Non-Rigid Dense Correspondence with Applications for Image Enhancement

Han-Wook Park

13 April 2012

Korea University
Computer Graphics Lab.
Abstract

• Presents a new efficient method for recovering reliable local sets of dense correspondences between two images with some shared content
  ▪ Using Generalized PatchMatch [Barnes et al. 2010]
  ▪ Dense
  ▪ Robust to geometry & photometric variations
Introduction

• Most existing correspondence methods are designed for one of two different scenarios

  ▪ Images are close to each other in time and in viewpoint
    • Dense correspondence field (ex. optical flow)

  ▪ Images are NOT close to each other in time and in viewpoint, but the scene consists of mostly rigid objects
    • Sparse feature matching (ex. SIFT)
Introduction

• We present a new method for computing a reliable dense set of correspondences between two images
  ▪ Input images share some common content
    • Non-rigid changes
    • Changes in Lighting and/or tone mapping
    • Different cameras and lenses

• Motivated by the recent proliferation of large personal digital photo collections
  ▪ Many digital photograph manipulation applications benefit from the ability to detect reliable correspondences between the input images
  ▪ Existing correspondence methods may often find this task challenging
Introduction

- A robust set of dense correspondences
- A global non-linear parametric color transformation model
Related Work

- Correspondence
  - Sparse matching
    - Only for rigid
  - Dense
    - Only for highly similar scenes
    - Not robust to scale and rotation changes
  - Generalized PatchMatch

- Example-based enhancement
  - Color transfer
    - Globally matching color statistics
      - Poor results
    - Automatic co-segmentation
      - The two images must be similar
    - Requires user assistance
      - Our approach is automatic
  - Etc.
Correspondence algorithm

Nearest neighbor search → Consistent region aggregation → Fit and apply global color transfer → Search range adjustment

- color curves
  - saturation
  - hue

- translation
- rotation
- scale
- gain
- bias
Correspondence algorithm

Algorithm 1 Non-Rigid Dense Correspondence Algorithm

1: for scale = coarse to fine do
2:   for each patch \( u \in S \) do
3:     Find a transformation \( T^u = \arg \min_T \| S_u - R_{T(u)} \|_2 \) (Sec. 3.1)
4:   end for
5: Aggregate consistent matches to regions (Sec. 3.2)
6: Connect adjacent patches \( u, v \) if \( C(u, v) < \tau_{local} \) (eq. 1)
7: Eliminate small regions
8: Eliminate regions for which \( C(Z) < \tau_{ratio} \) (eq. 2)
9: Fit and apply a global color transformation (Sec. 3.3)
10: (Optional) Estimate a blur kernel and deconvolve (Sec. 5)
11: Narrow search ranges (Sec. 3.4)
12: end for
Nearest-neighbor search

- Compute the Nearest Neighbor Field from source $S$ to reference $R$

  - For each patch $u \in S$, seek a transformation
    
    \[ T^u = \arg \min_T \| S_u - R_{T(u)} \|_2 \]

  - Each transformation consists of
    - Translation $T_x$, $T_y$
    - Rotation $T_{rotation}$
    - Scale $T_{scale}$

- 8 x 8 neighborhood
- Lab color + luminance gradient (4ch) per a pixel
- Using Generalized PatchMatch
Nearest-neighbor search

• Color transformations
  - Gain $g$, bias $b$
  - No need to extend the randomized search strategy

$$g(u) = \sigma(S_u)/\sigma(R_T(u))$$

$$b(u) = \mu(S_u) - g(u)\mu(R_T(u))$$

• Use Gaussian-weighted mean and variance to be rotation-invariant
  - Pre-computed for each scale using mipmaps
Aggregating consistent regions

- Apply a consistency criterion to calculate a coherence error for a group of matches together and accept sufficiently large regions if their coherence error is small.

- Adjacent patches are consistent if their nearest neighbor field transformations are similar.
Aggregating consistent regions

\[ C(u, v) = \frac{\| T^v(v_c) - T^u(v_c) \|_2}{\| T^u(u_c) - T^u(v_c) \|_2} \]
Aggregating consistent regions

- Construct a graph
  - Connect node $u$ and $v$ only when $C(u, v) < \tau_{\text{local}}$

- Eliminate small regions
  - When if $|Z| < \tau_{\text{size}}$ for a region $Z$

- Eliminate regions for which $C(z) < \tau_{\text{ratio}}$

$$C(Z) = \frac{|\{(u, v) \in J(Z) \text{ s.t. } C(u, v) > \tau_{\text{global}}\}|}{|J(Z)|}$$

