OLM Digital Inc.

CGで用いるいくつかの補間手法と
その数学的背景

安生健一
オー・エル・エム・デジタル

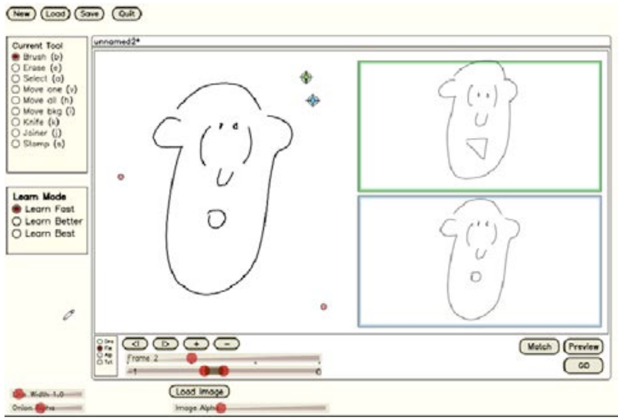
第10回「計算機を用いた数学研究セミナー」GCOE セミナー
コンピュータグラフィックスと数学の接点
京都大学理学部3号館 110講義室
2012.07.21

PSD, NPR Shading, Stereoscopic 3D

Application Examples

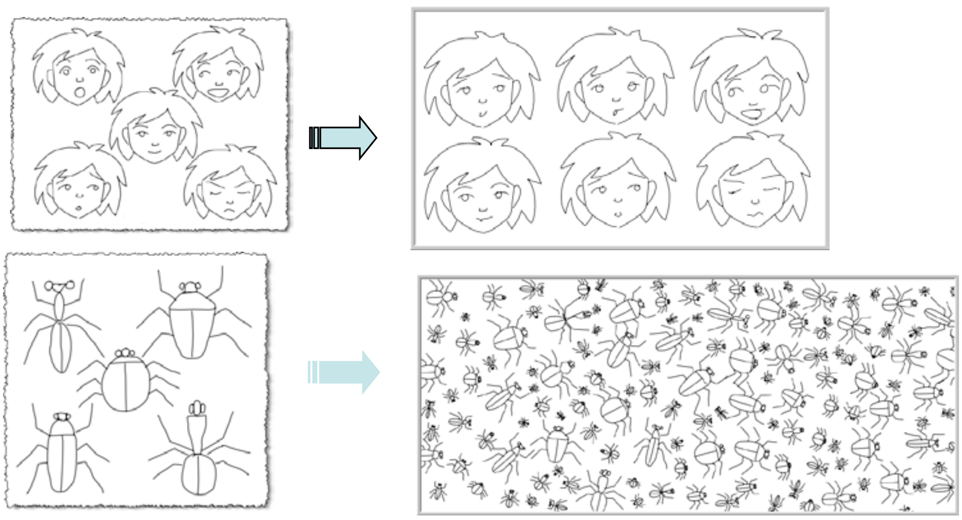
Learning Doodle by Example

Application Examples



Problem Definition

- Given *N* similar input drawings (doodles)
- Construct more doodles that resemble the *N* inputs
  - Example results



Proposed Solution

- Construct a space of doodles defined by the inputs
  - Use results from machine learning and statistics:
    - Consider drawings as sample points in some space.
    - Similar doodles should be located nearby in the space.
    - We wish to fit (or learn) a continuous function over the space that “explains” the examples as well.
- We call the result a “Latent Doodle Space”(LDS).

‣ See the paper in EUROGRAPHICS06 (by Baxter and Anjyo) for details.

Two Main Challenges

To build a latent doodle space (LDS):

- Find correspondences between two line drawings.
  - Hard problem. No perfect solution.
    - ‣ Do best possible, but still must have a good UI.
- Generate the space of similar drawings.
  - Use Bayesian techniques and statistical methods to improve results.

How to construct the LDS?

To generate the space of similar drawings (LDS):

- Two main tasks :
  - Finding latent coordinate system ⇐ PCA
  - Interpolating within LDS ⇐ RBF
- Three options in the EG06 paper: PCA+RBF, PCA+GP, GPLVM

➡ This talk focuses on the first strategy: PCA+ RBF

Dimension reduction by PCA

- A line drawing (doodle) has the structure:
  - line drawing l_k - stroke s_p - linear segments a_v^p
- After establishing correspondences among line drawings:
  - Construct the data matrix X (with zero-mean):

