Geometry in CV
part 2

Homography
Correspondence

Yaroslava Lochman
Dec 17, 2019

Credits

[1] Pajdla, Tomas. Elements of geometry for computer vision. FEE CTU, 2013

[2] Hartley, Richard, and Andrew Zisserman. Multiple view geometry in computer vision. Cambridge university press, 2003

[3] Maksym Davydov. Computer Vision – Image correspondence, 2019

[4] OpenCV Documentation

[5] VLFeat Documentation

Outline

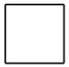
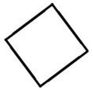
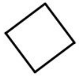
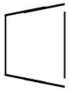
- Geometric Transformations
  - Affine Transformation
  - Homography
- Consistent vs. Overdetermined SLE
- Correspondence Problem
  - Feature, Detection & Description
  - Matching
  - Tentative Correspondences & RANSAC

Geometric Transformations

Consistent vs. Overdetermined SLE

Correspondence Problem

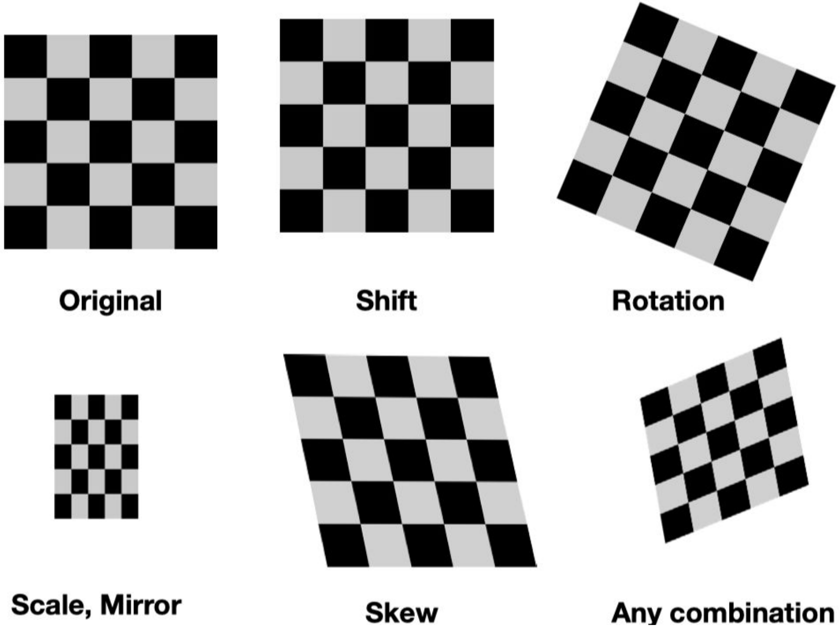
Hierarchy of 2D Transformations

Transformation	Matrix	# DoF	Preserves	Icon
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	2	orientation	
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	3	lengths	
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	4	angles	
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times 3}$	6	parallelism	
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{3\times 3}$	8	straight lines	

Affine Transformation

$$x' = a_{11}x + a_{12}y + b_1$$

$$y' = a_{21}x + a_{22}y + b_2$$



Affine Transformation: Task 1.1

Rewrite affine transformation in homogeneous form

$$x' = a_{11}x + a_{12}y + b_1$$
$$y' = a_{21}x + a_{22}y + b_2$$

$$\longrightarrow \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \; ? \; \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

6 DoF

Affine Transformation Matrix

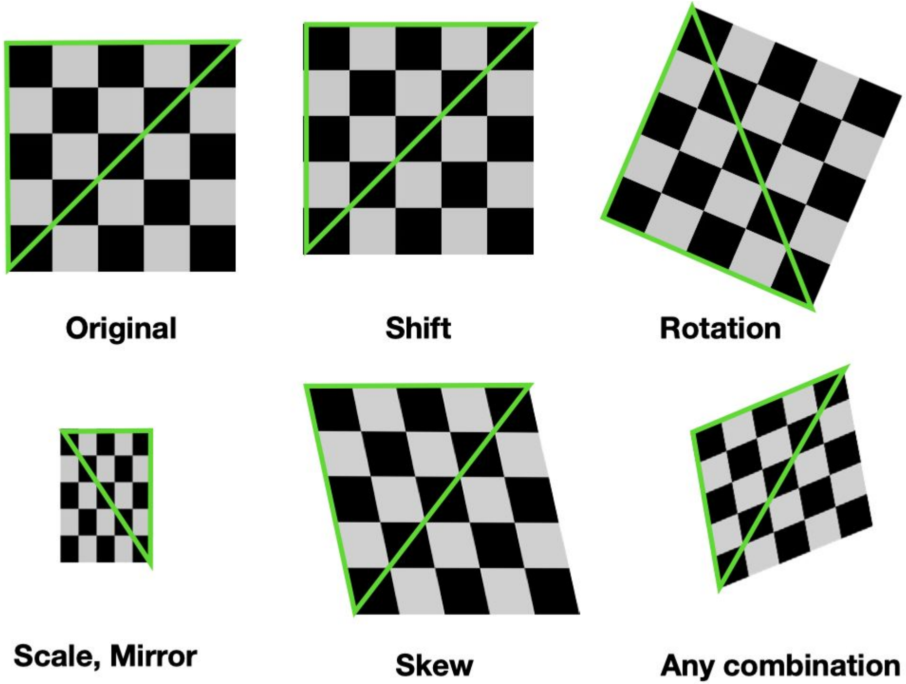
$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$A = \begin{pmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ 0 & 0 & 1 \end{pmatrix}$$

