Correspondence of Local Features for Wide-Baseline Matching, Image Retrieval, Stitching, 3D reconstruction and more ...

Jiří Matas, Ondra Chum, Jan Čech, Dmytro Mishkin
Center for Machine Perception, Czech Technical University Prague

Includes slides by:
- Darya Frolova, Denis Simakov, The Weizmann Institute of Science
- Martin Urban, Stepan Obdrzalek, Ondra Chum, Filip Radenovic, CTU CMP Prague
- Matthew Brown, David Lowe, University of British Columbia
- Daniel DeTone, MagicLeap
- Jerome Revaud, Naver Labs Europe
The Geometric Correspondence Problem
The Geometric Correspondence Problem

\[ O_{1,k} = \{x, d\} \quad \text{observation} \]
\[ x_{1,k} = (2290, 780) \quad \text{spatial info} \]
\[ d_{1,k} = (17, 21, 4) \quad \text{descriptor} \]

\[ V_1 = \bigcup_{k=1}^{n} O_{1,k}, \Theta_1 \]

\[ \Theta_1 = \left\{ (R_1, t_1), K_1, 10.12.2021, \text{Xiaomi Mi 9}, \text{daylight} \right\} \]
Sensor information

\[ M: f(V, O, C, \Omega) < \epsilon \]
World model

\[ x_k^T F_{12} x_l < \epsilon_{epi} \]

\[ C_{1,k,2,l} \quad \text{correspondence} \]

\[ O_{2,l} \]

\[ V_2 \]

\[ \Theta_2 = \left\{ (R_2, t_2), K_2, 11.12.2021, \text{iPhone 11, fairy lights} \right\} \]
We will be interested in local geometric correspondence

Not semantic correspondence
Correspondences are useful for 3D reconstruction

- 3D reconstruction – camera pose estimation
Correspondences are useful for 3D reconstruction

1. **matching distinguished regions**
   - tentative correspondences (verification)
   - two view geometry

2. **camera calibration**
   - camera positions
   - sparse reconstruction

3. **dense stereoscopic matching**
   - pixel/sub-pixel matching
   - depth maps, 3D point cloud

4. **surface reconstruction**
   - surface refinement
   - triangulated 3D model

Matas et al., IVC 2004.

MPV course 2022, CTU in Prague
Correspondences are useful for 3D reconstruction


MPV course 2022, CTU in Prague
Correspondences are useful for 3D reconstruction

Large scale 3D reconstruction – “Microsoft Photo Tourism”
57,845 downloaded images, 11,868 registered images.

The Old City of Dubrovnik
SLAM: simultaneous localization and mapping

Not only buildings: medical image registration

Fig. 2. An example of a set of lung lesion tissue images using Cc10, CD31, H&E, Ki67 and proSPC stains (from left to right) with landmarks shown as green dots.

3D reconstruction: not only buildings

Endoscopy

Learned dense descriptor

SIFT

Extremely Dense Point Correspondences using a Learned Feature Descriptor. Liu et. al CVPR 2020

MPV course 2022, CTU in Prague
How does it work? Panorama stitching
Correspondences in Action (1): Building a Panorama

- We need to match and align images = find (dense) correspondences
- Note: this can be done only if both images were taken from the same viewpoint
Problem 1:

- Detect the *same* feature *independently* in both images*
- Note that the set of “features” may be rather sparse

A repeatable detector needed.

* Some existing methods operate on both images simultaneously
Problem 2:

• how to correctly recognize the corresponding features?

Solution:

1. Find a discriminative and stable descriptor
2. Solve the matching problem
Correspondences in Action (1): Building a Panorama

Possible approach:

1. Detect features in both images
2. Find corresponding pairs
3. Estimate transformations (Geometry and Photometry)
4. Put all images into one frame, blend.
Correspondences in Action (1): Building a Panorama

Possible approach:

1. Detect features in both images
2. Find corresponding pairs
3. Estimate transformations (Geometry and Photometry)
4. Put all images into one frame, **blend**.
Correspondences in Action (2): Image Retrieval
Correspondences in Action (2): Image Retrieval
Correspondences in Action (2): Image Retrieval
Correspondences in Action (2): Image Retrieval

