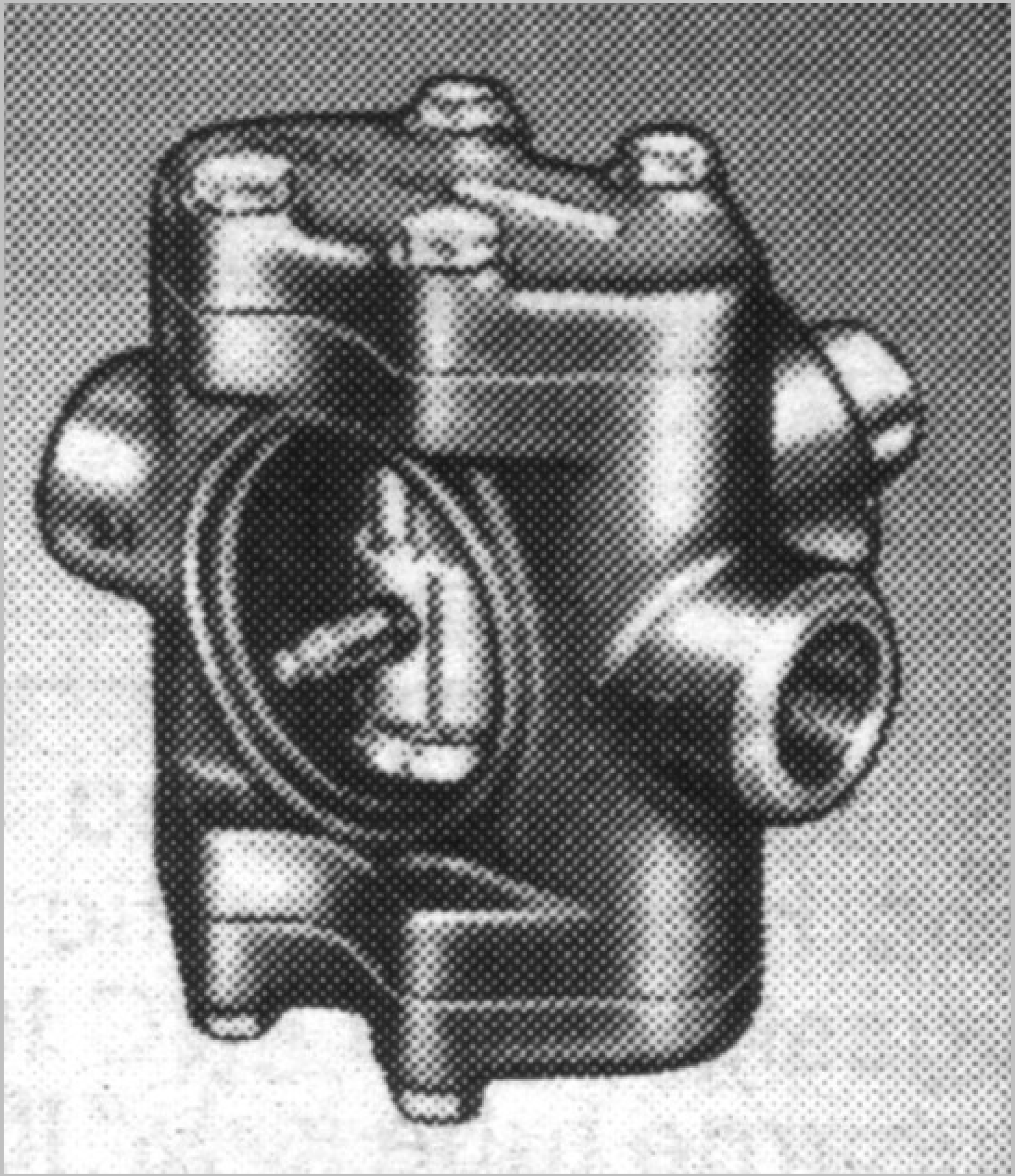
Louise Gordon



Vesalius (1514 - 1564)



David Suter

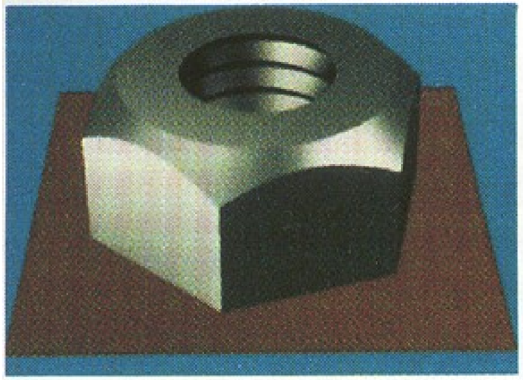


talk overview

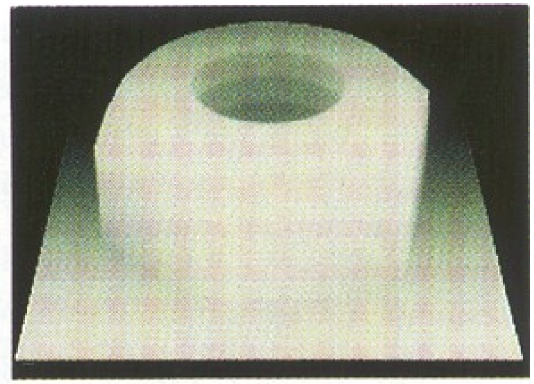
- motivation
- technical illustration
- pen & ink rendering
- painterly rendering
- graftals
- stroke-based rendering
- tonal art maps

Technical illustration

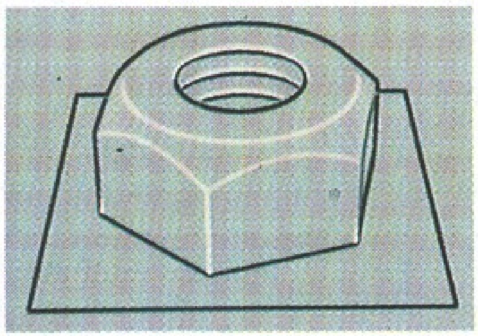
- Saito and Takahashi, Siggraph 90
- Purpose: render 3D models in styles that are more “comprehensible”
- Method:
  - Render various intermediate images
  - Do image-processing operations on them
  - Combine the results



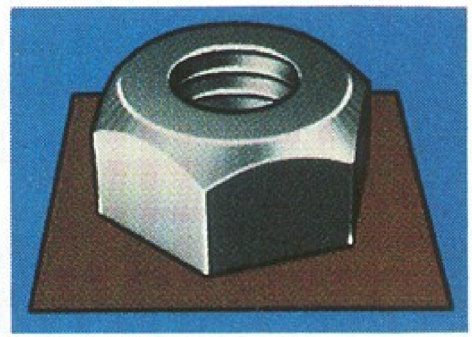
(a) shaded image



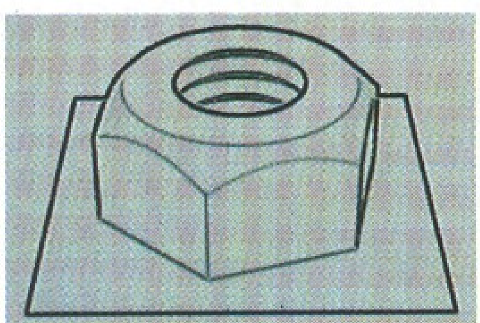
(b) depth image



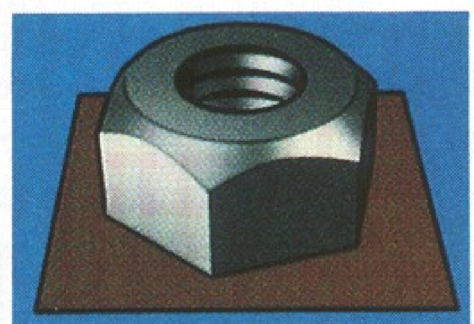
(c) edge image (1)



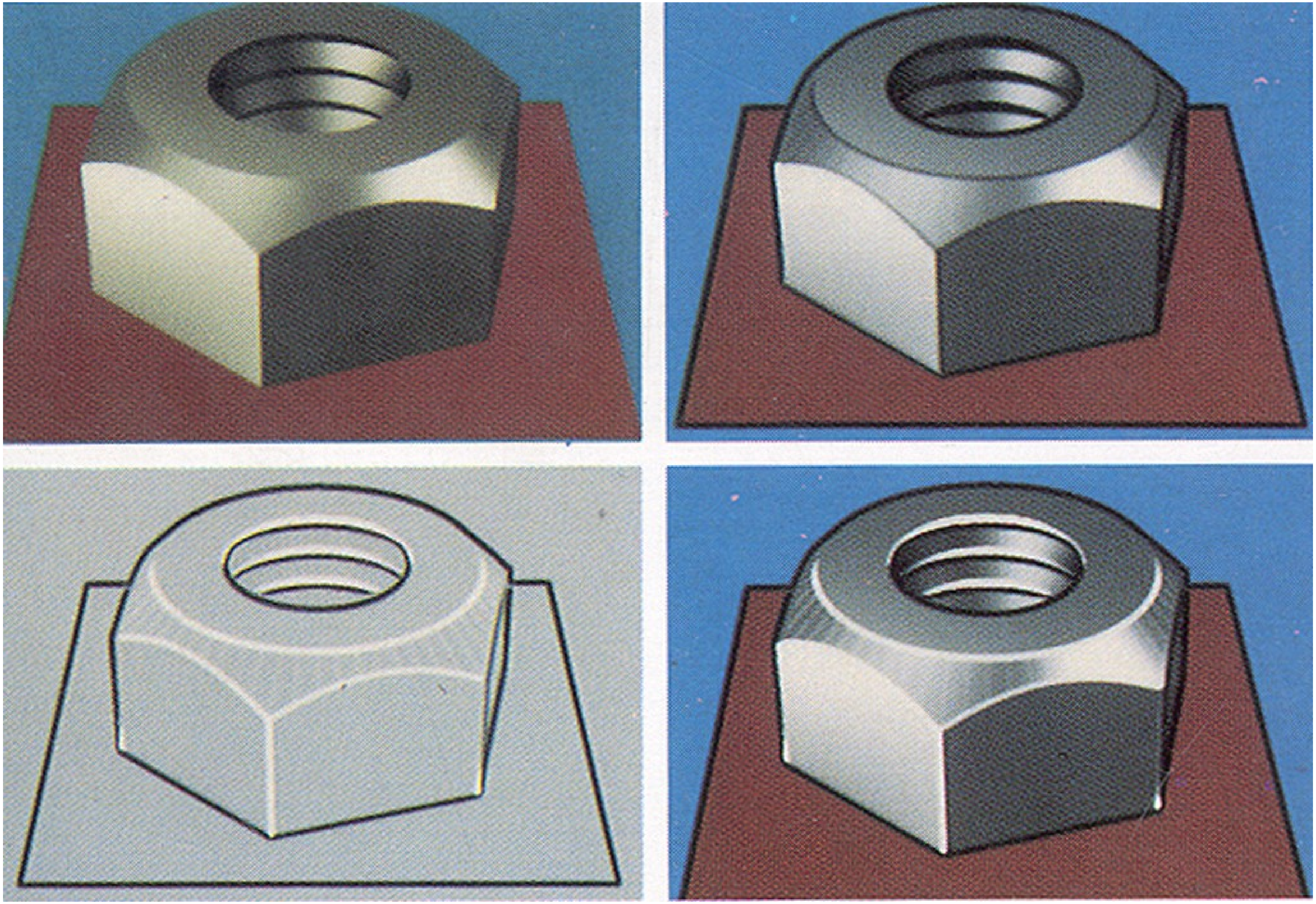
(d) enhanced image (1)



(c') edge image (2)



(d') enhanced image (2)



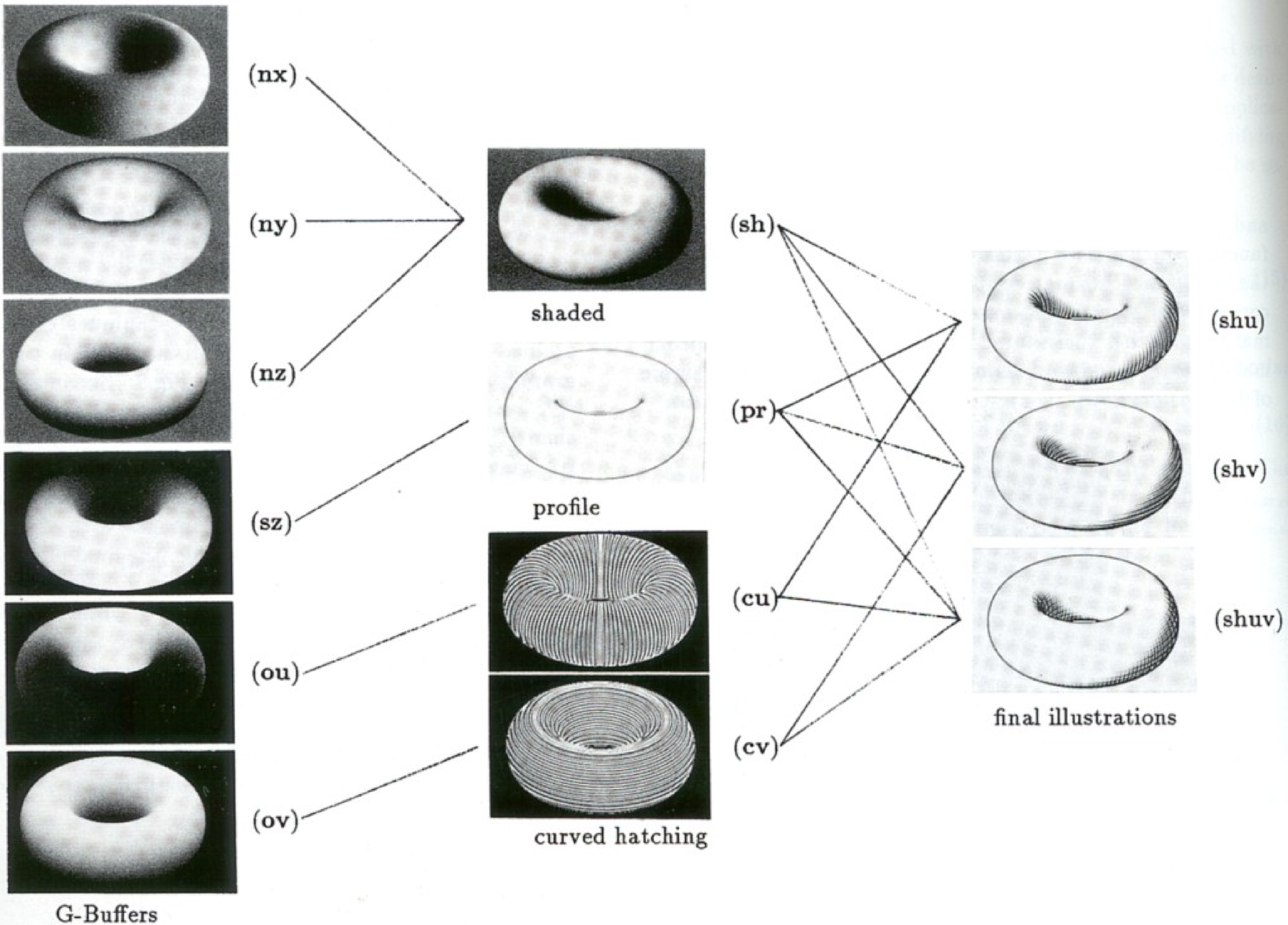
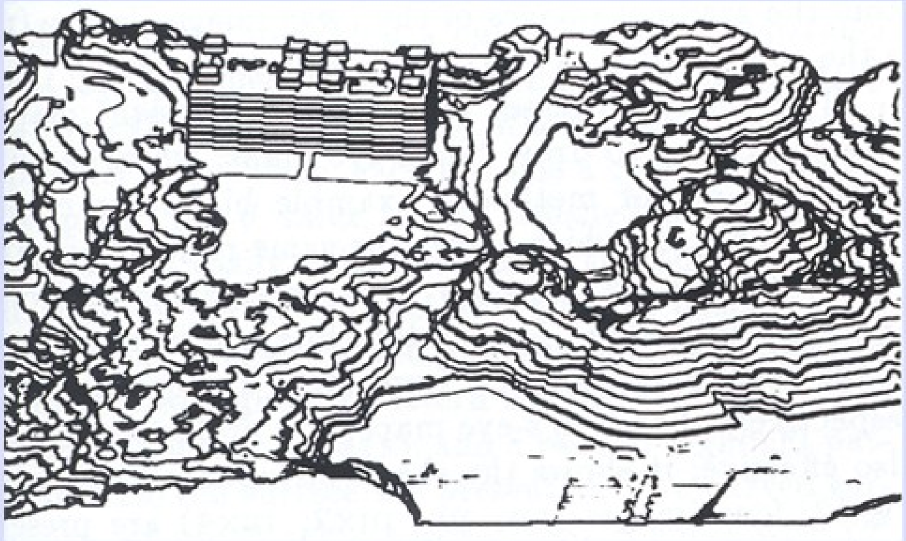
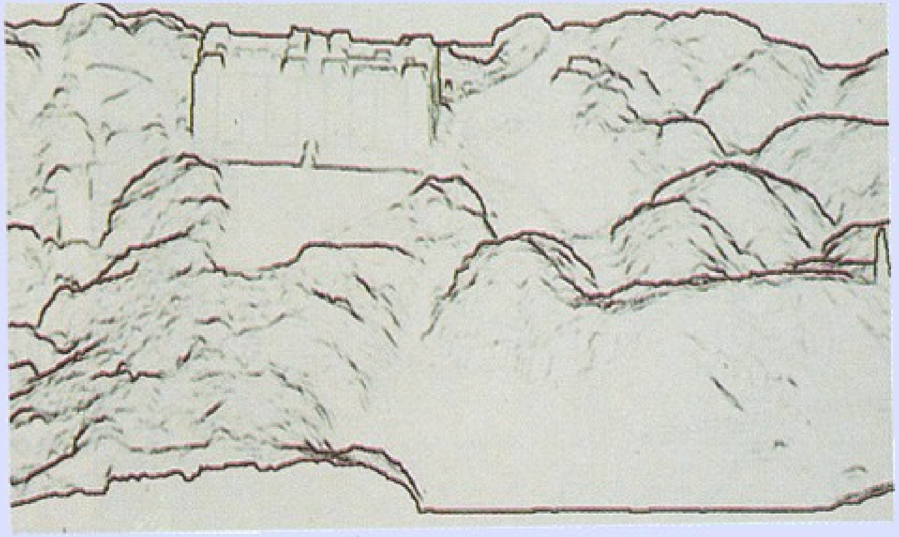


Fig.9 Process of drawing illustrations.

