Photogrammetry (30540)

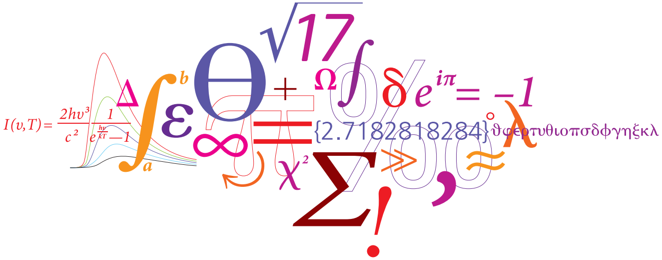
Feature Detection, Matching, RANSAC & Bundle Adjustment

Instructor: Daniel Olesen[1]

TA: Xiao Hu[2]

[1]: Assistant Professor, Dept. of Geodesy

2: PhD student



DTU Space
National Space Institute

Today's Lecture

- Image Feature
  - Feature Detection
  - Feature Description
  - Summary on Features
- Feature Matching
  - Principles
  - Matching strategy
- RANSAC
  - Principle
- Reference
  - Ending

Feature Definition

Features are recognizable structures of elements in the environment and serve as more compact and robust description of the environment.

- low-level features (geometric primitives): lines, points, blobs, circles or polygons.
- high-level features (sematic objects): signs, etc.

Low-level features: edges

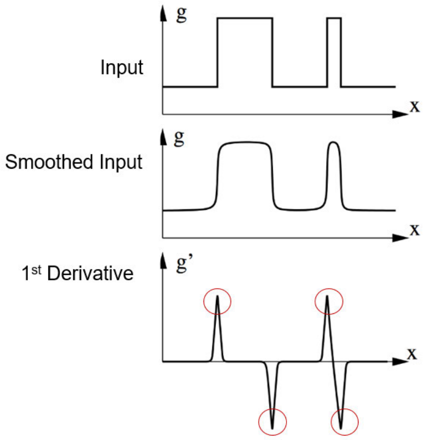
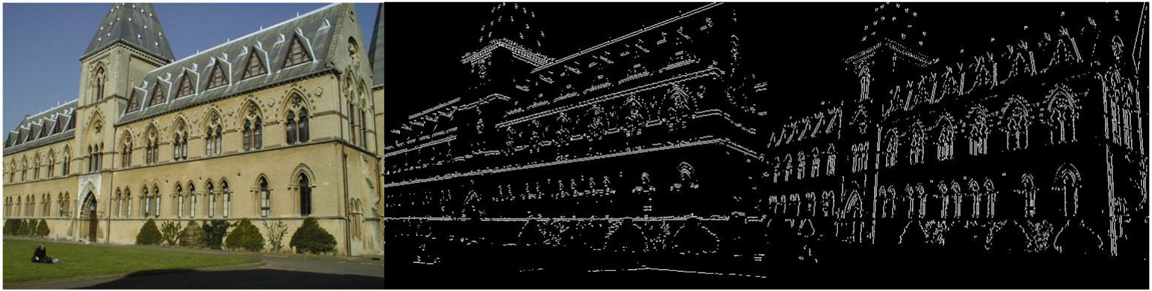
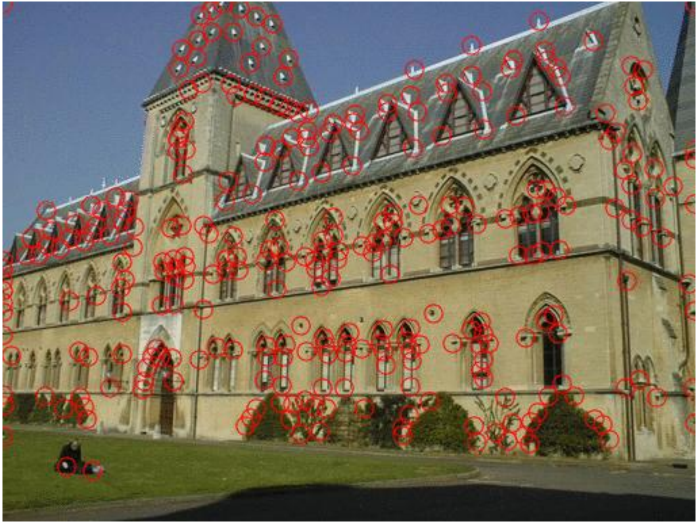


Figure: Illustration from[17].

