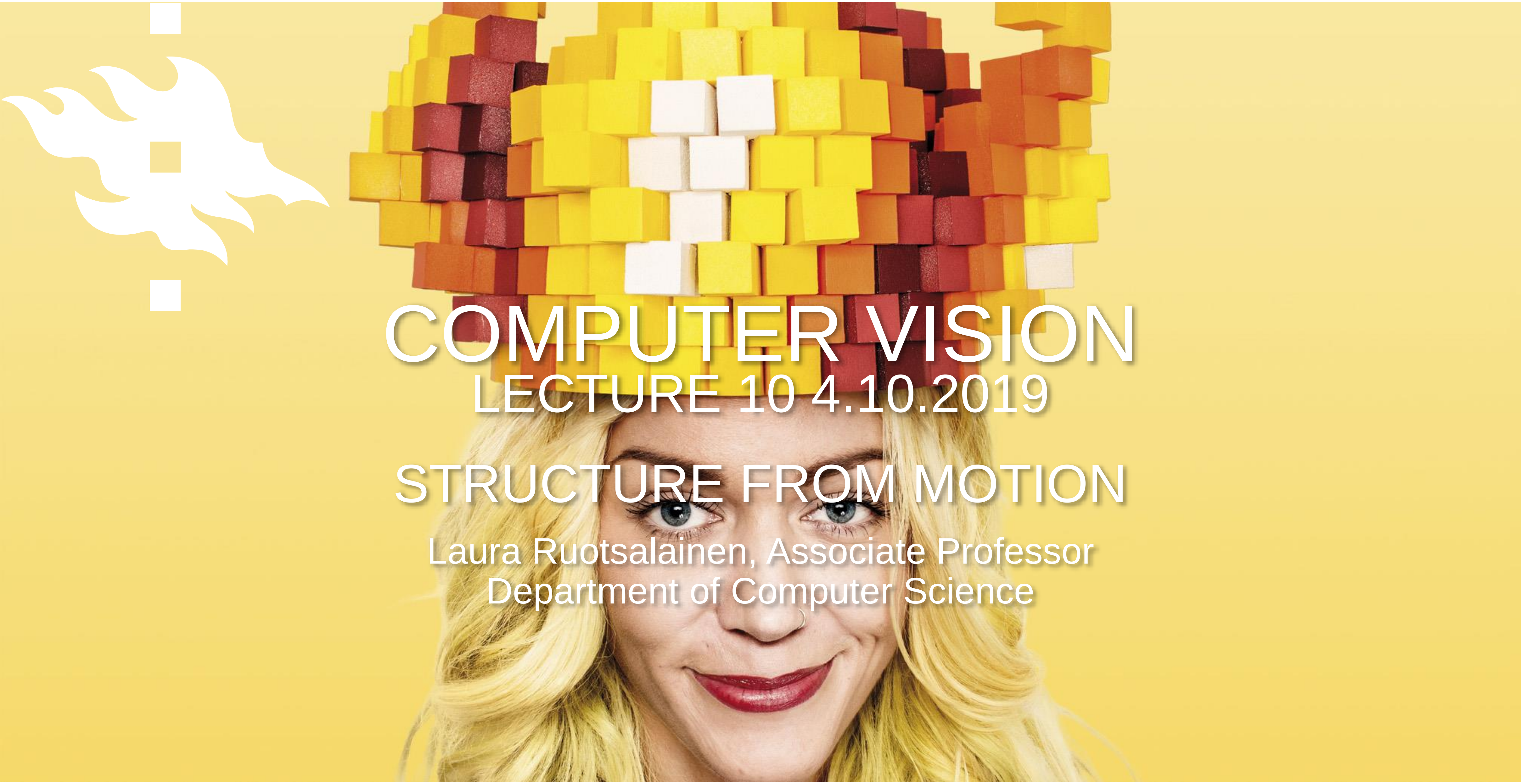


COMPUTER VISION

LECTURE 10 4.10.2019

STRUCTURE FROM MOTION

Laura Ruotsalainen, Associate Professor
Department of Computer Science

TODAY’S LECTURE

- Structure from Motion (SFM)
- Bundle adjustment
- SLAM basics, mapping
- Bag-of-Words, used both in
  - Simultaneous Localization and Mapping (SLAM, Lecture 11) and
  - Object recognition (Lecture 12)

- Basic reading:
  - Szeliski textbook, Chapters 7, 14
  - Hartley and Zisserman, Chapter 18

  - Durrant-Whyte, Bailey (2006), SLAM tutorial I, II

  https://people.eecs.berkeley.edu/~pabbeel/cs287-fa09/readings/Durrant-Whyte_Bailey_SLAM-tutorial-I.pdf
  https://people.eecs.berkeley.edu/~pabbeel/cs287-fa09/readings/Bailey_Durrant-Whyte_SLAM-tutorial-II.pdf

STRUCTURE FROM MOTION

- SFM solves both the 3D object locations and camera parameters at the same time
- However, it is impossible to recover the absolute scale of the scene!