6 DoF

Affine Transformation Matrix

is defined by **3** non-collinear points.



Affine Transformation: Task 1.2 (coding)

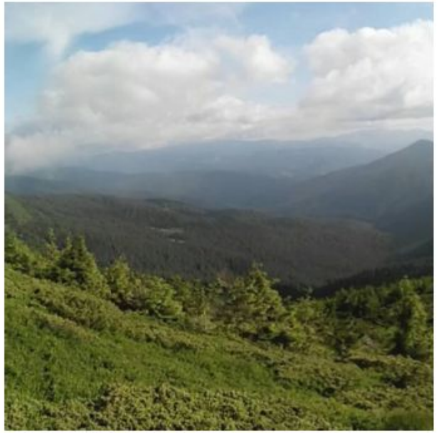
Given **3** pairs of points find affine transformation and apply it to an image.

P.S. Compare with cv2.getAffineTransform

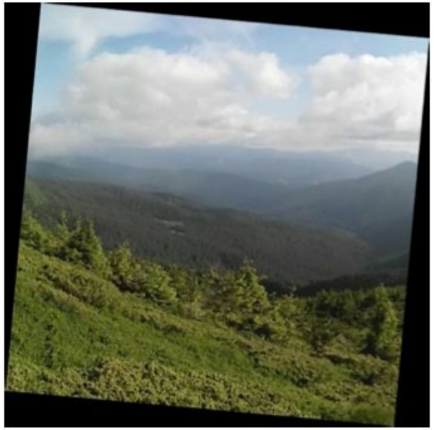
In:

$$(x_1, y_1) \to (x'_1, y'_1)$$
$$(x_2, y_2) \to (x'_2, y'_2)$$
$$(x_3, y_3) \to (x'_3, y'_3)$$

In:



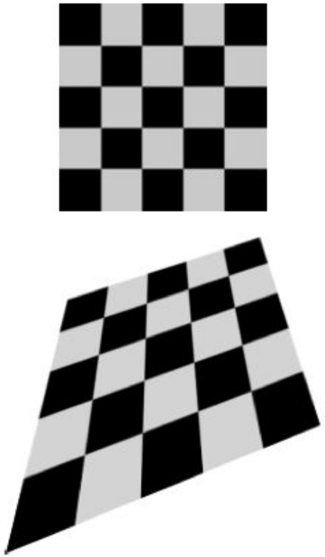
Out:



Homography a.k.a. Projective Transformation

$$x' = \frac{a_{11}x + a_{12}y + b_1}{a_{31}x + a_{32}y + b_3}$$

$$y' = \frac{a_{21}x + a_{22}y + b_2}{a_{31}x + a_{32}y + b_3}$$



Homography: Task 2.1

Rewrite projective transformation in homogeneous form

$$x' = \frac{a_{11}x + a_{12}y + b_1}{a_{31}x + a_{32}y + b_3}$$

$$y' = \frac{a_{21}x + a_{22}y + b_2}{a_{31}x + a_{32}y + b_3}$$

$$\longrightarrow \quad \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \sim ? \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

