CVPR’2017 Tutorials: Local Feature Extraction and Learning for Computer Vision

Part II: Modern Descriptors for High Matching Performance

Bin Fan
National Laboratory of Pattern Recognition
Chinese Academy of Sciences
Outline

■ Handcrafted Descriptors
  • Non-linear Scale Space
  • Intensity Order Pooling
  • Using Multiple Scales

■ Learned Descriptors
  • Traditional Learning
  • Convolutional Neural Networks (CNN)
Handcrafted Descriptors
  • Non-linear Scale Space
  • Intensity Order Pooling
  • Using Multiple Scales

Learned Descriptors
  • Traditional Learning
  • Convolutional Neural Networks (CNN)
Core idea: **Use non-linear scale space** to improve localization accuracy and distinctiveness

**Identical pipeline to SURF:**

1. Scale space construction
2. Keypoint detection: based on the determinant of Hessian matrix
3. Dominant orientation estimation: sliding circular segment of 60 degree is used to find the longest x and y derivative vector.
4. Descriptor construction

Pablo F. Alcantarilla, Adrien Bartoli, and Andrew J. Davison. KAZE Features. ECCV, 2012.
Scale space: Linear VS. Non-linear

Gaussian scale space (linear). When scale evolves, image structures become blurry.

Non-linear scale space. When scale evolves, image structures with strong edge responses are not affected.
Non-linear Scale Space

• Can be understood as something similar to SIFT’s Gaussian scale space discretization

\[ \sigma_i (o, s) = \sigma_0 2^{o+s/S} \]

\[ o \in [0 \ldots O - 1], \ s \in [0 \ldots S - 1], \ i \in [0 \ldots N] \]

• Each scale (in pixel unit) is transferred into a time unit for non-linear scale space construction.

\[ t_i = \frac{1}{2} \sigma_i^2, \ i = \{0 \ldots N\} \]
Non-linear scale space is constructed by iteratively applying diffusion filtering that depends on local gradient strength.

\[ c(x, y, t) = g(\| \nabla L_\sigma(x, y, t) \|) \]

\[ g_1 = \exp \left( -\frac{\| \nabla L_\sigma \|^2}{k^2} \right), \quad g_2 = \frac{1}{1 + \frac{\| \nabla L_\sigma \|^2}{k^2}} , \]

\[ g_3 = \begin{cases} 1, & \| \nabla L_\sigma \|^2 = 0 \\ 1 - \exp \left( -\frac{3.315}{(\| \nabla L_\sigma \|/k)^8} \right), & \| \nabla L_\sigma \|^2 > 0 \end{cases} \]

Additive Operator Splitting (AOS)

\[ L^{i+1} = \left( I - (t_{i+1} - t_i) \cdot \sum_{l=1}^{m} A_l(L^i) \right)^{-1} L^i \]
Making the computation fast

Additive Operator Splitting (AOS) is computational expensive

\[ L^{i+1} = \left( I - (t_{i+1} - t_i) \cdot \sum_{l=1}^{m} A_l (L^i) \right)^{-1} L^i \]

Using Fast Explicit Diffusion (FED) as an alternative

\[ L^{i+1,j+1} = (I + \tau_j A (L^i)) L^{i+1,j}, \quad j = 0, \ldots, n - 1 \]

\[ t_{i+1} - t_i = \tau_{\text{max}} \frac{n^2 + n}{3} \]

\[ \tau_j = \frac{\tau_{\text{max}}}{2 \cos^2 \left( \pi \frac{2j + 1}{4n + 2} \right)} \]

• Traditional methods rely on geometric locations to aggregate low-level features to construct the descriptor.
• Geometric locations need an (unreliable) estimated orientation to align.

Intuition Order Pooling

- Pooling based on intensity orders is rotationally invariant.
- Encode some spatial information too.
- Invariant to monotonic intensity changes.