Reconstruction
(2 view structure from motion)

Given a set of matched points

$$\{\boldsymbol{x}_i, \boldsymbol{x}'_i\}$$

Estimate the camera matrices

$$\mathbf{P}, \mathbf{P}'$$

Estimate the 3D point

$$\mathbf{X}$$

‘motion’
(of the cameras)

‘structure’

TWO-VIEW SFM

1. Compute the Fundamental Matrix **F** from points correspondences
   8-point algorithm

2. Compute the camera matrices **P** from the Fundamental matrix
   $\mathbf{P} = [\ \mathbf{I} \mid \mathbf{0}\ ]$ and $\mathbf{P}' = [\ [\mathbf{e}_{\times}]\mathbf{F} \mid \mathbf{e}'\ ]$

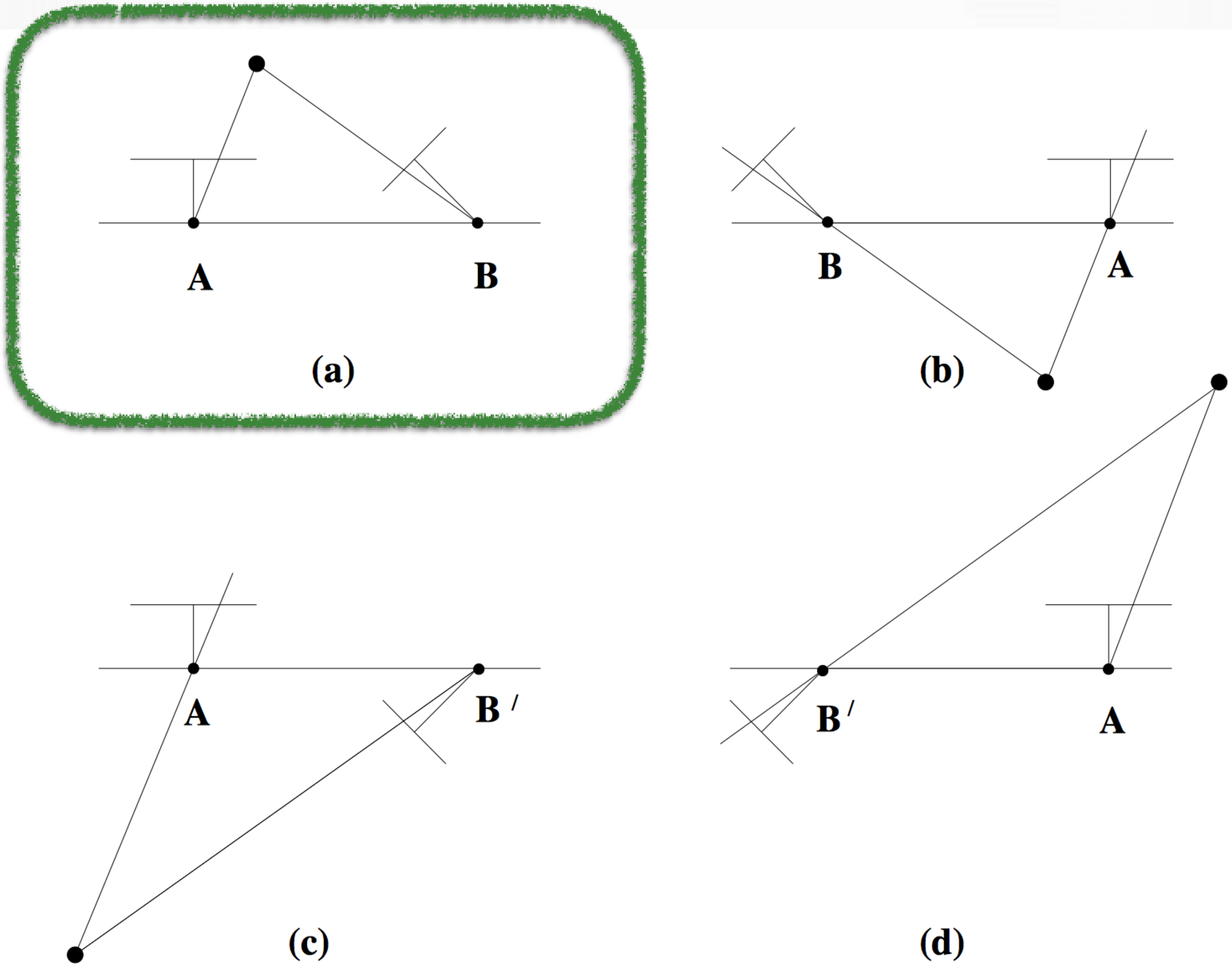
Slide credit: Kris Kitani

PROJECTIVE SFM: TWO-CAMERA CASE

- Compute fundamental matrix $\mathbf{F}$ between the two views
- First camera matrix: $[\mathbf{I}|\mathbf{0}]$
- Second camera matrix: $[\mathbf{A}|\mathbf{b}]$
- Then $\mathbf{b}$ is the epipole ($\mathbf{F}^T\mathbf{b} = 0$), $\mathbf{A} = -[\mathbf{b}_\times]\mathbf{F}$