Low-level features: edges



Low-level features: points



Low-level features: circles

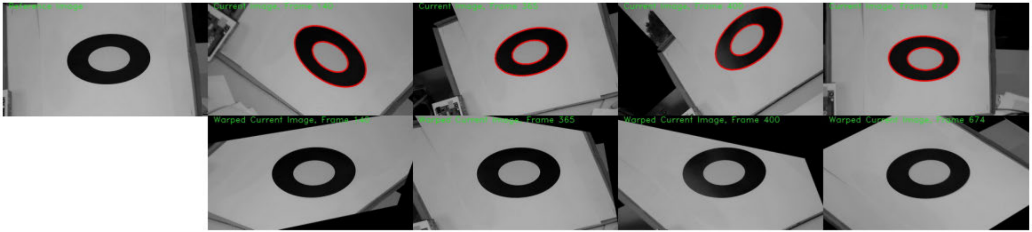


Figure: Conic detection from [16]

High-level features: sign

Figure: Sign detection from [15]

High-level features: road

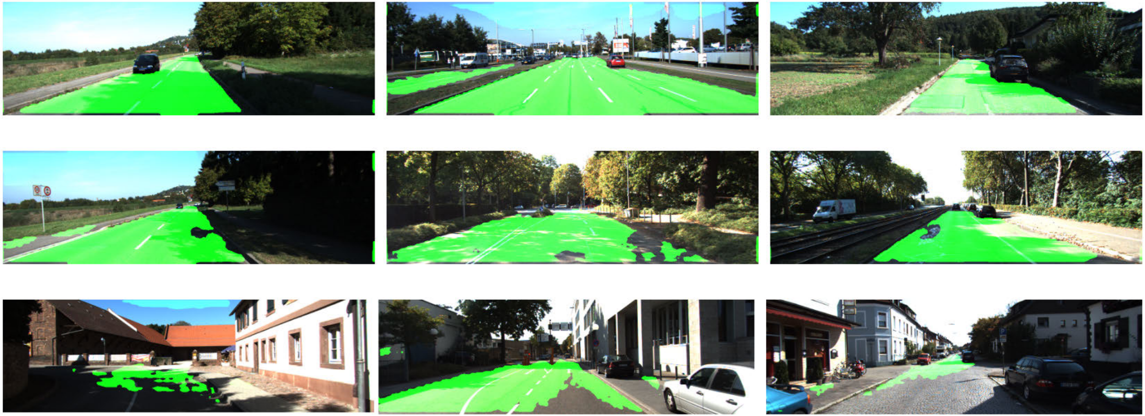
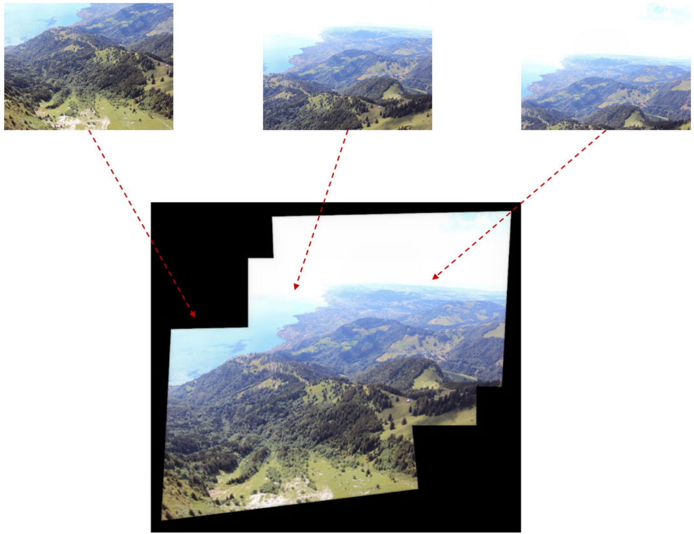


Figure: Sign detection from [14]

Application of Point features: Visual odometry

Application of Point features: Mosaicing



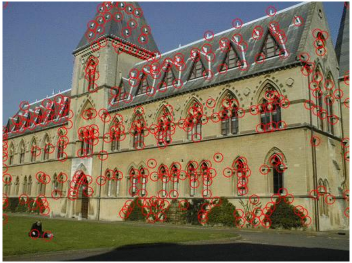
Application of Point features: 2D Mapping with Aerial images

Feature Definition

Features are recognizable structures of elements in the environment:

- low-level features (geometric primitives): lines, points, blobs, circles or polygons.
- high-level features (sematic objects): doors, tables, or trash cans.