MPV course 2022, CTU in Prague
Correspondences in Action (2): Image Retrieval
Local Features in Action (2): Image Retrieval
Correspondences in Action (2): Image Retrieval

“Zoom in”

“Zoom out”

Correspondences in Action (2): Image Retrieval

https://youtu.be/Dlv1aGKqSlk

Correspondences in action(3): Localization and Mapping

- Place recognition - retrieval in a structured (on a map) database

Correspondences in action (3): Localization and Mapping

Pion et al., 3DV 2020 Benchmarking Image Retrieval for Visual Localization

MPV course 2022, CTU in Prague
Challenges in the Correspondence Problem

Why is Establishing Correspondence Difficult?
Finding correspondences is not easy due to large viewpoint change (including scale)

=>

the wide-baseline stereo problem

Applications:
- pose estimation
- 3D reconstruction
- location recognition
Finding correspondences is not easy
due to large viewpoint change
(including scale)
=>
the wide-baseline (WBS) stereo problem
Finding correspondences is not easy
due to large illumination change
=>
wide “illumination-baseline” stereo problem

Applications:
- location recognition
- summarization of image collections
Find the matches (look for tiny colored squares...)

NASA Mars Rover images with SIFT feature matches
Figure by Noah Snavely
Finding correspondences is not easy

due to large

time difference

=>

wide temporal-baseline stereo problem

Applications:
- historical reconstruction
- location recognition
- photographer recognition
- camera type recognition
Finding Correspondences is not easy due to occlusion

Applications:
- pose estimation
- inpainting
Finding Correspondences is not easy

change of imaging modality

Applications:
- medical imaging

MPV course 2022, CTU in Prague
Outline

The classical method and its components:

1. Detection of Local invariant features:
   - Moravec, Harris, FAST
   - Scale invariant: SIFT, Hessian-Affine, MSER

2. Descriptors: Histogram-based, binary

3. Matching and filtering

4. Deep Learned detectors & descriptors: SuperPoint, R2D2, HardNet

5. Limitations

6. RANSAC (robust model fitting)
Motivation: Generalization of Local Stereo to Wide Baseline Stereo (WBS)

Narrow-baseline stereo

1. Local Feature (Region) = a rectangular “window”
   • robust to occlusion, translation invariant
   • windows matched by correlation, assuming small displacement
   • successful in narrow-baseline stereo matching

Brewster Stereoscope, 1856
A “photo” for both eyes
Motivation: Generalization of Local Stereo to Wide Baseline Stereo (WBS)

2. Widening of baseline or zooming in/out
   • local deformation is well modelled by affine or similarity transformations
   • how can the “local feature” concept be generalised? *The set of ellipses is closed under affine tr., but it’s too big to be tested*
   *window scanning approach becomes computationally difficult.*
Wide baseline stereo pipeline

Detector → Measurement region selector → Descriptor → Matching and filtering → Geometrical verification (RANSAC)

Images → covariant detector → feature frames → crop & normalize → normalized features → distinctive & invariant descriptor → descriptors → vector comparison

Detector

Descriptor

Image credit: Andrea Vedaldi, ICCVW 2017
Common Structure of “Local Feature” Algorithms

1. Detect affine- (or similarity-) covariant regions (=distinguished regions) = local features
Yields regions (connected set of pixels) that are detectable with high repeatability over a large range of conditions.

2. Description: Invariants or Representation in Canonical Frames
Representation of local appearance in a Measurement Region (MR). Size of MR has to be chosen as a compromise between discriminability vs. robustness to detector imprecision and image noise.

3. Indexing
For fast (sub-linear) retrieval of potential matches

4. Verification of local matches

5. Verification of global geometric arrangement
Confirms or rejects a candidate match
Local Features

- Methods based on “Local Features” are the state-of-the-art for a number of computer vision problems.