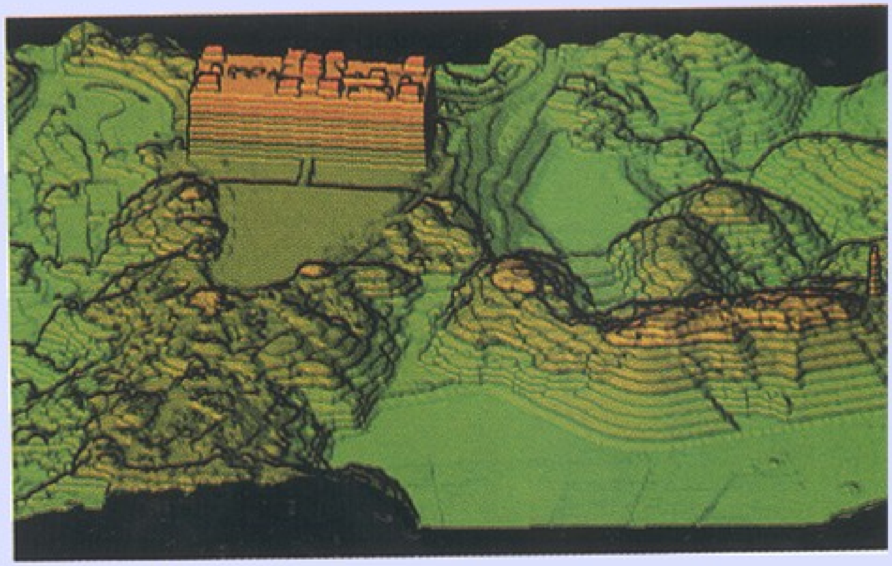


(cn) contour image



(pr) profile image

(mx2) combination of (pr) and (sh)



(mx4) combination of four enhanced images

Problem

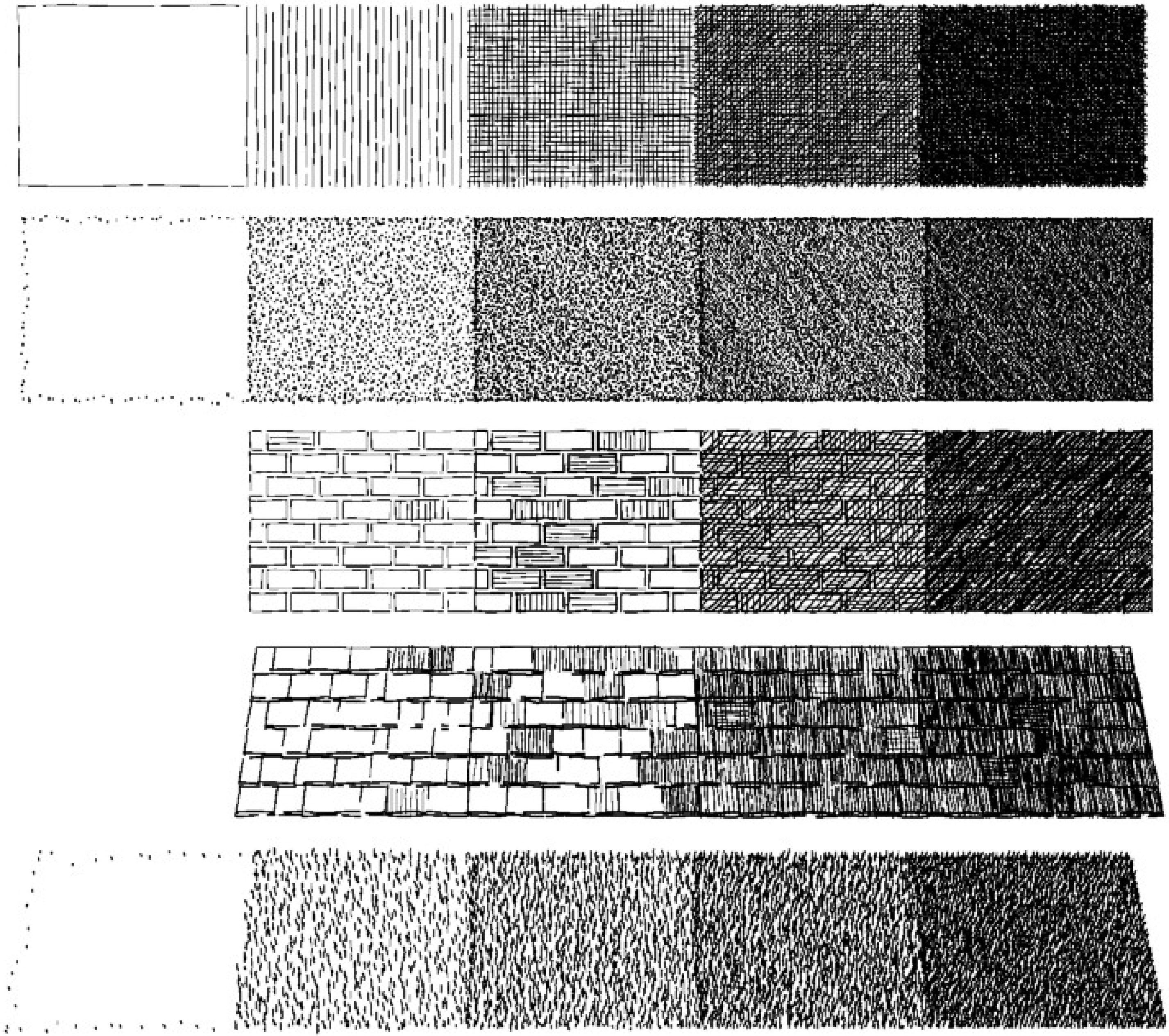
- Parameters need careful tuning for good results
- Can be a problem in animations if frequent parameter tuning is needed!

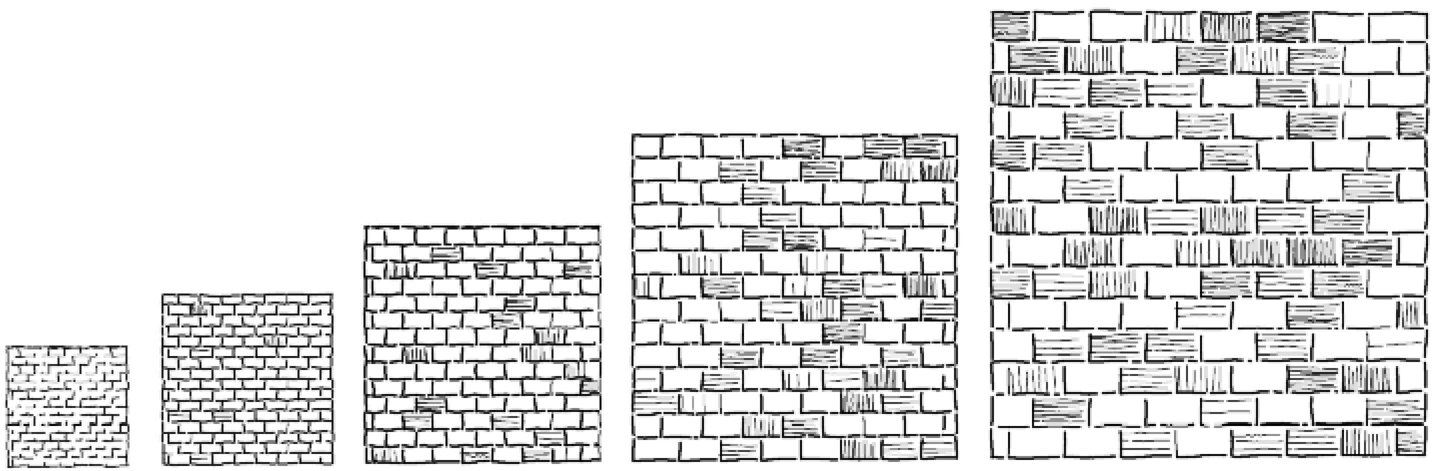
talk overview

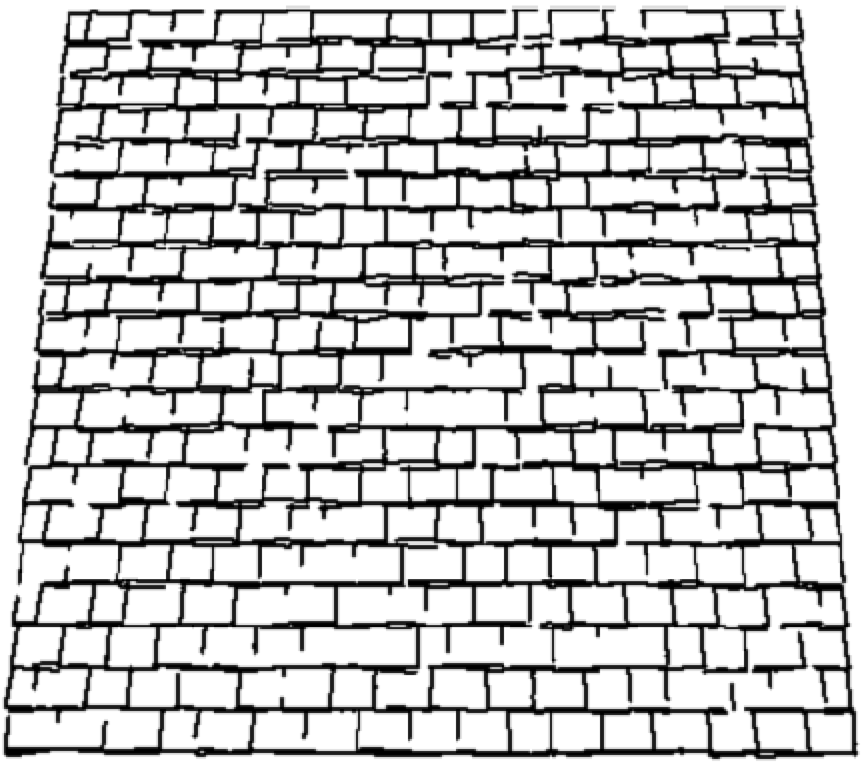
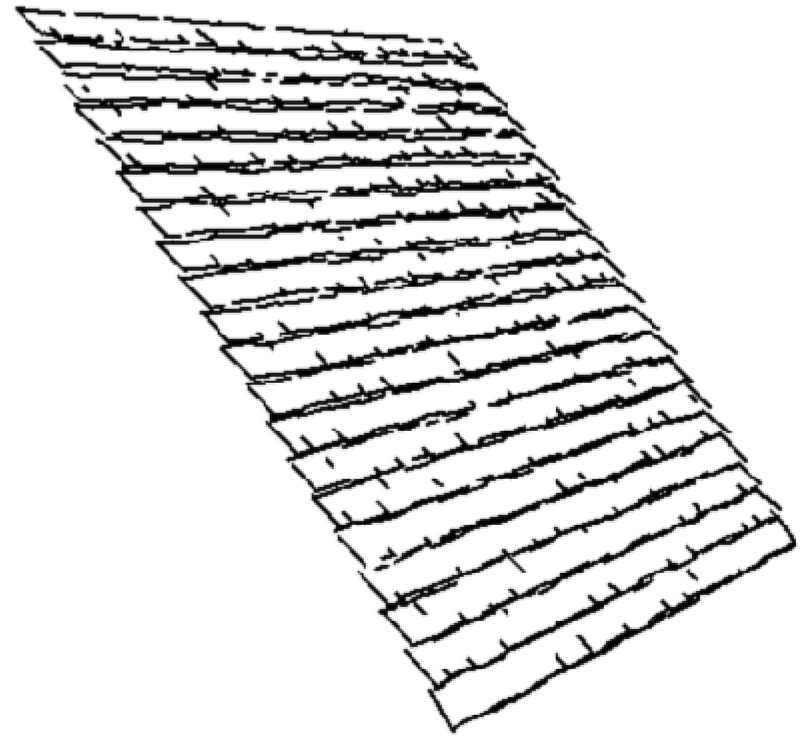
- motivation
- technical illustration
- pen & ink rendering
- painterly rendering
- graftals
- stroke-based rendering
- tonal art maps

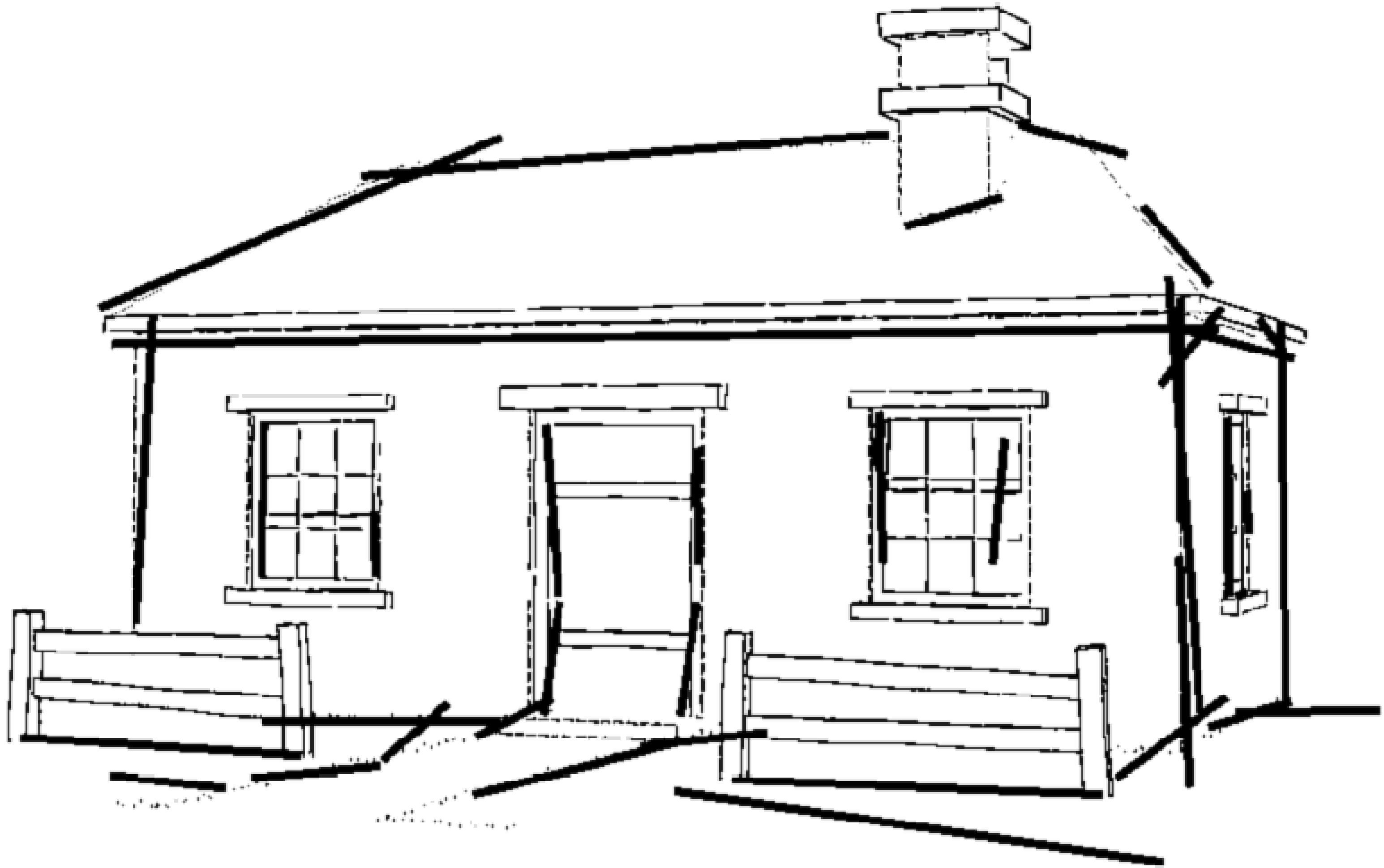
Pen and Ink

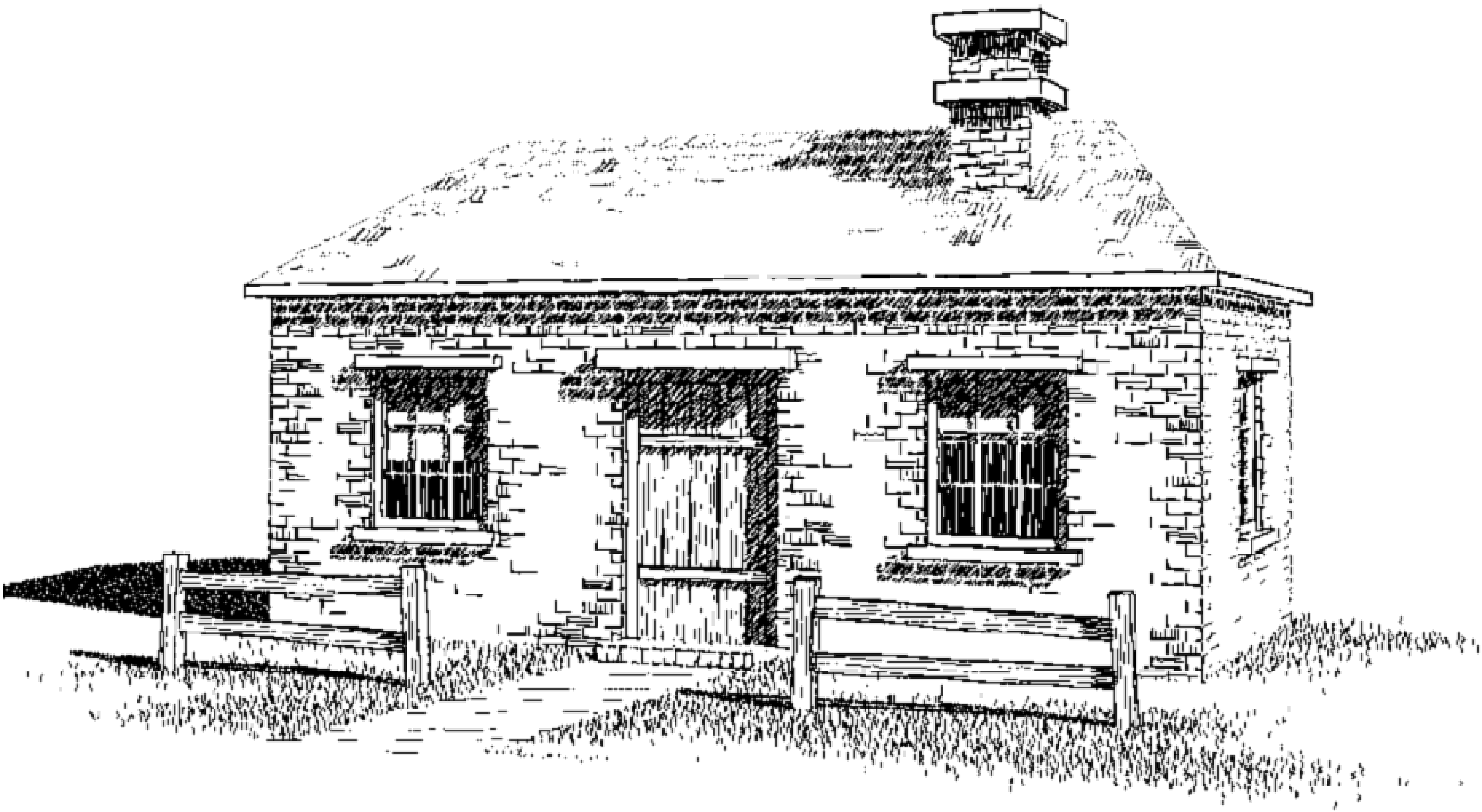
- Winkenbach and Salesin, Siggraph 94
- Purpose: render 3D models as pen & ink drawings
- Method:
  - annotate model with procedural “textures”
  - Render tonal “reference image”
  - Use it to guide pen and ink textures