Features serve as more compact and robust description of the environment.



Feature Detection: Keypoint Detectors

Properties of the ideal feature detector:

- Repeatability: can be redetected regardless view/illumination changes: rotation, scale (zoom), and illumination invariant.
- Distinctiveness: easy to distinguish and match.
- Localization accuracy.
- Computational efficiency.

Two major groups:

1. Corner detectors: Harris[2], FAST[3], etc.
2. Blob feature detectors: SIFT[4], etc.

Corner detector: Harris

A corner in an image can be defined as the intersection of two or more edges.

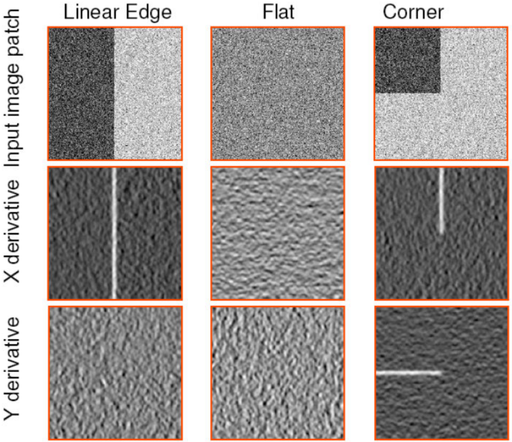


Figure: Illustration of Corner keypoint[18].

Corner detector: Harris

A corner in an image can be defined as the intersection of two or more edges.

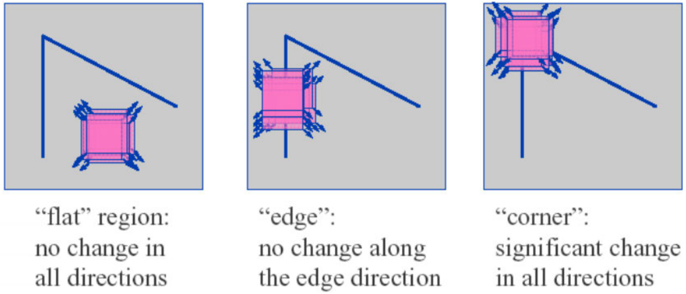


Figure: Illustration of Corner keypoint[18].

What should be inside the sliding window → the partial derivatives of the Sum of Squared Differences (SSD).

Corner detector: Harris

SSD:

$$\text{SSD}(x,y) = \sum_u \sum_v (I(u,v) - I(u+x, v+y))^2$$

$I(u+x, v+y)$ can be approximated by a first-order Taylor expansion using partial derivatives I_x and I_y:

$$I(u+x, v+y) \approx I(u,v) + I_x(u,v)x + I_y(u,v)y$$

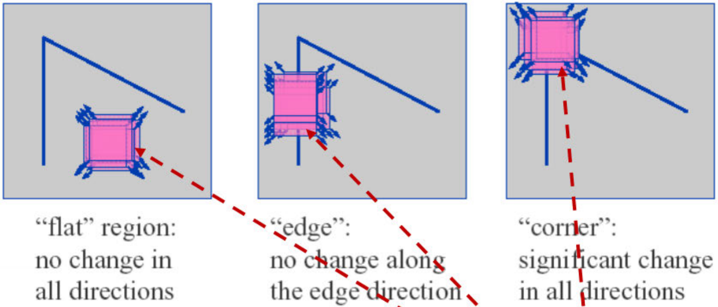
This produces the approximation of SSD:

$$\text{SSD}(x,y) \approx \sum_u \sum_v (I_x(u,v)x + I_y(u,v)y)^2 = [x,y] \underbrace{\left(\sum_u \sum_v \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \right)}_{\mathbf{M}} [x,y]^T$$

The Harris detector analyses the eigenvalues of to verify whether currecnt point is a corner or not.

