Most Frequently Used PDE

- Wave Equation

\[ \frac{\partial^2 u}{\partial t^2} = c^2 \nabla^2 u + f(t, x, y, z) \]

- Heat Equation

\[ \frac{\partial u}{\partial t} = c^2 \nabla^2 u + f(t, x, y, z) \]
Most Frequently Used PDE

- Poisson Equation, Steady State of Wave Equation and Heat Equation

\[ \nabla^2 u = f(x, y, z) \]

- Laplace’s Equation

\[ \nabla^2 u = 0 \]
Boundary Conditions

- **Dirichlet Boundary Conditions**
  - Specify the value of the function on a surface

- **Neumann Boundary Condition**
  - Specify the normal derivative of the function on a surface
Guided Interpolation

- $f$: to be solved, $f^*$: known region
- $v$: guided field, $g$: $v$ is prob. gradient of $g$
Simple Interpolation

- Maximize the Smoothness

\[
\begin{aligned}
\min_{f} \int_{\Omega} \left\| \nabla f \right\|^2 \\
\quad \text{subject to } f|_{\partial \Omega} = f^*|_{\partial \Omega}
\end{aligned}
\]

- Solution: Laplace Equation with Dirichlet Boundary Conditions

\[
\begin{aligned}
\nabla^2 f &= 0 \\
\quad \text{subject to } f|_{\partial \Omega} = f^*|_{\partial \Omega}
\end{aligned}
\]
Example: Laplace’s Equation. The functional corresponding to Laplace’s equation is

\[ I(\phi) = \frac{1}{2} \int_V \|\nabla \phi\|^2 dV \]

\[ = \frac{1}{2} \int_V \left( \phi_x^2 + \phi_y^2 \right) dx \, dy \] (7.16)

in the 2D case. The factor of 1/2 is included so that the functional can be identified as the energy stored by the field \( \phi \) (for a particular physical problem there may also be an additional constant to make the units work out).

The Euler-Lagrange equation is

\[ \frac{1}{2} \frac{\partial}{\partial \phi} \left( \phi_x^2 + \phi_y^2 \right) - \frac{1}{2} \frac{\partial}{\partial \phi_x} \phi_x^2 - \frac{1}{2} \frac{\partial}{\partial \phi_y} \phi_y^2 = 0 \] (7.17)

Working the partial derivatives gives

\[ 0 - \frac{\partial}{\partial x} \phi_x + \frac{\partial}{\partial y} \phi_y = 0 \] (7.18)

Notice that the partial derivatives are evaluated symbolically as if \( \phi, \phi_x, \) and \( \phi_y \) were all independent variables. This can be understood intuitively by thinking of forming a parameterized variation of each of these functions with respect to a different small parameter and then using (7.10) separately for each function. Since the subscripts on \( \phi_x \) and \( \phi_y \) represent partial derivatives, Eq. (7.18) becomes

\[ \frac{\partial \phi}{\partial x^2} + \frac{\partial \phi}{\partial y^2} = 0 \] (7.19)

which is Laplace’s equation. The functional can easily be modified to include a source term, so that Poisson’s equation is obtained.
Guided Interpolation

- Interpolation $\rightarrow$ minimization
\[
\min_f \int\int_\Omega \| \nabla f - \mathbf{v} \|^2
\]
\[f|_{\partial\Omega} = f^*|_{\partial\Omega}
\]
- Solution: Poisson Equation with Dirichlet Boundary Conditions
\[
\nabla^2 f = \nabla \cdot \mathbf{v}
\]
\[f|_{\partial\Omega} = f^*|_{\partial\Omega}
\]
- Relationship with Laplace case?
Or from vector field decomposition

- Helmholtz-Hodge decomp.

\[ w = \nabla \phi + \nabla \times v + h \]

\[ \min_{\phi} \int \int_T \| \nabla \phi - w \|^2 dA, \]

\[ \nabla^2 \phi = \nabla \cdot w. \]
Discrete Poisson Solver

- Discretize the Minimization Directly
  \[
  \min_{f|\Omega} \sum_{\langle p,q \rangle \cap \Omega \neq \emptyset} (f_p - f_q - v_{pq})^2 \text{ with } f_p = f_p^* \text{ for } \forall p \in \partial \Omega
  \]

