Overview

- Problem Statement
- Examples
- Algorithm
- Results
- Limitations
- Solutions
- Discussion
Problem Statement

• Fill holes in images caused by removal of unwanted objects
• Motivation: ability of our visual system to “fill in” missing areas
• Given an image and an inverse matte → complete unknown regions based on known regions
Examples – Photographs
Examples - Paintings
Algorithm - Overview

**Input:** image C, inverse matte $\bar{\alpha}$ ( $\exists$ pixel with $\bar{\alpha} < 1$ )

**Output:** completed image, $\bar{\alpha} = 1$

**Algorithm:**
For each scale from coarse to fine
- approximate image from color and coarser scale
- compute confidence map from $\bar{\alpha}$ and coarser scale
- compute level set from confidence map
while mean confidence $< 1 - \varepsilon$
  for next target position p
    - compute adaptive neighborhood N(p)
    - search for most similar and frequent source match N(q)
    - composite N(p) and N(q) at p, updating color and $\bar{\alpha}$
    - compute approximation, confidence map and update level set
Algorithm – Fast Approximation

- Estimate colors of unknown region with iterative filtering of known values
- ‘Smear’ colors into unknown region
Algorithm – Fast Approximation

- Simple iterative filtering method
- Build pyramid with image at different scales
- Down-sample and up-sample image hierarchically with kernel at multiple resolutions
Algorithm – Fast Approximation

- Use this region for the approximation at the next level
Algorithm – Fast Approximation

- Illustration:
**Algorithm - Overview**

**Input:** image C, inverse matte $\bar{\alpha}$ ( $\exists$ pixel with $\bar{\alpha} < 1$ )

**Output:** completed image, $\bar{\alpha} = 1$

**Algorithm:**

For each scale from coarse to fine

- approximate image from color and coarser scale
- compute confidence map from $\bar{\alpha}$ and coarser scale
- compute level set from confidence map
- while mean confidence $< 1 - \varepsilon$
  
  for next target position $p$
  
  - compute adaptive neighborhood $N(p)$
  - search for most similar and frequent source match $N(q)$
  - composite $N(p)$ and $N(q)$ at $p$, updating color and $\bar{\alpha}$
  - compute approximation, confidence map and update level set
Algorithm – Confidence Map

• Define confidence map $\beta$ by assigning a value in $[0, 1]$ to each pixel $i$

$$\beta_i = \begin{cases} 
1 & \text{if } \overline{\alpha}_i = 1 \\
\sum_{j \in N(i)} g_j \overline{\alpha}_j^2 & \text{otherwise}
\end{cases}$$

• This tells us how confident we are in our approximation (1 = most confident)
Algorithm – Level Set

- Compute Level Set with similar confidence values:

\[ v_i = \begin{cases} 
0 & \text{if } \beta_i > \mu(\beta) \\
\beta_i + \rho[0, \sigma(\beta)] & \text{otherwise}
\end{cases} \]

- Next target position = Largest value in level set
Algorithm – Confidence Map, Level Set

Inverse matte  Confidence map  Level set
**Algorithm - Overview**

**Input:** image C, inverse matte $\overline{\alpha}$ ( $\exists$ pixel with $\overline{\alpha} < 1$)

**Output:** completed image, $\overline{\alpha} = 1$

**Algorithm:**

For each scale from coarse to fine

- approximate image from color and coarser scale
- compute confidence map from $\overline{\alpha}$ and coarser scale
- compute level set from confidence map
- while mean confidence $< 1 - \varepsilon$
  - for next target position p
    - compute adaptive neighborhood $N(p)$
    - search for most similar and frequent source match $N(q)$
    - composite $N(p)$ and $N(q)$ at p, updating color and $\overline{\alpha}$
    - compute approximation, confidence map and update level set
Algorithm – Adaptive neighborhood

• Determine size of neighborhood with a contrast criterion:
  ➜ Absolute of difference between extreme values across color channels
Algorithm – Search for best Fragment

• For each target fragment T, search for best source fragment S over
  – all positions (x, y)
  – 5 scales l
  – 8 orientations θ

• Add detail only to kernel approximated pixels, do not modify known pixels
**Algorithm – Search for best Fragment**
Algorithm – Search for best Fragment

- Find position, scale and orientation of source fragment by minimizing

\[ r^* = \arg \min_r \sum_{s \in S_r(i), t \in T(i), i \in N} (d(s, t) \beta_s \beta_t + (\beta_t - \beta_s) \beta_t) \]

Penalizes different values in corresponding pixels with high confidence in both source and target fragment

Rewards pixels with higher confidence in source than in target, while penalizing pixels with lower confidence in source than in target
Algorithm – Search for best Fragment

- Complete structured texture in perspective by searching in different scales
Algorithm – Search for best Fragment

- Complete symmetric shapes by searching under rotations and reflections
Algorithm - Overview

Input: image C, inverse matte $\bar{\alpha}$ ( $\exists$ pixel with $\bar{\alpha} < 1$)
Output: completed image, $\bar{\alpha} = 1$
Algorithm:
For each scale from coarse to fine
    approximate image from color and coarser scale
    compute confidence map from $\bar{\alpha}$ and coarser scale
    compute level set from confidence map
while mean confidence $< 1 - \varepsilon$
    for next target position $p$
        compute adaptive neighborhood $N(p)$
        search for most similar and frequent source match $N(q)$
        composite $N(p)$ and $N(q)$ at $p$, updating color and $\bar{\alpha}$
        compute approximation, confidence map and update level set
Algorithm – Composite Fragment

• Laplacian pyramid for smooth merging of source and target image using binary masks
  – Color components decomposed into Laplacian pyramids $L$
  – Binary masks decomposed into Gaussian pyramids $G$

$$L_k(C_{out}) = L_k(C_F)G_k(\alpha_F) + L_k(C_B)G_k(\alpha_B)G_k(1 - \alpha_F)$$
Algorithm – Composite Fragment
Algorithm - Overview
Results

• Computation time on 2.4 GHz processor initial mean confidence $\mu(\beta) > 0.7$
  – 120 to 419 seconds for 192 x 128 images
  – 83 to 158 minutes for 384 x 256 images

• 90% of the total computation time is spent on search for matching fragments

• Computation time is quadratic in the number of pixels
Limitations

• Example-based approach
  – performance is directly dependent on richness of available fragments
  – In all presented examples, training set is known region of single image → rather limited

• 2D image-based method
  – No knowledge of underlying 3D structure in image

• No distinction between figure and ground
  – Limitation for completion when inverse matte is on boundary of a figure
Limitations
Limitations

• Does not handle ambiguities in which missing area covers intersection of two perpendicular regions
Solutions

• User specifies point of interest
Solutions

• User specifies region bridges
Discussion

???