Combined with various low-level cues

- **MROGH** [Fan et al. CVPR’11]
  - With gradient orientation distribution
- **MRRID** [Fan et al. PAMI’12]
  - With center symmetric local binary patterns
- **LIOP** [Wang et al. ICCV’11]
  - With LIOP features
- **OIOl** [Wang et al. PAMI‘16]
  - With OIOl features
Rotation Invariant Gradient

\[ D_x(X_i) = I(X_i^1) - I(X_i^3) \]
\[ D_y(X_i) = I(X_i^2) - I(X_i^4) \]

Magnitude is linearly assigned to two neighbor orientation bins like SIFT

Rotation Invariant CS-LBP

- Using 8 neighbor points
- Intensity test among opposite points
- 16D CS-LBP vector

MRRID

Using Multiple Regions

\[ D = \begin{bmatrix} D_1 & D_2 & D_3 & D_4 \end{bmatrix} \]
Local Intensity Order Pattern

\[ Y(X_i) = (I(X^1_i), I(X^2_i), I(X^3_i), I(X^4_i)) \]
\[ = (86, 217, 102, 151) \]
\[ \gamma(Y(X_i)) = (1, 4, 2, 3) \]

Relative intensity relationship

24 possible ordering

Zhenhua Wang, Bin Fan, and Fuchao Wu. Local Intensity Order Pattern for Feature Description. ICCV, 2011.
Overall Intensity Order Pattern

**Problem:** LIOP relies on neighbor points’ intensity order, which is not robust to noise.

**OIOP solution:** According to the *overall* intensity distribution in a local patch, quantize neighbor points’ intensities, and use the quantization results to construct the descriptor.

**Key:** How to quantize?
Quantization methods

Naïve average quantization

Quantization distribution of each ordinal bin. It is largely related to the ordinal pooling region. Limited discriminative ability.
Quantization methods

Learn the quantization percentiles

For each ordinal pooling region, learning the best quantization. **Enhanced discriminative ability.**
1. For each $X_i$ in the patch, sampling $N$ neighbor points
2. Each neighbor point’s intensity is quantized to $M$ numbers
3. Represent $X_i$ by a $N \times M$-based number
4. $M^N$ possible numbers, one-one correspond to a $M^N$-D vector
5. Intensity order based pooling for these $M^N$-D vector, obtain the OIOP descriptor

$M = 4, \quad N = 3$, suggested by the authors
Using Multiple Scales

Remember that MROGH/MRRID use multiple scales to compute descriptors separately and concatenate them together.

\[ D = \begin{bmatrix} D_1 & D_2 & D_3 & D_4 \end{bmatrix} \]
Using Multiple Scales

DSP-SIFT: pooling SIFT descriptors across scales

\[ D = \begin{bmatrix} D_1 & + & D_2 & + & D_3 & + & D_4 \end{bmatrix} \]

Using Multiple Scales

Scale-Less SIFT (SLS): pursuit a subspace representation of SIFT across scales

\[ D = \begin{bmatrix} D_1 & D_2 & D_3 & D_4 \end{bmatrix} \]

Handcrafted Descriptors
- Non-linear Scale Space
- Intensity Order Pooling
- Using Multiple Scales

Learned Descriptors
- Traditional Learning
- Convolutional Neural Networks (CNN)
Affine Subspace Representation Descriptor

Core idea: learn a **subspace** representation of the patch **under different viewpoints**. Incorporate robustness to viewpoint changes in descriptor.

Making the method feasible

• Affine warping is expensive
  – Extremely computational expensive to do for all patches!!!