Slide credit: Kris Kitani

Find the configuration where the points is in front of both cameras



Slide credit: Kris Kitani

TWO-VIEW SFM

1. Compute the Fundamental Matrix **F** from points correspondences
   8-point algorithm
2. Compute the camera matrices **P** from the Fundamental matrix
   $\mathbf{P} = [\ \mathbf{I} \mid \mathbf{0}\ ]$ and $\mathbf{P}' = [\ [\mathbf{e}'_{x}]\mathbf{F} \mid \mathbf{e}'\ ]$
3. For each point correspondence, compute the point **X** in 3D space (triangularization)
   DLT with $x = \mathbf{P}\,X$ and $x' = \mathbf{P}'\,X$

Slide credit: Kris Kitani

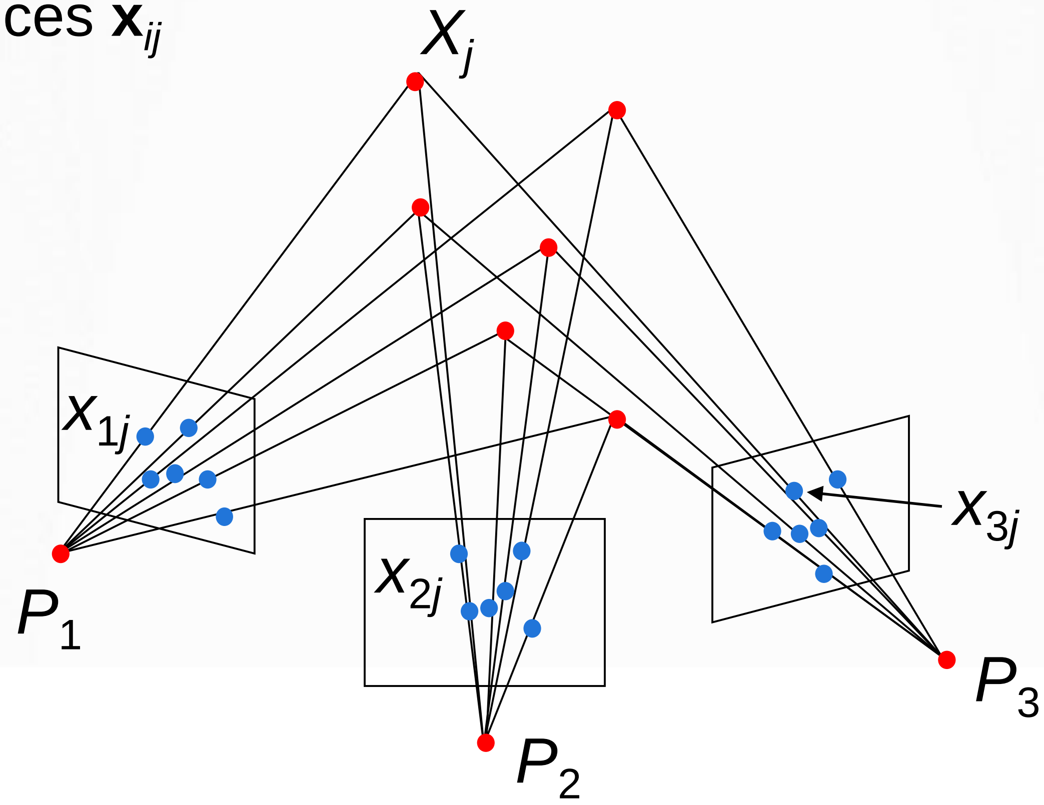
MULTI-VIEW PROJECTIVE STRUCTURE FROM MOTION

PROJECTIVE STRUCTURE FROM MOTION

- Given: m images of n fixed 3D points

$$z_{ij}\,\mathbf{x}_{ij} = \mathbf{P}_i\,\mathbf{X}_j\,, \quad i = 1, \dots, m, \quad j = 1, \dots, n$$

- Problem: estimate m projection matrices $\mathbf{P}_i$ and n 3D points $\mathbf{X}_j$ from the mn correspondences $\mathbf{x}_{ij}$



Slide credit: Kris Kitani

PROJECTIVE STRUCTURE FROM MOTION

- Given: *m* images of *n* fixed 3D points

$$z_{ij}\,\mathbf{x}_{ij} = \mathbf{P}_i\,\mathbf{X}_j\,, \quad i = 1,\ldots, m, \quad j = 1, \ldots, n$$