(x, y)

$$\begin{array}{r l}
 & \begin{array}{c c} \overbrace{\hspace{10em}}^{\textit{stroke } s_1} & \overbrace{\hspace{10em}}^{\textit{stroke } s_2} \end{array} \\
\begin{array}{r} \textit{line drawing } l_1 \rightarrow \\ \textit{line drawing } l_2 \rightarrow \\ \\ \end{array} & \left(\begin{array}{cccccccc} a^1{}_0 & a^1{}_1 & \cdots & a^1{}_{s_1} & b^1{}_0 & \cdots & b^1{}_{s_2} & \cdots \\ a^2{}_0 & \cdots & & a^2{}_{s_1} & b^2{}_0 & \cdots & b^2{}_{s_2} & \\ \vdots & & & & & & & \end{array} \right) = X
\end{array}$$

  - Applying PCA: Eigen decomposition of the covariance matrix $X^T \cdot X$

2-d LDS and RBF

- Make the LDS 2-dim by taking the eigenvectors with the first two largest eigenvalues.

- Interpolate each of s, t (scalar) values of the drawing samples by *thin plate spline*:

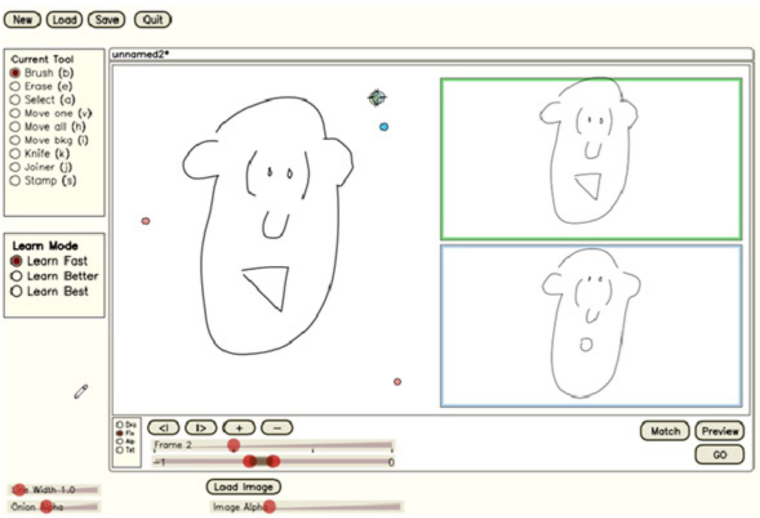
  - space dimension n = 2 and smoothness m = 2 (explain later!)

  - The spline function is: $\phi(r) = r^2 \log r$ where $r := \|(s,t)\| = \sqrt{s^2 + t^2}$

  - The interpolant is of the form:

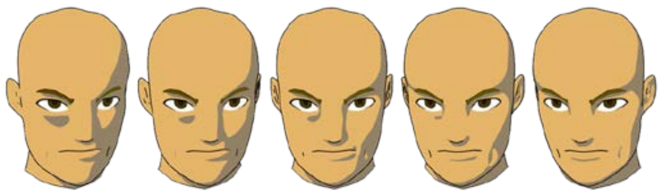
$$f(s,t) := \sum_i w_i \phi(s - s_i, t - t_i) + as + bt + c$$