8 DoF

Homography Matrix

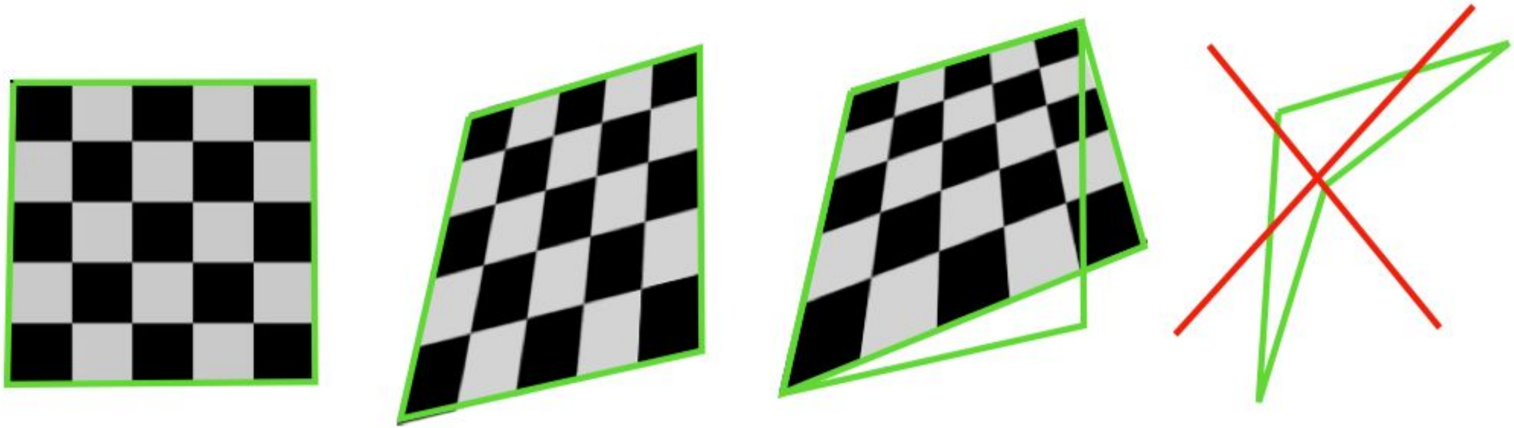
$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \sim \begin{pmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ a_{31} & a_{32} & b_3 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$H = \begin{pmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ a_{31} & a_{32} & b_3 \end{pmatrix}$$

H and ***αH*** define the same homography

Homography Matrix

is defined by **4** noncollinear points that do not break convex polygons



Homography: Task 2.2 (coding)

Given **4** pairs of points find projective transformation and apply it to an image.

P.S. Compare with cv2.getPerspectiveTransform

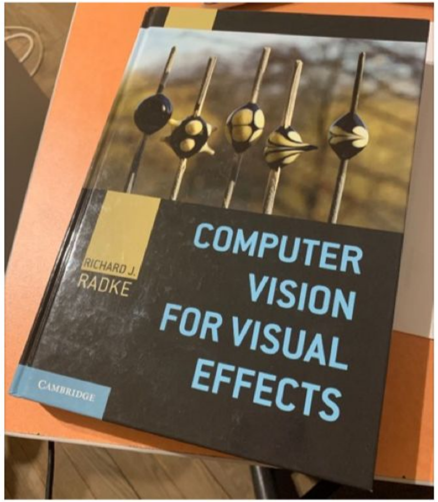
In: $(x_1, y_1) \to (x'_1, y'_1)$

$(x_2, y_2) \to (x'_2, y'_2)$

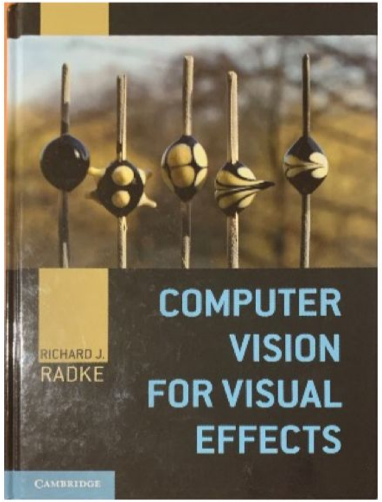
$(x_3, y_3) \to (x'_3, y'_3)$

$(x_4, y_4) \to (x'_4, y'_4)$

In:

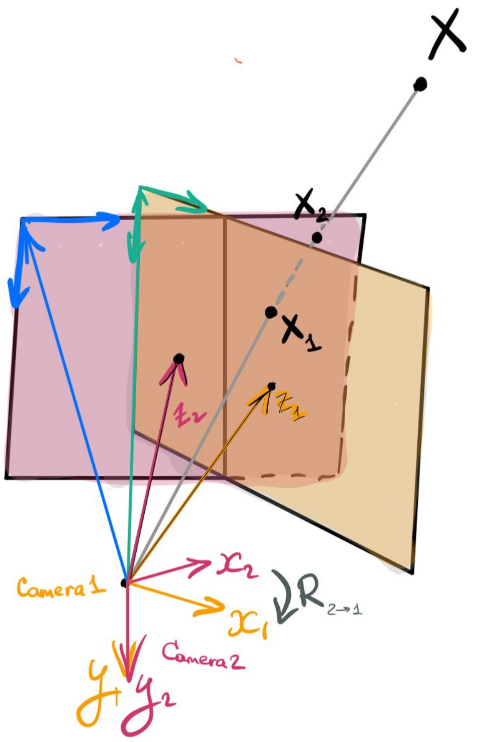


Out:



Homography: Image Planes (Conjugate Rotation)