• Efficient extraction of the subspace through linear approximation and offline learning
ASR Descriptor

Feasible Implementation

Patch Reconstruction

\( P_i, \bar{L} \)

\( L \approx \alpha_1 + \alpha_2 + \cdots + \alpha_n + \bar{L} \)

Decompose Patch (cheap)

Pre-computed offline

For \( A_1 \)

Affine Warping

Cropping

PCA Projection

Offline Training

Subspace Representation

\[ \begin{bmatrix} d_{A_1} \\ d_{A_2} \\ \vdots \\ d_{A_m} \end{bmatrix} \]
Brown’s Descriptor Learning

Discriminative Descriptor Learning by Optimizing Low-level Feature, Spatial Pooling, and Feature Embedding

- Normalized Patch
- Smooth
- Low-level feature extraction
- Spatial pooling
- Post process
- Projection
- Descriptor

- Pre-defined low level features: gradient-based, filter bank based
- Pre-defined spatial poolings: SIFT-like, DAISY-like, GLOH-like
- Optimized combination of low level feature + spatial pooling
- Objective: maximize the area under ROC
- Projection: PCA, LDE …

<table>
<thead>
<tr>
<th></th>
<th>Low level feature</th>
<th>Spatial pooling</th>
<th>dim</th>
<th>PCA dim</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st</td>
<td>Steerable filters bank</td>
<td>DAISY-like</td>
<td>272</td>
<td>26</td>
</tr>
<tr>
<td>2nd</td>
<td>Gradient orientation map</td>
<td>DAISY-like</td>
<td>136</td>
<td>53</td>
</tr>
</tbody>
</table>

PCA is better than LDE for projecting descriptor!
VGG’s Descriptor Learning

- Pre-defined units for spatial pooling
- Each pooling unit contributes to a part of the descriptor
- Select a small number of pooling units to construct the descriptor

\[ d(x, y) + 1 < d(u, v) \quad \forall (x, y) \in \mathcal{P}, (u, v) \in \mathcal{N} \]

Convex Learning objective

MatchNet

- Simultaneously learn the descriptor and the metric
- Siamese feature descriptor network
- Metric network on top
- Cross-entropy loss, transfer matching problem to classification problem

MatchNet

Two steps for usage
1. Using feature network to extract descriptors for all patches
2. Combining descriptor pairs, input to metric network

Computational efficient compared to directly input all patch pairs.
DeepCompare

Core idea: Using CNN + metric network to learn similarity function for a pair of patches

DeepCompare

- **No direct notion of descriptor**: Simply considers the two patches of an input pair as a 2-channel image.
- **Computational expensive**: Pair-wise operation, cannot re-use descriptor of each patch.

2 channel(2ch)
DeepCompare

- Siamese feature descriptor network
- Metric network on top
Consists of two separate streams, central and surround (CS), allowing the network to process at two different resolutions.
Core idea: Reduce the proportion of false positive and false negative, i.e., blue shaded area.

**Objective:**
1. Minimize the variance of two distributions.
2. Maximize the margin between the mean values of two distributions.

Triplet samples: \((x_i, x_i^+, x_i^-), i = 1, \cdots, N\), \((x_i, x_i^+)\) match pair, \((x_i, x_i^-)\) non-matching pair

①Global Loss (L2 distance):
\[
J_1 = \delta_+^2 + \delta_-^2 + \lambda \max(0, \mu_+ - \mu_- + t)
\]
\[
d_i^+ = \|f(x_i) - f(x_i^+))\|, d_i^- = \|f(x_i) - f(x_i^-)\|
\]
\(\delta_+^2, \mu_+\) are variance and mean of \(d_i^+\), \(\delta_-^2, \mu_-\) are variance and mean of \(d_i^-\)

②Triplet Local Loss:
\[
J_2 = \max(0, 1 - \frac{d_i^-}{d_i^+ + m})
\]

③Global Loss (Similarity):
\[
J_3 = \delta_+^2 + \delta_-^2 + \lambda \max(0, \mu_- - \mu_+ + m), d_i^+ = g(x_i, x_i^+), d_i^- = g(x_i, x_i^-)
\]

- Without metric net: \(\text{TNet-Tloss(②)} < \text{TNet-TGLoss(② + ①)}\)
- With metric net: \(\text{SNet-Gloss(③)} < \text{CS SNet-Gloss(③, 2ch-2stream)}\)
DeepDesc