- E.g.: Wide-baseline stereo, image retrieval, 3D reconstruction

- Terminology (diverse, unfortunately):
  Local Feature = Interest “Point” = The “Patch” =
  = Feature “Point”
  = Distinguished Region
  = (Transformation) Covariant Region
Detecting
Local Invariant Features
“Local Features” are regions, i.e. in principle arbitrary sets of pixels (not necessarily contiguous) with:

- High **repeatability**, (ideally invariance) under
  - Illumination changes
  - Changes of viewpoint => geometric transformations
    i.e. are **distinguishable** in an image regardless of viewpoint/illumination => are **distinguished regions**

- Are **robust to occlusion** => must be **local**

- Must have discriminative neighborhood => they are “**features**”

Methods based on local features/distinguished regions (DRs) formulate computer vision problems as matching of some representation derived from DR (as opposed to matching of entire images)
Two interpretations of the local features:

- "Detector":
  the task is to find all instances of the "object" == "good feature", e.g. corner, blob, face.

- "Sorter":
  rank all the patches (distinguished regions) from the "best" to the worst, such that the ranking is consistent under nuisance factor. Then take top-K.

Keypoints here are the centers of the local features
Drop the detector completely? Possible, but

The main reason we need the local feature detector is efficiency:

- Computation-wise: even small image contains A LOT of pixels, e.g. 640x480 = 307 200. And even more regions (required for the descriptors), if we consider scale, orientation, etc.

- Memory-wise: storing 307k x 128 dim float descriptors = 157 Mb

- If any-to-any matching is possible, i.e. wide baseline setup, we have 307k x 307k = 94 Billion tentative correspondences
“Local Features” are regions, i.e. in principle arbitrary sets of pixels (not necessarily contiguous) with high repeatability, (invariance in theory) under illumination changes
Changes of viewpoint => geometric transformations

Local feature detector is a form of attention

Methods based on local features/distinguished regions (DRs) formulate computer vision problems as matching of some representation derived from DR (as opposed to matching of entire images)

MPV course 2022, CTU in Prague
Drop the detector completely? Possible, but

- Yet, if we can afford pairwise running (no separate feature extraction step)
- AND the correspondence search space is constrained:
  - Scene Flow (lecture in 2 month)
  - Optical Flow
  - Coarse-to-fine matching
  - Sparse-to-dense matching (asymmetric)

MPV course 2022, CTU in Prague
To summarize: Classical local feature detectors have to be **designed** in a such way, to have the properties, we discussed before:

- Repeatability
- Distinguishability
- Locality
- Consistent ranking
- Robustness to the nuisance factors
- ...
Two core ideas (in “modern terminology”):

1. To be a distinguished region, a region must be \textit{at least} distinguishable from \textit{all} its neighbours.

2. Approximation of Property 1. can be tested very efficiently, without explicitly testing.

Note: both properties were proposed before Harris paper, (1) by Moravec, (1)+(2) by Foerstner.
Harris Detector: Basic Idea

“flat” region: no change in all directions

“edge”: no change along the edge direction

“corner”: significant change in all directions

- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give a *large change*
Harris Detector: Basic Idea

The Harris Detector: Basic Idea involves three fundamental ideas:

1. **f1**: This is typically a feature vector that captures the intensity changes in the image.
2. **f2**: Represents a gradient vector that captures the orientation changes.
3. **f3**: A combination of f1 and f2, which together form the core of the Harris Detector.

These features are used to detect interest points in an image, which are locations where the intensity changes significantly. The Harris Detector is widely used in computer vision for tasks such as object recognition and feature matching.
Harris Detector: Preliminaries (Moravec)

Measure similarity of the image function $I(x_0, y_0)$ at point $(x_0, y_0)$ of and its neighbors around $I(x_0 + u, y_0 + v)$ by weighted SSD:

$$E(x_0, y_0; u, v) = \sum_{(x, y) \in W(x_0, y_0)} w(x, y) (I(x, y) - I(x + u, y + v))^2$$  \hspace{1cm} (1)

- $W(x_0, y_0)$ is a window centered at point $(x_0, y_0)$
- $w(x, y)$ can be constant or (better) Gaussian

We are interested in “corner” points $I(x_0, y_0)$ where $E(x_0, y_0; u, v)$ is large for all $(u, v)$, i.e. $\min(E(x_0, y_0; u, v))$ is locally maximal.
Harris Detector: Preliminaries (Moravec)

\[ E(x_0, y_0; u, v) = \sum_{(x, y) \in W(x_0, y_0)} w(x, y)(I(x, y) - I(x + u, y + v))^2 \]