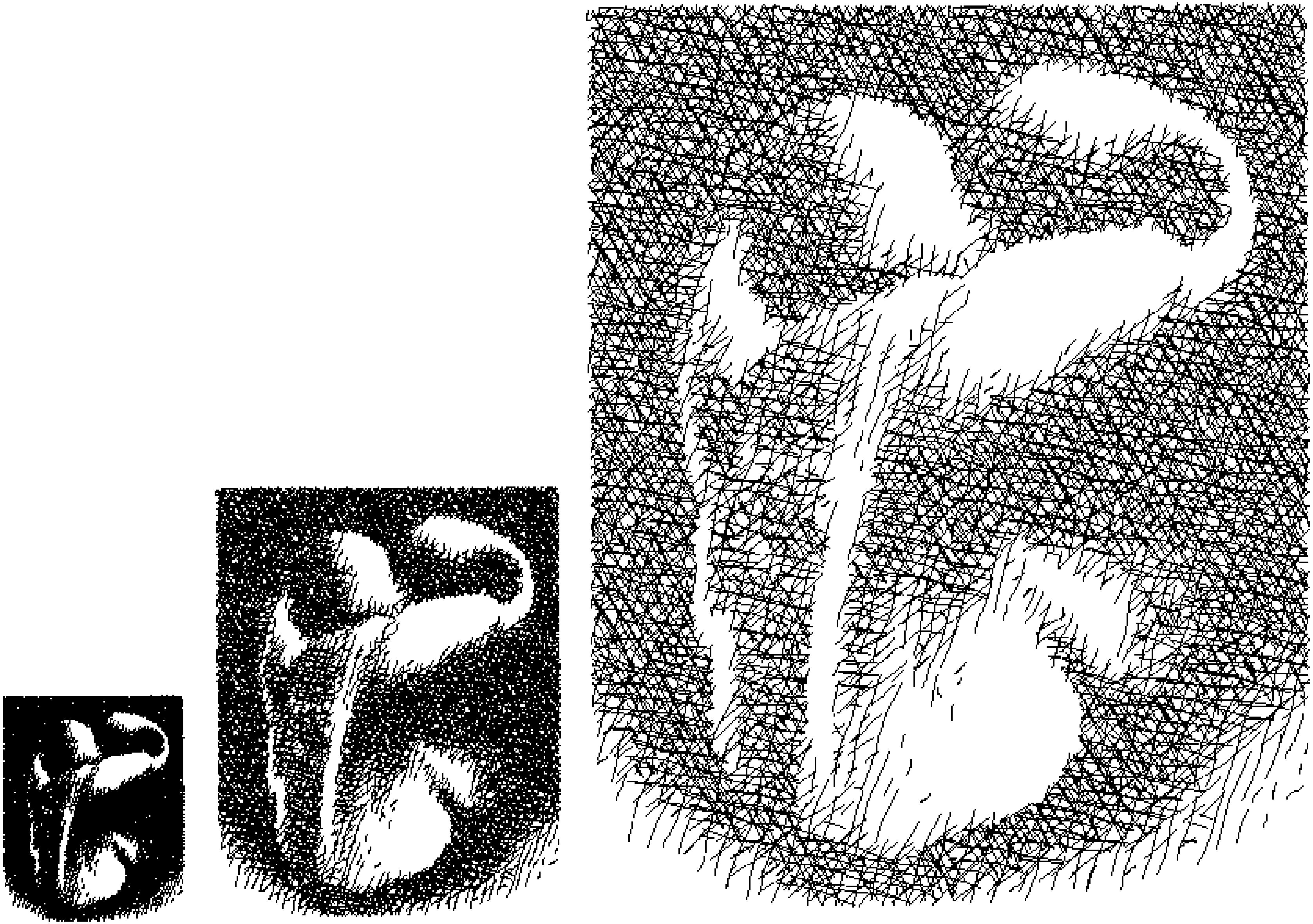




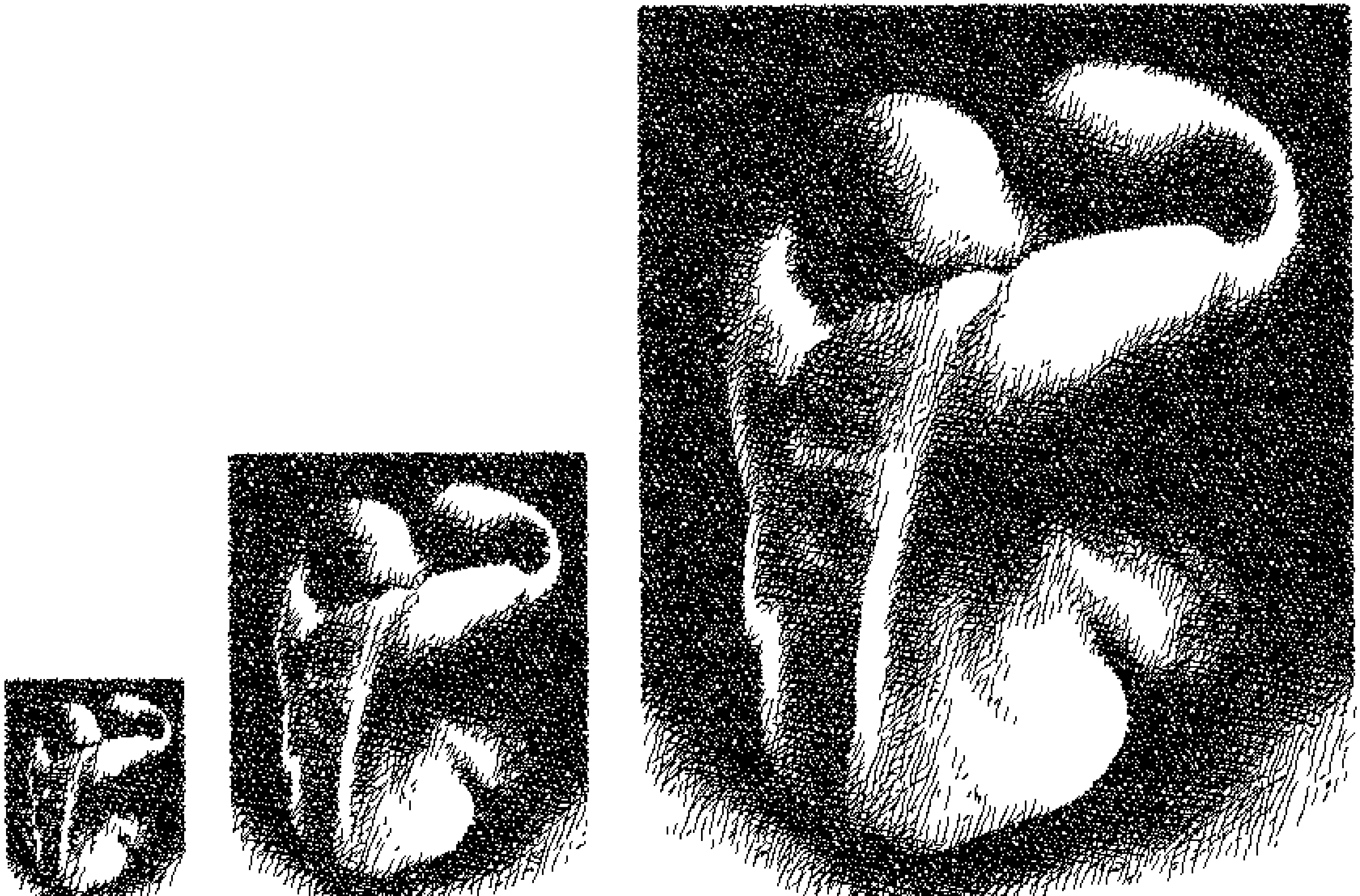
Pen and Ink

- Salisbury, Anderson, Lischinski and Salesin, Siggraph 96
- Purpose: define a scale-independent representation for pen & ink images

Scaling with fixed number of strokes: bad



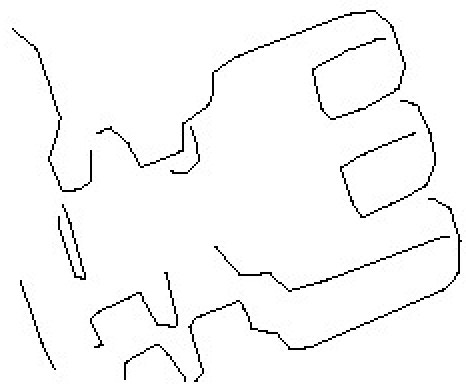
Scaling with variable number of strokes: good



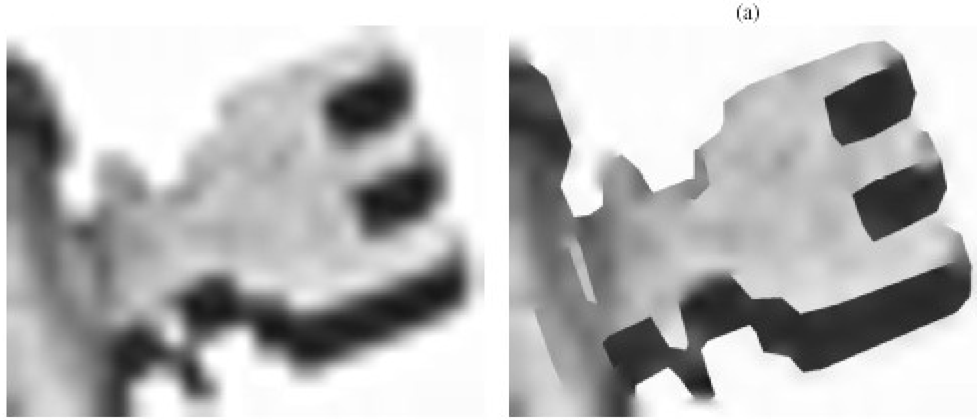
Salisbury *et al.*: method

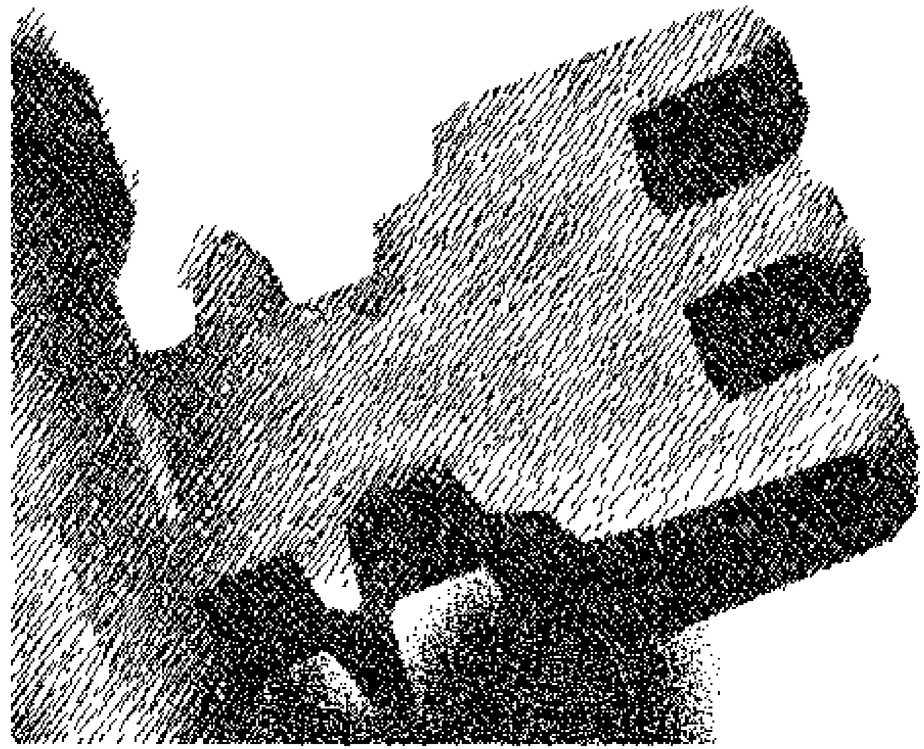
- store lo-res greyscale image annotated with discontinuities
- filter greyscale image to desired size, run stroke generation algorithm on it
- repeatedly try to generate a stroke at random location in image
- keep stroke if it does not exceed target darkness

Detect sharp features, then modify blurring algorithm to preserve them



(a)





Problems

- Only produces still images
  - Would not provide temporal coherence
- What's the application?

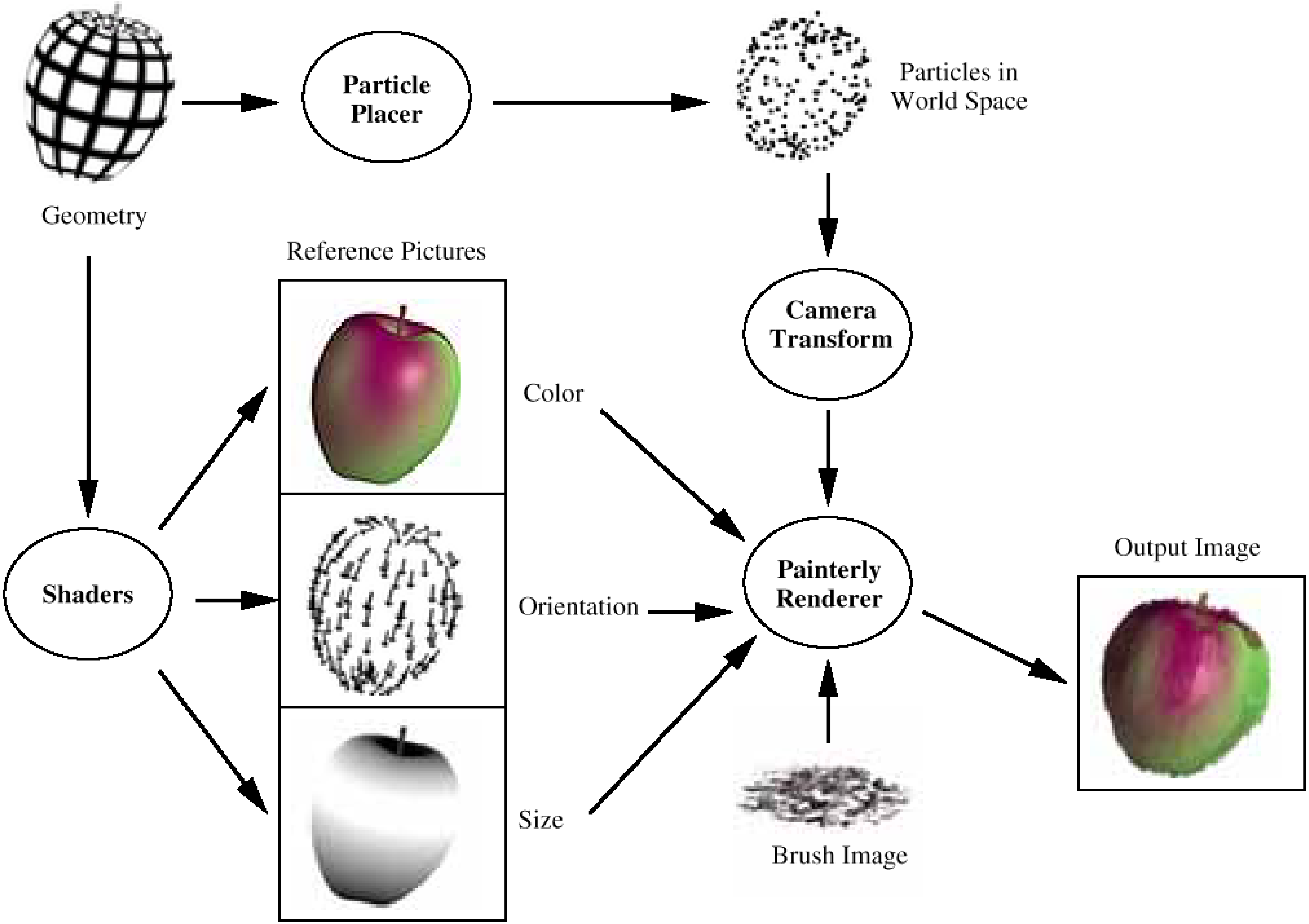
talk overview

- motivation
- technical illustration
- pen & ink rendering
- painterly rendering
- graftals
- stroke-based rendering
- tonal art maps

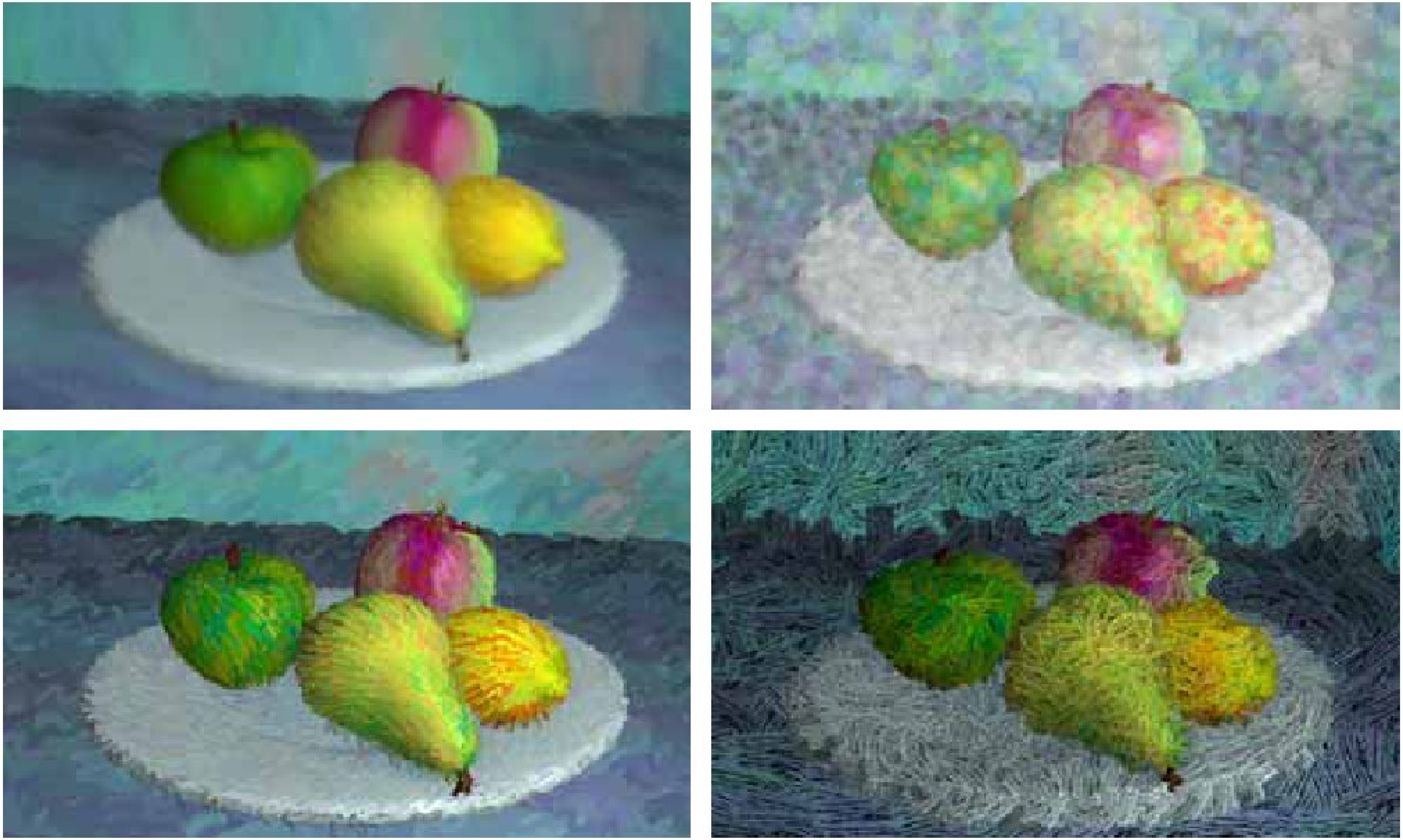
Painterly rendering: Meier 1996

Painterly rendering

- Meier, Siggraph 96
- Problem: produce animations in a "painterly" style with temporal coherence of strokes
- Method:
  - populate surfaces with stroke "particles"
  - render with the help of reference images
  - does not handle zooming!



Particle Placer
Particles in World Space
Geometry
Reference Pictures
Camera Transform
Color
Output Image
Painterly Renderer
Shaders
Orientation
Size
Brush Image



video

Problem

- Particles have fixed distribution
  - Need prescribed camera path

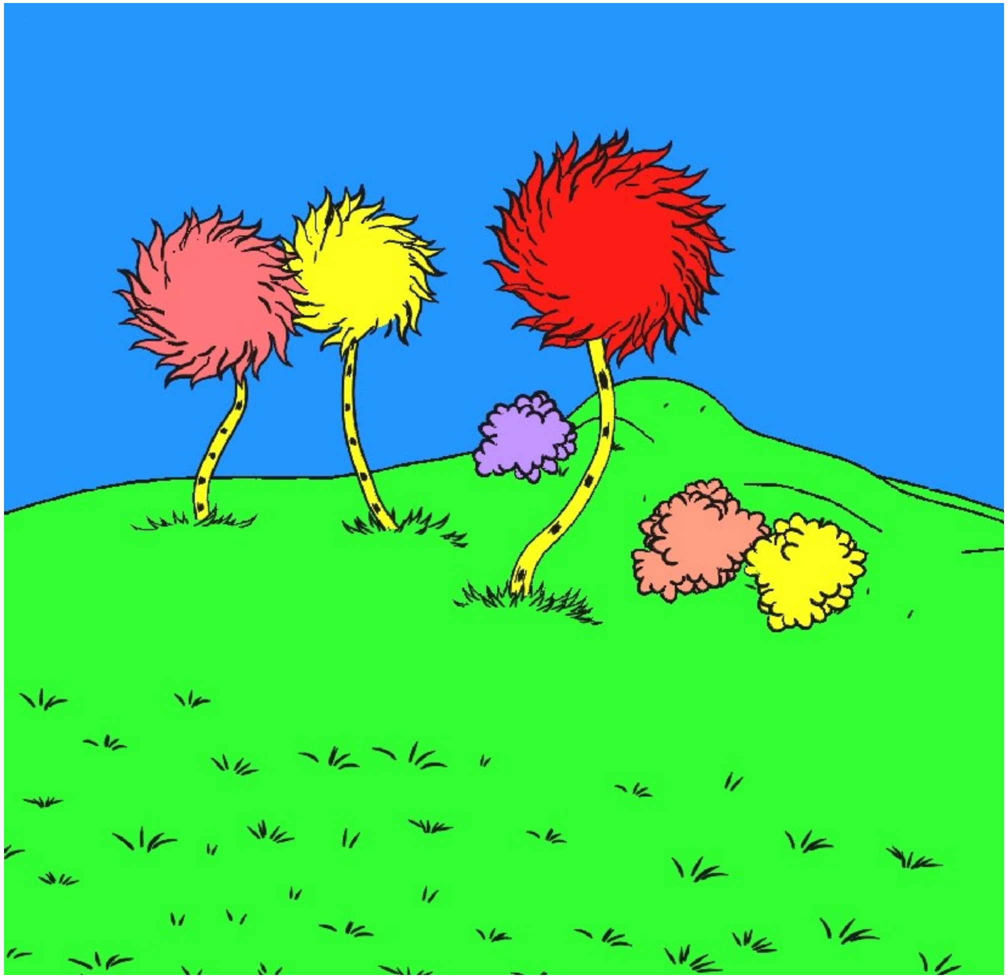
- newer work addresses that:

  A dynamic drawing algorithm for interactive painterly rendering:

  http://artis.imag.fr/Publications/2006/VBTS06/

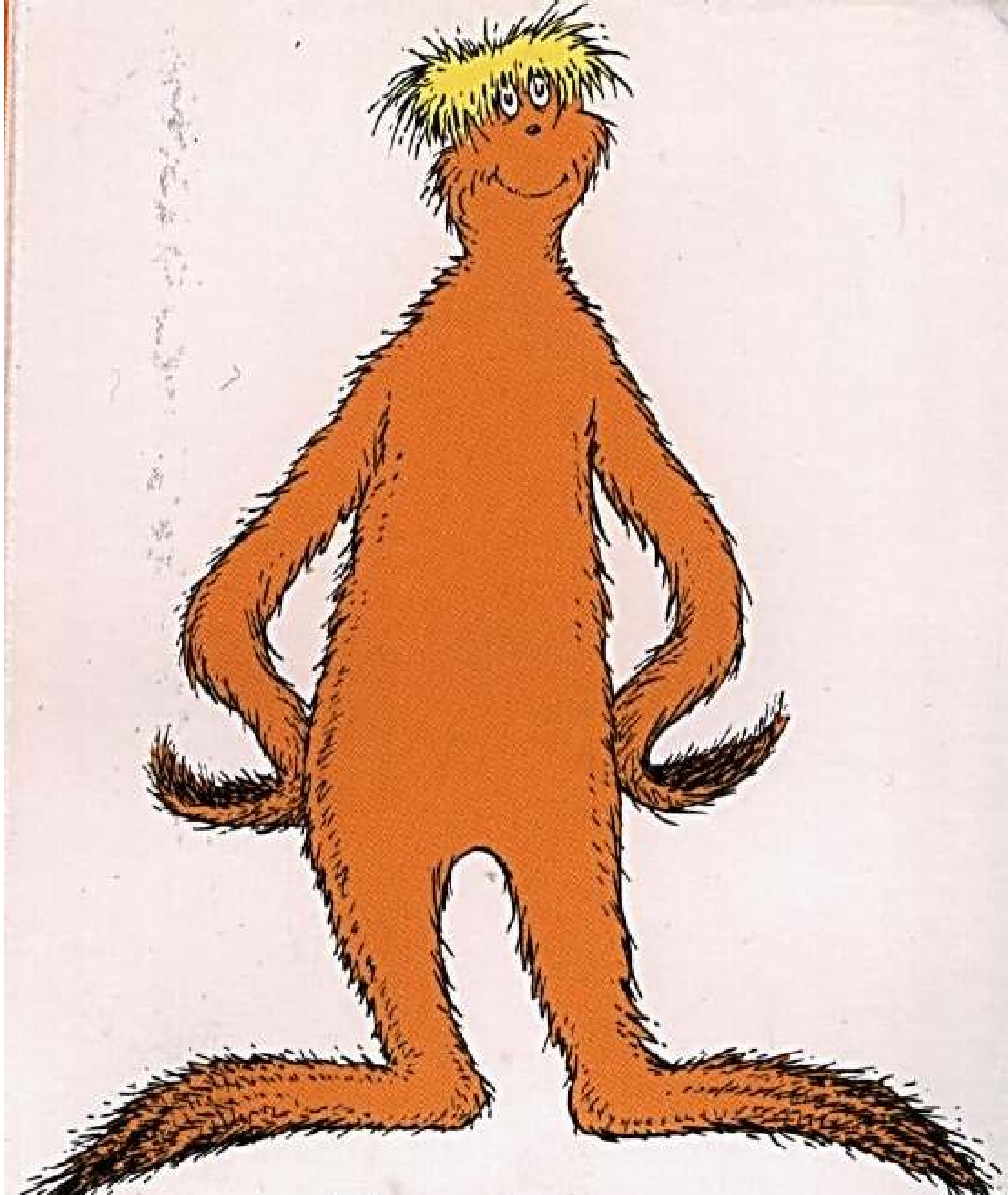
talk overview

- motivation
- technical illustration
- pen & ink rendering
- painterly rendering
- graftals
- stroke-based rendering
- tonal art maps



- <u>Art-based Rendering of Fur, Grass and Trees</u>. Kowalski, Markosian, Northrup, Bourdev, Barzel, Holden & Hughes. SIGGRAPH 1999.