- Use Euclidean distance, direct substitution of SIFT
- **loss**: minimize pairwise hinge loss

\[
l(x_1, x_2) = \begin{cases} 
\|D(x_1) - D(x_2)\|_2, & p_1 = p_2 \\
\max(0, C - \|D(x_1) - D(x_2)\|_2), & p_1 \neq p_2 
\end{cases}
\]

• Only 3 convolutional layers, simple.
• Use hard negative mining to alleviate the problem of imbalanced positive and negative samples, key to good performance.
Core idea: Smallest negative distance within the triplet should be larger than the positive distance.

Ranking-based: $\lambda(\Delta_+, \Delta^*) = \max(0, \mu + \Delta_+ - \Delta^*)$

Ratio-based: $\lambda(\Delta_+, \Delta^*) = \left(\frac{\Delta_+}{\Delta_+ + \Delta^*}\right)^2$

L2-Net

L2-Net Architecture

- Fully convolutional architecture
- Batch normalization after convolution
- Local response normalization before output
- Input 32x32 patch, output 128D vector
- CS L2-Net: two separate L2-Net deal with the central and the whole patch

Core ideas

1. Progressive sampling: in training stage, a batch contains lots of negatives while only a few positives, in accord with matching scenario.

2. Using the concept of relative minimal distance of matching pairs, which is the characteristics of NN matching.

3. Incorporate the supervision information from intermediate layers into the learning objective, improving generalization.

4. Constrain the compactness of the learned descriptor, reducing overfitting.
Progressive sampling: each batch, randomly select one patch from matching pairs to form non-matching pairs, improving the number of non-matching pairs.

**Advantage:** consisting of more negatives compared to pair-wise and triplet samples.
Learning Objectives

1. Relative minimal distance of matching pairs

\[ E_1 = \frac{1}{2} \left( \sum_i \log s_{ii}^c + \sum_i \log s_{ii}^r \right) \]
\[ s_{ij}^c = \exp(2 - d_{ij})/\sum_m \exp(2 - d_{mj}) \]
\[ s_{ij}^r = \exp(2 - d_{ij})/\sum_n \exp(2 - d_{jn}) \]

2. Less correlation among different dimensions

\[ E_2 = \frac{1}{2} \left( \sum_{i \neq j} (r_{ij}^1)^2 + \sum_{i \neq j} (r_{ij}^2)^2 \right) \]
\[ r_{ij}^s = \frac{(b_i^s - \bar{b}_i^s)^T (b_j^s - \bar{b}_j^s)}{\sqrt{(b_i^s - \bar{b}_i^s)^T (b_i^s - \bar{b}_i^s)} \sqrt{(b_j^s - \bar{b}_j^s)^T (b_j - \bar{b}_j^s)}} \]

3. Relative maximal similar intermediate feature maps of matching pairs

\[ E_3 = -\frac{1}{2} \left( \sum_i \log v_{ii}^c + \sum_i \log v_{ii}^r \right) \]
\[ v_{ij}^c = \exp(g_{ij})/\sum_m \exp(g_{mj}) \]
\[ v_{ij}^r = \exp(g_{ij})/\sum_n \exp(g_{jn}) \]

\[ G = (F_1)^T F_2 \]
Ideally, $d_{ii}$ should be the smallest one in the $i$th row, and also for the $i$th column.

$$
S_i^c = \frac{\exp(2 - d_{ii})}{\sum_m \exp(2 - d_{mi})}
$$

$$
S_i^r = \frac{\exp(2 - d_{ii})}{\sum_m \exp(2 - d_{im})}
$$

$$
E_1 = -\frac{1}{2} \left( \sum_i \log s_i^r + \sum_i \log s_i^c \right)
$$
L2-Net

