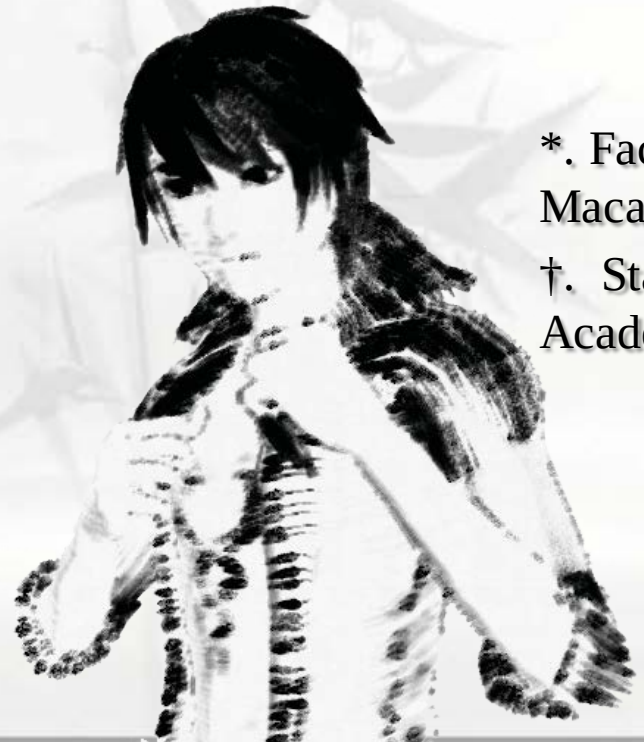
Stroke-based Real-time Rendering of Ink Wash Style for Geometric Models



Tian-Chen Xu*, Li-Jie Yang*, En-Hua Wu*†

*. Faculty of Science and Technology, University of Macau, Macao, China

†. State Key Lab of CS, Institute of Software, Chinese Academy of Sciences, Beijing, China

SIGGRAPH ASIA 2012

The 5th ACM SIGGRAPH Conference and Exhibition on Computer Graphics and Interactive Techni ques in Asia

Outline

- Introduction
- Related Work
- Ink Rendering
- Stroke Generation
- Results & Demos
- Summary

SIGGRAPH ASIA 2012

Introduction

- **Ink wash painting**
  - Originated in China
  - Soul vs. appearance
  - Various compounds of water and ink
  - Introduced to Korea and Japan (Sumi-e etc.)
- **What can be done by computer graphics**
  - Replace the position of real artwork?
  - Ink wash stylistic animations
    - Save time & labor consumption
    - Eg. “The Cow Boy’s Flute” (1963)
      - 2 years using traditional craftwork
      - Reproduce within only one month by CG

Introduction (Cont')

- **Current applications**
  - Hard to convey the feature of strokes on 3D models
  - Low perception of painting
    - Hierarchical vs. stereoscopic perception
- **Challenges & our motivations**
  - High-quality real-time rendering
  - Feature convey: core ink wash techniques
    - Contouring ("gou")
    - Wrinkling (texturing, "cun")
      - Rubbing (supplement of wrinkling, "ca")
    - Coloring (dyeing, "ran")
      - Dotting ("dian")