- Partial Derivative
  \[
  \forall p \in \Omega, \ |N_p| f_p - \sum_{q \in N_p \cap \Omega} f_q = \sum_{q \in N_p \cap \partial \Omega} f_q^* + \sum_{q \in N_p} v_{pq}
  \]

- Partial Derivative for Interior Points
  \[
  |N_p| f_p - \sum_{q \in N_p} f_q = + \sum_{q \in N_p} v_{pq}
  \]
Discrete Poisson Solver

- Linear System of Equations
- Gauss-Seidel Method with Successive Overrelaxation
- V-cycle Multigrid
- Discretize Laplacian with Discrete Laplacian of Gaussian

- [http://www.tau.ac.il/~stoledo/taucs/] Taucs
Seamless Cloning: Importing Gradients

- Importing Gradients from a Source Image
  \[ g \]
  \[ \nabla g \]
- Discretize

for all \( (p, q) \), \( v_{pq} = g_p - g_q \)
Seamless Cloning Results
Seamless Cloning Results

Texture

Alignment
Transfer intensity only

source/destination

color transfer

monochrome transfer
Seamless Cloning: Mixing Gradients

- Two Proposals
  - Define \( v \) as Linear Combination of Source and Destination Gradients
  - Select Stronger one from Source and Destination Gradients (not conservative!)

  \[
  \text{for all } x \in \Omega, \quad v(x) = \begin{cases} 
  \nabla f^*(x) & \text{if } |\nabla f^*(x)| > |\nabla g(x)| \\
  \nabla g(x) & \text{otherwise}.
\end{cases}
\]

- Discretization

  \[
  v_{pq} = \begin{cases} 
  f^*_p - f^*_q & \text{if } |f^*_p - f^*_q| > |g_p - g_q| \\
  g_p - g_q & \text{otherwise},
\end{cases}
\]
Mixing Gradients Results

(a) color-based cutout and paste
(b) seamless cloning
(c) seamless cloning and destination averaged
(d) mixed seamless cloning
Mixing Gradients Results

source

destination
Mixing Gradients Results

source/destination  seamless cloning  mixed seamless cloning

Just reduce terrible things! Another solution
Texture Flattening

- Remain Only Salient Gradients

\[ \text{for all } x \in \Omega, \ v(x) = M(x) \nabla f^*(x) \]

- Discretization

\[ v_{pq} = \begin{cases} 
  f_p - f_q & \text{if an edge lies between } p \text{ and } q \\
  0 & \text{otherwise}, 
\end{cases} \]
Texture Flattening

Edge mask
Local Illumination Changes

- Fattal Transformation

\[ \mathbf{v} = \alpha^\beta \left| \nabla f^* \right|^{-\beta} \nabla f^* \]
Local Color Changes

- Mix two different colored version of original image
  - One provide $f^*$ outside
  - One provide $g$ inside
Local Color Changes

Monochrome of background
White in flower
Seamless Tiling

- Select original image as $g$
- Boundary condition:
  - $f^*_{\text{north}} = f^*_{\text{south}} = 0.5(g_{\text{north}} + g_{\text{south}})$
  - Similarly for the east and west
Seamless Tiling
Discussion
Discussion
Discussion
Discussion
Drag-and-Drop Pasting

SIGGRAPH 2006

Leo Jiaya Jia  The Chinese University of Hong Kong
Jian Sun      Microsoft Research Asia
Chi-Keung Tang The H.K. University of Sci. & Tech.
Heung-Yeung Shum Microsoft Research Asia

Slides by the authors
Introduction to our method

- Our method improves the Poisson image editing with
  - a new boundary optimization algorithm,
  - an easier user interface,
  - and an integration of alpha values.
Poisson equations in images

A case study

\[ f_s + f_t \]
Poisson equations in images

A case study
Poisson equations in images

The optimization problem in image blending [Perez et al. 2003]

\[ \min_f \int_{p \in \Omega} | \nabla f - \nabla f_s |^2 \, dp \quad \text{with} \quad f |_{\partial \Omega} = f_t |_{\partial \Omega} \]

Taking \( f' = f - f_s \), we have

\[ \min_{f'} \int_{p \in \Omega} | \nabla f' |^2 \, dp \quad \text{with} \quad f' |_{\partial \Omega} = f_t - f_s |_{\partial \Omega} \]

What does it imply?
Poisson equations in images

The minimization problem equals to solving the Laplace equation:

\[ \Delta f' = 0 \text{ with } f'\big|_{\partial \Omega_0} = f_t - f_s \big|_{\partial \Omega_0} \]

Image blending should take both the source and the target images into consideration.