Images from cameras with the same projection center

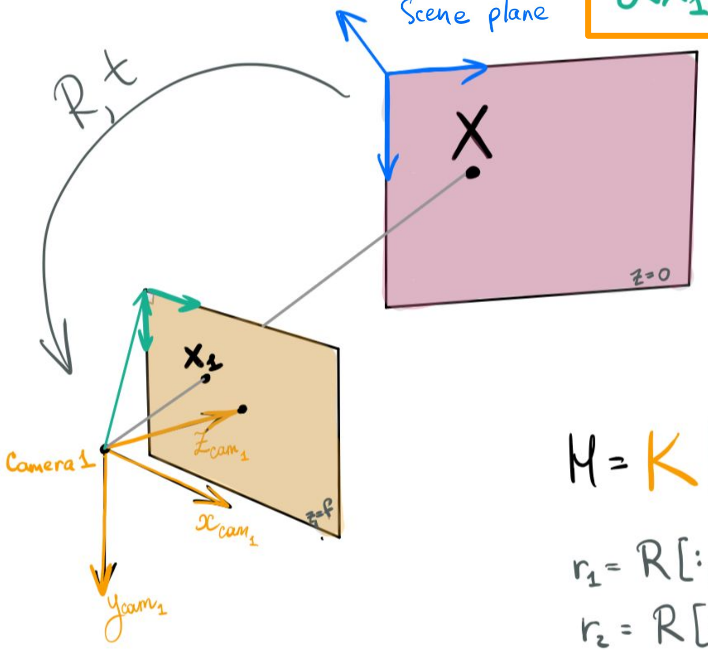


$$\boxed{\lambda x_1 = H x_2}$$

$$H = K_1 R_{2\to1} K_2^{-1}$$

Homography: Scene Plane <-> Image Plane

$$\boxed{\lambda x_1 = H X}$$

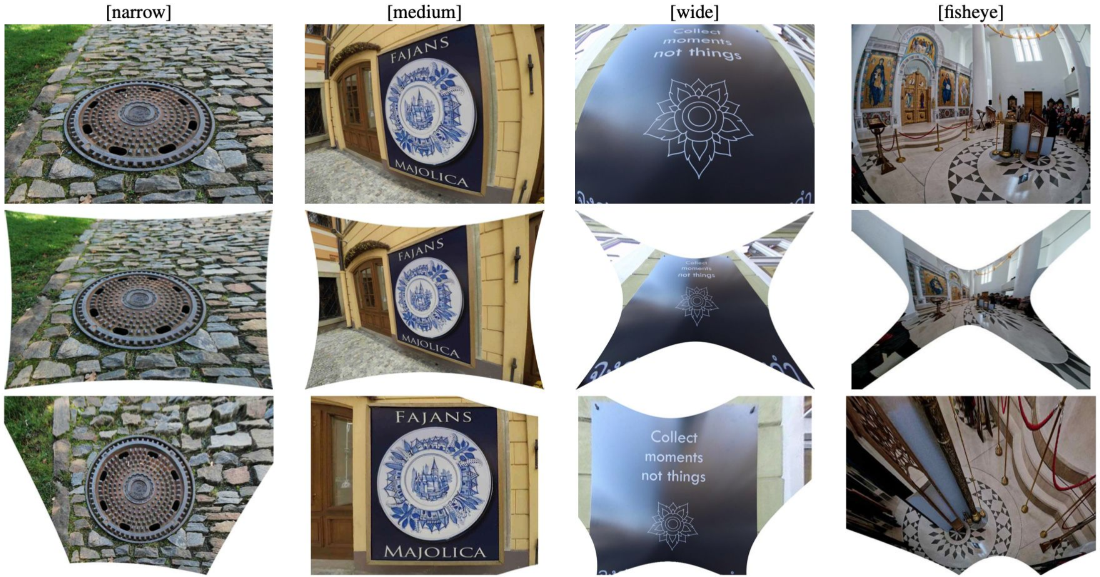


$$H = K \begin{bmatrix} r_1 & r_2 & t \end{bmatrix}$$

$$r_1 = R[:, 1]$$

$$r_2 = R[:, 2]$$

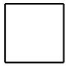
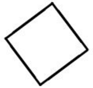
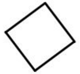
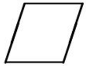
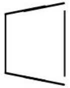
Homography: Scene Plane <-> Image Plane
Use Case: Rectification



https://arxiv.org/abs/1907.11539

https://arxiv.org/abs/1911.01507

Hierarchy of 2D Transformations

Transformation	Matrix	# DoF	Preserves	Icon
translation	$\left[\, \boldsymbol{I} \mid \boldsymbol{t} \,\right]_{2\times 3}$	2	orientation	
rigid (Euclidean)	$\left[\, \boldsymbol{R} \mid \boldsymbol{t} \,\right]_{2\times 3}$	3	lengths	
similarity	$\left[\, s\boldsymbol{R} \mid \boldsymbol{t} \,\right]_{2\times 3}$	4	angles	
affine	$\left[\, \boldsymbol{A} \,\right]_{2\times 3}$	6	parallelism	
projective	$\left[\, \tilde{\boldsymbol{H}} \,\right]_{3\times 3}$	8	straight lines	

Geometric Transformations

Consistent vs. Overdetermined SLE

Correspondence Problem

What if we do not have exact correspondences?

$$(x_i, y_i) \rightarrow (x_i', y_i') + \delta_i$$

How can we use more matches to make estimation error smaller?