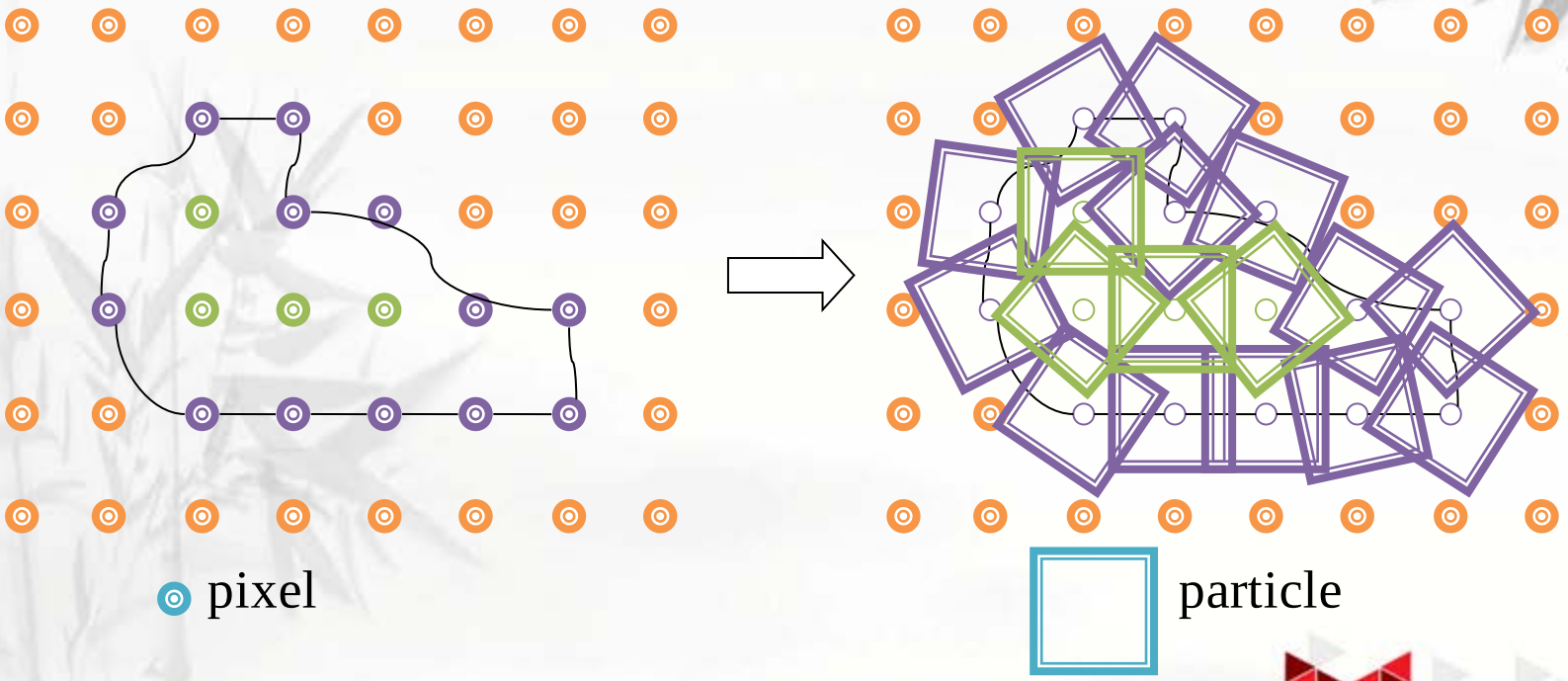
Related Work

- **Virtual brush models**
  - E-brush [Xu et al. 2004]
  - Mass-spring skeletal model [Chu and Tai 2004], …
- **Ink simulation & rendering**
  - Ink dispersion in absorbent paper using LBM [Chu and Tai 2004]
  - Real photo to sumi-e [Xie et al. 2010], …
- **Line drawing**
  - Per-triangle contour [Hertzmann and Zorin 2000]
  - Suggestive contour [DeCarlo et al. 2003], …
- **GPU acceleration**
  - Chinese landscape painting [Chen et al. 2005], …

Our Method Overview

- **Stroke extraction**
  - Contour extraction
  - Texture stroke representation (wrinkling/texturing)
  - Shade ink generation (coloring/dyeing)
  - Object to dot with depth
- **Ink rendering**
  - Stroke pixels → particles
    - Pressure
    - Orientation
  - Particle rendering
    - Density (ink intensity)
    - Pressure
    - Dryness

Our Method Overview (Cont')