Corner detector: Harris

Now we know what's inside the sliding window:



$$\mathrm{SSD}(x,y) \approx \sum_u \sum_v (I_x(u,v)x + I_y(u,v)y)^2 = [x,y] \underbrace{\left(\sum_u \sum_v \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \right)}_{\mathrm{M}} [x,y]^T$$

Corner detector: Harris

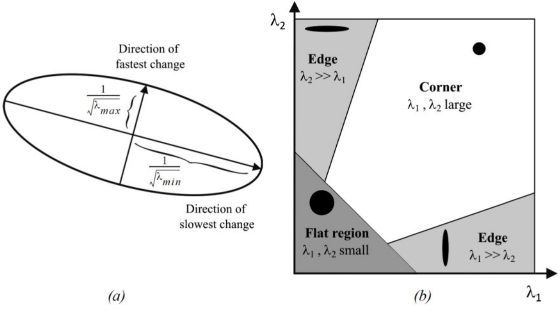


Figure: The classification of corner and edges according to Harris and Stephens[6].

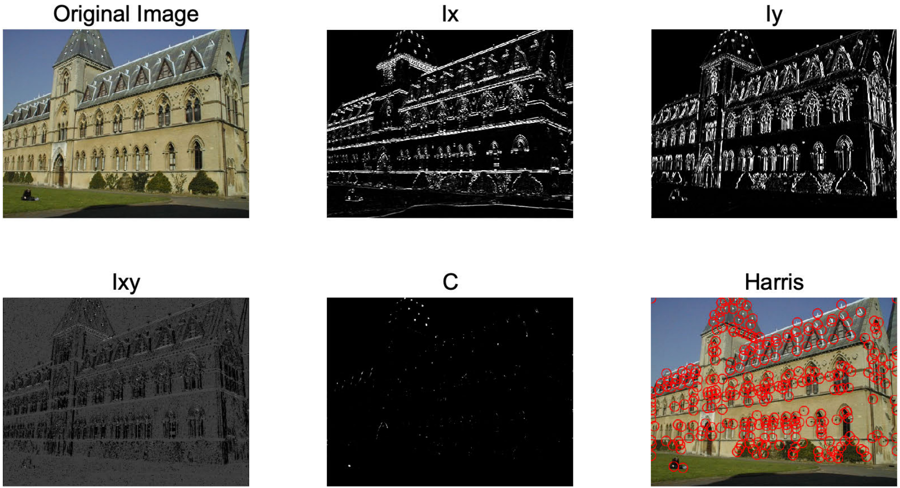
For easy computation, the following "cornerness function" is actually used:

$$C = \lambda_1\lambda_2 - k(\lambda_1+\lambda_2)^2 = \text{det}(\mathbf{M}) - k\cdot\text{trace}(\mathbf{M})^2$$

Harris corner point is then found as the local maximum by using nonmaxima suppression. In conclusion, Harris keypoint is rotation/translation invariant, but not scale invariant.

Corner detector: Harris

Let's see an example:



Other corner detectors

- Shi-Tomasi corner detector or GFTT: use $min(\lambda_1, \lambda_2)$.
- FAST: compares 16 pixels on a circle around the candidate corner. Not rotation invariant, but super fast.

Figure: Comparison of Harris and FAST.

Blob Detectors

Blob is an pattern which differs from its immediate neighborhood in terms of intensity, color, and texture.

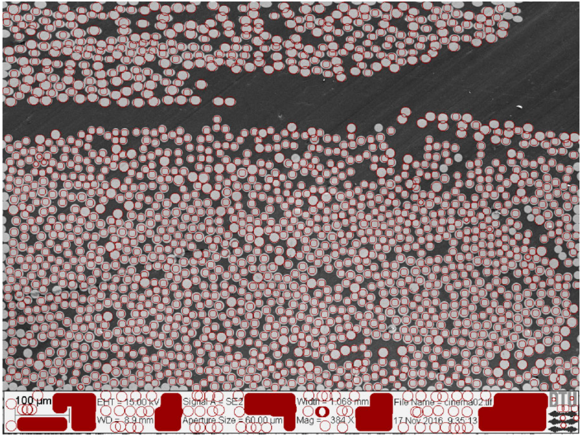


Figure: Blobs.