- Problem: estimate *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the *mn* correspondences $\mathbf{x}_{ij}$
- With no calibration info, cameras and points can only be recovered up to a 4x4 projective transformation **Q**:

$$\mathbf{X} \rightarrow \mathbf{QX},\ \mathbf{P} \rightarrow \mathbf{PQ}^{-1}$$

- We can solve for structure and motion when

$$2mn >= 11m + 3n - 15$$

- For two cameras, at least 7 points are needed

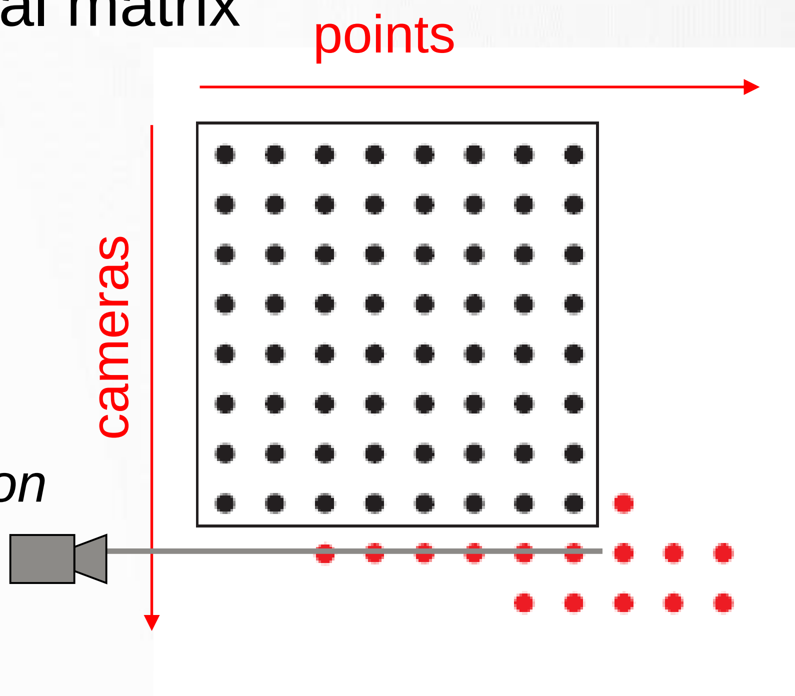
Slide credit: Kris Kitani

SEQUENTIAL STRUCTURE FROM MOTION

- Initialize motion from two images using fundamental matrix
- Initialize structure by triangulation

- For each additional view:

  Determine projection matrix of new camera using all
  the known 3D points that are visible in its image – *calibration*



Slide credit: Kris Kitani

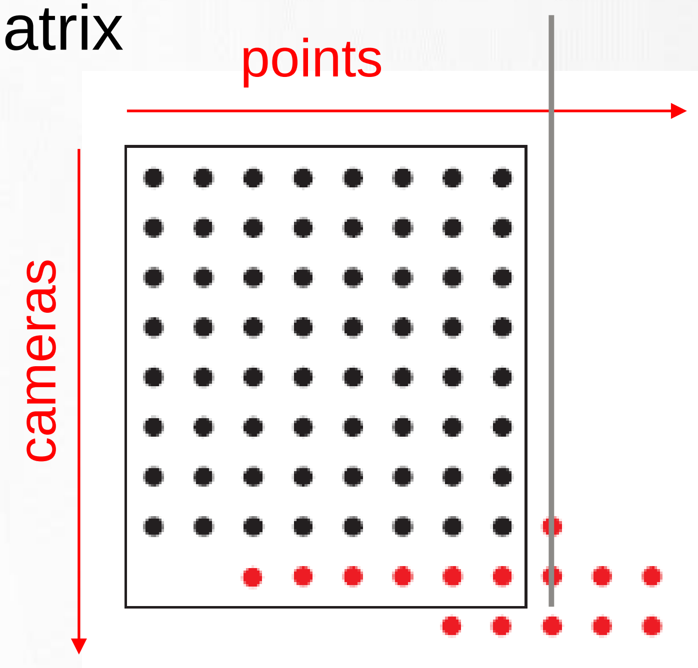
SEQUENTIAL STRUCTURE FROM MOTION

- Initialize motion from two images using fundamental matrix
- Initialize structure by triangulation

- For each additional view:

  Determine projection matrix of new camera using all the known 3D points that are visible in its image – *calibration*

  Refine and extend structure: compute new 3D points, re-optimize existing points that are also seen by this camera – *triangulation*



Slide credit: Kris Kitani

SEQUENTIAL STRUCTURE FROM MOTION

- Initialize motion from two images using fundamental matrix

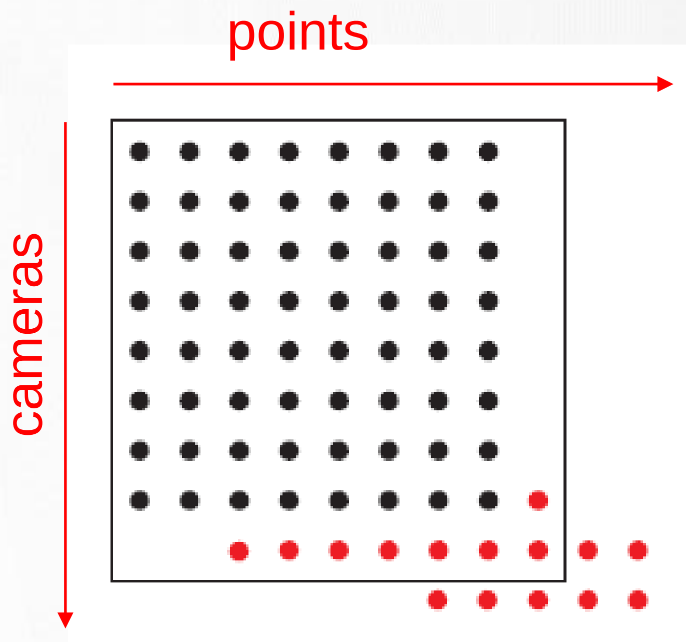
- Initialize structure by triangulation

- For each additional view:

  Determine projection matrix of new camera using all the known 3D points that are visible in its image – *calibration*

  Refine and extend structure: compute new 3D points, re-optimize existing points that are also seen by this camera – *triangulation*

- Refine structure and motion: bundle adjustment

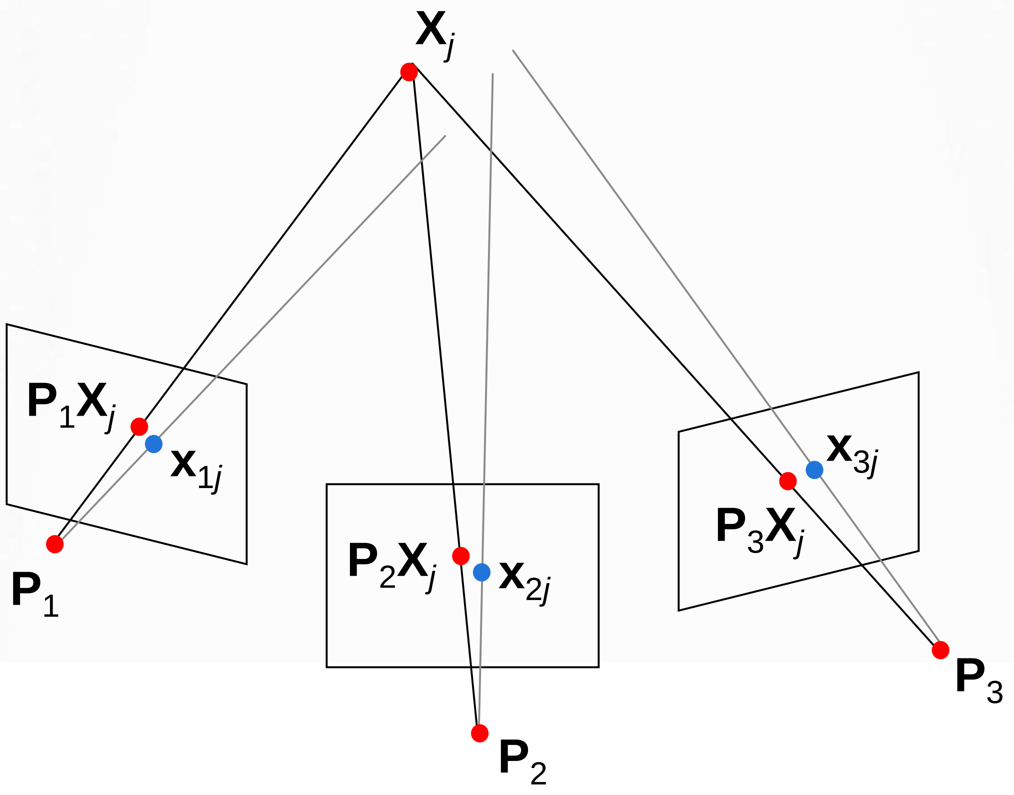


Slide credit: Kris Kitani

BUNDLE ADJUSTMENT

- Non-linear method for refining structure and motion
- Minimizing reprojection error

$$E(\mathbf{P},\mathbf{X}) = \sum_{i=1}^{m}\sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, \mathbf{P}_i\mathbf{X}_j\right)^2$$



Slide credit: Kris Kitani

GLOBAL STRUCTURE FROM MOTION

Minimize sum of squared reprojection errors:

$$g(\mathbf{X}, \mathbf{R}, \mathbf{T}) = \sum_{i=1}^{m} \sum_{j=1}^{n} \underbrace{w_{ij}} \cdot \left\| \underbrace{\mathbf{P}(\mathbf{x}_i, \mathbf{R}_j, \mathbf{t}_j)} - \underbrace{\begin{bmatrix} u_{i,j} \\ v_{i,j} \end{bmatrix}} \right\|^2$$

indicator variable:
is point *i* visible in image *j* ?

predicted
image location

observed
image location

Minimizing this function is called *bundle adjustment*

Slide credit: Kris Kitani

DOING BUNDLE ADJUSTMENT

- Minimizing *g* is difficult
  - *g* is non-linear due to rotations, perspective division
  - lots of parameters: 3 for each 3D point, 6 for each camera
  - difficult to initialize
  - gauge ambiguity: error is invariant to a similarity transform (translation, rotation, uniform scale)
- Bundle adjustment requires non-linear least-squares (NLLS) optimization (*bundle adjustment*)
  - Levenberg-Marquardt is one common algorithm for NLLS
  - Lourakis, **The Design and Implementation of a Generic Sparse Bundle Adjustment Software Package Based on the Levenberg-Marquardt Algorithm**, http://www.ics.forth.gr/~lourakis/sba/