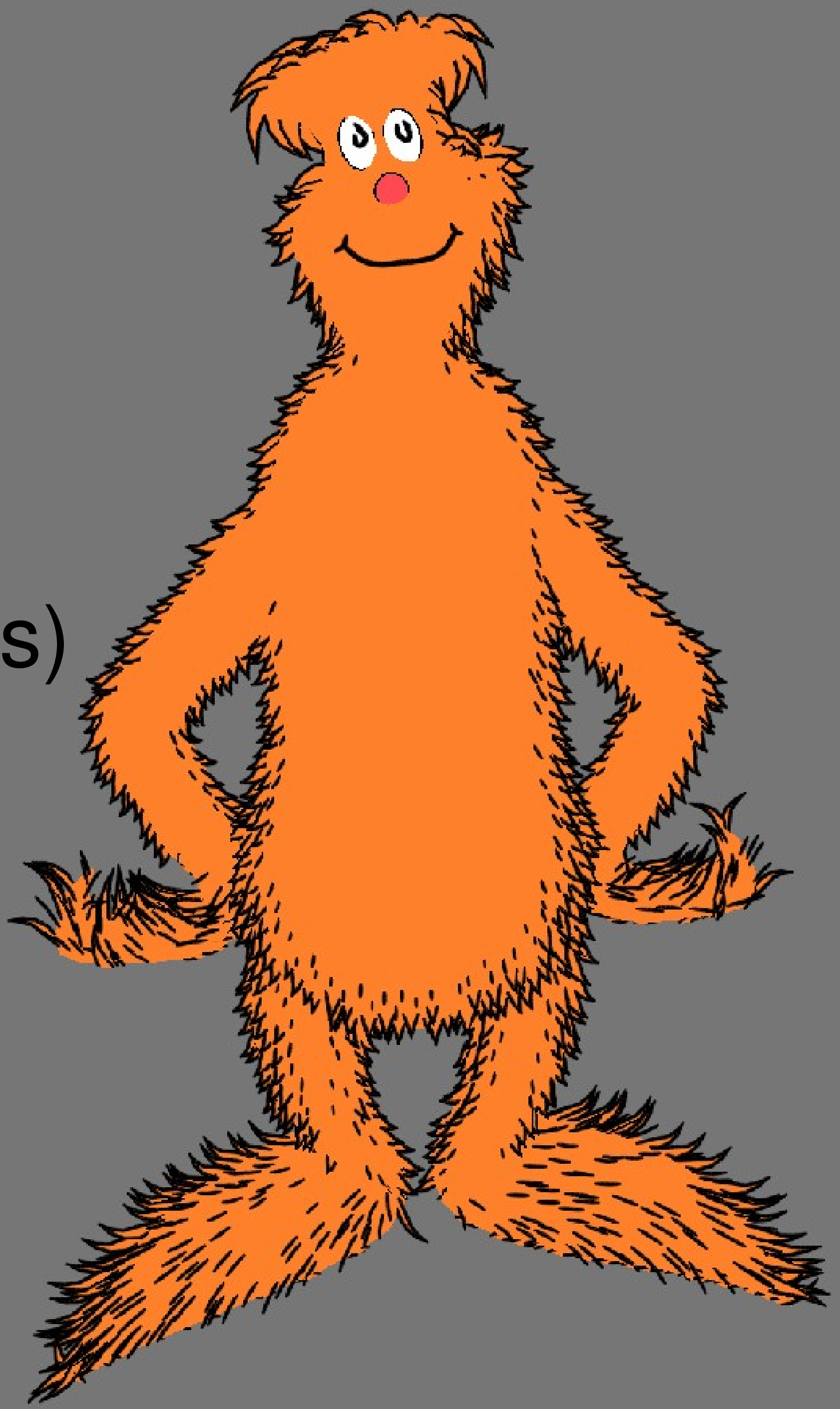
Dr. Seuss



Fuzzy fur feet

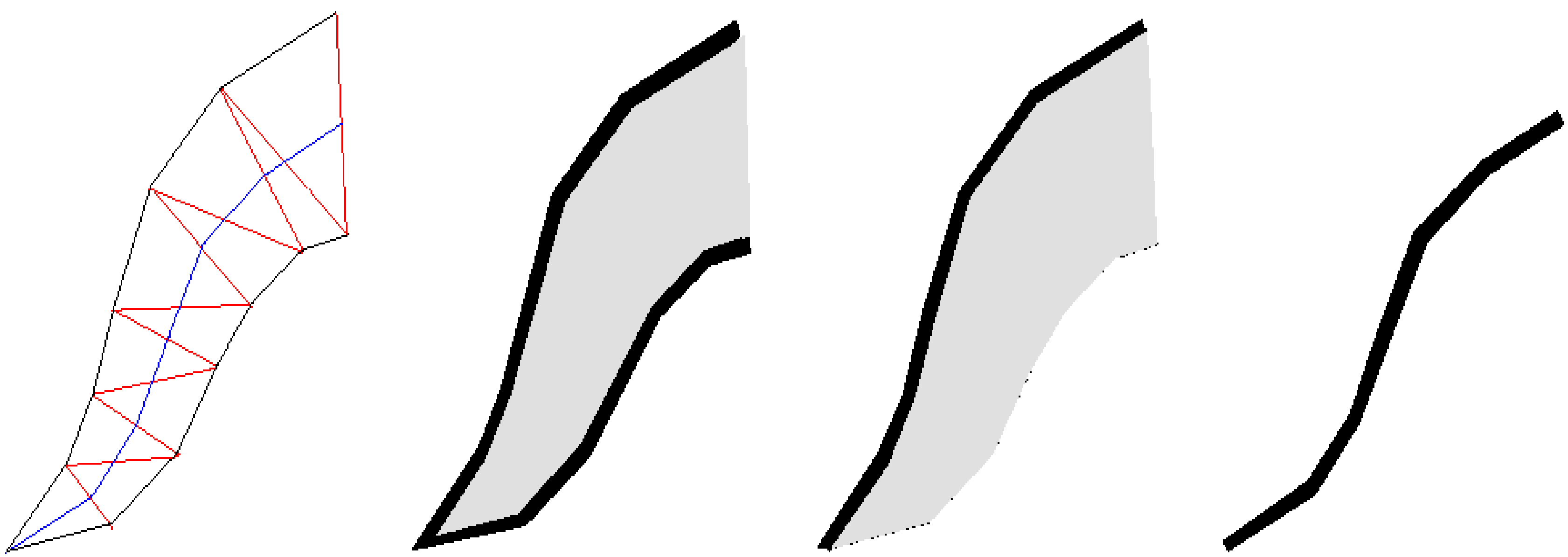
Graftal textures

Detail elements (graftals) generated as needed



Graftals

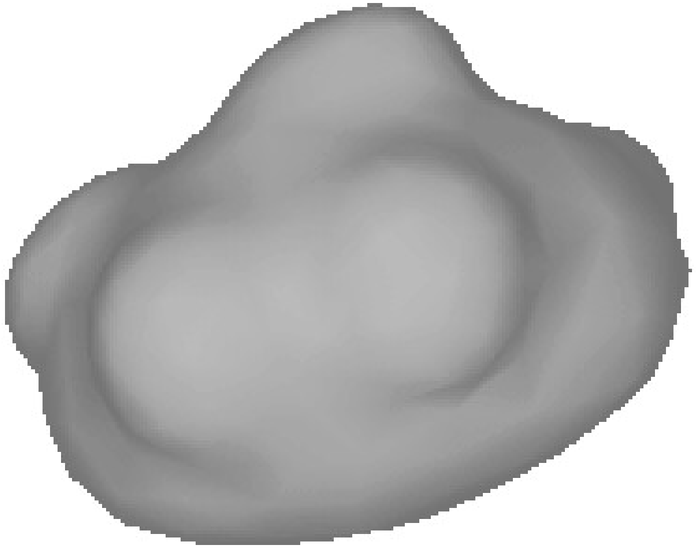
- simple bit of geometry (e.g. leaf or "tuft")
- oriented in local frame

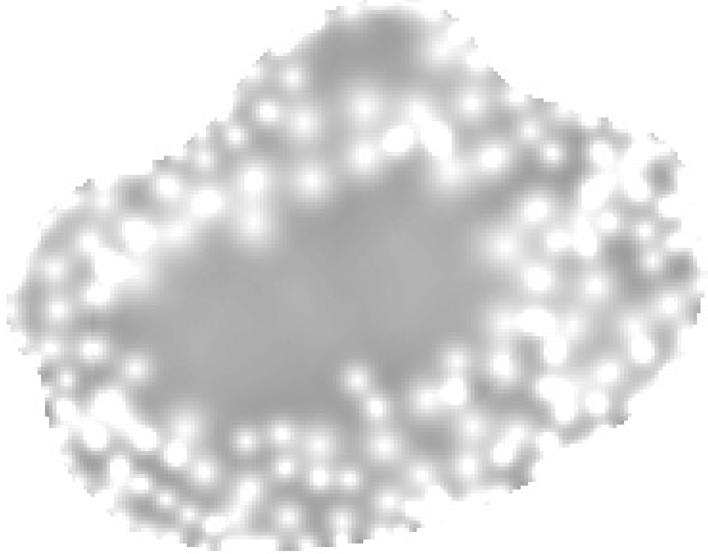


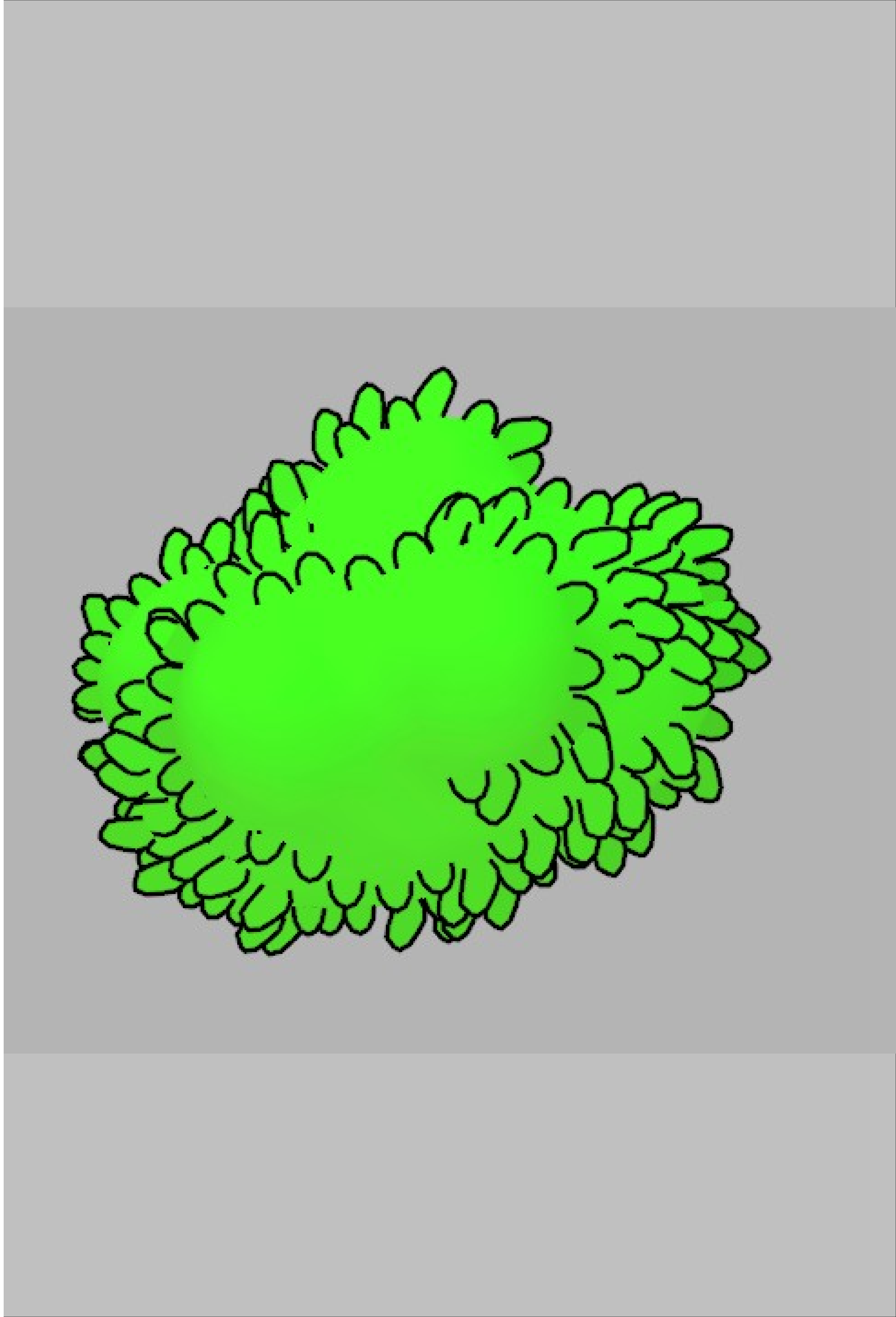
(a) (b) (c) (d)

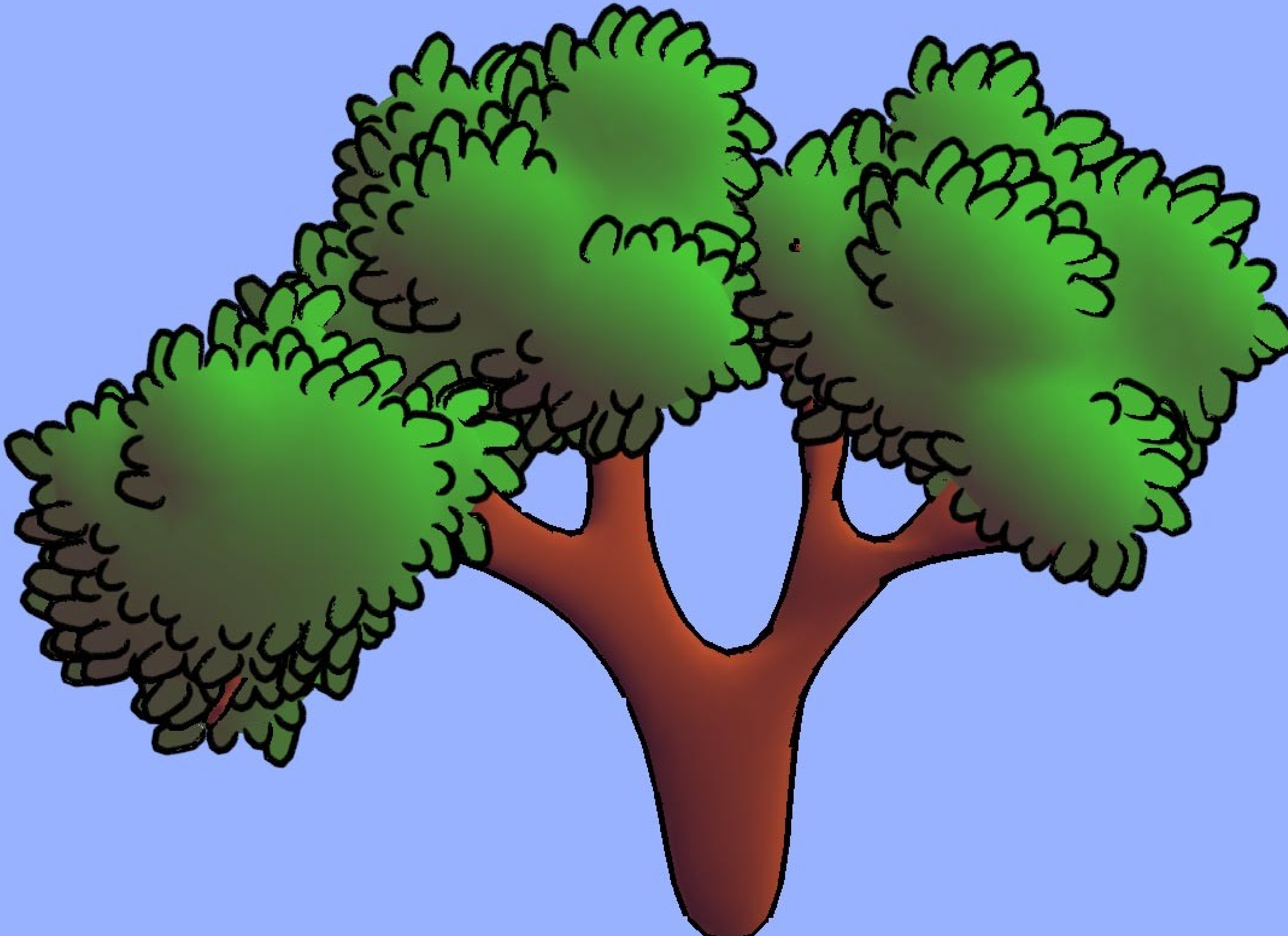
Needed for placement of graftals:

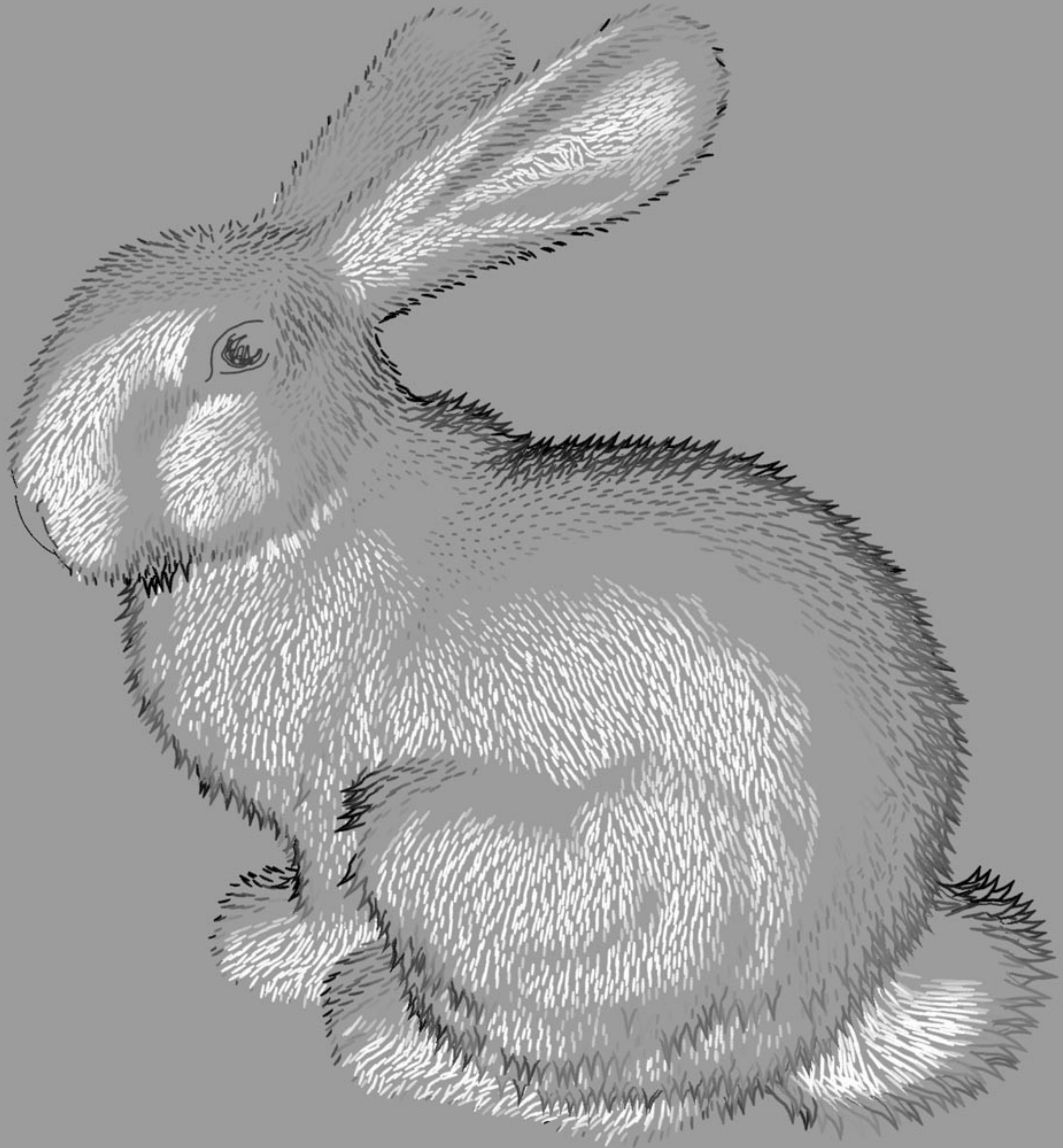
- Controlled *screen-space* density
- Placement on surfaces
- Controlled placement
  (e.g. only near silhouettes)
- Persistence between frames





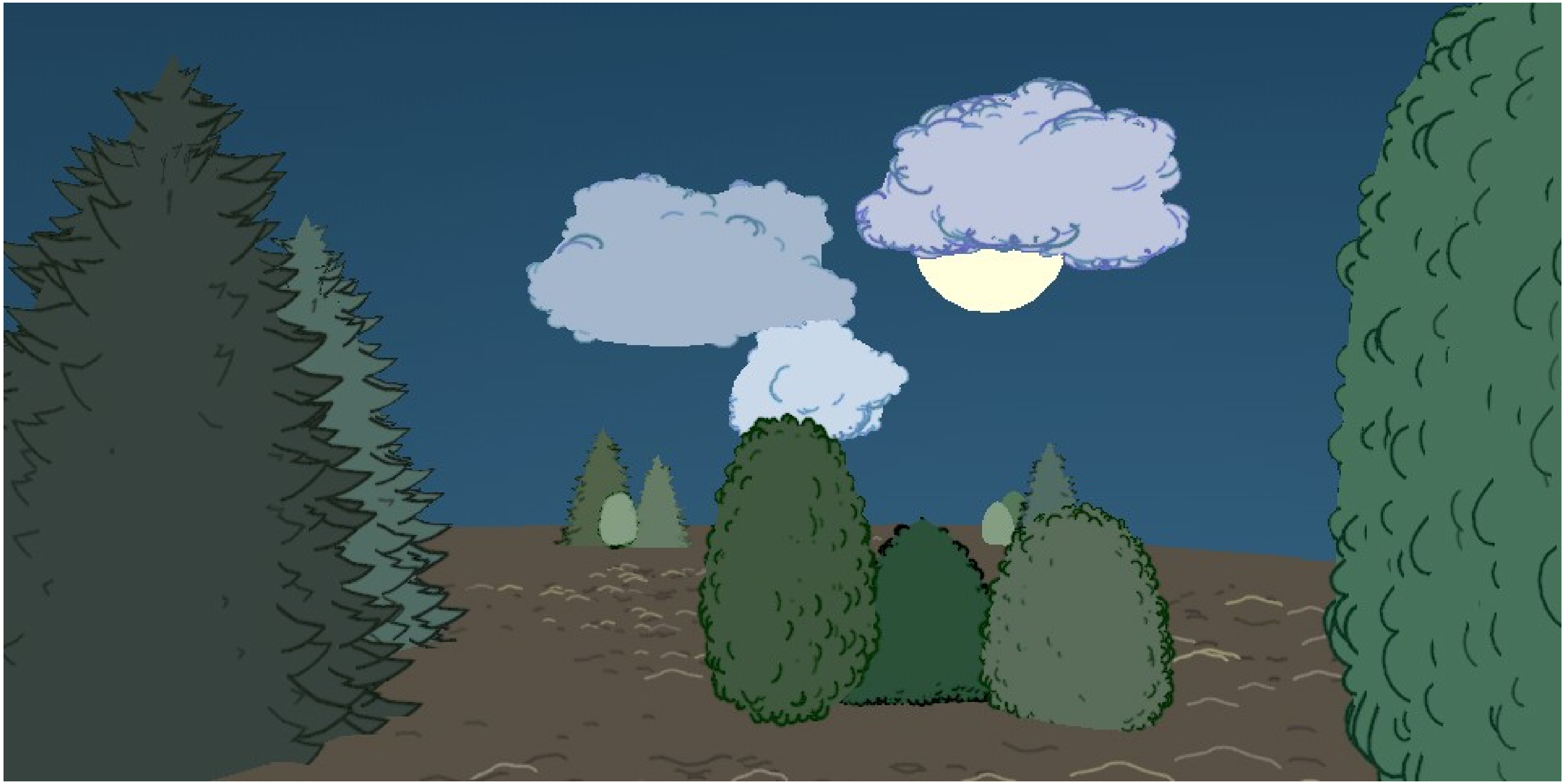






Problems

- Graftal textures defined in code
  - hard to edit
  - how to integrate with UI?
- Coherence
  - Graftals still appear/disappear suddenly
  - Better at low frame rates!



<u>Art-based Rendering w/ Continuous Levels of Detail</u>.

Markosian, Meier, Kowalski, Holden, Northrup, & Hughes.

NPAR 2000.

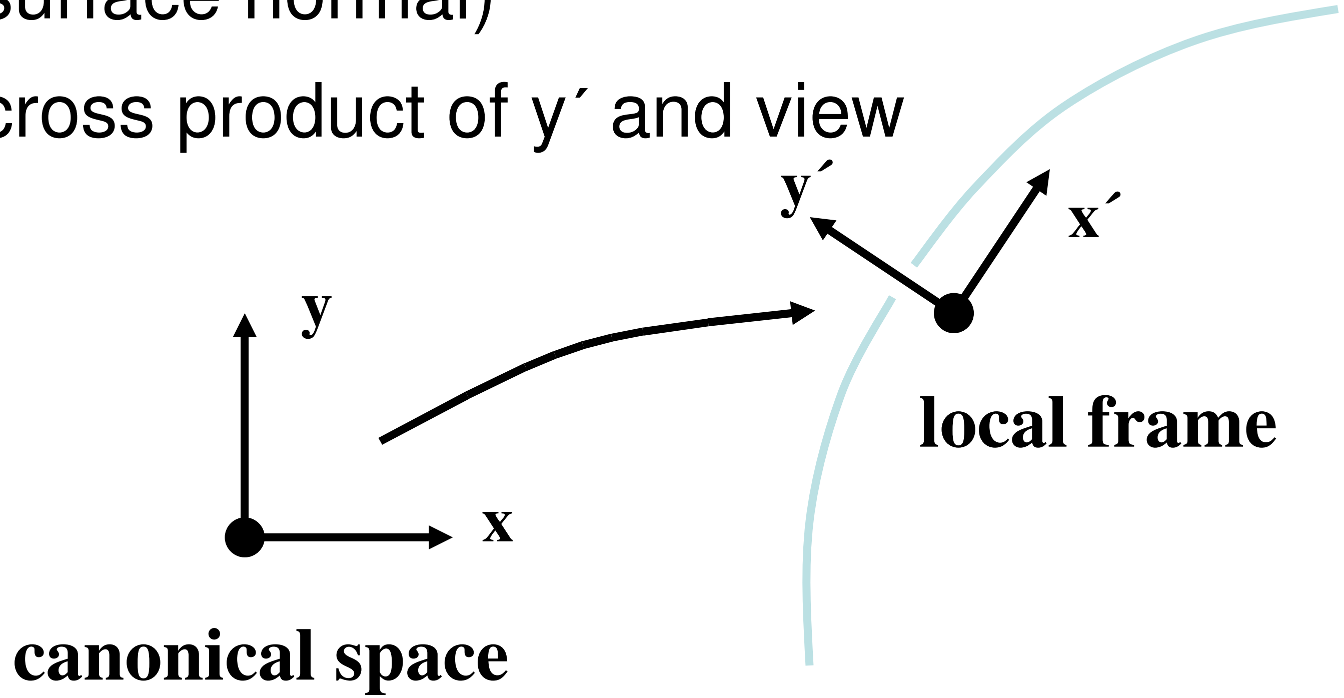
Basic graftals

- Collection of drawing primitives
  - triangle strips / fans
  - plus strokes
- Shared vertices
- Local coordinate frame

- Tuft: hierarchy of graftals

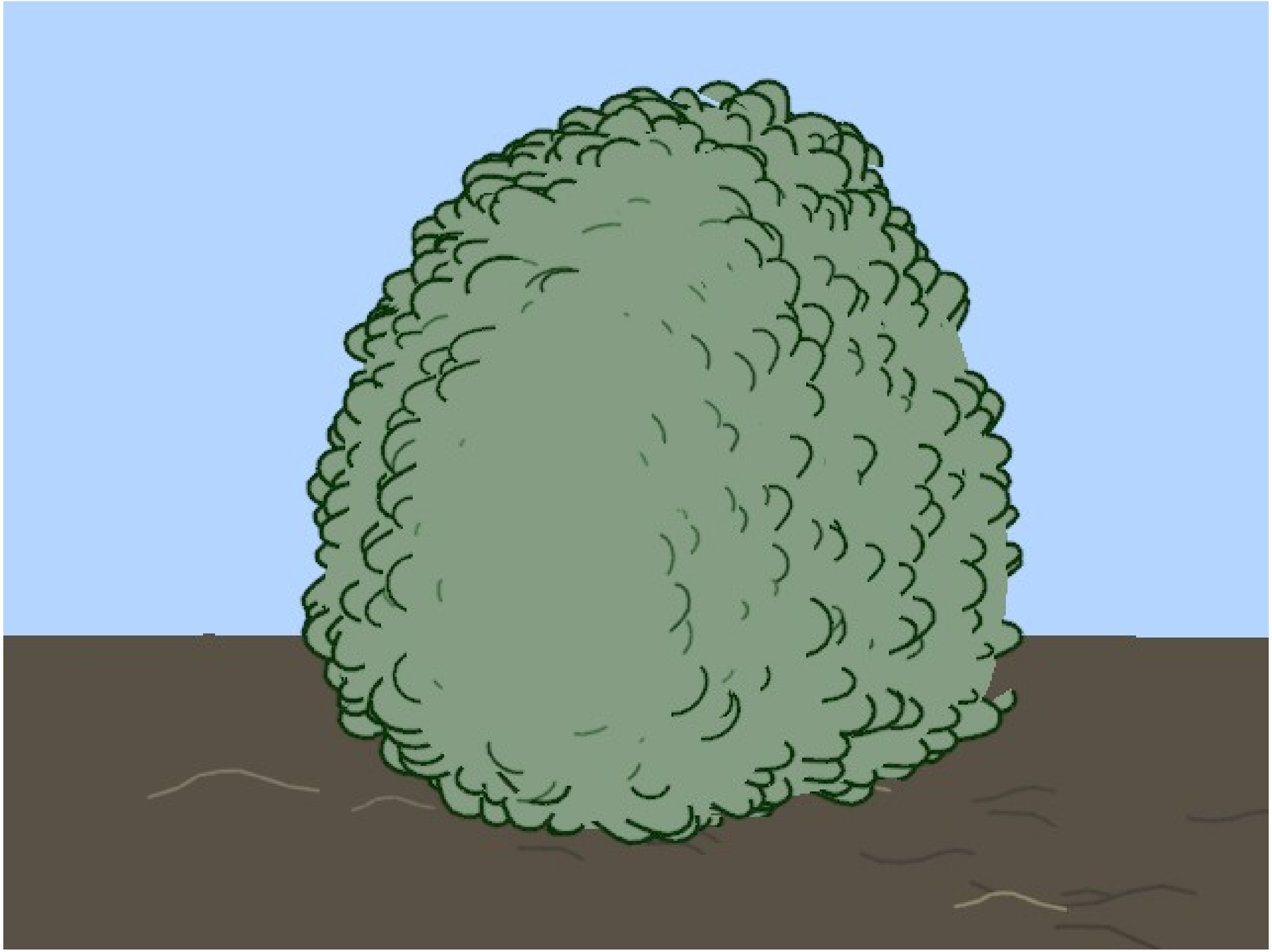
The local frame

- Base position (e.g. on surface)
- y´ (e.g. surface normal)
- x´ (e.g. cross product of y´ and view vector)



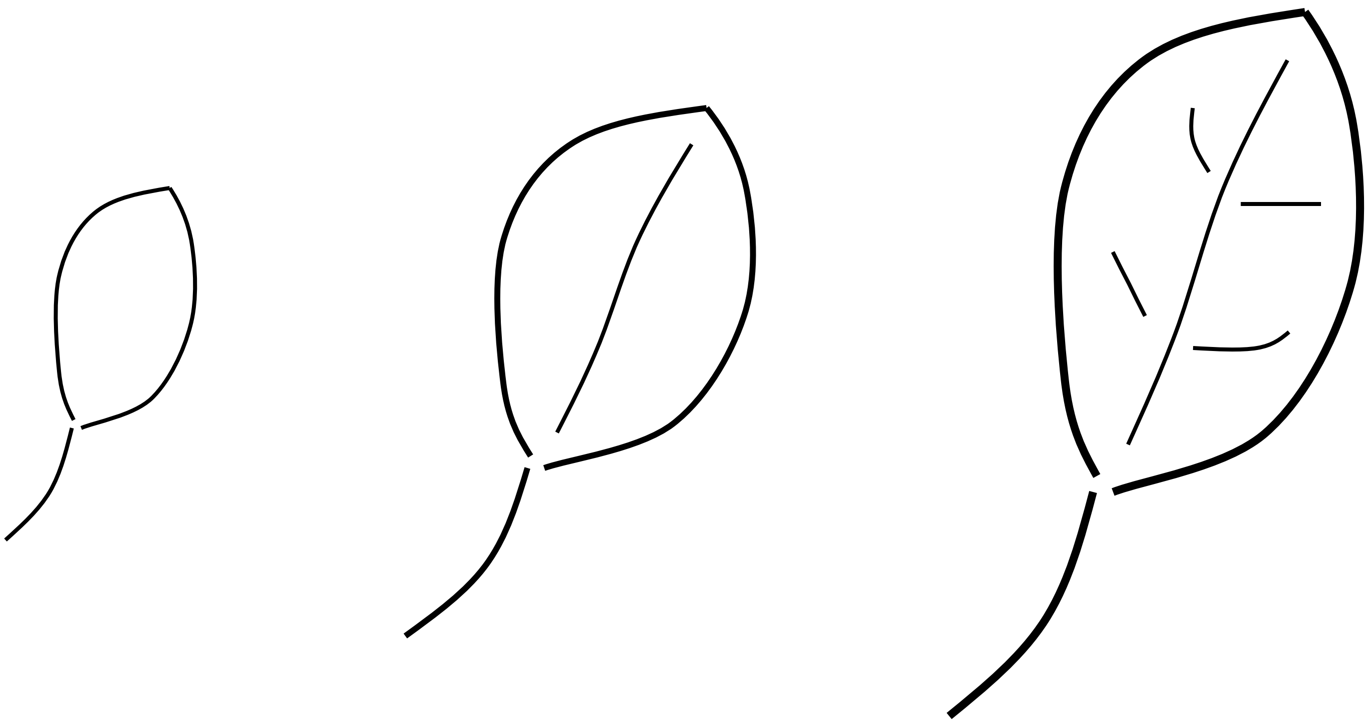
Placement and duplication

- Designer creates a few “example graftals”
- Copies can be generated on surfaces
  - explicitly
  - procedurally
- Random variation can be used
  - copies are not exact



Level of detail (LOD)

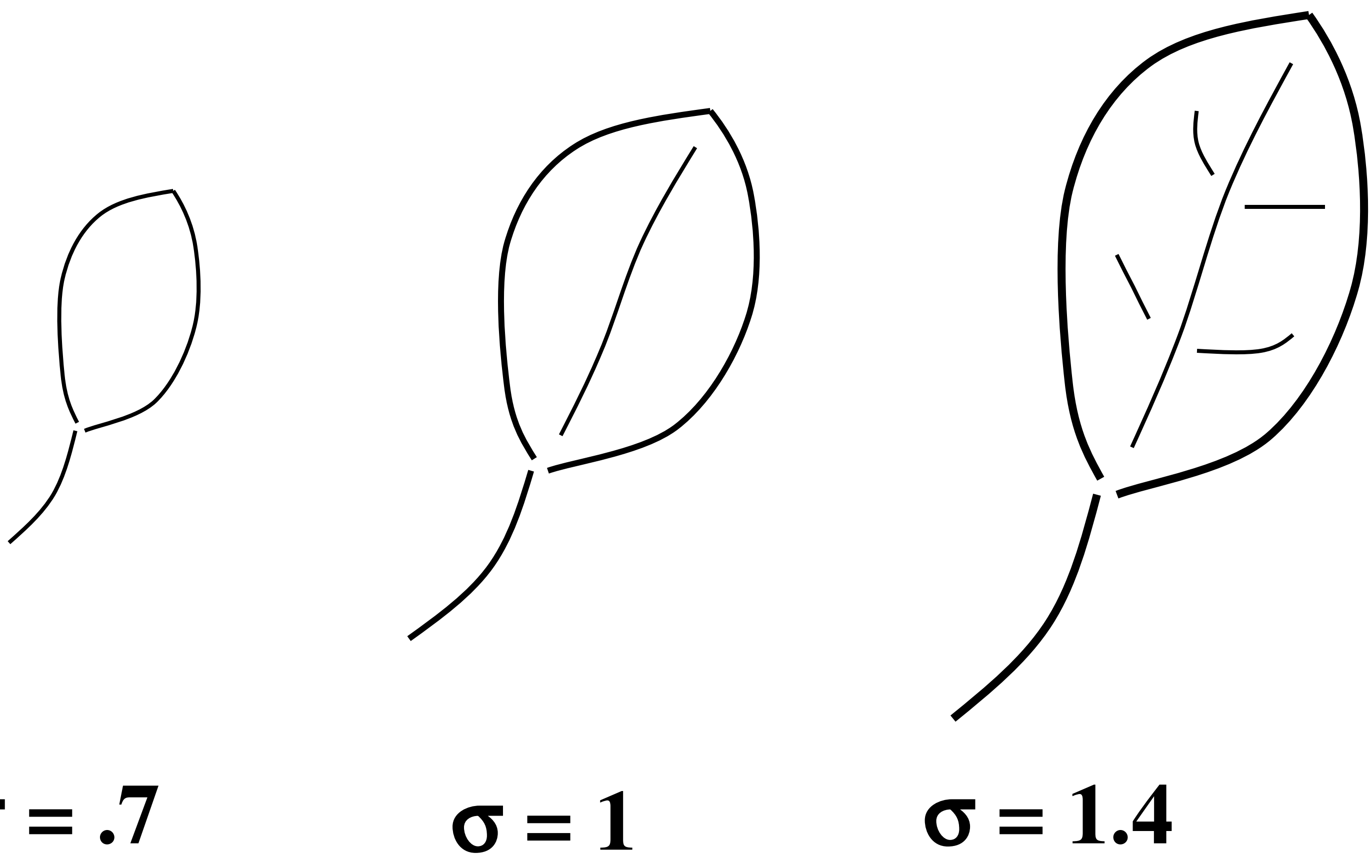
- Graftal computes current LOD
- Decides which primitives to draw