Property of solving the Laplace equation:

The variational energy \( \int_{\Omega_0} |\nabla f'|^2 \) will approach zero if and only if all boundary pixels satisfy \( (f_t - f_s)\big|_{\partial \Omega_0} = k \), where \( k \) is a constant value.
Poisson equations in images

- Where is the optimal boundary $\partial \Omega$?
  - Inside the user drawn region
  - Outside the object of interest

- How is the object extracted?
  - Lazy snapping or Grabcut [Rother et al. 2004] (September 23)

$$\min \sum_{p \in \partial \Omega} ((f_r(p) - f_s(p)) - k)^2, \text{ s.t. } \partial \Omega \in \text{blue}$$

- How to optimize it?
  - Minimum color variance
Boundary optimization

\[ E(\partial \Omega, k) = \sum_{p \in \partial \Omega} \left( (f_i(p) - f_s(p)) - k \right)^2, \text{ s.t. } \partial \Omega \in \text{blue} \]

- \( \partial \Omega \) and \( k \) are all unknowns
- An iterative optimization
  - Initialize \( \partial \Omega \) as the user drawn boundary.
  - Given new \( \partial \Omega \), the optimal \( k \) is computed:
    \[ \frac{\partial E(\partial \Omega, k)}{\partial k} = 0 \]
  - Given new \( k \), optimize the boundary \( \partial \Omega \).
  - Repeat the previous two steps until convergence.
Boundary optimization

- In 2D graph, computing the shortest path between any two points: Dynamic Programming

- Our problem is to compute a closed path
Boundary optimization

A shortest closed-path algorithm

Breaking closed boundary
Boundary optimization

A shortest closed-path algorithm
Boundary optimization

A shortest closed-path algorithm

Computation complexity $O(N)$
Boundary optimization

A shortest closed-path algorithm

- Total computation complexity $O(NM)$
Boundary optimization discussion

- **Optimality**
  - Avoiding that the path twists around the cut by selecting the initial cut position.

- **How to select the initial cut?**
  - Making it short to reduce $O(MN)$
  - Passing smooth region
One example
Integrating fractional boundary

- Fractional boundary is important in image composting: (transparency)
Matting

- Composition Image $I$ is generated by foreground and background with alpha matte
- $I = \alpha F + (1-\alpha)B$
- Matting is a problem to get $\alpha, F, B$ from a given image $I$.

User have to divide the image into three regions: Foreground, Background and Unknown area. In foreground area, $F = I$, $\alpha = 1$, $B = 0$; In background area, $F = 0$, $\alpha = 0$, $B = I$. Our task is to get $F, B, \alpha$ in unknown area.
Integrating fractional boundary

- Where to use the fractional values?
  - only the pixels where the optimized boundary is near the blue ribbon
Integrating fractional boundary

- Where to use the fractional values?
  - only the pixels where the optimized boundary is near the blue ribbon

  fractional integration:
  - the green region
  otherwise:
  - the yellow region
Integrating fractional boundary

How to integrate the fractional values in Poisson blending?

- A blended guidance field

\[ \nabla'_{x}(x, y) = \begin{cases} 
\nabla_{x}f_{s}(x, y) & (x, y), (x+1, y) \in \text{yellow} \\
\nabla_{x}(\alpha f_{s} + (1-\alpha)f_{t}) & (x, y), (x+1, y) \in \text{green} \\
0 & \text{otherwise}
\end{cases} \]
Integrating fractional boundary

How to integrate the fractional values in Poisson blending?

A blended guidance field

\[
v'_x(x, y) = \begin{cases} 
\nabla_x f_s(x, y) & (x, y), (x+1, y) \in \text{yellow} \\
\n\nabla_x (\alpha f_s + (1-\alpha)f_t) & (x, y), (x+1, y) \in \text{green} \\
0 & \text{otherwise}
\end{cases}
\]

\[\n\]
Integrating fractional boundary

How to integrate the fractional values in Poisson blending?