SIFT

Scale Invariant Feature Transform(SIFT)[4] is the most popular blob feature detector, which is widly used in mapping and navigation, etc thanks to its robustness to rotation and small changes of illumination, scale.

Comparison: SIFT V.S. Harris+BRIEF

SIFT

Main steps of the SIFT algorithm:

- find keypoint location and scale.
- find dominant orientation.
- generate keypoint descriptor.

So, SIFT = keypoint + descriptor.

Find keypoint location and scale

- Build up scale space.
- Compute Difference of Gaussian (DoG).
- Find local extrema across adjacent scales, then additional substeps like elimination points on edges or in very flat regions, interpolation of feature's location.

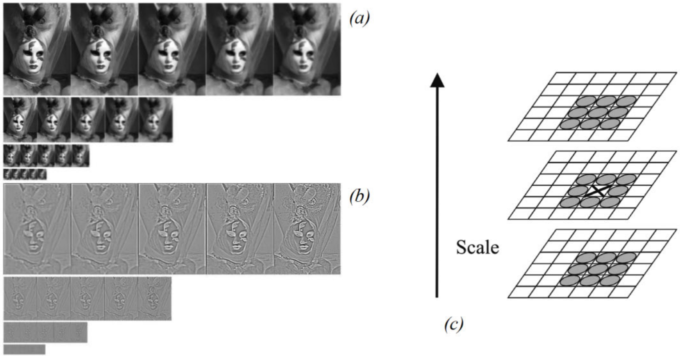


Figure: Scale space and DoG images.[6].

Find dominant orientation.

The dominant orientation is used to make the descriptor rotation invariant.

- Compute the gradient magnitude and orientation in the image with closet scale.
- Build a gradient magnitude weighted histogram of orientations within a Gaussian-weighted circular window.
- Find the peaks as the dominant orientation. If the second peak is within 80% of the first peak, an additional keypoint is created.

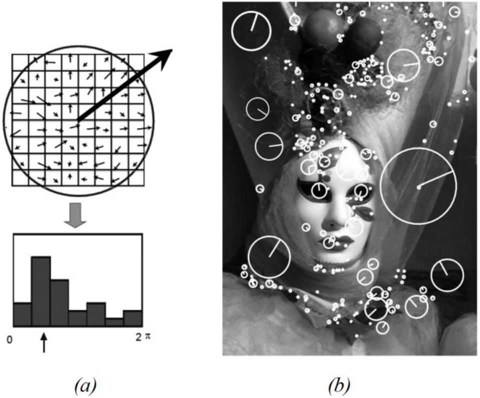


Figure: Histogram of orientation and examples of SIFT features with different scales and dominant orientations[6].

Generate feature descriptor

- Rotate the gradient orientations relative to the keypoint dominant orientation.
- The Gaussian window is then divided into 4×4 regions. A second histogram of gradient with eight orientation bins is computed within each region.
- The descriptor is built by flattening the orientation histograms and concatenating them to form a $4 \times 4 \times 8 = 128$ vector.
- Normalization, clipping and renormalization for illumination invariant.

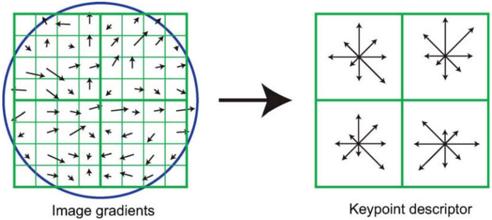


Figure: Gaussian circular windown and 4×4 regions with histograms of gradient with 8 bins[4].

Why descriptor

Keypoints alone are difficult to match, only (u, v) and scale.
Just like human being, we need the name, CPR-number, student-id, etc. to match the corresponding data or affairs belonging to you.
Feature descriptor serves as the CPR-number for keypoint. For feature matching, we actually match their descriptors.

Descriptor

Descriptor is often computed from the neighbours of the keypoint. Normally the neighbouring information is utilized in the following ways:

- Histogram of Gradients (HoG) descriptors, like SIFT.
- Binary descriptors, like BRIEF[8].