- \( W(x_0, y_0) \) is a window centered at point \((x_0, y_0)\)
- \( w(x, y) \) can be constant or (better) Gaussian

```python
def ssd_min(img, ksize = 3, wsize=7):
    b, ch, h, w = img.size()
    out = torch.zeros(b, ch, ksize, ksize, h, w)
    pd = int(ksize/2)
    img_padded = F.pad(img, (pd, pd, pd, pd), mode='replicate')
    # u, v are possible shifts
    # wsize is the size of the window W
    for u in range(ksize):
        for v in range(ksize):
            diff_shift_sq = (img - img_padded[:, :, v:h+v, u:w+u]).pow(2)
            out[:,:, u, v, :, :] = K.gaussian_blur2d(diff_shift, (wsize, wsize), (1.0, 1.0))
            out[:,:,pd, pd, :, :] = 100000. # (pd, pd) is window center, so difference is 0
    out = out.min(dim=2)[0].min(dim=2)[0] # Minimum over x and y shifts
    return out
```

Code example available [here](#)
Harris Detector: Preliminaries (Moravec)

\[ E(x_0, y_0; u, v) = \sum_{(x,y) \in W(x_0,y_0)} w(x, y)(I(x, y) - I(x + u, y + v))^2 \]

Code example available [here](#).

Note, that response is high not only at the corners, but also at the “L” endpoints.

Input image \(\min_{(u,v)} E(x_0, y_0; u, v)\)
Harris Detector: Preliminaries (Moravec)

Code example available here
Harris Detector: Preliminaries (Moravec)

\[ E(x_0, y_0; u, v) = \sum_{(x,y) \in W(x_0,y_0)} w(x, y)(I(x, y) - I(x + u, y + v))^2 \]

Code example available [here](#)

![Input image](#) ![min_{(u,v)}E(x_0, y_0; u, v)](##)
Harris Detector: Mathematics

Approximate intensity function in shifted position by the first-order Taylor expansion:

\[ I(x + u, y + v) \approx I(x, y) + [I_x(x, y), I_y(x, y)] \begin{bmatrix} u \\ v \end{bmatrix} \]

where \( I_x, I_y \) are partial derivatives of \( I(x, y) \).

\[
\sum_{(x, y) \in W(x_0, y_0)} w(x, y) (I(x, y) - I(x + u, y + v))^2 \\
\approx \sum_{(x, y) \in W(x_0, y_0)} w(x, y) (I(x, y) - (I(x, y) + [I_x(x, y), I_y(x, y)] \begin{bmatrix} u \\ v \end{bmatrix}))^2 \\
= \sum_{(x, y) \in W(x_0, y_0)} w(x, y) [I_x(x, y), I_y(x, y)] \begin{bmatrix} u \\ v \end{bmatrix})^2
\]
Harris Detector: Mathematics

\[
E(x_0, y_0; u, v) \approx \sum_{(x,y) \in W(x_0, y_0)} w(x, y) \begin{bmatrix} I_x(x, y), I_y(x, y) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}^2
\]

\[
= [u, v] \sum_{(x,y) \in W(x_0, y_0)} w(x, y) \begin{bmatrix} I_x(x_0, y_0)^2 & I_x(x_0, y_0)I_y(x_0, y_0) \\ I_x(x_0, y_0)I_y(x_0, y_0) & I_y(x_0, y_0)^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}
\]

\[
M(x_0, y_0) = \sum_{(x,y) \in W(x_0, y_0)} w(x, y) \begin{bmatrix} I_x(x_0, y_0)^2 & I_x(x_0, y_0)I_y(x_0, y_0) \\ I_x(x_0, y_0)I_y(x_0, y_0) & I_y(x_0, y_0)^2 \end{bmatrix}
\]

Note that \( M \) is symmetric and positive semi-definite, Remember how \( E \) was designed, eq. (1), slide 52.
Harris Detector: Mathematics

\[ E(x_0, y_0; u, v) \approx [u, v]M(x_0, y_0) \begin{bmatrix} u \\ v \end{bmatrix} \]

Iso-contours of \( E, \ E(x_0, y_0; u, v) = \text{const} \), form an ellipse with a center at (0,0) and axes aligned with eigenvector of \( M \)

