Poisson Image Editing

Presented by Yunjun Zhang

How to write a paper for SIGGRAPH

- The applications are plentiful and the result is amazing.
- Idea could be simple but it really works and it is robust.
- The method is explained in good detail without too much fancy formulas.
Definitions

- Poisson’s Equation: \[ \Delta F = 4\pi \rho \]
- Laplacian’s Equation: \[ \Delta F = 0 \]
- Dirichlet boundary condition: specify the value of the function on a surface.

Notations used in paper and this presentation

- \( f \): unknown scalar image.
- \( f^* \): known scalar image.
- \( g \): known scalar image of an object which will be cloned.
- \( \Omega \): the selected area.
- \( \partial \Omega \): the boundary of the area.
**Contribution of this paper**

- Based on membrane interpolation
  \[ \min f \int \int_{\Omega} \| \nabla f \|^2 \text{ with } f \big|_{\partial \Omega} = f^* \big|_{\partial \Omega} \]
  \[ \Delta f = 0 \text{ over } \Omega, \text{ with } f \big|_{\partial \Omega} = f^* \big|_{\partial \Omega} \]

- Extend the above minimization with a vector field \( v \)
  \[ \min f \int \int_{\Omega} \| \nabla f - v \|^2 \text{ with } f \big|_{\partial \Omega} = f^* \big|_{\partial \Omega} \]

- Solution is a Poisson Equation with a boundary condition
  \[ \Delta f = \text{div}(v) \text{ over } \Omega, \text{ with } f \big|_{\partial \Omega} = f^* \big|_{\partial \Omega} \]

**Implementation**

- \( N_p \) is the set of 4-connected neighbors which are in the image, \(|N_p|\) is the Count Number. __4

- Discrete minimization function
  \[
  \min f \sum_{p,q \in N_p \cap \Omega} (f_p - f_q - v_{pq})^2, \text{ with } f_p = f^*_p, \text{ for all } p \in \Omega
  \]
  \[
  \text{for all } p \in \Omega, |N_p| f_p - \sum_{q \in N_p \cap \Omega} f_q = \sum_{q \in N_p \cap \Omega} f^*_q + \sum_{q \in N_p} v_{pq}
  \]
Implementation

- Gauss-Seidel iteration

$$\hat{g}_i^{(k+1)} = \sum_{j \neq i} C_{ij} g_j^{(k)} = \sum_{j \neq i} C_{ij} g_j^{(k-1)}$$

Explanation of the formulas

- Assume we work on one dimensional image
Applications

- Seamless cloning for opaque objects

\[ v = \nabla g \]
\[ \Delta f = \Delta g \text{ over } \Omega, \text{ with } f|_{\partial \Omega} = f^*|_{\partial \Omega} \]

- The texture is maintained, but not necessarily for the color.
Applications

- Seamless cloning for objects with transparent parts or holes
  \[ v(x) = \begin{cases} \nabla f^*(x) & \text{if } |\nabla f^*(x)| > |\nabla g(x)| \\ \nabla g(x) & \text{otherwise} \end{cases} \]
- Mixing gradient on different location can not really handle the transparency. Since a pixel with transparent foreground and opaque background should be a combinational value.
Applications

- Selection Editing
  - Texture flattening
    The vector field has value only on edge pixel. Thus the detailed texture information is removed and better segmentation can be achieved.
Applications

- Selection Editing
  - Local illumination changes
    - The gradient field is enlarged or reduced by logarithm. But the texture itself will not be changed.
Problems ? !

- Color intensity of the cloned object may be changed by the boundary color value.
- Check this poor ‘pink’ seagull out.
Problems

- Independency on color channels?
  - The author claims it is true, but we need a formal prove.
  - Recall something we discussed in last paper

Extended Usage

- Inpainting
  - Set the vector field to zeros. Then apply the method.
  - Can’t handle textured area.
Extended Usage

Shadow Removal
- The problem is how to handle the edge of the shadow gradient field.