Computing LOD

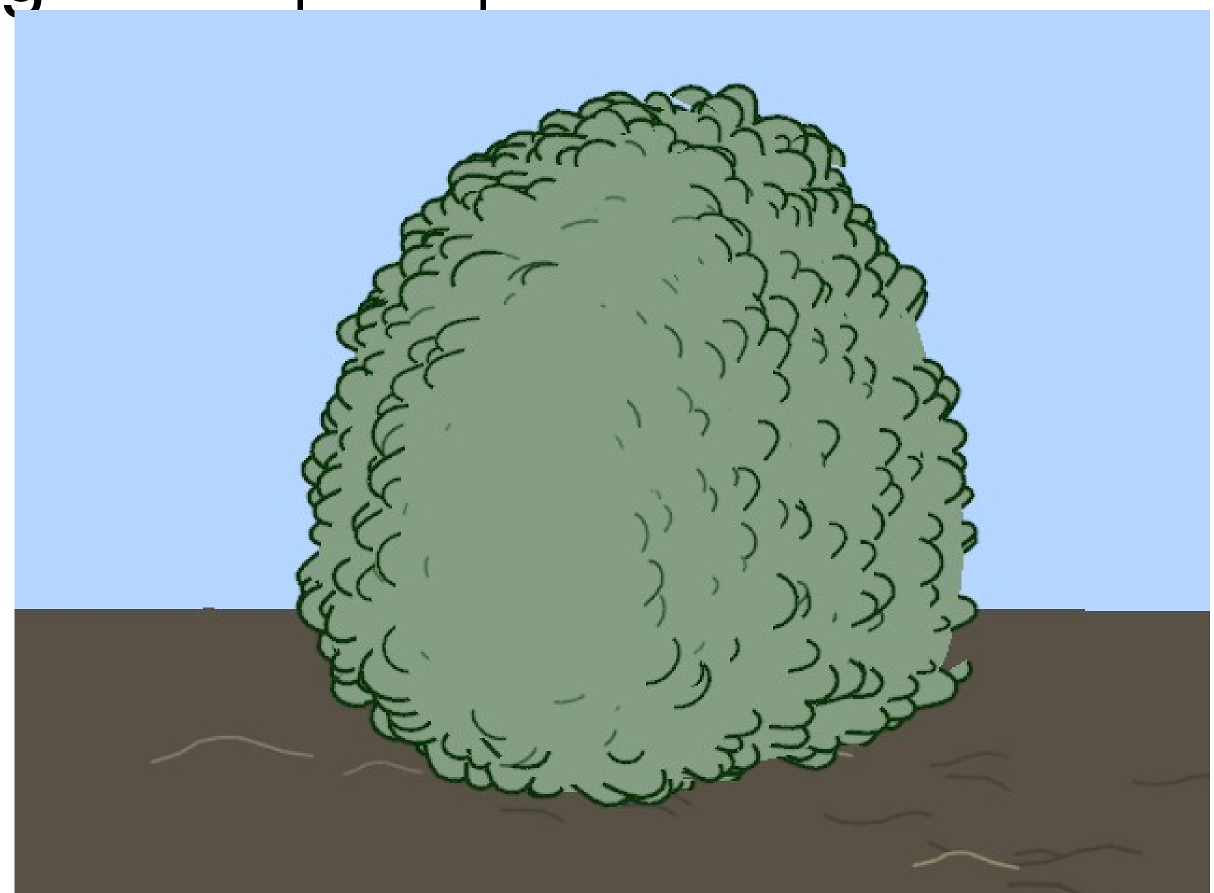
- LOD can be derived from:
  - apparent size
  - orientation
  - elapsed time

σ: ratio of current size to "rest" size



Orientation

- Value used to selectively suppress LOD
- E.g.: $1 - |v \cdot n|$



Movie

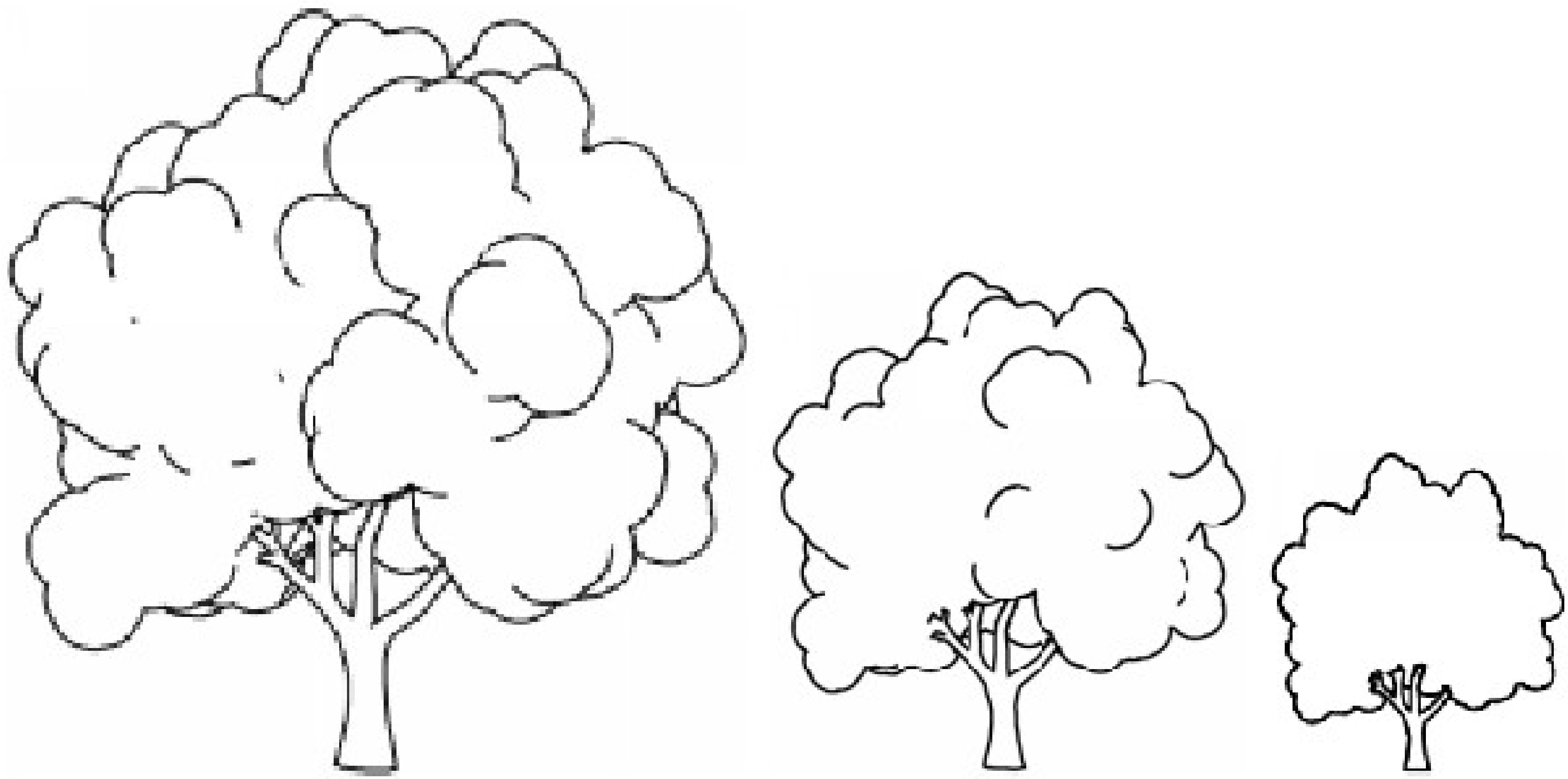
Discussion

- Coherence: much better!
- Slower
- Introducing / removing elements
  - Fading & thinning work well
  - Growing looks creepy
- LOD mechanism too inflexible
- Need direct UI

Pen & Ink: trees

- Deussen and Strothotte, Siggraph 00
- Problem: temporally coherent pen and ink rendering of trees
- Method:
  - Draw leaf entities w/ controlled size/abstraction
  - Do image processing on depth buffer

Tree I Tree II Tree III



talk overview

- motivation
- technical illustration
- pen & ink rendering
- painterly rendering
- graftals
- stroke-based rendering
- tonal art maps

WYSIWYG NPR: Drawing Strokes Directly on 3D Models.
Kalnins, Markosian, Meier, Kowalski, Lee, Davidson,
Webb, Hughes & Finkelstein. SIGGRAPH 2002.

Contributions

- Direct user-control for NPR
- Better silhouettes
- New media simulation
- Stroke synthesis by example
- Hatching with LODs

Overview of Components

Base Coat

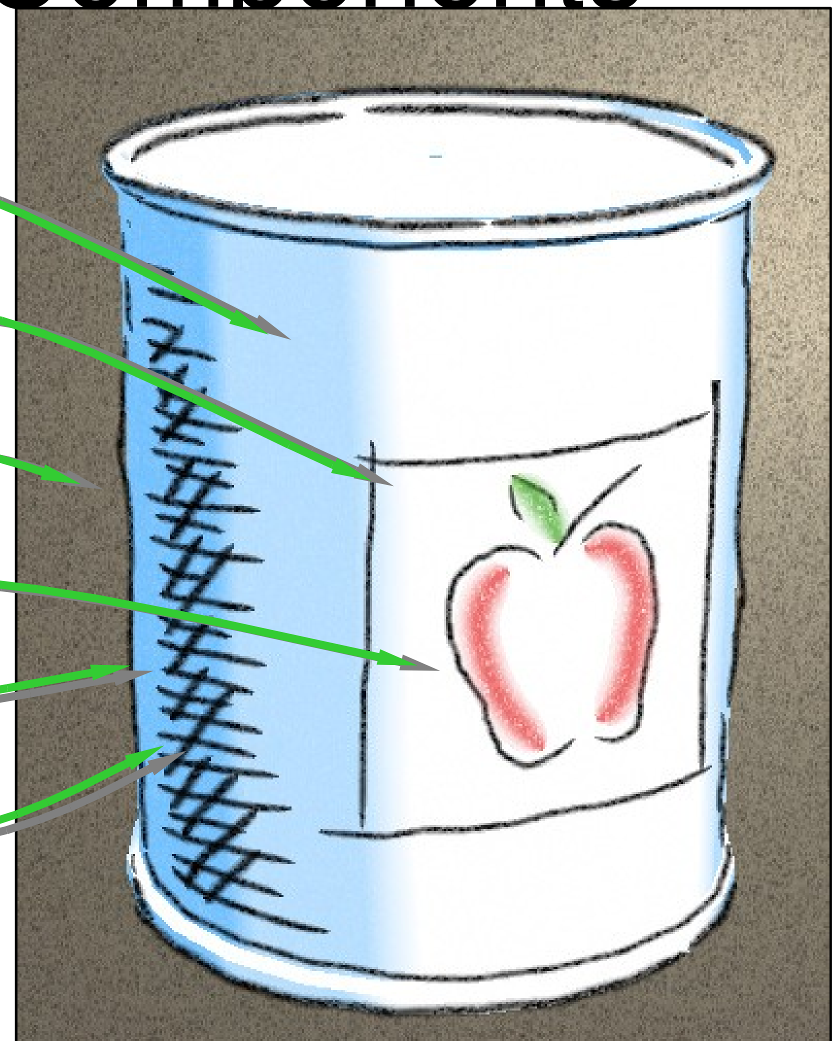
Brush Style

Paper Effect

Decals

Outlines

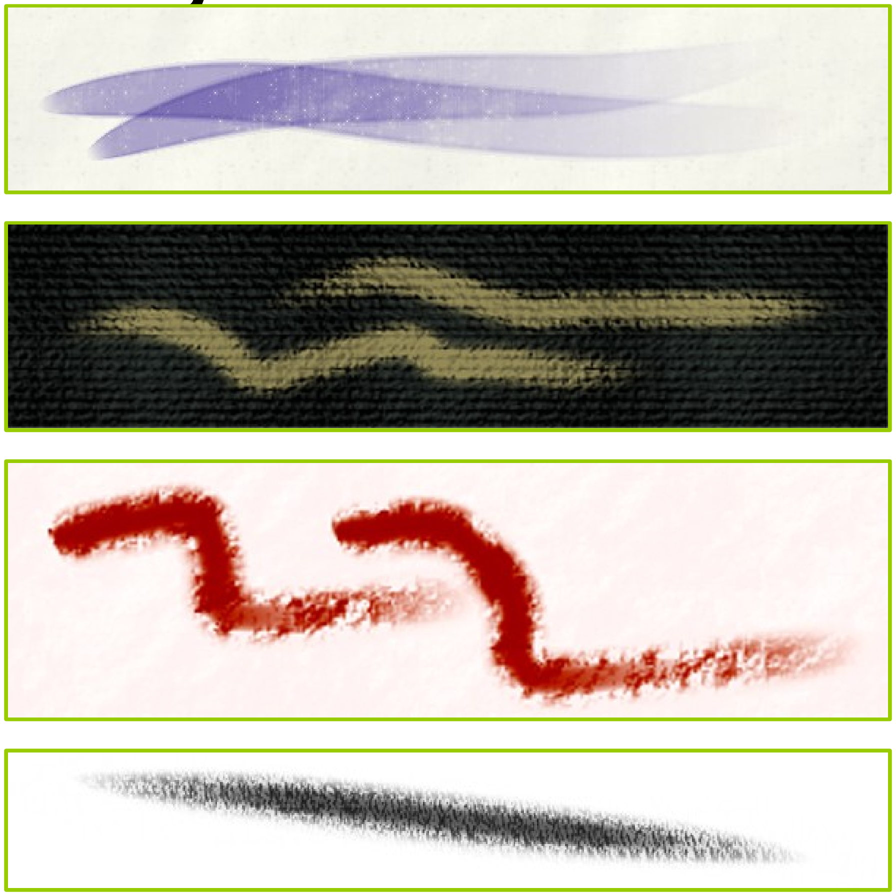
Hatching



Brush Style

Per stroke:

- Color
- Width
- Paper effect

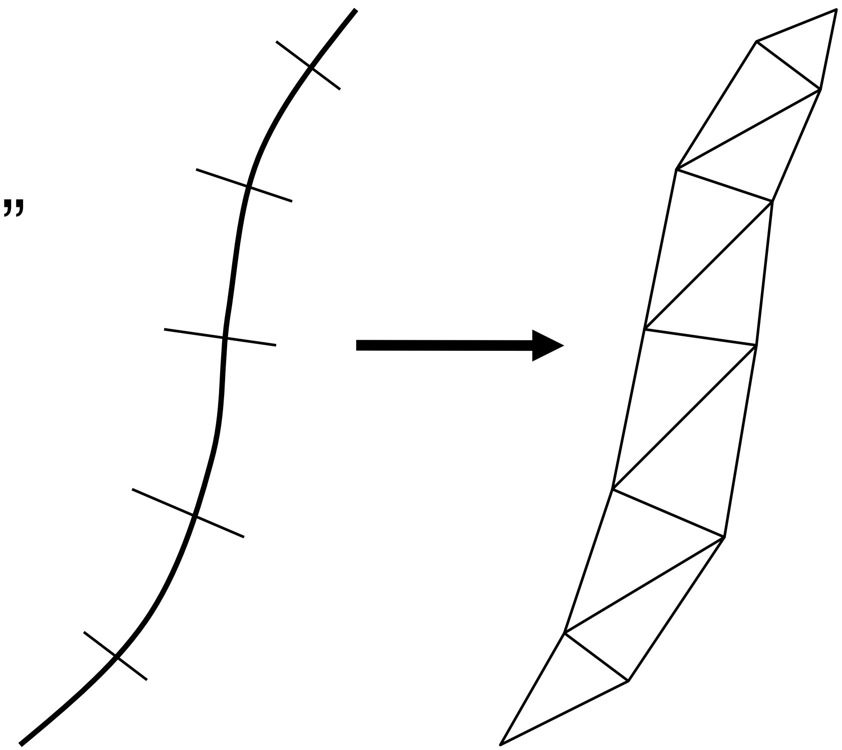


Rendered as triangle strips.

Strokes in OpenGL

Based on "Skeletal strokes"

Hsu *et al.*, UIST '93



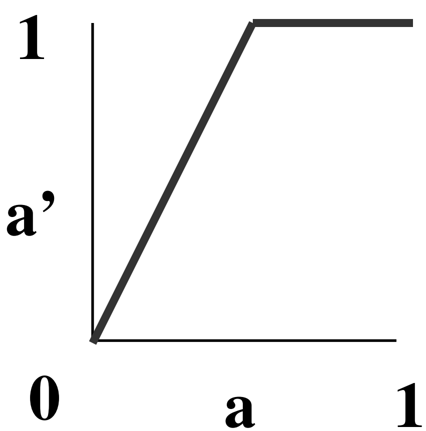
Paper Effect

- Height field texture:
- Peaks catch pigment
- Valleys resist pigment

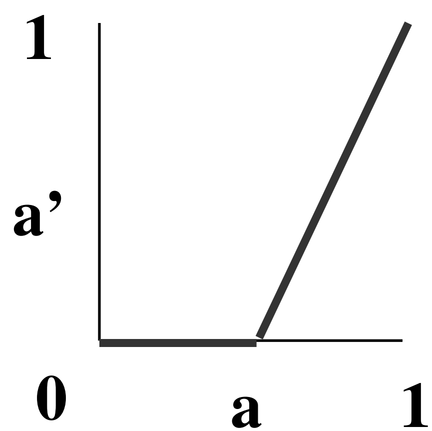
Implementation:

- Pixel shader

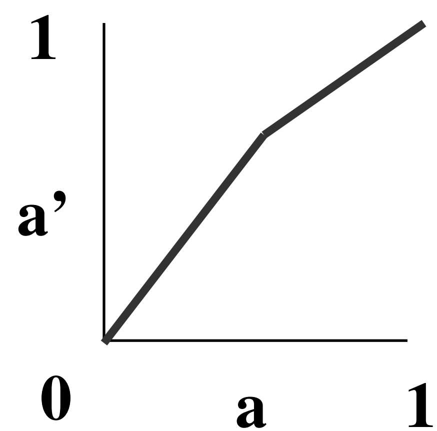
Re-map alpha with a “paper texture” heightfield



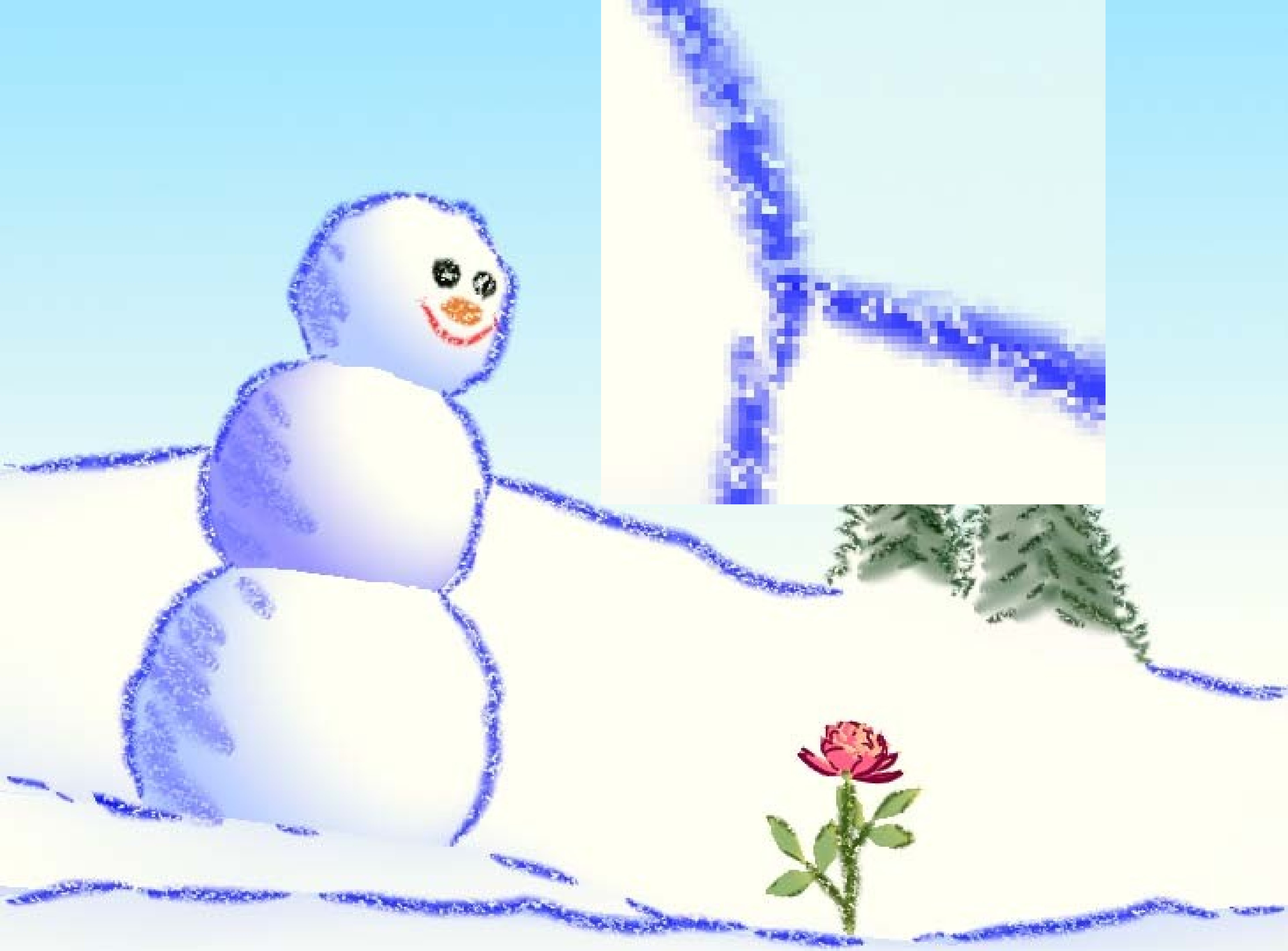
peak



valley



intermediate



talk overview

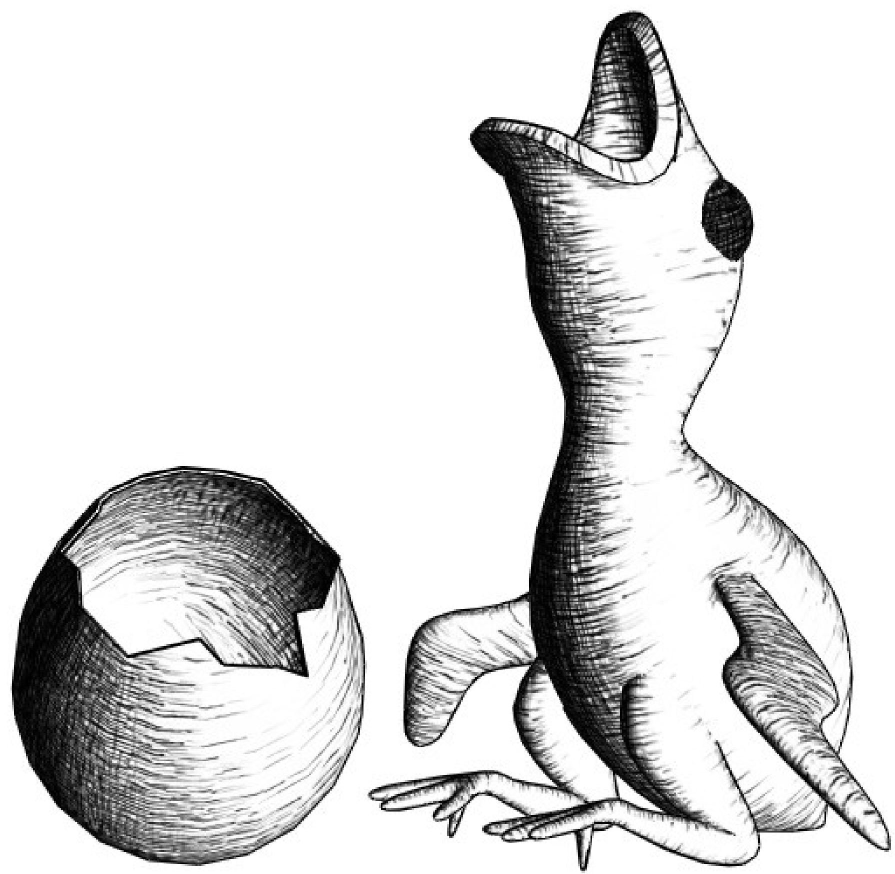
- motivation
- technical illustration
- pen & ink rendering
- painterly rendering
- graftals
- stroke-based rendering
- tonal art maps