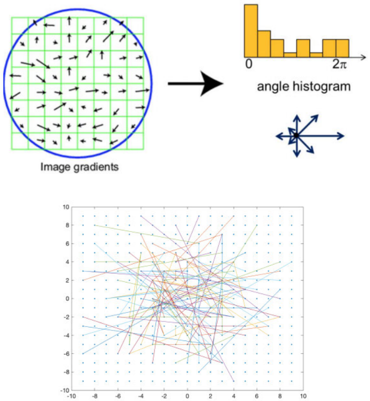


Figure: Histogram of Gradients (HoG) descriptor and Binary descriptor.

Summary

	Corner detector	Blob detector	Rotation invariant	Scale invariant	Affine invariant	Repeatability	Localization accuracy	Robustness	Efficiency
Harris	x		x			+++	+++	++	++
Shi-Tomasi	x		x			+++	+++	++	++
Harris-Laplacian	x	x	x	x		+++	+++	++	+
Harris-Affine	x	x	x	x	x	+++	+++	++	++
SUSAN	x		x			++	++	++	+++
FAST	x		x			++	++	++	++++
SIFT		x	x	x	x	+++	++	+++	+
MSER		x	x	x	x	+++	+	+++	+++
SURF		x	x	x	x	++	++	++	++

Figure: Comparison of feature detectors: properties and performance[6].

Summary

- SIFT is the most used feature in modern Photogrammetry.
- ORB and Harris are often used in visual odometry, VSLAM, tracking.
- Hand-crafted features are now challenged by deep learning based features, e.g. LIFT[9] V.S. SIFT. More learning based features are emerging. However, their usages in Photogrammetry are still in development (could be an research interest).

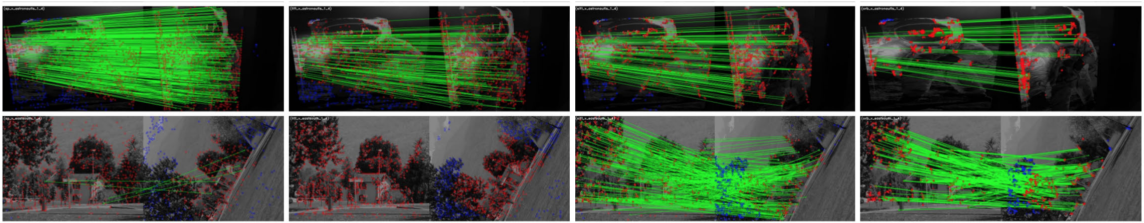


Figure: Comparison of Superpoint, LIFT, SIFT, and ORB[10].

Principles

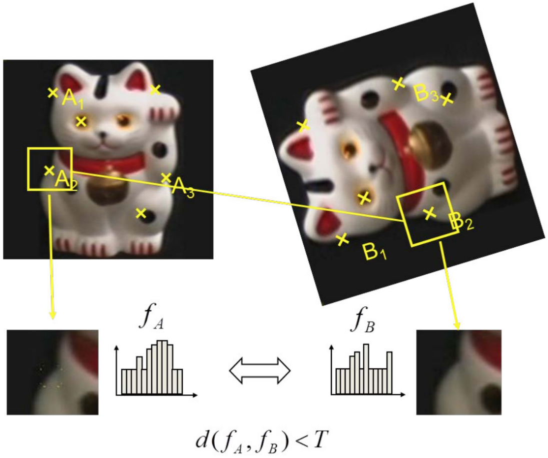


Figure: Principle of feature matching[11].

Different distance functions for comparing the similarity of two descriptors:

- L_2 distance, for descriptors with floating value (SIFT):

$$d(f_a, f_b) = \sum_i ||(f_a(i) - f_b(i))||_2$$

- Hamming distance, for descriptors with binary value (BRIEF):

$$d(f_a, f_b) = \sum_i \text{XOR}(f_a(i), f_b(i))$$

- others: L_1 distance, etc.

Matching strategy

Comparing strategy:

- Brute-force matching: for a in image 1, compare all b in image 2.
- Kd tree: binary search in Kd tree.

Selecting strategy:

- Take the nearest.
- Take the nearest with the distance below a threshold.
- Take the two closest and perform ratio test:

$$\frac{d_{closet}}{d_{second\_closet}} < \tau$$

  τ is usually set to 0.7.
- Cross-check: f_b is the best match for f_a in image 2 and f_a is the best match for f_b in image 1.