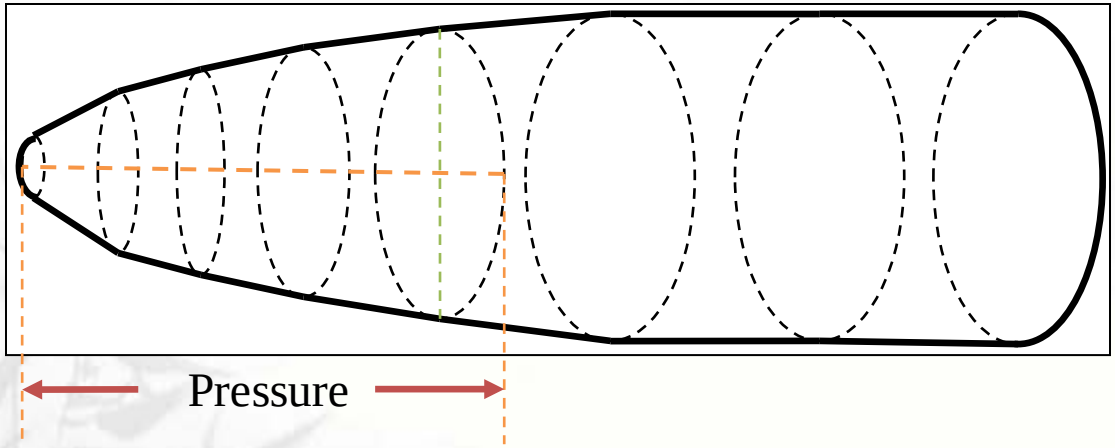


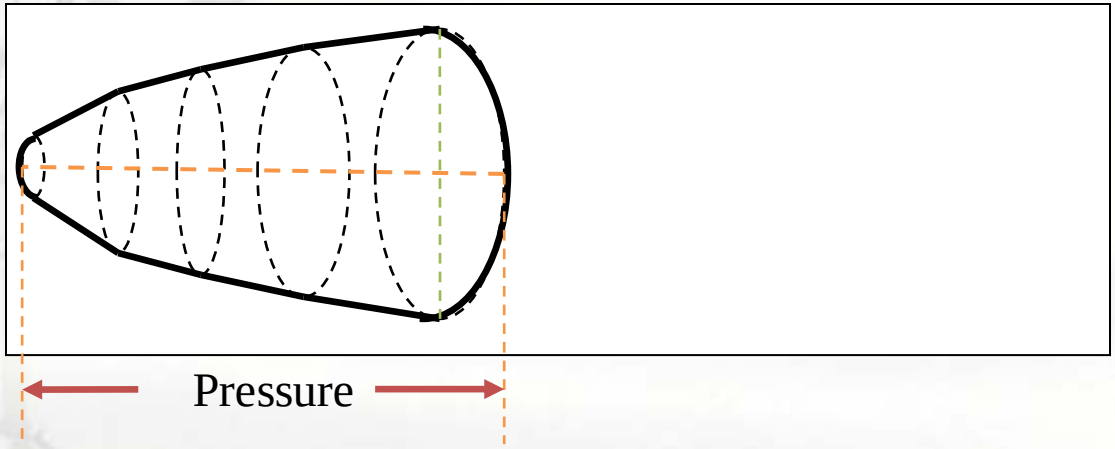
Ink Particle I/O Structure

- **Input: *PADI* structure**
  - Pressure (P)
  - Orientation (angle, A)
  - Dryness (D)
  - Density (ink intensity, I)
- **Output: *RGBA***
  - RGB for ink color
  - A channel: alpha value, $\alpha = (kF + k_{diff}F_{diff})I - k_{dry}\xi D$
    - F: footprint mask value
    - k: coefficients (weight factor)
    - ξ : random threshold (noise)
    - *diff*: subscript for dispersed ink