What if we do not have exact correspondences?

$$(x_i, y_i) \rightarrow (x_i', y_i') + \delta_i$$

How can we use more matches to make estimation error smaller?

Use **SVD**

Ordinary Least Squares Problem

$$||A\mathbf{x} - \mathbf{y}||^2 \rightarrow min$$

where A **is a** $r \times c$ **matrix** $(r > c)$

$\mathbf{x}$ **is a vector-column of size** c

$\mathbf{y}$ **is a vector-column of size** r

$$\hat{\mathbf{x}} = \arg\min_{x} ||A\mathbf{x} - \mathbf{y}||^2$$

Let's find $\mathbf{x}$ **when** $\nabla ||A\mathbf{x} - \mathbf{y}||^2 = 0$

Ordinary Least Squares Problem

$$d(\|A\mathbf{x} - \mathbf{y}\|^2) = d((A\mathbf{x} - \mathbf{y})^T(A\mathbf{x} - \mathbf{y})) =$$

$$d(((A\mathbf{x})^T - \mathbf{y}^T)(A\mathbf{x} - \mathbf{y})) =$$

$$d((\mathbf{x}^T A^T - \mathbf{y}^T)(A\mathbf{x} - \mathbf{y})) =$$

$$d(\mathbf{x}^T A^T A\mathbf{x}) - d(\mathbf{x}^T A^T \mathbf{y}) - d(\mathbf{y}^T A\mathbf{x}) + d(\mathbf{y}^T \mathbf{y}) =$$

$$\mathbf{x}^T A^T A\mathbf{dx} + \mathbf{dx}^T A^T A\mathbf{x} - \mathbf{dx}^T A^T \mathbf{y} - \mathbf{y}^T A\mathbf{dx} =$$

are scalars, thus we can simplify $\mathbf{x}^T A^T A\mathbf{dx} = (\mathbf{x}^T A^T A\mathbf{dx})^T = \mathbf{dx}^T A^T A\mathbf{x}$

$\mathbf{y}^T A\mathbf{dx} = (\mathbf{y}^T A\mathbf{dx})^T = \mathbf{dx}^T A^T \mathbf{y}$

$$2\mathbf{dx}^T A^T A\mathbf{x} - 2\mathbf{dx}^T A^T \mathbf{y} = 2\mathbf{dx}^T(A^T A\mathbf{x} - A^T \mathbf{y})$$

$A^T A\mathbf{x} = A^T \mathbf{y}$ **makes gradient=0**

$$\mathbf{x} = (A^T A)^{-1} A^T \mathbf{y}$$

What if we have outliers?

$$(x_i, y_i) \rightarrow (x'_i, y'_i) + \delta_i$$

$$\exists i : \|\delta_i\| > \Delta$$

What if we have outliers?

$$(x_i, y_i) \rightarrow (x_i', y_i') + \delta_i$$

$$\exists i : ||\delta_i|| > \Delta$$

Use **RANSAC**

Select random **4** points, check how many constraints are satisfies, repeat.

Choose homography matrix that satisfies maximum number of constraints.

Geometric Transformations

Consistent vs. Overdetermined SLE

Correspondence Problem

What is a Feature

- Find the exact location of the patches in the original image
- How many correct results can you find?

What is a Feature

- Find the exact location of the patches in the original image
- How many correct results can you find?
- A and B are flat surfaces and they are spread over a lot of area.