Slide credit: Kris Kitani

LEVENBERG-MARQUARDT

- Least-squares fitting is used in multiple computer vision problems
- E.g. alignment of images by matching features

$$E_{\mathrm{LS}} = \sum_i \|\boldsymbol{r}_i\|^2 = \sum_i \|\boldsymbol{f}(\boldsymbol{x}_i;\boldsymbol{p}) - \boldsymbol{x}_i'\|^2,$$

- $E_{\mathrm{LLS}} = \sum_i |\boldsymbol{a}_i \boldsymbol{x} - b_i|^2 = \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2$ minimum by solving $(\boldsymbol{A}^T\boldsymbol{A})\boldsymbol{x} = \boldsymbol{A}^T\boldsymbol{b}.$

  General version

- Function needs to be linear in unknown parameters => for non-linear functions non-linear least-squared fillting is required, Levenberg-Marquardt is the most used method for that (Szeliski A.3, http://en.wikipedia.org/wiki/Levenberg-Marquardt_algorithm)
  - First-order Taylor series expansion
  - Damping parameter λ, changing according to residual's decreasing speed

SFM APPLICATIONS

- 3D modeling
- Surveying
- Robot navigation and mapmaking
- Visual effects (“Match moving”)
- When the position of the camera needs to be tracked for a long time (e.g. autonomous systems), drift will be a problem

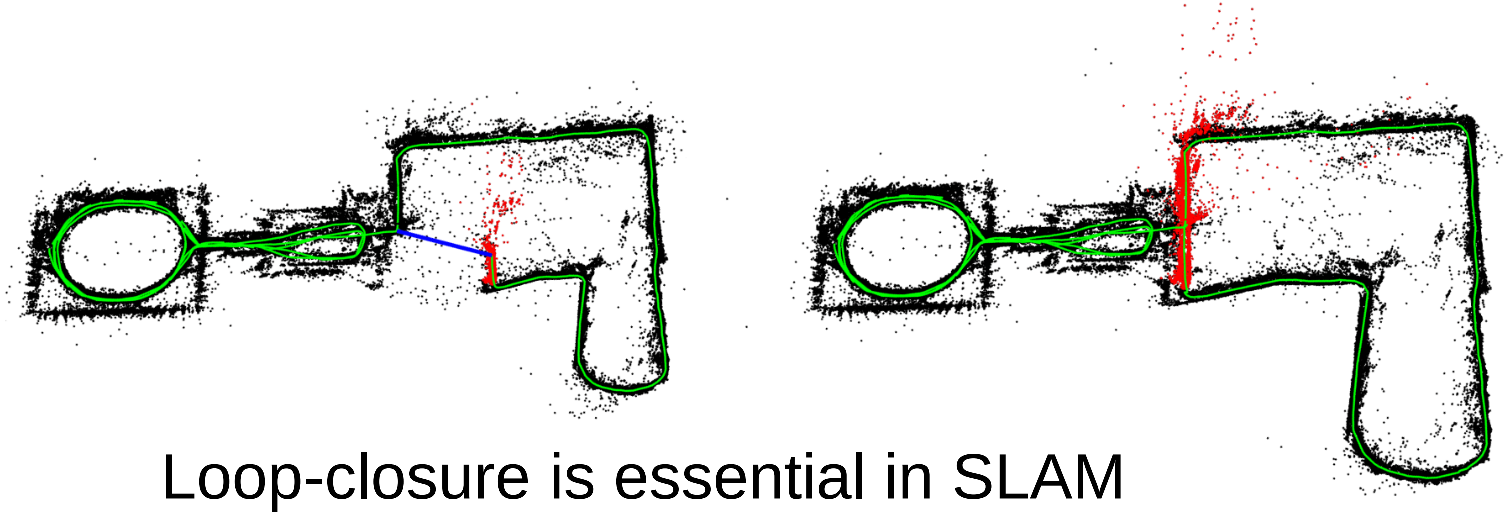
SIMULTANEOUS LOCALIZATION AND MAPPING (SLAM)

- The act of finding one's location against a map is known as localization
- Mapping is the act or process of making a map
- SLAM is a procedure where a map of the unknown environment is produced simultaneously while positioning the user in this newfound map
- Both the trajectory of the platform and the location of all landmarks are estimated on-line without the need for any a priori knowledge of location => autonomous robots

SIMULTANEOUS LOCALIZATION AND MAPPING (SLAM)

- Aims at building a globally consistent representation of the environment, ego-motion and loop-closure
- Vision is the most used technique in SLAM, but it can be realized with different sensors (INS, odometer, ranging, ...)
- If positioning is accurate and reliable (e.g. satellite positioning in good conditions) SLAM is not needed => mapping only