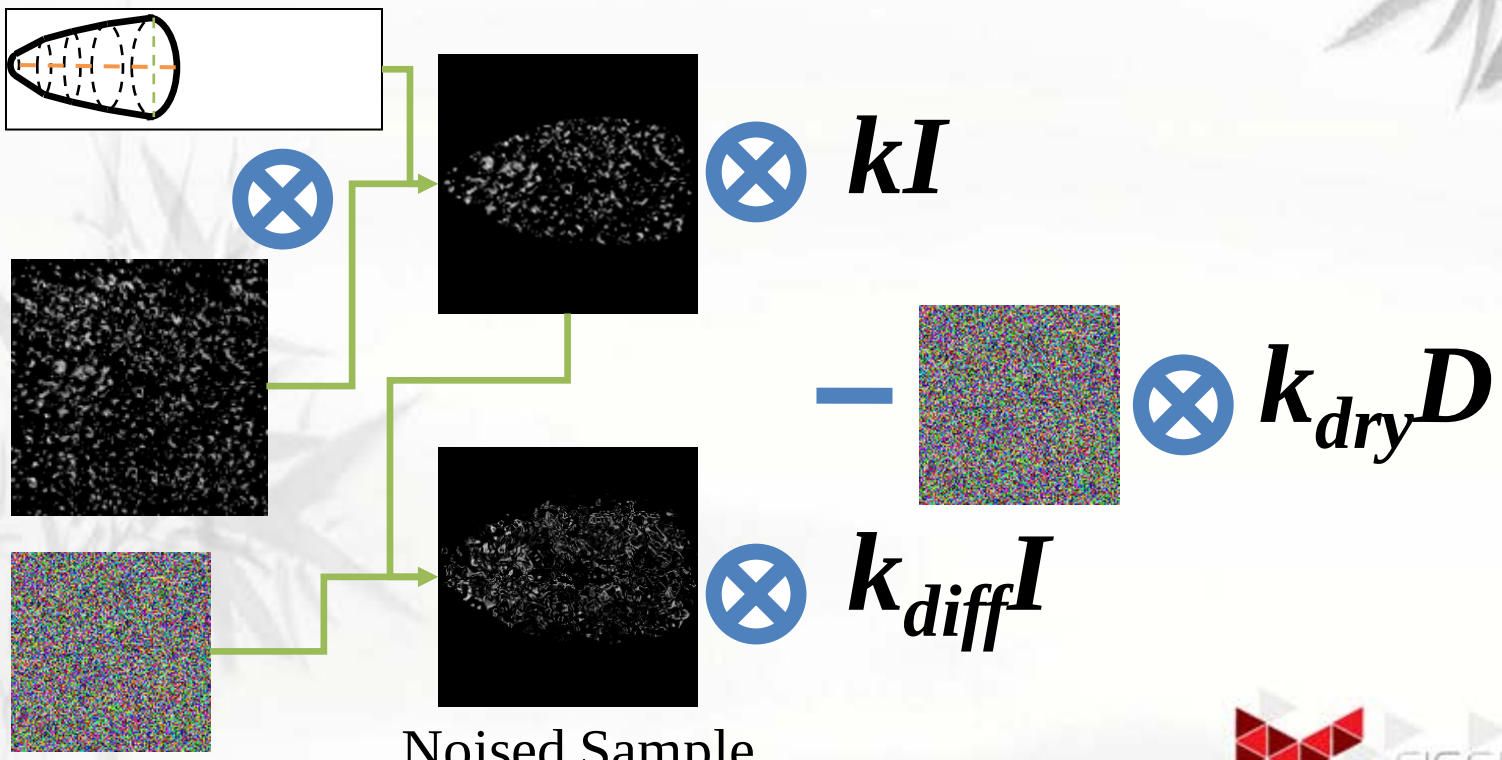
Flying white effect

Footprint Mask



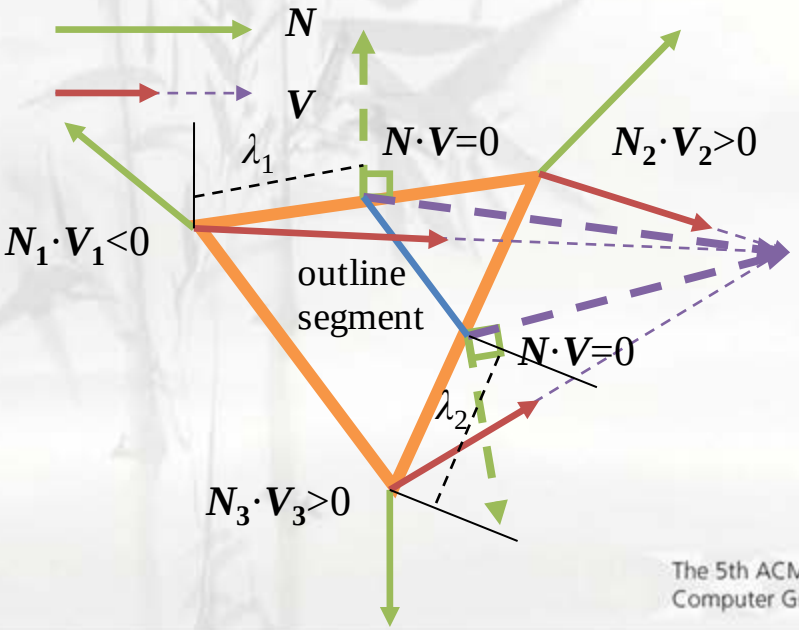


Ink Footprint Generation



Contouring

- **Stroke extraction**
  - Intrinsic contour: crease & boundary
    - Preprocessing: traverse and find edge with no shared triangles
  - Silhouette: per-triangle detection [Hertzmann and Zorin 2000]
    - Solve $[(1-\lambda)N_1 + \lambda N_2] \cdot [(1-\lambda)V_1 + \lambda V_2] = 0$

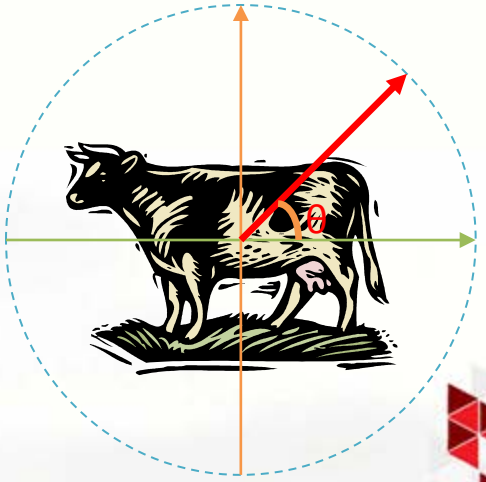


$$\lambda = \begin{cases} \dfrac{\vec{N}_1 \cdot \vec{V}}{\vec{N}_1 \cdot \vec{V} - \vec{N}_2 \cdot \vec{V}}, & \left(\vec{N}_1 \cdot \vec{V}\right)\left(\vec{N}_2 \cdot \vec{V}\right) \le 0 \\ undefined, & otherwise \end{cases}$$

$$\vec{P}_{silhouette} = (1-\lambda)\vec{P}_1 + \lambda \vec{P}_2$$

Contouring (Cont')