Real–Time Hatching

Emil Praun	Princeton University
Hugues Hoppe	Microsoft Research
Matthew Webb	Princeton University
Adam Finkelstein	Princeton University

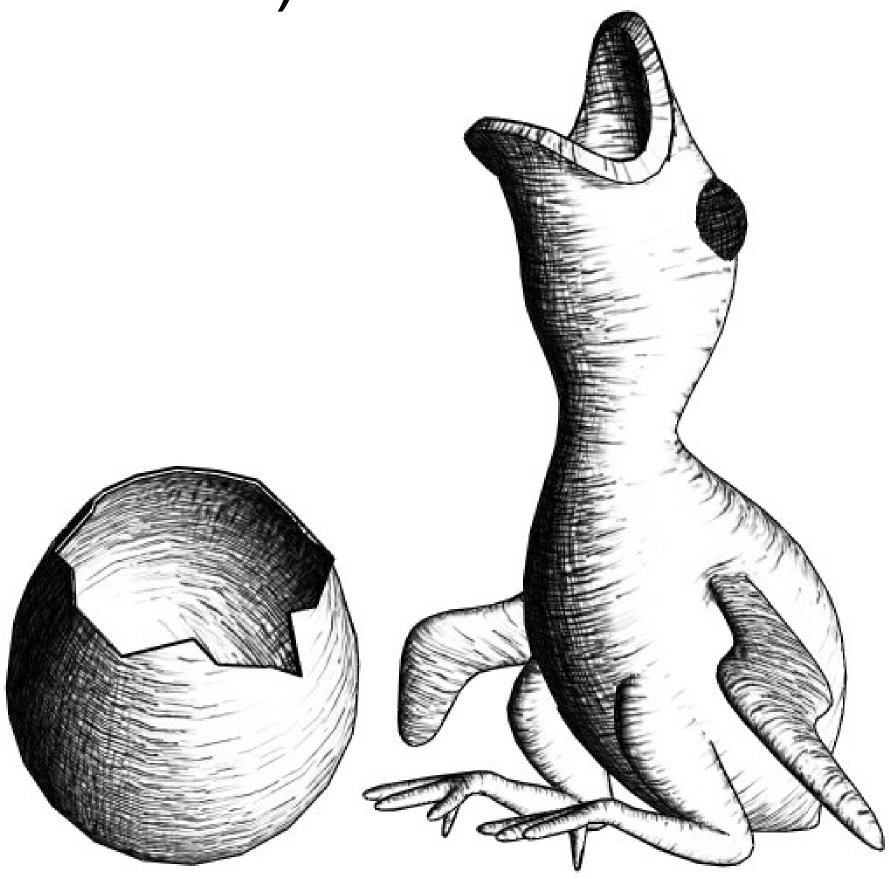
Goal

- Stroke-based rendering of 3D models
- Strokes convey:
  - tone
  - material
  - shape

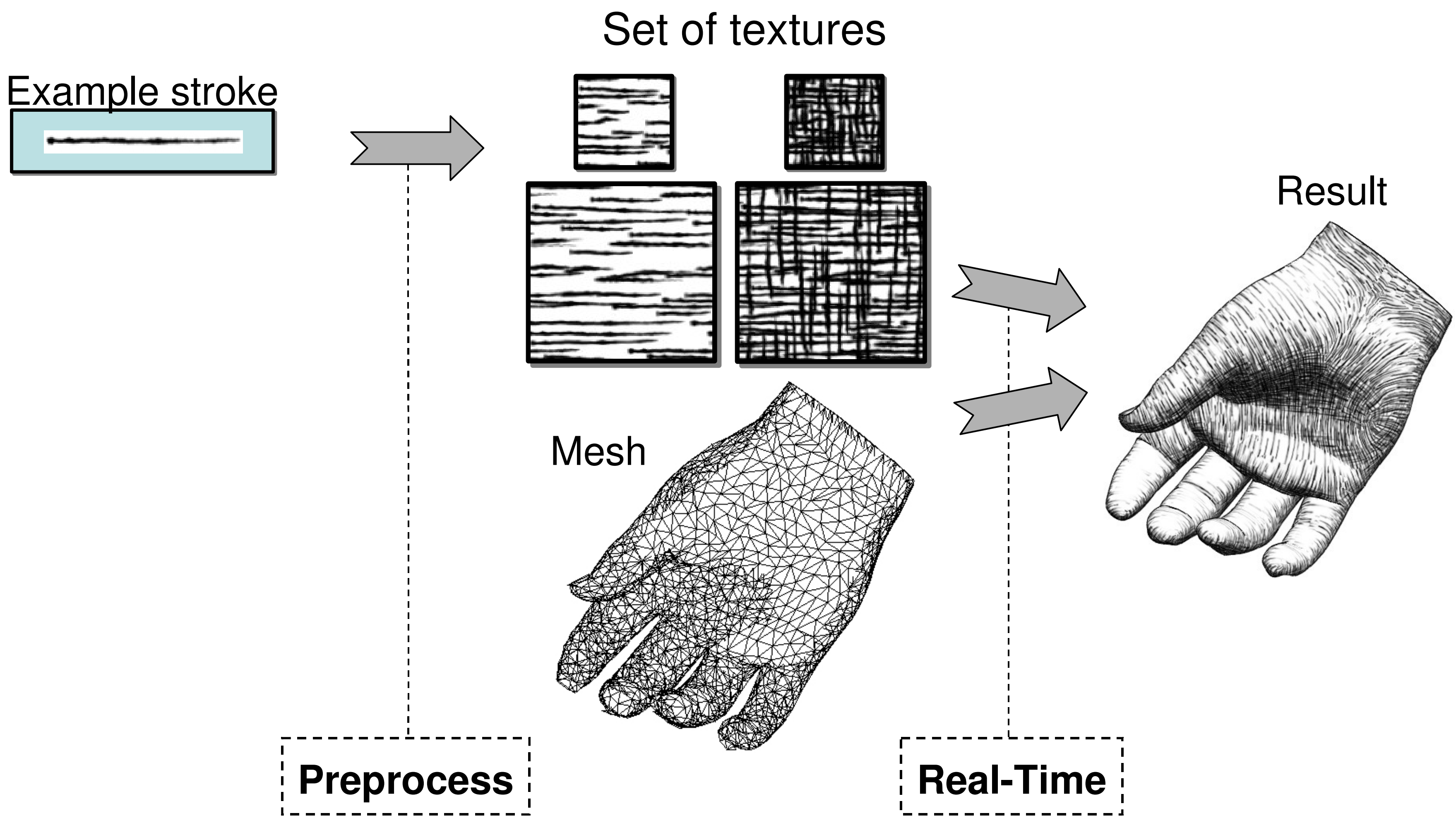


Challenges

- Interactive camera and lighting control
- Temporal (frame to frame) coherence
- Spatial continuity
- Artistic freedom

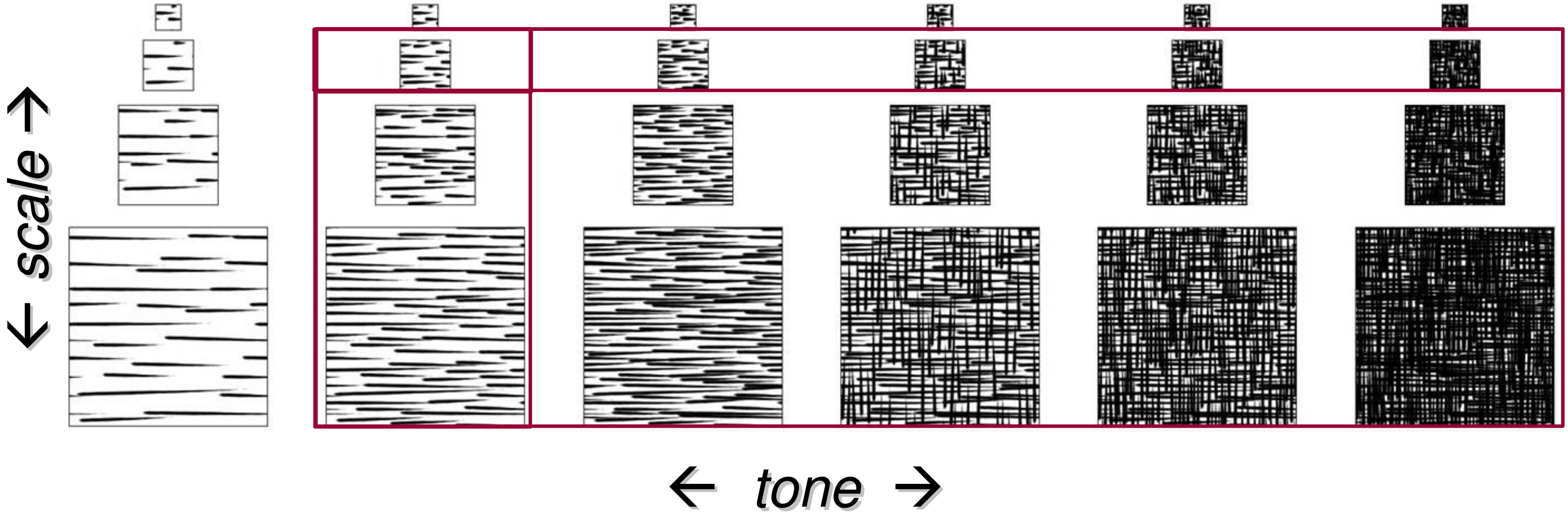


Approach


Set of textures
Example stroke
Result
Mesh
Preprocess
Real-Time

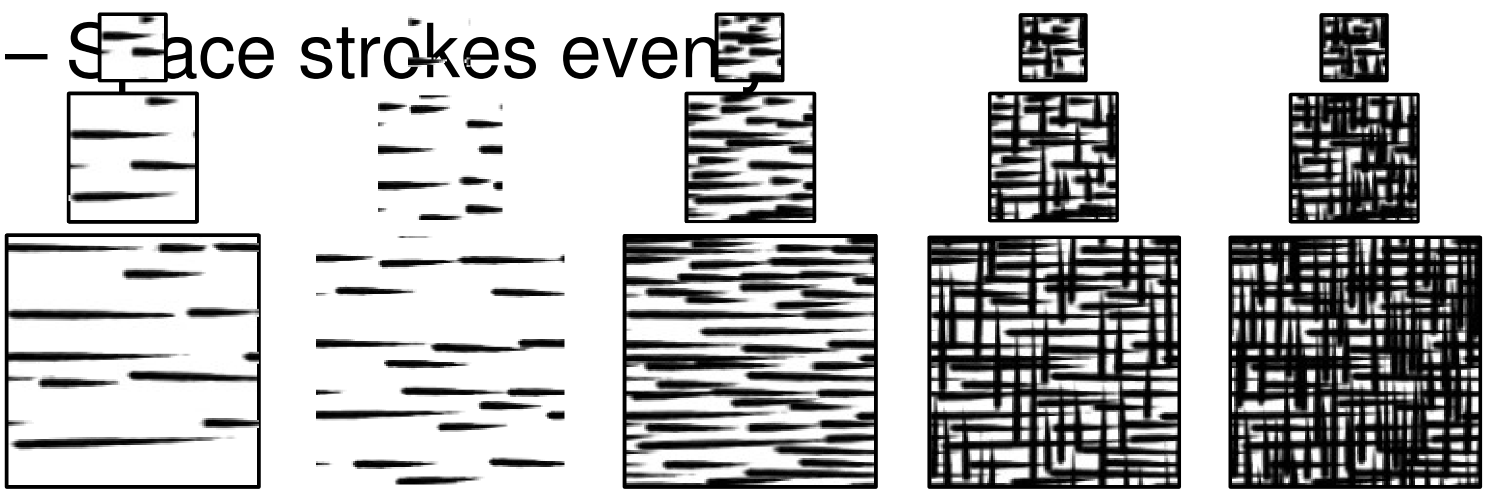
Tonal Art Maps

- Collection of stroke images
- Will blend → design with high coherence
- Stroke nesting property



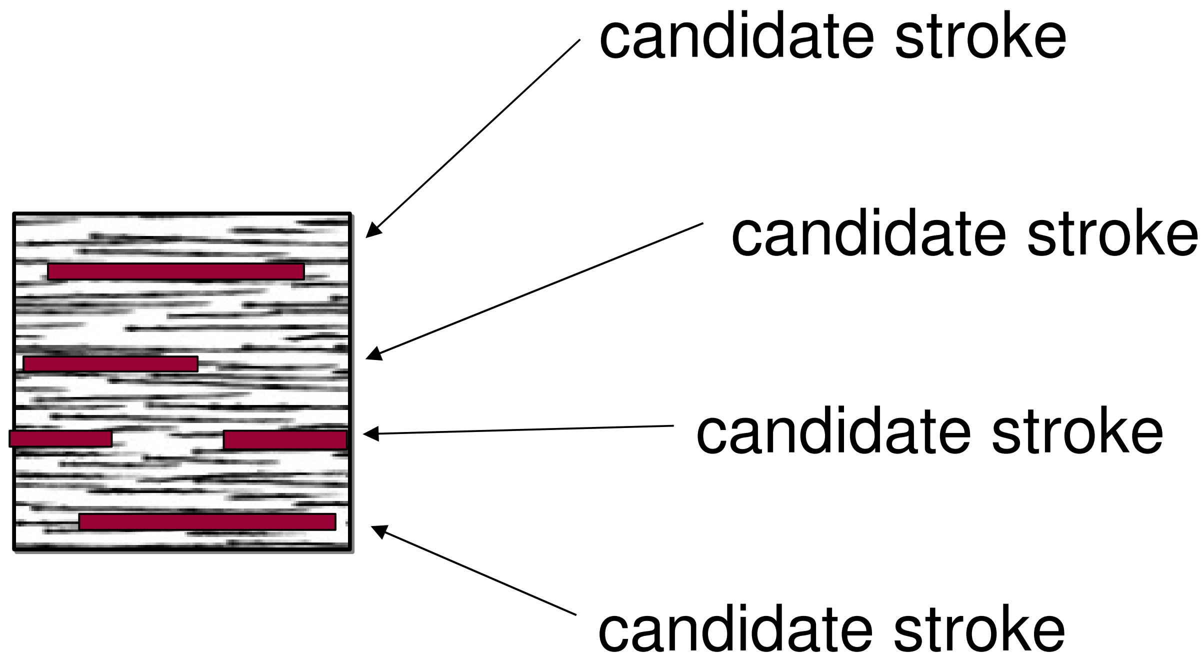
Generating Tonal Art Maps

- Draw or import bitmap for one stroke
- Automatically fill TAM with strokes
  - When placing stroke in an image, add it to all finer & darker images
  - Fill table column by column, coarse to fine
  - Space strokes evenly



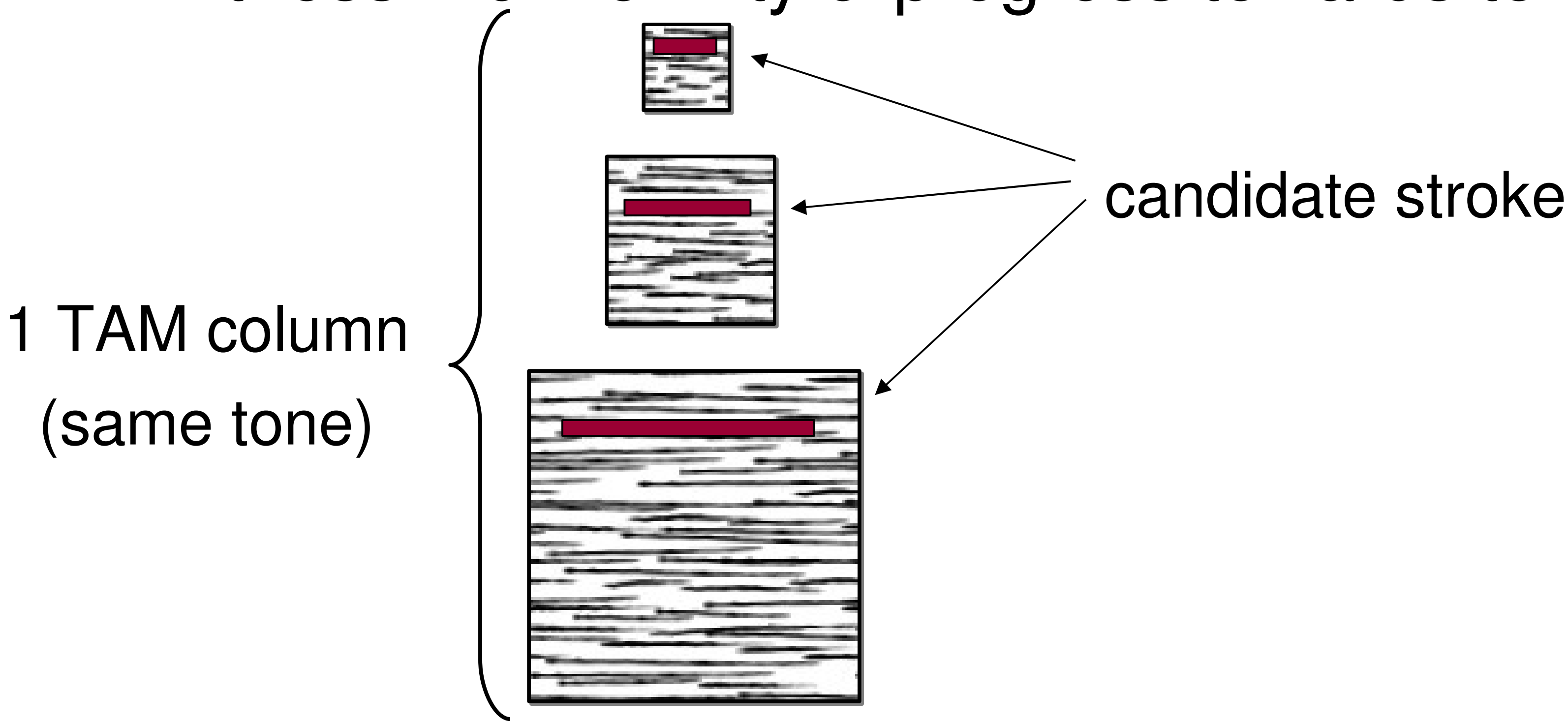
Even Spacing of Strokes

- Choose best stroke from large candidate pool
- Fitness = uniformity & progress towards tone