Mur-Artal, R., Montiel,J., Tardos. J (2015). ORB-SLAM: a Versatile and Accurate Monocular SLAM System IEEE TRANSACTIONS ON ROBOTICS



Loop-closure is essential in SLAM

SLAM'S HISTORY

- Started 1986 from the introduction of probabilistic methods into robotics and Artificial Intelligence (AI)
  - Extended Kalman filters
  - Rao-Blackwellised Particle Filters
  - Maximum likelihood estimation
- 2004 – 2015 algorithm analysis age
  - Efficiency
  - Open-source SLAM libraries
- 2016 – Robust-perception age

SLAM'S TWO PHASES

Frontend

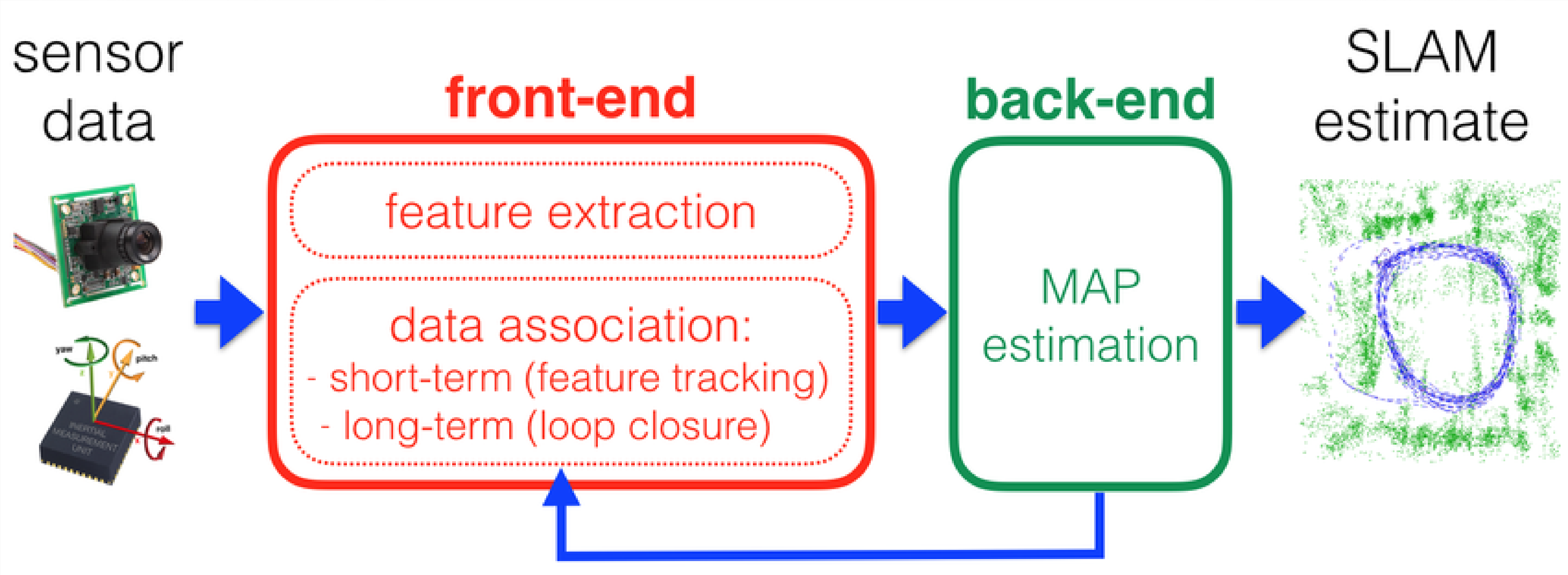
- Uses sensor data (camera, laser)
- Calculates spatial relations between poses
- In practice would be enough for the task, but solution would degrade fast
- NOTE! Without loop-closures the method degrades to visual odometry
- Computer vision, signal processing
- Loop-closure (returning to a previously visited area)

Backend

- Optimizes the estimated pose to keep the resulting map coherent
- Geometry, graph theory, optimization