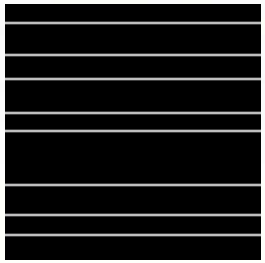
- ***PADI* parameterization**
  - Pressure
    - Circular mapping (similar to sphere environmental mapping)
    - Trigonometric function $f(\theta) = A\sin(\omega\theta + \varphi) + T$
  - Angle
    - Along the silhouette segment
    - Orient to increasing direction of θ
  - Dryness
    - Medium value
  - Itensity
    - Heavy value
- **Stroke visibility**
  - Depth testing with Z-bias



$A\sin(\omega\theta+\varphi) + T$

Wrinkling (Texturing)

- **Stroke extraction**
  - Using stripe mask s by texture mapping



- ***PADI* parameterization**
  - Sample the intrinsic diffuse map
  - Convert sampled value to grey value G
  - $\rho = 1 - G$
  - $h(\boldsymbol{N} \cdot \boldsymbol{V})$: adjusting function (from silhouette to center)
  - k, m (medium), q (high), α, β, γ: scalar and exponential factors
  - Pressure: $P = [k\rho h(\boldsymbol{N} \cdot \boldsymbol{V})]^{\alpha}$
  - Intensity: $I = (m\rho)^{\beta} s$
  - Dryness: $D = [qGh(\boldsymbol{N} \cdot \boldsymbol{V})]^{\gamma}$
  - Angle: gradient of $h(\boldsymbol{N} \cdot \boldsymbol{V})$

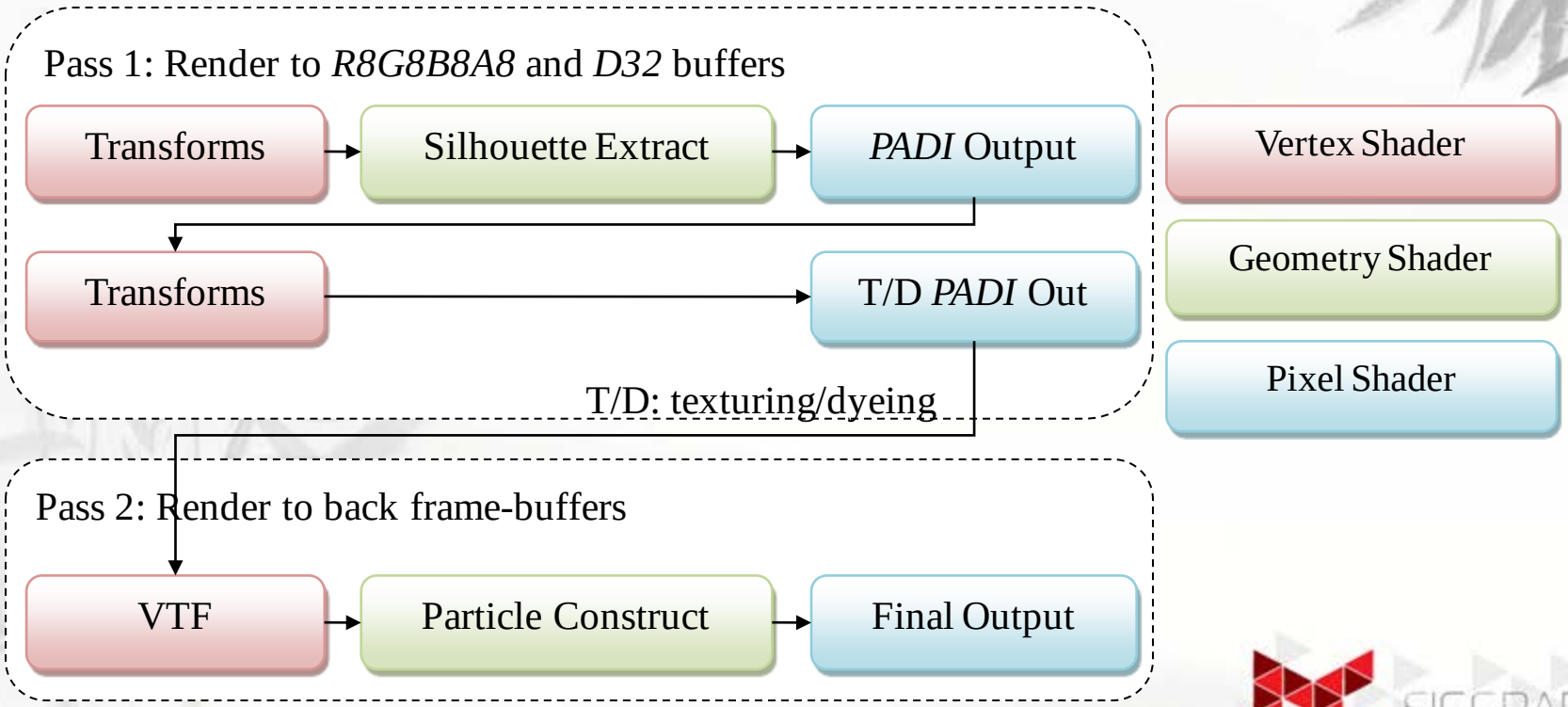
Coloring (Dyeing)