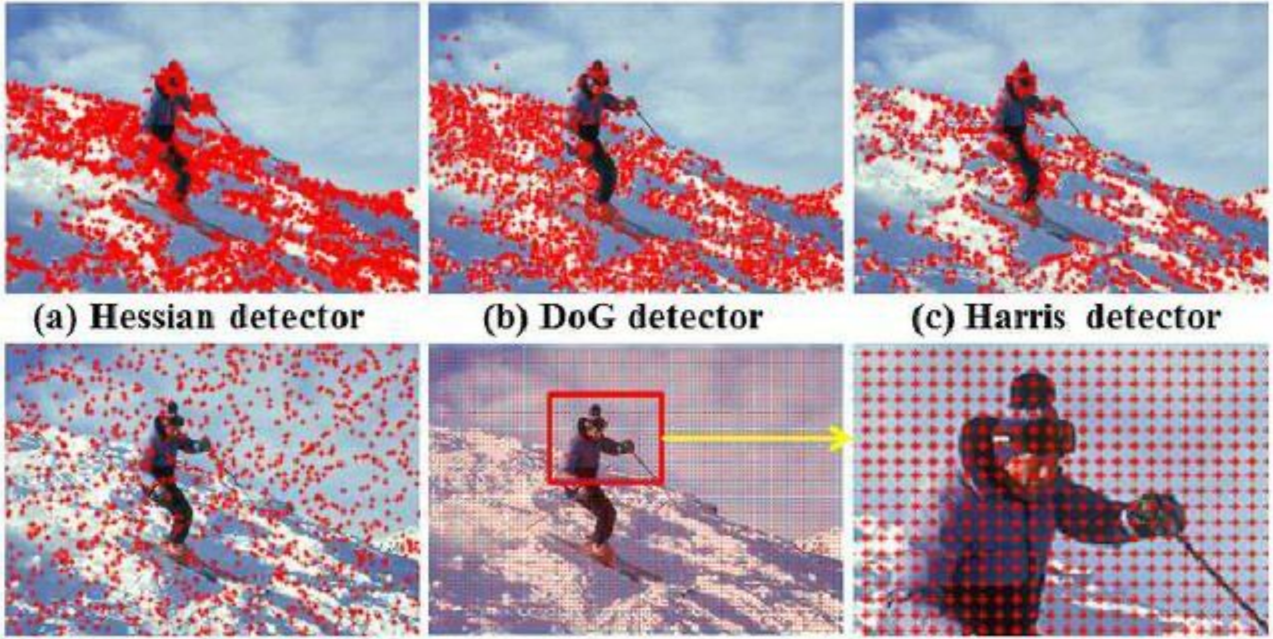
What is a Feature

- Find the exact location of the patches in the original image
- How many correct results can you find?
- A and B are flat surfaces and they are spread over a lot of area.
- C and D are edges of the building. Better, but an approximate location.

What is a Feature

- Find the exact location of the patches in the original image
- How many correct results can you find?
- A and B are flat surfaces and they are spread over a lot of area.
- C and D are edges of the building. Better, but an approximate location.
- E and F are some corners of the building, and can be easily found. Wherever you move this patch, it will look different. So they can be considered as **good features**.