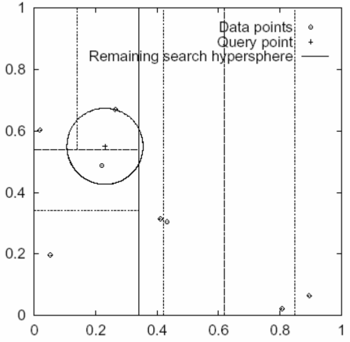


Figure: Feature matching using Kd tree[12].

Example

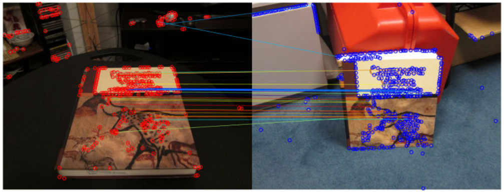


Figure: FAST+BRIEF matching with ratio test.

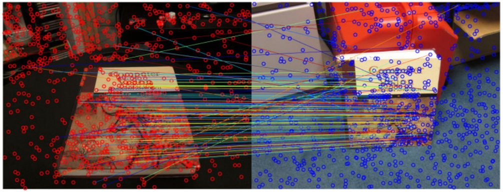


Figure: SIFT matching with ratio test.

Principle

Why need RANSAC?
Feature matching result often contains a certain amount of false matches (**Outliers**). Outlier will severally deteriorate the estimation result.
Generally speaking, **RANdom SAmple Consensus (RANSAC)[13] is an iterative method for estimating model parameters from observations containing outliers**.
The model could be a line, a parabola, an ellipse, etc.

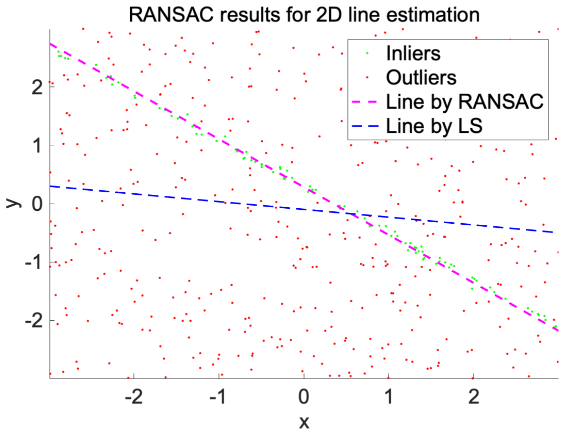


Figure: Line estimation results.

RANSAC Pipeline

- Randomly select n samples from observations, n is the minimum number of samples needed for model estimation, e.g. $n = 2$ for 2D line, $n = 3$ for 3D plane, etc.
- Model estimation, could be LS, SVD, etc.
- Compute Consensus: apply a model-specific loss function to each observation and the model obtained, the response could serve as the consensus, e.g. point-line distance, point-plane distance.
- Classifier inliers and outliers using a predefined threshold and log the model with the maximum number of inliers.
- Iterate.

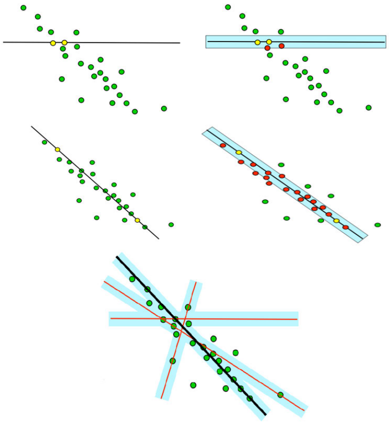


Figure: RANSAC Procedure.

How many iterations to choose

Let p_{out} be the probability that one point is an outlier, n be the minimum number of samples for model estimation, N be the number of iterations, p be the desired probability that we get a good sample:

$$p = 1 - (1 - (1 - p_{out})^n)^N$$

- $1 - p_{out}$: Probability of an inlier.
- $(1 - p_{out})^n$: Probability of choosing n inliers.
- $(1 - (1 - p_{out})^n)$: Probability of at least one items in the sample being outliers for one iteration.
- $(1 - (1 - p_{out})^n)^N$: Probability that N iterations are contaminated.
- $1 - (1 - (1 - p_{out})^n)^N$: Probability that at least one iteration is not contaminated.