- **Stroke extraction**
  - Shade directly
- ***PADI* parameterization**
  - Deploy Phong illumination model
  - Convert illuminated value to grey value G
  - Pressure, dryness, angle: similar to wrinkling
  - Intensity: $I = (m\rho)^{\beta}$
    - Without stripe mask
    - To fill the remaining area
  - m: low value
  - q: low value

Dotting

- **Stroke extraction**
  - Ratio of size and depth < threshold
  - Convert the entire mesh to one particle
  - Similar to some strategies of LOD
- ***PADI* parameterization**
  - Pressure, dryness, intensity: predefined constants
  - Angle: the ratio of width and height

Rendering Pipeline

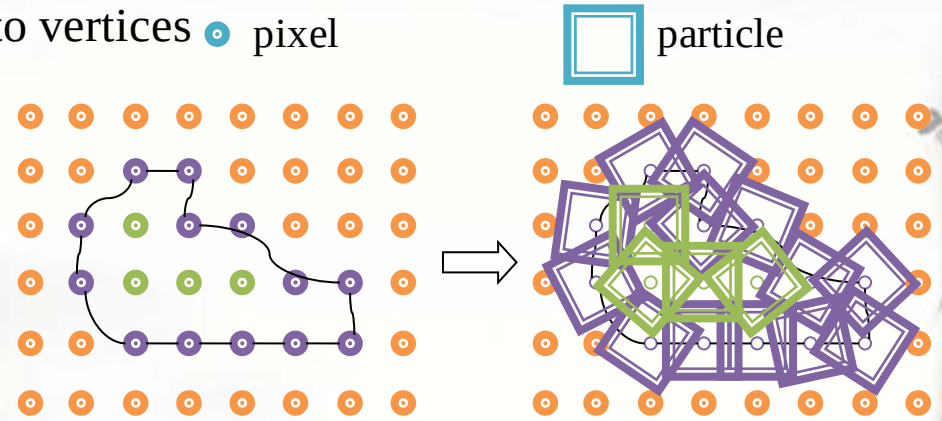


Rendering Pipeline (Cont')

- **Pass 1**
  - Contouring
    - Vertex shader
      - Geometric transformations
      - Circular mapping
    - Geometry shader
      - per-triangle silhouette detection
      - Triangle in and line out
    - Pixel shader: *PADI* output
  - Wrinkling & coloring
    - Vertex shader: geometric transformations
    - Pixel shader:
      - Wrinkle stroke extraction (texture mapping)
      - *PADI* calculation for wrinkle and coloring strokes respectively

Rendering Pipeline (Cont")

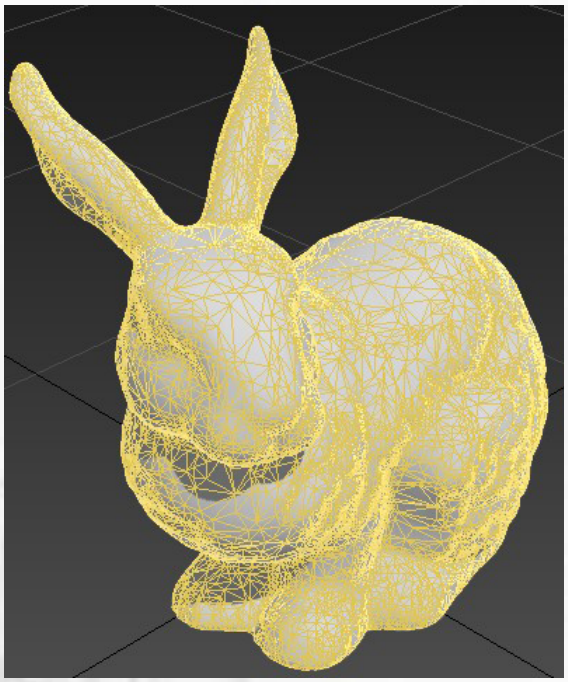
- **Pass 2**
  - Emit vertices to fill the screen
  - Vertex shader: vertex texture fetch
    - Store *PADI* data from pixels to vertices
  - Geometry shader (point mode)
    - Filter vertices
    - Vertex → particles
  - Pixel shader
    - Footprint mask mapping
    - Calculate the final ink intensity of footprint
      - $\alpha = (kF + k_{diff}F_{diff})I - k_{dry}\xi D$