What is a Feature

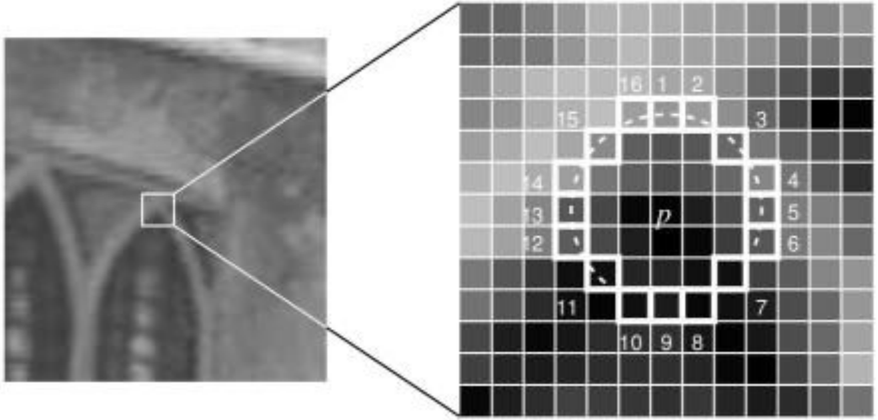


(a) Hessian detector (b) DoG detector (c) Harris detector

(d) Random sampling (e) Dense sampling (d) Zooming

Feature Detection

Features from Accelerated Segment Test — FAST Detector

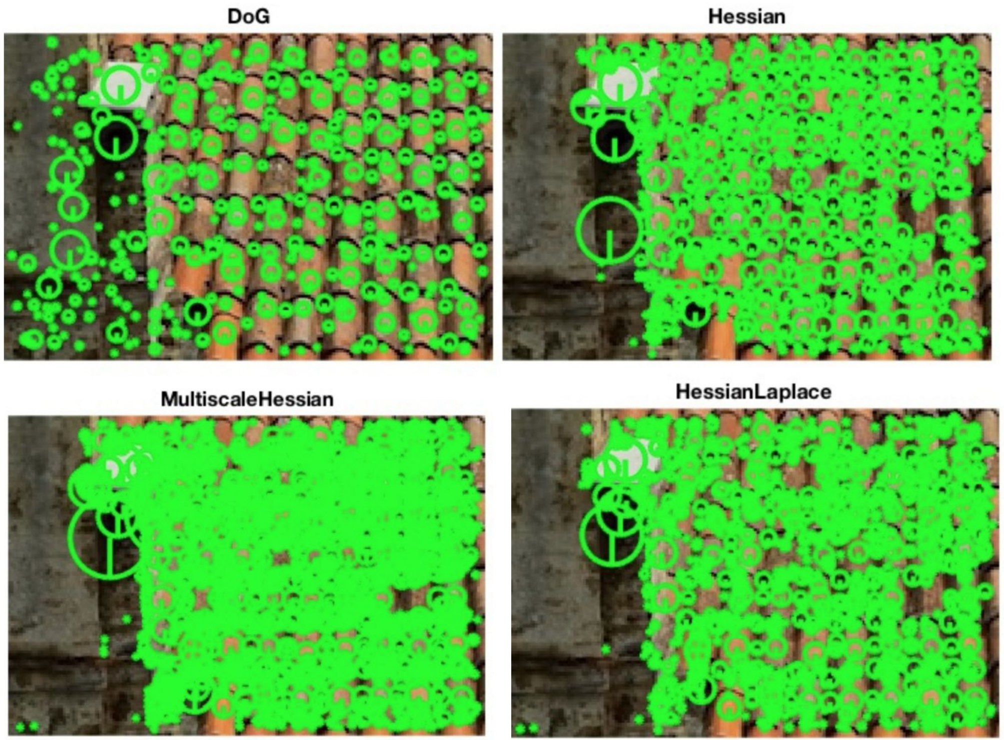


- The pixel **x** is a corner if there exists a set of **12** contiguous pixels in the circle (of 16 pixels) which are all brighter than ***I(x)+t***, or all darker than ***I(x)−t***.
- There exists a high speed test
- See more: https://docs.opencv.org/3.4/df/d0c/tutorial_py_fast.html

Feature Detection

Covariant Feature Detectors

- Difference of Gaussian (used in SIFT) — uses the local extrema trace of the multiscale Laplacian operator to detect features in scale and space.
- Hessian — uses the local extrema of the mutli-scale determinant of Hessian operator.
- Hessian Laplace — uses the extrema of the multiscale determinant of Hessian operator for localisation in space, and the extrema of the multiscale Laplacian operator for localisation in scale.



http://www.vlfeat.org/overview/covdet.html

Feature Detection

Covariant Feature Detectors

- Harris Laplace — uses the multiscale Harris cornerness measure instead of the determinant of the Hessian for localization in space, and is otherwise identical to the previous detector.
- Hessian Multiscale — detects features spatially at multiple scales by using the multiscale determinant of Hessian operator, but does not attempt to estimate their scale (it’s like the previous one, but uses the multiscale Harris measure instead).

HarrisLaplace

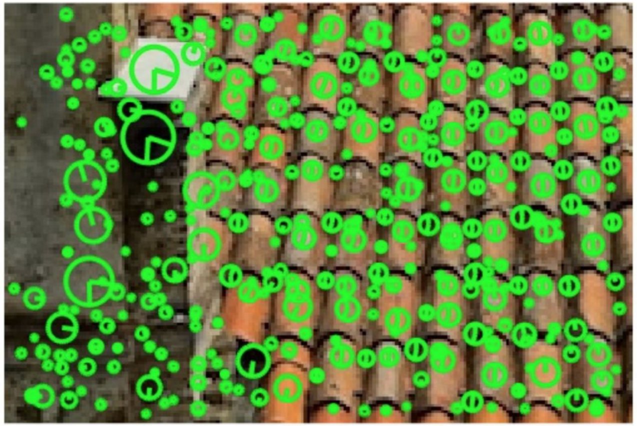
MultiscaleHarris

http://www.vlfeat.org/overview/covdet.html

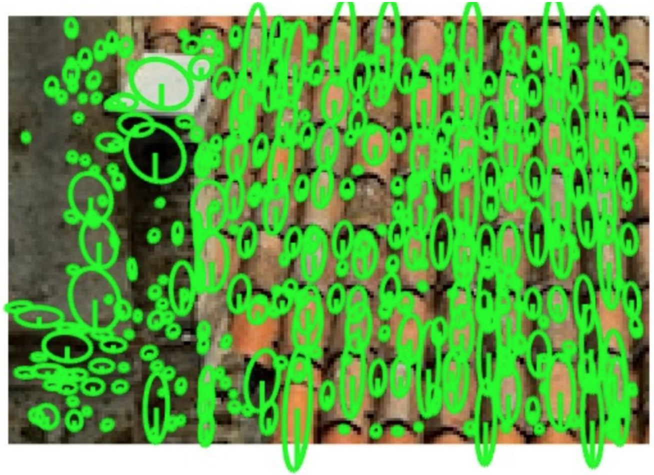
Feature Detection

Feature orientation. Estimating and removing the effect of rotation from a feature frame is needed to compute rotationally invariant descriptors.

Affine adaptation — the process of estimating the affine shape of an image region in order to construct an affinely covariant feature.



Features with orientation detection.