Less correlation among dims

$$Y_s = [y_1^s, \ldots, y_i^s, \ldots, y_p^s]_{q \times p}, (s = 1, 2)$$

$$Y_s^T = [b_1^s, \ldots, b_i^s, \ldots, b_q^s]$$

The correlation between the $i$th and $j$th dims:

$$r_{ij}^s = \frac{(b_i^s - \bar{b}_i^s)^T (b_j^s - \bar{b}_j^s)}{\sqrt{(b_i^s - \bar{b}_i^s)^T (b_i^s - \bar{b}_i^s)} \sqrt{(b_j^s - \bar{b}_j^s)^T (b_j^s - \bar{b}_j^s)}}$$

Minimize these correlations:

$$E_2 = \frac{1}{2} \left( \sum_{i \neq j} (r_{ij}^1)^2 + \sum_{i \neq j} (r_{ij}^2)^2 \right)$$
For intermediate feature maps:

\[ F_s = [f_1^s, f_2^s, \ldots, f_p^s], s = 1, 2 \]

Their similarity is defined by inner product:

\[ G = (F_1)^T F_2 \]

Ideally, similarity of matching pairs should be the largest, i.e., \( g_{ii} \) should be the largest one in the \( i \)th row and \( i \)th column.

\[
\begin{align*}
\nu_i^f &= \frac{\exp(g_{ii})}{\sum_m \exp(g_{im})} \\
\nu_i^c &= \frac{\exp(g_{ii})}{\sum_m \exp(g_{mi})}
\end{align*}
\]

\[ E_3 = -\frac{1}{2} \left( \sum_i \log \nu_i^f + \sum_i \log \nu_i^c \right) \]
The 3rd objective should be only used with intermediate feature maps of the first and last conv layers.
The output of L2-Net approximates a Gaussian distribution, which is very suitable for binary descriptors. By applying \texttt{sign()} to L2-Net, we obtain the Binary L2-Net.
Learn to assign orientations

Is considering only descriptors wise?

Learn to assign orientations

Learning Framework

Orientation Estimator

Minimize descriptor distance, e.g. SIFT

Feature Descriptor

Feature Descriptor

Feature Descriptor
Learn to assign orientations

Improved results with good orientations

Average performance

mAP (higher the better)
Training requires various patches

Keypoint VS Non-Keypoint

Matching Keypoints VS Non-Matching Keypoints
Quadruplet Siamese Network

$P_1, P_2$: corresponding keypoints.
$P_3$: non-corresponding keypoint.
$P_4$: non-keypoint.
Loss function

\[
\min_{\{f_\mu, g_\phi, h_\rho\}} \sum_{(P_1, P_2, P_3, P_4)} \gamma \mathcal{L}_{\text{class}}(P^1, P^2, P^3, P^4) + \mathcal{L}_{\text{pair}}(P^1, P^2)
\]

\[
\mathcal{L}_{\text{class}}(P^1, P^2, P^3, P^4) = \sum_{i=1}^{4} \alpha_i \max(0, (1 - \text{softmax}(f_\mu(P^i))^2)
\]

\[
\mathcal{L}_{\text{pair}}(P^1, P^2) = \| h_\rho(G(P^1, \text{softargmax}(f_\mu(P^1)))) - h_\rho(G(P^2, \text{softargmax}(f_\mu(P^2)))) \|_2
\]

\[
G(P, x) = \text{Rot}(P, x, g_\phi(\text{Crop}(P, x)))
\]
Each component is *meant for* each other
Take home messages

• For patch level datasets (e.g. Brown’s dataset), learning based methods generally outperform hand-crafted ones.
• For image level dataset (e.g. VGG dataset), performance gap between learning based methods and hand-crafted methods is not significant.
• CNN based methods are dominant in the learning based methods.
• CNN based methods operating on the Euclidean space is highly required for wider application.
• Learning for feature detector has been largely ignored, but it is of significant important.
Thanks!

Coffee Break