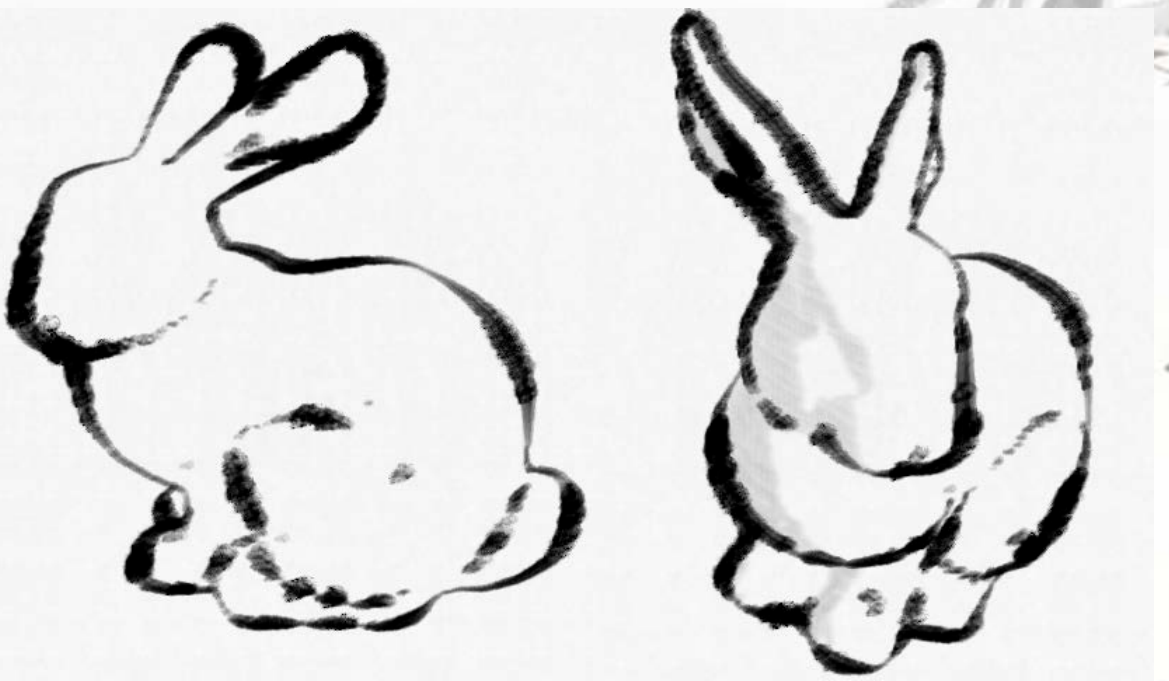


Results

- **Testing machine**
  - Intel(R) Core(TM) 2 Duo CPU P7450
  - NVIDIA(R) Geforce GT240M
- **Input geometric model**
  - Ordinary 3D triangle meshes
  - With texture coordinates
  - With/without diffuse texture
  - Static mesh/deformable mesh
- **Output image**
  - Rendered ink wash stylistic image

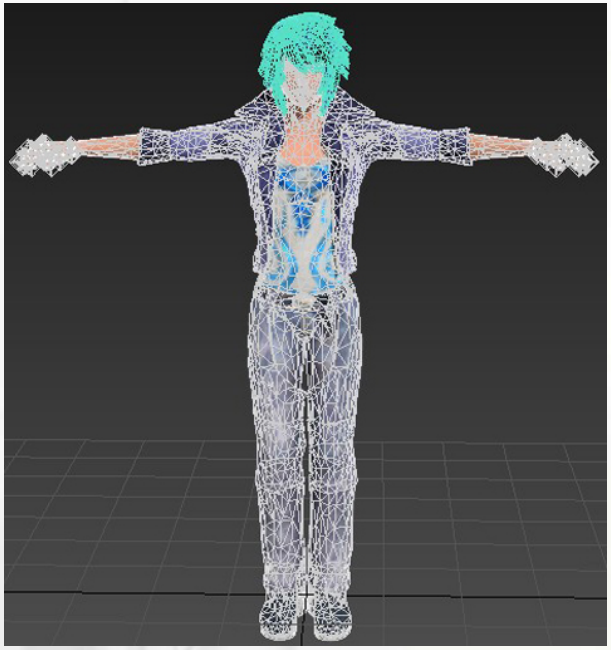
Results (Cont')





- **Bunny rendering**

Results (Cont”)