Demo



Locally Controllable Stylized Shading

Application Examples



Background: Cartoon shading

- Based on thresholded **N•L** shading model

N•L intensity distribution

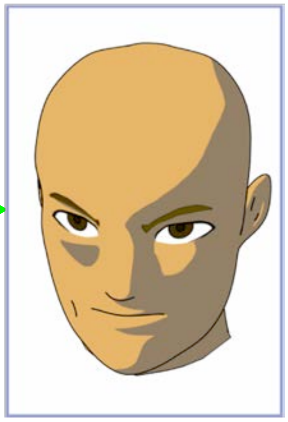
Motivation: Fake but expressive shading

With conventional shader

Artists want neat shading

Our Goal

conventional



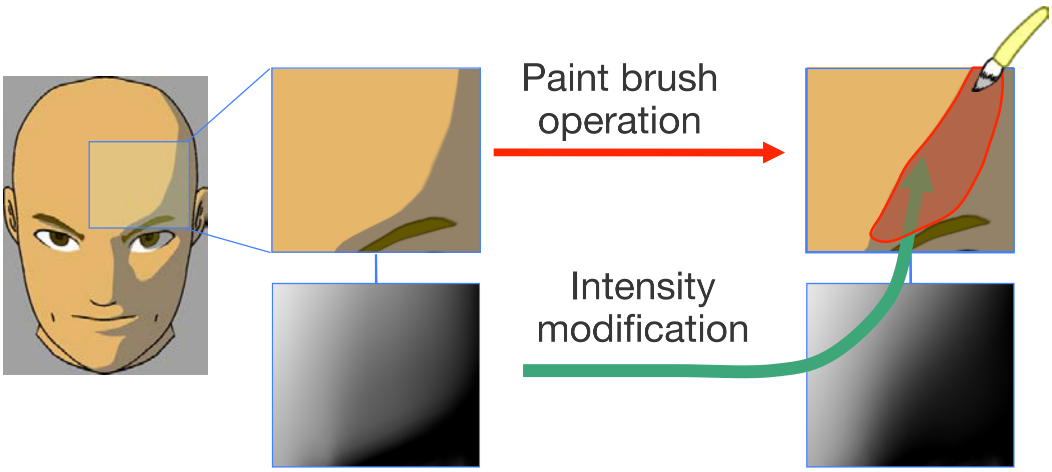
desired

- Make toon shader artist-friendly
  - Edit undesirable shaded area
  - Add artistic light and shade to original 3D lighting

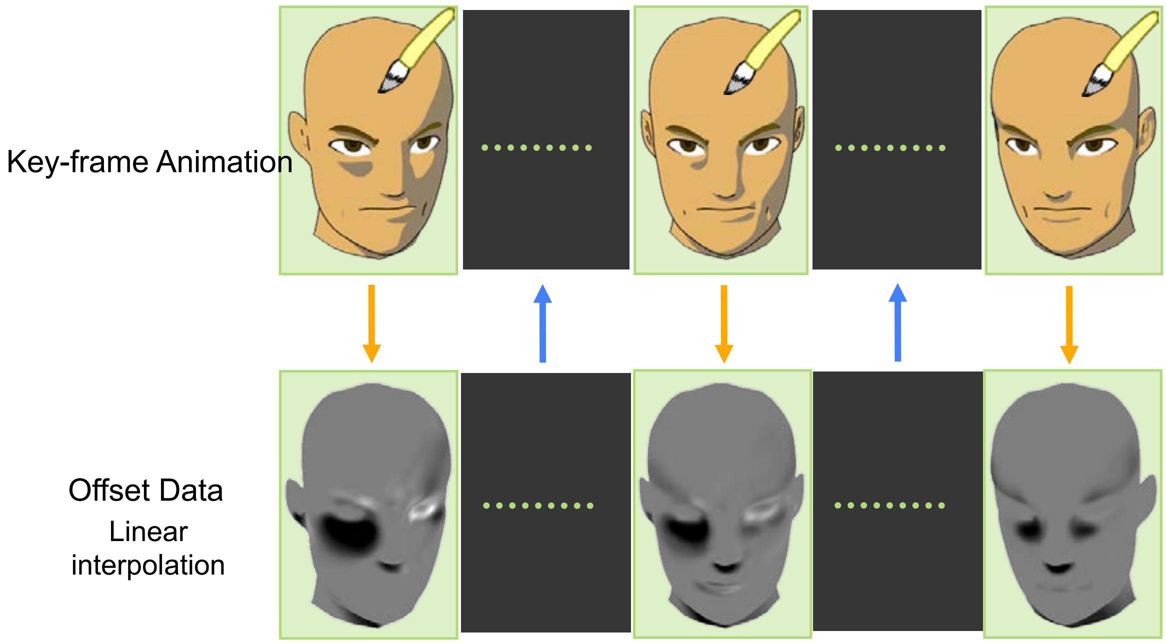
Video

The shade painter (our approach)

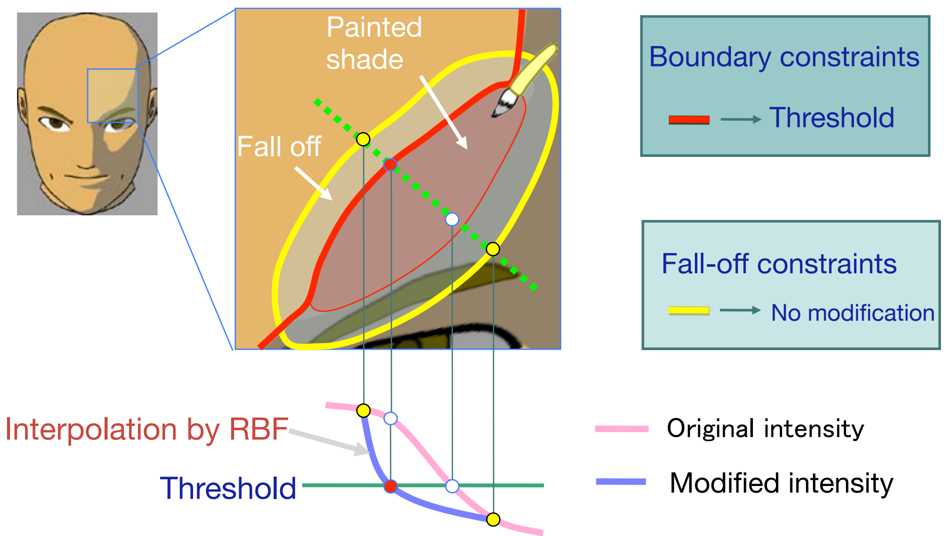
- The main idea:
  Modify intensity according to painted strokes