Even Spacing of Strokes

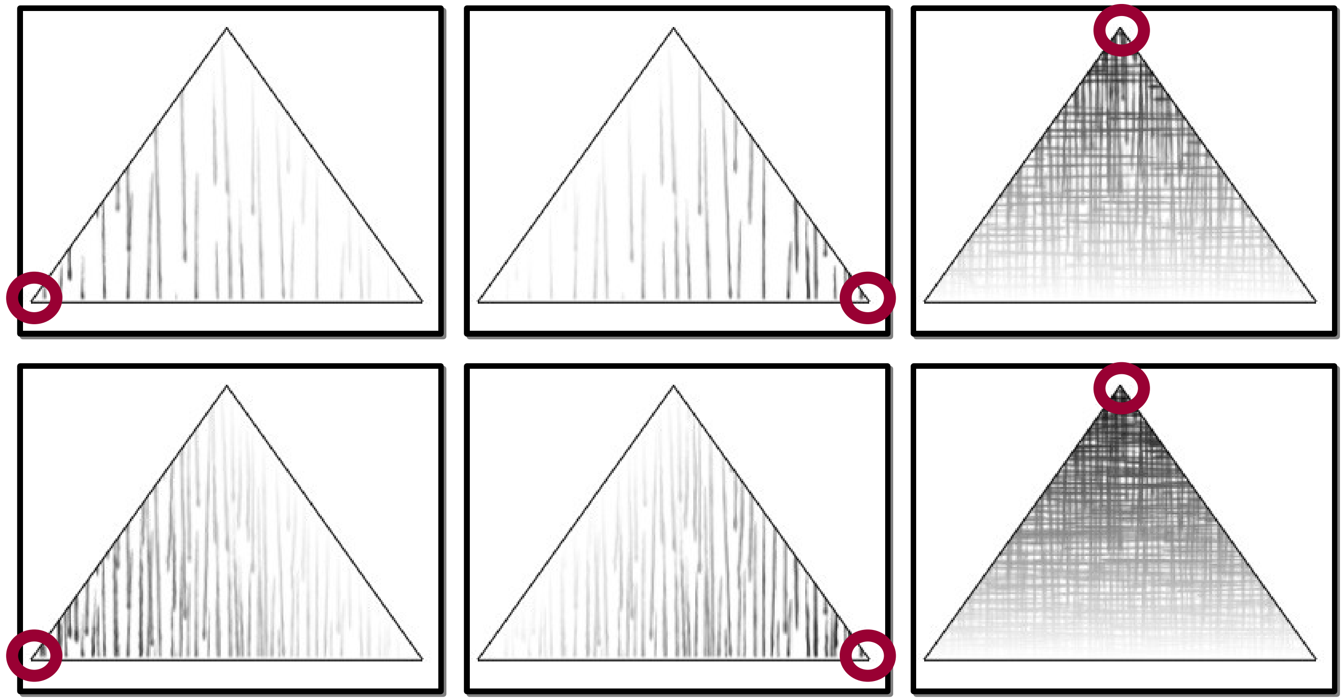
- Choose best stroke from large candidate pool
- Fitness = uniformity & progress towards tone



Texture Blending

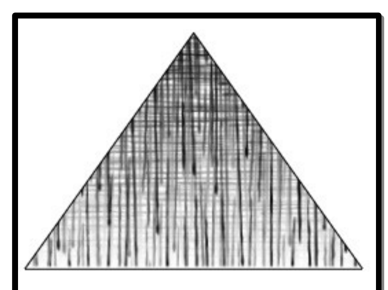
$\lfloor \textit{tone} \rfloor$

$\lceil \textit{tone} \rceil$



v_1 v_2 v_3

6-way blend → *final*



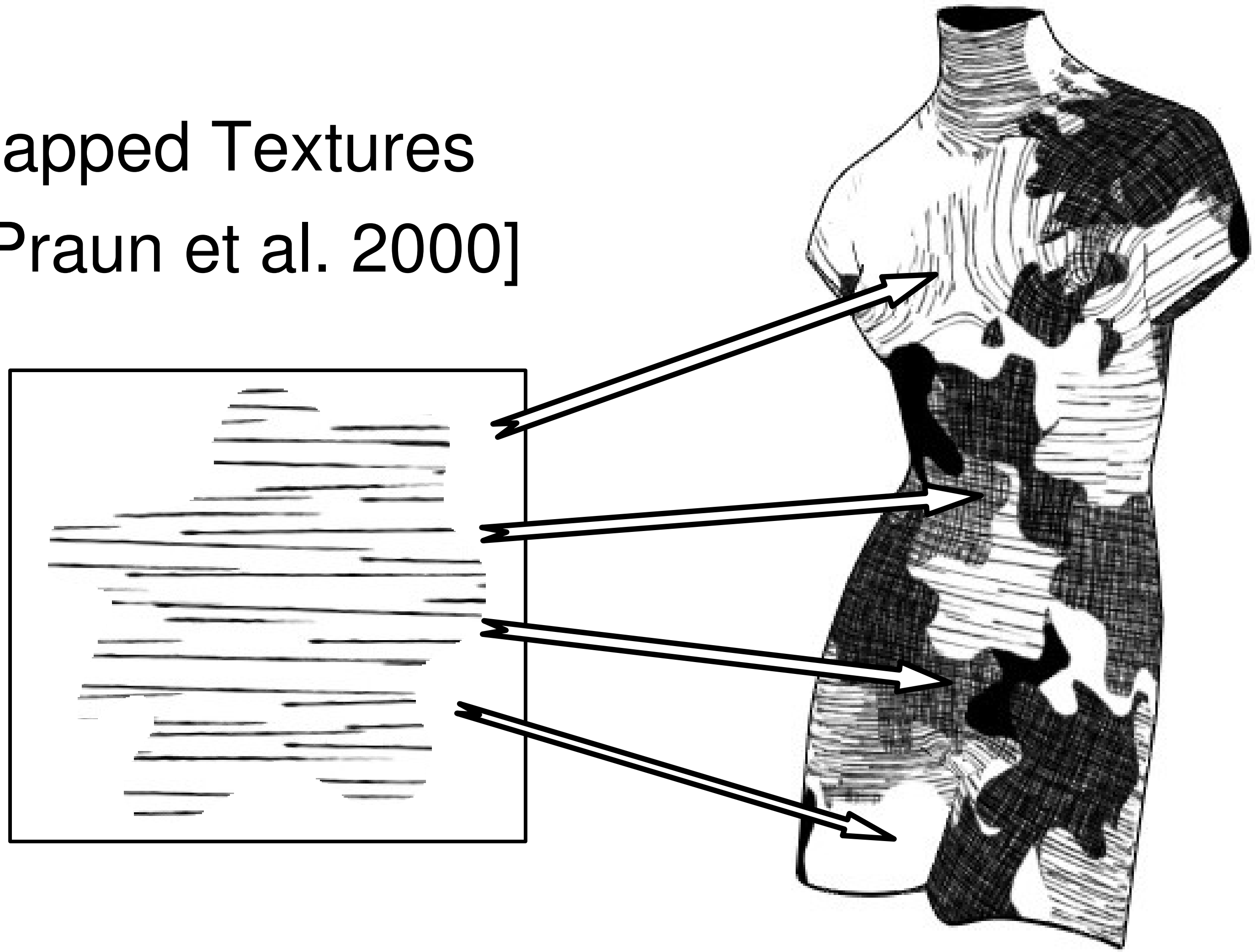
Texture Blending

- Pack grayscale tones in R,G,B channels
  - 6 tones in 2 textures
- Use multitexture engine
  - single-pass 6-way blend
- Vertex programs compute blend weights
  - static vertex data

```
!!VP1.0 #Vertex Program for Real-Time Hatching.
//output vertex homogeneous coordinates
DP4   R2.x, c[0], v[OPOS];
DP4   R2.y, c[1], v[OPOS];
DP4   R2.z, c[2], v[OPOS];
DP4   R2.w, c[3], v[OPOS];
MOV   o[HPOS], R2;
//stroke texture coordinates, transformed
DP3   o[TEX0].x, c[4], v[TEX0];
DP3   o[TEX0].y, c[5], v[TEX0];
DP3   o[TEX1].x, c[4], v[TEX0];
DP3   o[TEX1].y, c[5], v[TEX0];
// splotch mask coordinates
MOV   o[TEX2], v[TEX0];
//get the Gouraud shade
DP3   R1, c[8], v[NRML];
//apply clamp-linear tone transfer function
MUL   R1, R1, c[9].x;
ADD   R1, R1, c[9].y;
MAX   R1, R1, c[9].z;
MIN   R1, R1, c[9].w;
//now look up the weights for the TAMs blending
EXP   R2.y, R1.x; //frac(tone)
ARL   A0.x, R1.x;
MOV   R3, c[A0.x + 10];
MAD   R3, -R2.y, R3, R3;
MAD   o[COL1], R2.y, c[A0.x + 11], R3;
MOV   R4, c[A0.x + 20];
MAD   R4, -R2.y, R4, R4;
MAD   o[COL0], R2.y, c[A0.x + 21], R4;
END
```

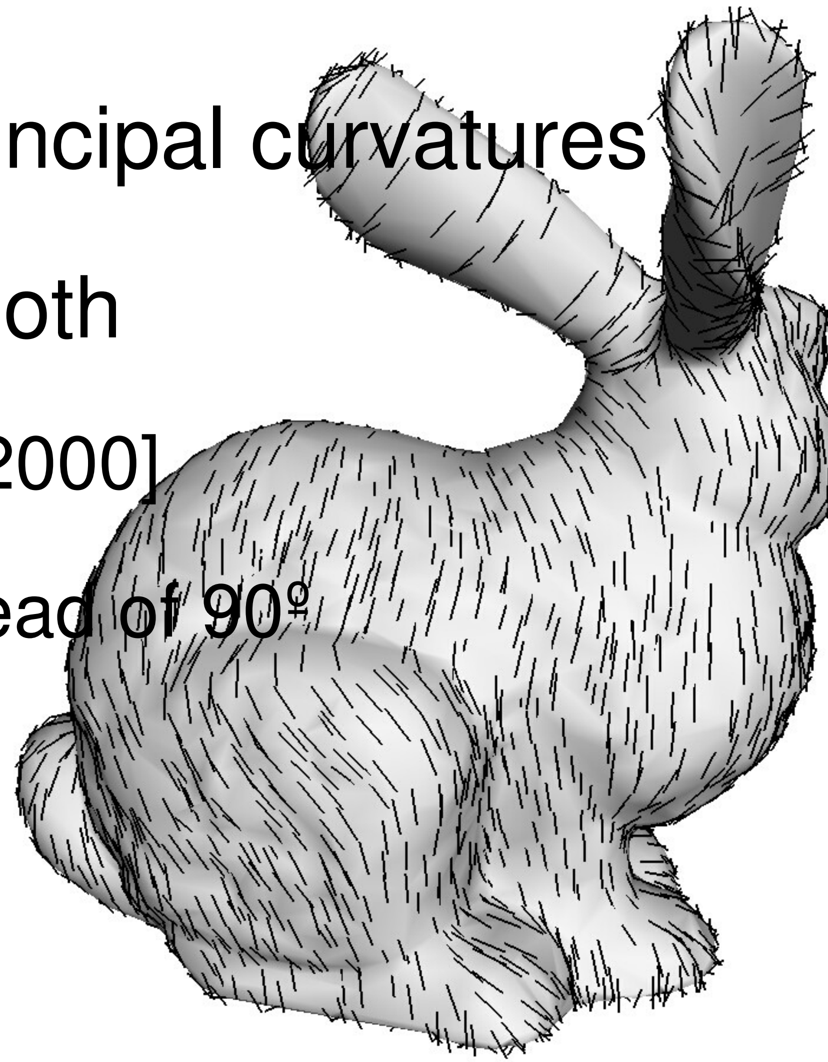
Texturing Arbitrary Surfaces

- Lapped Textures
  [Praun et al. 2000]



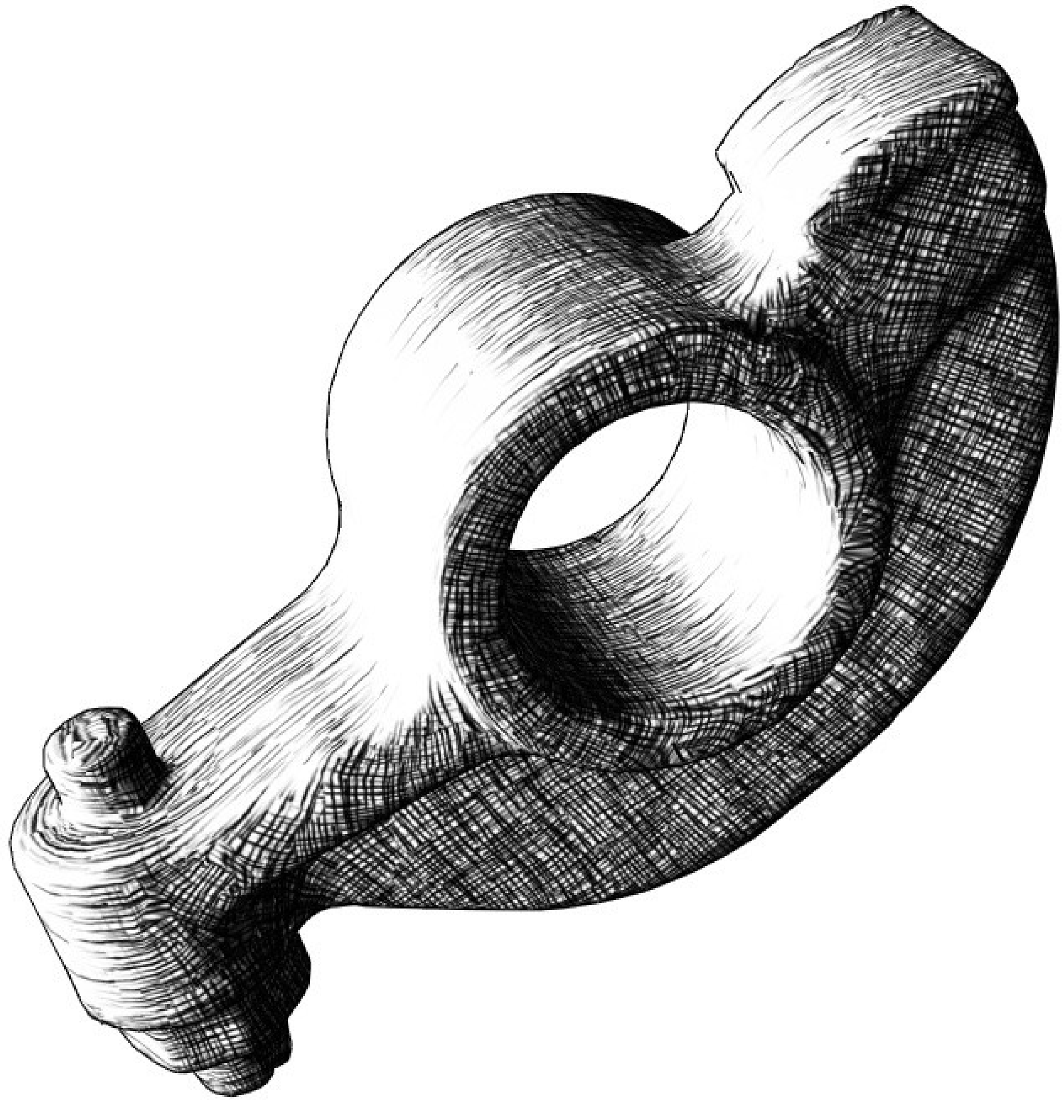
Direction Field

- Based on surface principal curvatures
- Optimized to be smooth
  - [Hertzmann & Zorin 2000]
  - Symmetry: 180º instead of 90º

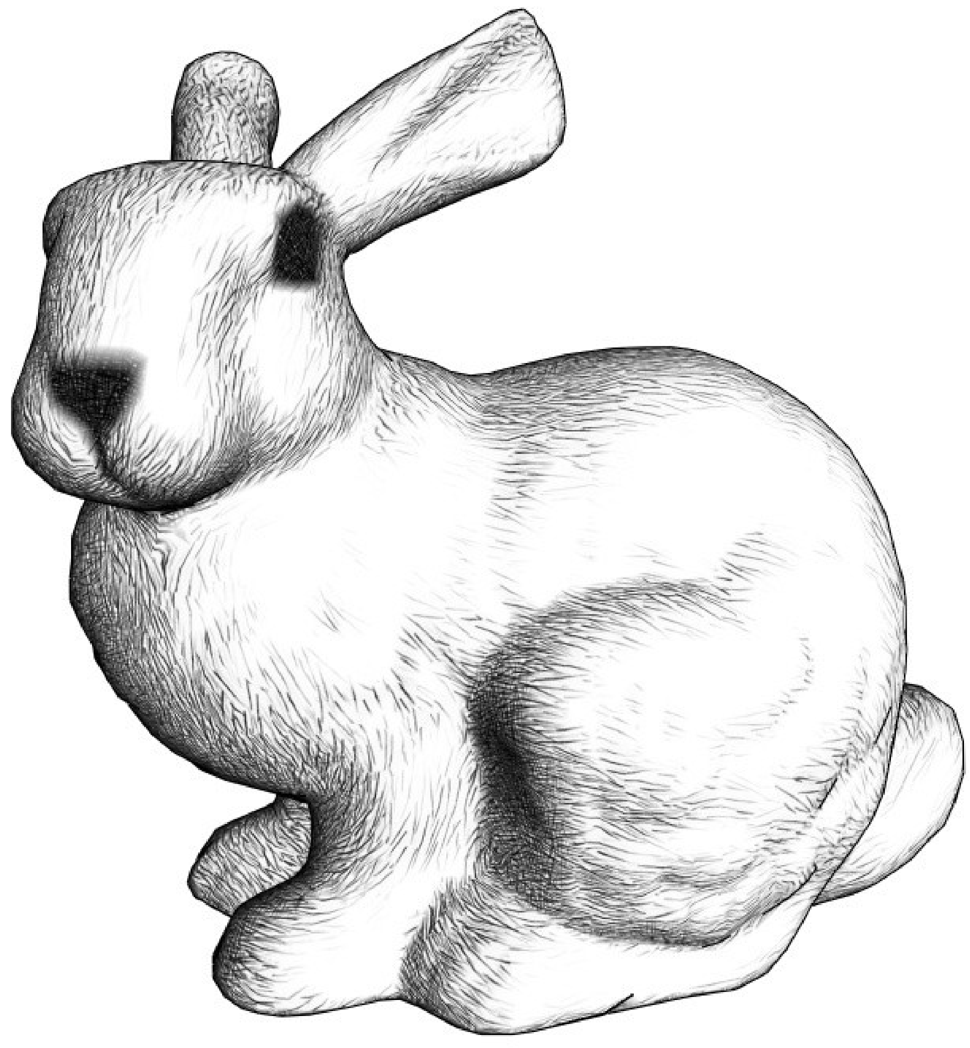


Result

Result



Result

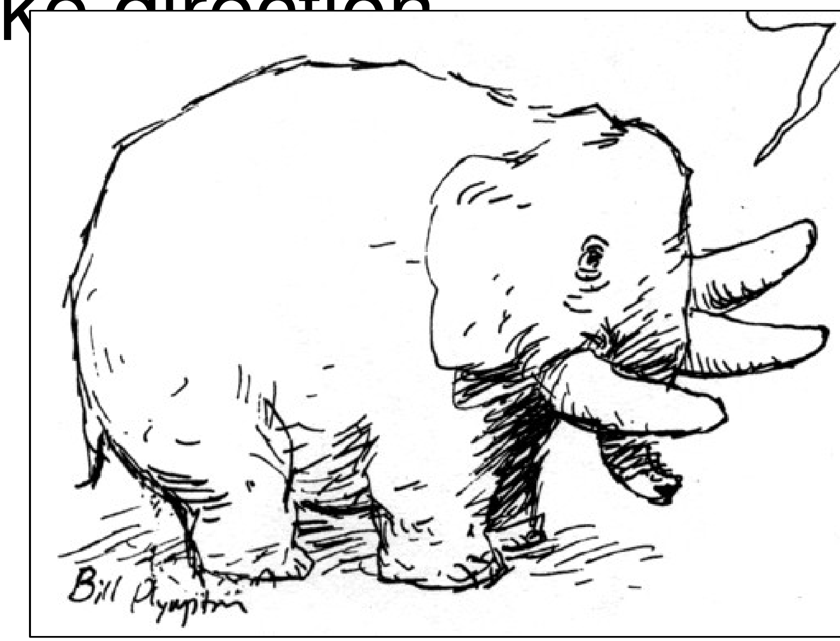


Summary

- Real-time hatching for NPR
- Strokes rendered as textures
- High coherence TAMs prevent blend artifacts
- 6-way blend very fast on modern graphics

Future Work

- More general TAMs
- View-dependent stroke direction
- Automatic indication



Bill Plympton

Wednesday

- last class
- course evaluations
- review for final
- homework due
- final exam:
  - December 21, 1:30 – 3:30 pm
  - EECS 1301 - 1303