- \( \lambda_{\text{min}}, \lambda_{\text{max}} \) – eigenvalues of \( M \),
  \( \lambda_{\text{min}} = \lambda_1, \lambda_{\text{max}} = \lambda_2 \)

- \( M \) symmetric, positive definite \( \rightarrow \) orthogonal eigenvectors, positive eigenvalues
Classification of image points using eigenvalues of $M$:

- **“Corner”**
  - $\lambda_1$ and $\lambda_2$ are large,
  - $\lambda_1 \sim \lambda_2$;
  - $E$ increases in all directions

- **“Edge”**
  - $\lambda_1 \gg \lambda_2$

- **“Flat” region**
  - $\lambda_1$ and $\lambda_2$ are small;
  - $E$ is almost constant in all directions
Harris Detector: Avoiding eigenvalue calculation

Measure of “corner” response ("cornerness"): 

\[ R = \det M - k(\text{trace } M) \]

- \[ M = \begin{bmatrix} A & B \\ B & C \end{bmatrix} \]
- \[ \det M = \lambda_1 \lambda_2 = AC - B^2 \]
- \[ \text{trace } M = \lambda_1 + \lambda_2 = A + C \]
- \[ k \ldots \text{empirical constant, } k \in (0.04, 0.06) \]

Find “corner” points as local maxima of corner response \( R \) in their neighborhood (3×3, or 5×5)
Harris Detector: Mathematics

- $R$ depends only on eigenvalues of $M$
- $R$ is large for a “corner”

- $R$ is negative with large magnitude for an edge
- $|R|$ is small for a flat region

\[ \lambda_1 \quad \lambda_2 \]

```
“Edge”
$R < 0$

“Corner”
$R > 0$

“Flat”
$|R|$ small

```

MPV course 2022, CTU in Prague
Harris Detector: Mathematics

[16] artifical_img = torch.zeros(1, 1, 21, 21)
    artifical_img[:,:, 7:14, 8] = 1
    artifical_img[:,:, 14, 8:17] = 1

in_img = artifical_img
out_img = ssd_min(in_img, 3, 7)
out_img_2 = k.feature.harris_response(in_img, 0.05)
imshow_torch_channels(torch.cat([in_img, out_img, out_img_2], 0), 0, colorbar=True, cmap='hot')

Input image  \( \min_{(u,v)} E(x_0, y_0; u, v) \)  Harris response
Harris Detector: Mathematics

```
[19] in_img = img_blured.float()
    out_img = ssd_min(in_img, 3, 7)
    out_img2 = K.feature.harris_response(in_img, 0.05)

    imshow_torch_channels(torch.cat([in_img, out_img, out_img2], 0)[:, :, 80:130, 260:310], 0,
                          colorbar=True, cmap='hot')
```

Input image  \( \min_{(u,v)} E(x_0, y_0; u, v) \)  Harris response
Harris Detector

The Algorithm:

1. Compute partial derivatives $I_x, I_y$
2. Compute: $A = \sum_W I_x^2$, $B = \sum_W I_x I_y$, $C = \sum_W I_y^2$
3. Compute “corner” response $R$
4. Find local maxima in $R$ above user defined threshold $T$

Parameters:

- Threshold $T$ on $R$
- Scale of the derivative operator (standard setting: very small, just enough to filter anisotropy of the image grid)
- Size of window $W$ (“integration scale”)
- Non-maximum suppression (NMS) algorithm setting, e.g. 8-neighbourhood
Harris Detector: Workflow

Compute corner response $R$
Harris Detector: Workflow

Find points with large corner response: $R > \text{threshold}$
Harris Detector: Workflow

Take only the points of local maxima of $R$
Harris Detector: Properties

- **Rotation invariance**

Ellipse rotates but its shape (i.e. eigenvalues) remains the same

*Corner response* $R$ is invariant to image rotation
Rotation Invariance of Harris Detector

ImpHarris: using Gaussian deriv (as we do)
Harris: finite differences without Gauss filtering

Repeatability rate:

\[
\frac{\text{# correspondences}}{\text{# possible correspondences}}
\]

Harris Detector: Intensity change

- Partial invariance to additive and multiplicative intensity changes

✓ Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$

? Intensity scale: $I \rightarrow a \cdot I$.