A blended guidance field

\[ v'_x(x, y) = \begin{cases} \nabla_x f_s(x, y) & (x, y), (x + 1, y) \in \text{yellow} \\ \nabla_x (\alpha f_s + (1 - \alpha)f_t) & (x, y), (x + 1, y) \in \text{green} \\ 0 & \text{otherwise} \end{cases} \]
Integrating fractional boundary

- Final minimization:
  \[ \min_f \int_{p \in \Omega^*} |\nabla f - v'|^2 \, dp \quad \text{with} \quad f |_{\partial \Omega^*} = f_t |_{\partial \Omega^*} \]

- New boundary: \( \Omega \cup \Phi \)

- Solving the corresponding Poisson equation.
Results and comparison

Alpha blending
Results and comparison

Our method

Poisson blending
Results and comparison

Our method  Alpha blending
Results and comparison

Our method

Poisson blending
Poisson Image Editing Extended

SIGGRAPH 2006 sketch

Daniel Leventhal Brown Univ.
Bernard Gordon Brown Univ.
Peter G. Sibley Brown Univ.
Problem caused by texture
Alpha control

- Alpha in $[0, 1]$, background 0, and foreground 1.
- Obtain unselected area’s Alpha by blurring.

\[ \forall x \in \Omega, v(x) = \begin{cases} 
\nabla f^*(x) & \text{if } ||\nabla f^*(x)|| > \alpha ||\nabla g(x)|| \\
\alpha \nabla g(x) & \text{otherwise}
\end{cases} \]

- Implemented in YUV rather than RGB
Result

Luminance rescaling
Far from solved...

- Only pure texture, how about mixture of texture and non-texture
- Far from perfect...
Parallel Method on Mesh

- Poisson based mesh editing
- Optical Boundaries for mesh merging
- Extend from image space to mesh manifold
Mesh Editing with Poisson-Based Gradient Field Manipulation

SIGGRAPH 2004
Yizhou Yu  UIUC
Kun Zhou  MSRA
Dong Xu  Zhejiang Univ, MSRA
Xiaohan Shi  Zhejiang Univ, MSRA
Hujun Bao  Zhejiang Univ.
Baining Guo  MSRA
Heung-Yueng Shum  MSRA
Mesh merging
Deformation
Basic idea

- Quite similar to the case of image
- Vector field decomposition can be extended to manifold

\[ w = \nabla \phi + \nabla \times v + h \]

\[ \min_{\phi} \int \int_T \| \nabla \phi - w \|^2 dA, \]

\[ \nabla^2 \phi = \nabla \cdot w. \]
Discretization

- NOT real gradient and divergence, but it works

\[(\text{Div}w)(v_i) = \sum_{T_k \in N(i)} \nabla B_{ik} \cdot w |T_k|\]

\[
\text{Div}(\nabla \phi) = \text{Div}w
\]

- Sparse linear system
Mesh Merging

- Some ‘ugly’ details
  - Boundary interaction
  - Boundary correspondence
  - Re-parameterization
Mesh deformation

- Interactively change guided vector field
  - Change normal on a curve
  - Propagate to other areas

- Smooth normal field – smooth
(a)  (b)  (c)

Our algorithm  Naive Poisson  WIRE
Optimal Boundaries for Poisson Mesh Merging

SPM 2007
Xiaohuang Huang Zhejiang Univ. & HUST
Hongbu Fu HUST
Oscar Kin-Chung Au HUST
Chiew-Lan Tai HUST
Basic idea

- Best boundary for Poisson Merging
- Similar to ‘Drag-and-Drop Pasting’.
Optimal boundary

\[ E(\partial \Omega, k) = \sum_{p \in \partial \Omega} \left( (f_t(p) - f_s(p)) - k \right)^2, \text{ s.t. } \partial \Omega \in \text{blue} \]

\[ E(\partial \Omega, T) = \sum_{e \in \partial \Omega} \|Te - e^*\| \cdot \text{length}(e), \quad \partial \Omega \subset \Omega_0 \setminus \Omega_{\text{feature}} \]
Mesh ‘ugly’ limitation

- Correspondence between source and target images are trivial
- Meaningful correspondence between source and target meshes are VERY difficult.
Results
Failure?