- **Deformable character rendering**

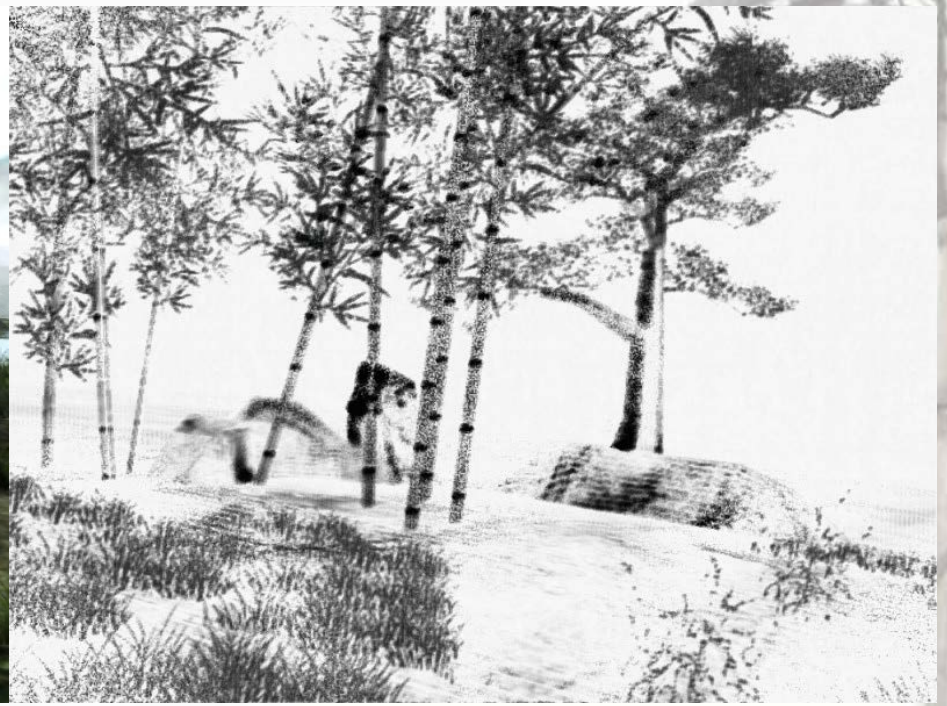
Results (Cont'")

- **Character rendering (different parameters)**
  - Wrinkling in thick ink
  - Wrinkling in light ink

Results (Cont'4)

- **Character rendering**
  - With coloring strokes

Results (Cont'5)



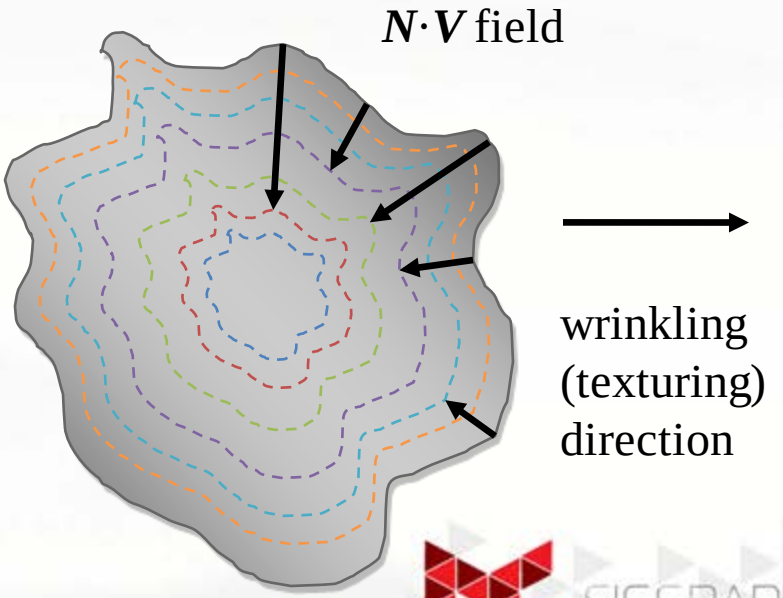
- **Complex scene rendering**

Summary

- **Generate ink wash stylistic images**
  - For fast 3D mesh rendering in real-time
  - Stroke extraction based on ink wash techniques
  - Ink rendering with ink wash features
  - GPU accelerated rendering pipeline

Summary (Cont')

- **Future work**
  - Improve stroke parameterization
    - Fast stroke tracing
    - Procedurally mapping
      - Scalar/vector field visualization
  - Improve stability
    - Frame coherence
    - Better animation
  - More immersive ink
    - Dynamics (Fluid simulation, etc.)



Acknowledgements

- The associate editor and all the referees
- National Basic Research 973 program of China (2009CB320802, 2011CB302801)
- National Natural Science Foundation of China (60833007)
- The grant of University of Macau

The End
Thank you!
Q & A

SIGGRAPH ASIA 2012

The 5th ACM SIGGRAPH Conference and Exhibition on Computer Graphics and Interactive Techni ques in Asia