But ordering of $R$ is preserved
Moravec detector in 2020: D2D

\[ E(x_0, y_0; u, v) = \sum_{(x,y) \in W(x_0, y_0)} w(x, y) (I(x, y) - I(x + u, y + v))^2 \]

Idea. Replace pixel intensity \( I \) with descriptor \( F \) in \( \mathbb{R}^d \):

\[ RS = \sum_{(x,y) \in W(x_0, y_0)} w(x, y) \| F(x, y) - F(x + u, y + v) \|_2 \]

Input image \( \min_{(u,v)} E(x_0, y_0; u, v) \) \( \min_{(u,v)} RS(x_0, y_0; u, v) \)
Realtime: working at >24 fps

Harris:
- 37.5 ms per 400x300 image
  - = 26.6 fps
  - without descriptor, matching, etc.
- Too slow for SLAM

*Laplace – scale selection method, see around slide 94
Realtime: working at >24 fps

FAST (detector part of ORB):
- 4.41 ms per 400x300 image
  = 226.7 fps
- Good enough for SLAM
FAST Feature Detector

- Considers a circle of 16 pixels around the corner candidate $p$
- $\geq 12$ contiguous pixels brighter/darker than $I_p \pm t$, $t$... threshold
- Rapid rejection by testing 1, 9, 5 then 13
- Only if at least 3 of those are brighter/darker than $I_p \pm t$ the full segment test is applied
FAST: Weaknesses

- Corners are clustered:
  - Use non-maximal suppression:

\[
V = \max \left( \sum_{q \in S_b} |I_q - I_p| - t, \sum_{q \in S_d} |I_p - I_q| - t \right)
\]

where \( S_b = \{ q | I_q \geq I_p + t \} \), \( S_d = \{ q | I_q \leq I_p - t \} \)

- High speed test does not generalize well for \( n < 12 \)
- Knowledge from the first 4 tests is discarded
- Multiple features are detected adjacent to one another
FAST: running times

<table>
<thead>
<tr>
<th>Detector</th>
<th>Opteron 2.6GHz</th>
<th>Pentium III 850MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fast ( n = 9 ) (non-max suppression)</td>
<td>1.33 ms, 6.65%</td>
<td>5.29 ms, 26.5%</td>
</tr>
<tr>
<td>Fast ( n = 9 ) (raw)</td>
<td>1.08 ms, 5.40%</td>
<td>4.34 ms, 21.7%</td>
</tr>
<tr>
<td>Fast ( n = 12 ) (non-max suppression)</td>
<td>1.34 ms, 6.70%</td>
<td>4.60 ms, 23.0%</td>
</tr>
<tr>
<td>Fast ( n = 12 ) (raw)</td>
<td>1.17 ms, 5.85%</td>
<td>4.31 ms, 21.5%</td>
</tr>
<tr>
<td>Original FAST ( n = 12 ) (non-max suppression)</td>
<td>1.59 ms, 7.95%</td>
<td>9.60 ms, 48.0%</td>
</tr>
<tr>
<td>Original FAST ( n = 12 ) (raw)</td>
<td>1.49 ms, 7.45%</td>
<td>9.25 ms, 48.5%</td>
</tr>
<tr>
<td>Harris</td>
<td>24.0 ms, 120%</td>
<td>166 ms, 830%</td>
</tr>
<tr>
<td>DoG</td>
<td>60.1 ms, 301%</td>
<td>345 ms, 1280%</td>
</tr>
<tr>
<td>SUSAN</td>
<td>7.58 ms, 37.9%</td>
<td>27.5 ms, 137.5%</td>
</tr>
</tbody>
</table>

Table 1. Timing results for a selection of feature detectors run on fields (768 x 288) of a PAL video sequence in milliseconds, and as a percentage of the processing budget per frame. Note that since PAL and NTSC, DV and 30Hz VGA (common for webcams) have approximately the same pixel rate, the percentages are widely applicable. Approximately 500 features per field are detected.
FAST is a part of the ORB feature

- FAST is a backbone of the ORB-SLAM2
  - Uses Harris cornerness function for NMS
- Detects corner orientation (covered later in slides)

[Image Description]

https://webdiis.unizar.es/~raulmur/orbslam/