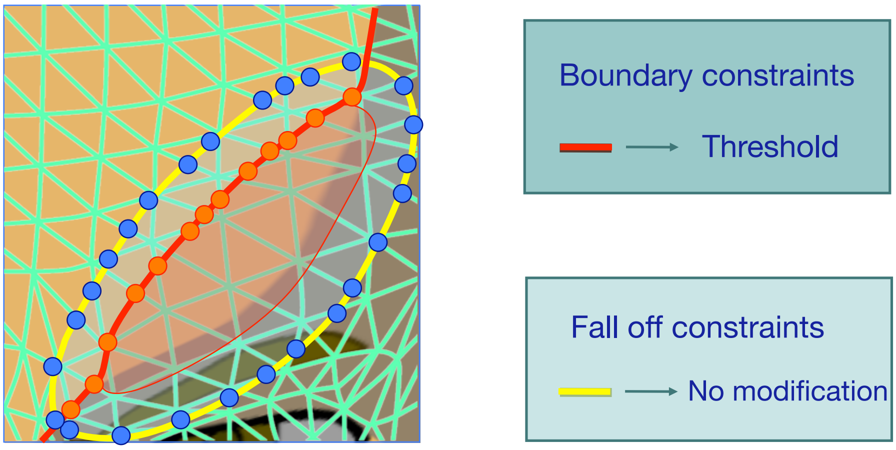
Keyframing



Intensity modification by painting



RBF interpolation



- The discretized constraints for the RBF are specified for the points ● and ●.

The unknown values

- We employ the following interpolation function:

$$f(x) = \sum_{i=1}^{l} w_i \phi(x - \mathrm{c}_i) + P(x)$$

where $x = (x_1, x_2, x_3)$, and $\phi(x) := \|x\| \equiv \sqrt{x_1^2 + x_2^2 + x_3^2}$; $P(x)$ is a linear polynomial of x_1, x_2, x_3; c_i means the constraint points (● and ●); l is the number of all the constraint points.

- We want to determine the weights $\{w_k\}$ and the four coefficients of $P(x)$

  → Totally $l + 4$ unknown values.

For the given values $h_{j\ (1 \le j \le l)}$, we have:

$$\sum_{i=1}^{l} w_i \phi(\mathrm{c}_j - \mathrm{c}_i) + P(\mathrm{c}_j) = h_j, \quad \mathit{for}\ 1 \le j \le l.$$

The unknown values

- We further add the *vanishing moment* condition:

$$\sum_{i=1}^{l} w_i Q(\mathrm{c}_i) = 0. \quad \text{for any linear polynomial Q.}$$

→ Taking $\mathrm{Q} = 1, x_1, x_2,$ and x_3 , we know that the above condition means:

$$\sum_{i=1}^{l} w_i = 0 \qquad \sum_{i=1}^{l} w_i c_{ij} = 0, \ (j = 1, 2, 3)$$

The linear equation for RBF

- By putting $P(x) = p_0 + p_1x_1 + p_2x_2 + p_3x_3$ and $\phi_{ij} := \phi(\mathrm{c}_i - \mathrm{c}_j)$, we have the following *linear* equation:

$$\begin{pmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1l} & 1 & c_{11} & c_{12} & c_{13} \\ \phi_{21} & \ddots & & & \vdots & \vdots & \ddots & \vdots \\ \vdots & & & & & & & \\ \phi_{l1} & & & \phi_{ll} & 1 & c_{l1} & c_{l2} & c_{l3} \\ 1 & 1 & \cdots & 1 & 0 & 0 & 0 & 0 \\ c_{11} & & \cdots & c_{l1} & \vdots & \ddots & & \vdots \\ c_{12} & & \cdots & c_{l2} & \vdots & & \ddots & \vdots \\ c_{13} & & \cdots & c_{l3} & 0 & \cdots & \cdots & 0 \end{pmatrix} \begin{pmatrix} w_1 \\ \\ \\ w_l \\ p_0 \\ p_1 \\ p_2 \\ p_3 \end{pmatrix} = \begin{pmatrix} h_1 \\ \vdots \\ h_l \\ 0 \\ \vdots \\ \vdots \\ 0 \end{pmatrix}$$

Locally Controllable Stylized Shading

System Overview

Locally Controllable Stylized Shading

Example Animations