Usually, we set p and then compute N backwardly as $N = \frac{\log(1-p)}{\log(1-(1-p_{out})^n)}$.

How many iterations to choose

A look-up table for N when p i set to 0.99.

			p_{out}				
n	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

Example of RANSAC

Figure: Feature matching before applying RANSAC.

As you can see, false matches get hypothesized.Let's apply RANSAC + epipolar constraint.

Example of RANSAC

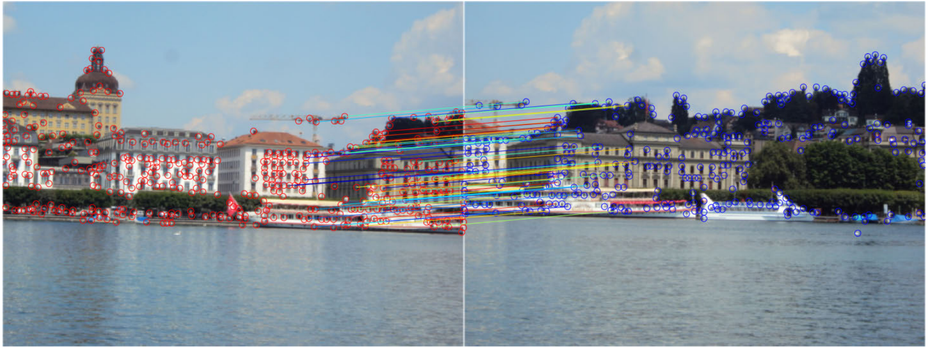


Figure: Feature matching after applying RANSAC.

As you can see, false matches are removed.

Thank You!

- https://en.wikipedia.org/wiki/Mars_Exploration_Rover.
- Chris Harris and Mike Stephens (1988). "A Combined Corner and Edge Detector". Alvey Vision Conference. 15.
- Rosten, Edward, and Tom Drummond. "Machine learning for high-speed corner detection." European conference on computer vision. Springer, Berlin, Heidelberg, 2006.
- Lowe, David G. "Distinctive image features from scale-invariant keypoints." International journal of computer vision 60.2 (2004): 91-110.
- https://www.uio.no/studier/emner/matnat/its/UNIK4690/v16/forelesninger/lecture_3_2_1_corner_features.pdf.
- Siegwart, Roland, Illah Reza Nourbakhsh, and Davide Scaramuzza. Introduction to autonomous mobile robots. MIT press, 2011.
- J. Shi and C. Tomasi. Good Features to Track,. 9th IEEE Conference on Computer Vision and Pattern Recognition. Springer. June 1994.

- Calonder, Michael, et al. "BRIEF: Computing a local binary descriptor very fast." IEEE Transactions on Pattern Analysis and Machine Intelligence 34.7 (2012): 1281-1298.
- Yi, Kwang Moo, et al. "Lift: Learned invariant feature transform." European Conference on Computer Vision. Springer, Cham, 2016.
- DeTone, Daniel, Tomasz Malisiewicz, and Andrew Rabinovich. "Superpoint: Self-supervised interest point detection and description." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2018.
- https://www.uio.no/studier/emner/matnat/its/UNIK4690/v16/forelesninger/lecture_4_2_feature_matching.pdf
- https://www.cs.ubc.ca/~nando/nipsfast/slides/fast04.pdf
- Fischler, Martin A., and Robert C. Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography." Communications of the ACM 24.6 (1981): 381-395.

- Mendes, Caio César Teodoro, Vincent Frémont, and Denis Fernando Wolf. "Exploiting fully convolutional neural networks for fast road detection." 2016 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2016.
- Wang, Bihao, et al. "Landmarks based human-like guidance for driving navigation in an urban environment." 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2017.
- Hua, Minh-Duc, et al. "Explicit complementary observer design on Special Linear Group SL (3) for homography estimation using conic correspondences." 2017 IEEE 56th Annual Conference on Decision and Control (CDC). IEEE, 2017.
- Förstner, Scriptum Photogrammetry I, Chapter “Kantenextraktion".
- Robert Collins, CSE486, Penn State, Lecture 06: Harris Corner Detector.