- $J(Z)$: A set of random pairs in the region $Z$
  - $|J(Z)| = |(Z)|^{1/2}$
  - $\tau_{\text{small}} < \|u_c - v_c\| < \tau_{\text{large}}$
Global color mapping

• Simple adjustment of mean and variance cannot reproduce complex variations

• More sophisticated statistics-based methods might fail to produce a meaningful mapping for colors lie in the outliers

• We chose a parametric model that can be applied to predict a reasonable mapping for colors in the outliers
  ▪ Fits three monotonic curves, one per channel (RGB) + linear transformation
Global color mapping

- RGB curves, using piecewise cubic splines
  - 7 breaks, uniformly distributed
  - \( y(-0.1) = -0.1 \)
  - \( y(1.1) = 1.1 \)
  - \( y'(x) \geq 0.1 \)

- Optimize saturation scale factor \( s \)
  - Eliminate luminance variation of both images
    \[
    \begin{pmatrix}
      s - w_r & w_g & w_b \\
      w_r & s - w_g & w_b \\
      w_r & w_g & s - w_b \\
    \end{pmatrix}
    \]

  - \((w_r, w_g, w_b) = (1, 1, 1) / 3\) or \((.2989, .587, .114)\)
  - Choose the one that best minimizes
Search constraints

- There may be many low-cost but incorrect matches
  - Limit search range of transformations

- At initial step
  - $T_x \in [0, R_w]$, $T_y \in [0, R_h]$, $T_{scale} \in [0.33, 3]$, $T_{rotation} \in [-45, 45]$, $T_{Lbias} \in [-30, 20]$, $T_{Lgain} \in [0.2, 3]$, $T_{Ggain} \in [0.5, 2]$
  - Resolution of image = $(R_w, R_h)$
  - $T_{Gbias} = 0$, $T_{again} = T_{bgain} = 1$

- For the reliable regions in the upcoming iterations
  - Radius of 4 pixels for the translation
  - 10% for the scale
  - 4 degrees for the rotation
Search constraints

• Range of gain and the bias correspondences within reliable regions, combined with the global color correction, can capture the gain and bias that are required for the rest of the image
  ▪ $[\min, \max]$ of reliable matches w.r.t color corrected image using global color correction
  ▪ Use initial search range if the total area of the reliable regions is less than 1%
Evaluation

- One coarse-to-fine sweep of our basic algorithm on a 640 x 480 pixel image takes between 4 and 9 seconds
  - 2.3GHz Intel Core i7 (2820qm) Mac-Book Pro
  - MATLAB/C++ implementation

- In most cases we obtain a very good estimate of the transfer model at the second coarsest scale
  - Around 0.9 seconds
Qualitative comparison

(a) Inputs
(b) SIFT
(c) GPM
(d) Our
Correspondence evaluation

(a) Inputs  (b) SIFT  (c) SIFT-Flow  (d) GPM  (e) Our
Comparison to Co-recognition
[Cho et al. 2008]

(a) Inputs
(b) Co-recognition
(c) Our
Evaluation

(a) Reference    (b) Source    (c) Pitié et al.    (d) Our
Global color transfer evaluation
Limitations

• Has difficulty finding reliable correspondences in very large smooth regions
  ▪ Ex. Clear sky

• Cannot match an object’s boundaries which appears in different backgrounds

• Single global color model cannot handle some cases
  ▪ Ex. Strong lighting changes, local user edits

• Only handles saturation changes
Applications - Local color transfer

• When global color transfer is not appropriate

• Use external algorithm to locally match color
  
  ▪ Locally adaptive histogram equalization [Pizer et al. 1987]
  ▪ Extrapolate using Poisson blending [Perez et al. 2003]
Applications - Local color transfer

(a) Reference  (b) Source  (c) Matches  (d) Our global  (e) Our local  (f) Zoom-in (global)  (g) Zoom-in (local)
Applications - Deblurring

• Synthesize a sharp image by assigning colors in the consistent regions of the source image using the corresponding reference locations
  - ‘Reconstructed source’
  - Used to kernel estimation

• Interleave the kernel estimation and deconvolution steps in the inner loop of our correspondence algorithm
  • Sparse deconvolution [Levin et al. 2007]
  • Progressively refine kernel and source image
Applications - Deblurring

(a) Sharp example  (b) Blurry  (c) [Cho and Lee 2009]  (d) [Levin et al. 2011]  (e) Our
Applications - Mask transfer

- Masking the same objects across different images
  - Transfer one mask to the another

- Transfer mask using correspondence
  - ‘Grey’ for outliers

- Grabcut - use transferred mask as a trimap
Applications - Mask transfer

(a) Reference  (b) Input mask  (c) Source  (d) Trimap  (e) Output mask