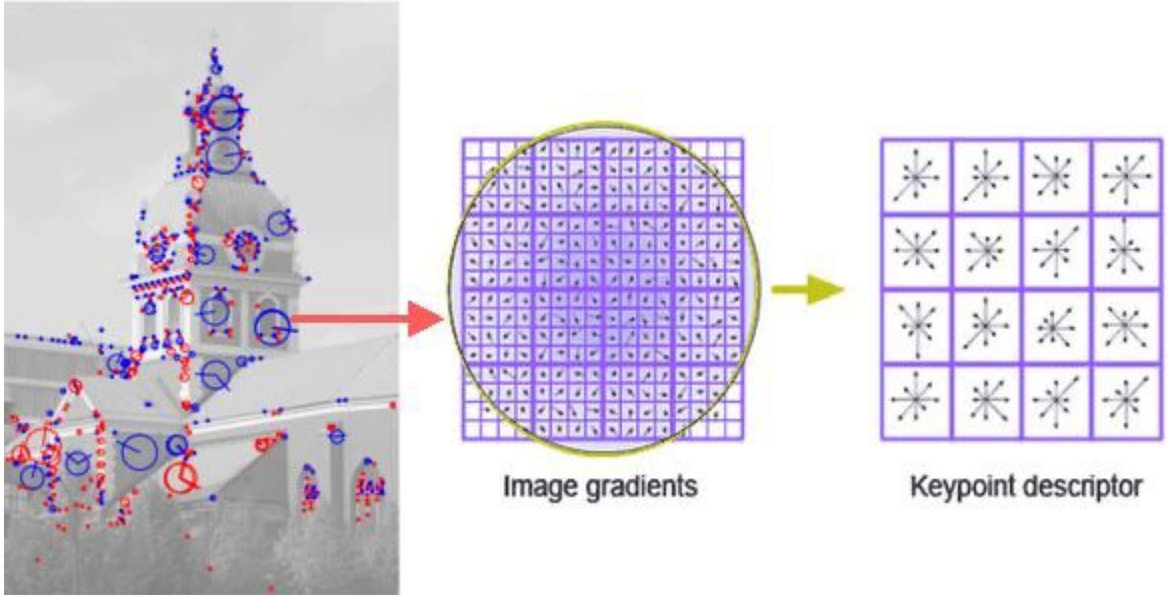


Affinely adapted features.

http://www.vlfeat.org/overview/covdet.html

Feature Description

Scale-Invariant Feature Transform (SIFT) (with detection, actually)

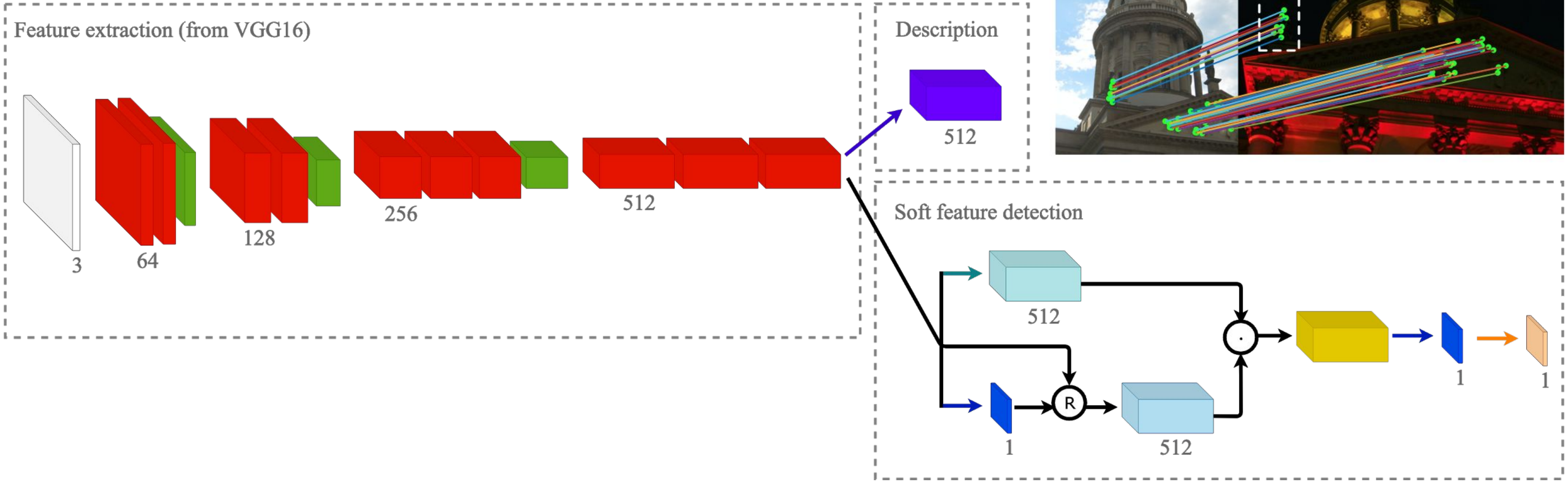


Feature Description

- **Discrete Cosine Transformation (DCT)**
- **Binary Robust Independent Elementary Features (BRIEF)**
- **Speeded-Up Robust Features (SURF)**

Feature Description

CNNs. D2-Net e.g. (with detection, actually)



Homography from matched features: Task 3.1

Given **2 images** of the **scene plane**, detect features, describe and match them (using OpenCV or PyTorch or whatever), and find homography transformation from matched features

P.S. Compare with cv2.findHomography