Mathworks, Matlab's SLAM
before loop-closure

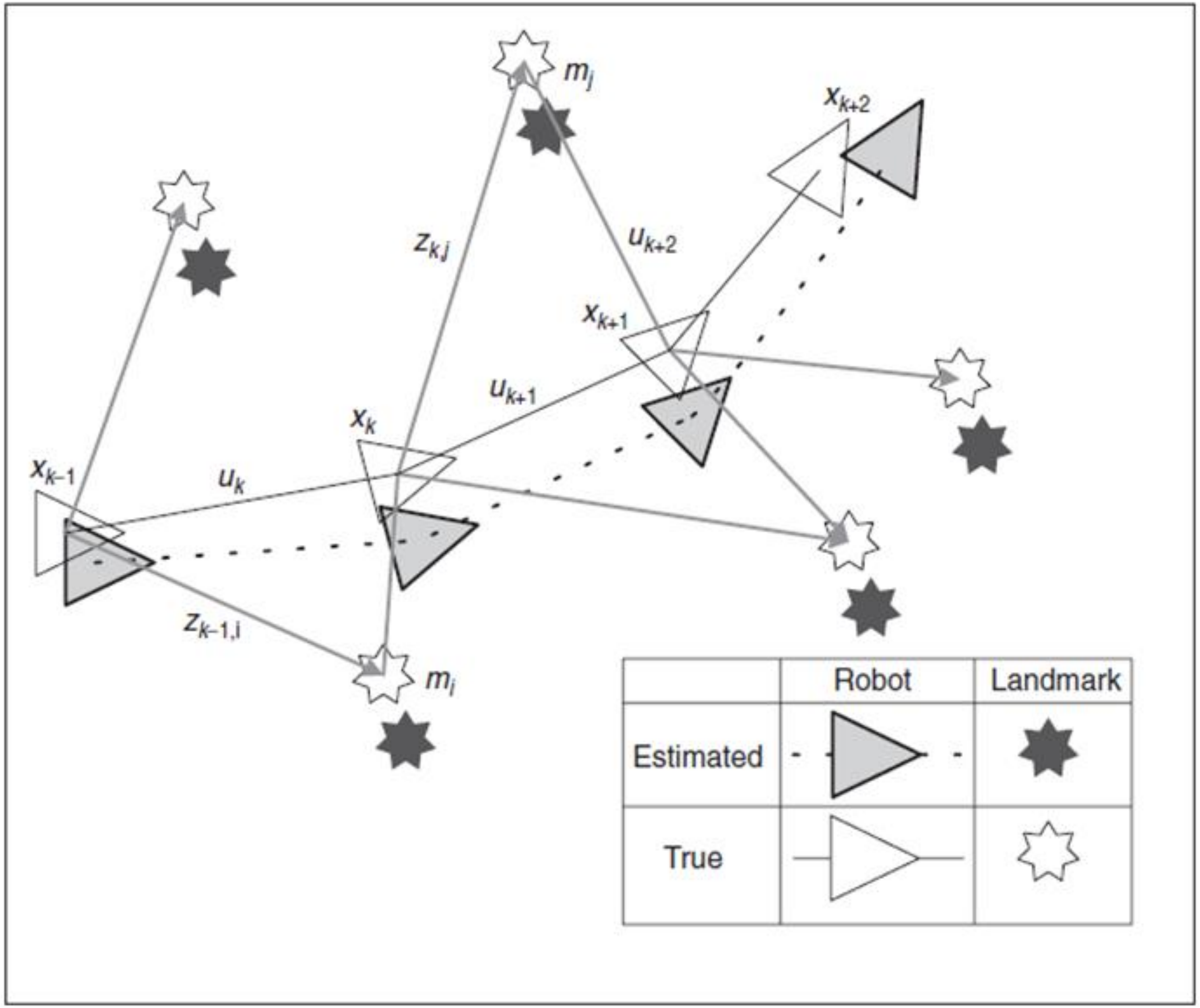
FRONT-END AND BACK-END IN VISUAL SLAM



Cadena et al (2016) [Simultaneous Localization And Mapping: Present, Future, and the Robust-Perception Age]

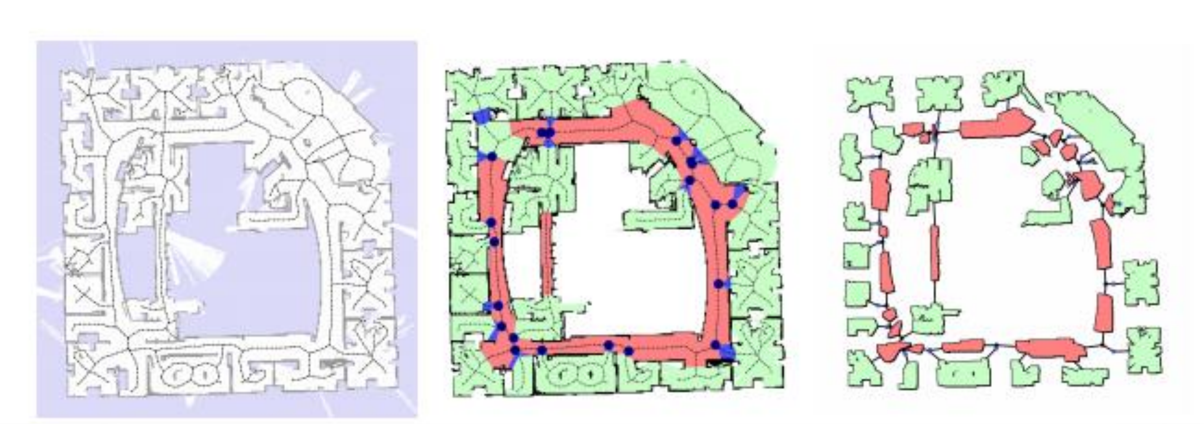
SLAM - PRINCIPLES

Without loop-closure error accumulates out-of-bound



SLAM - MAPS

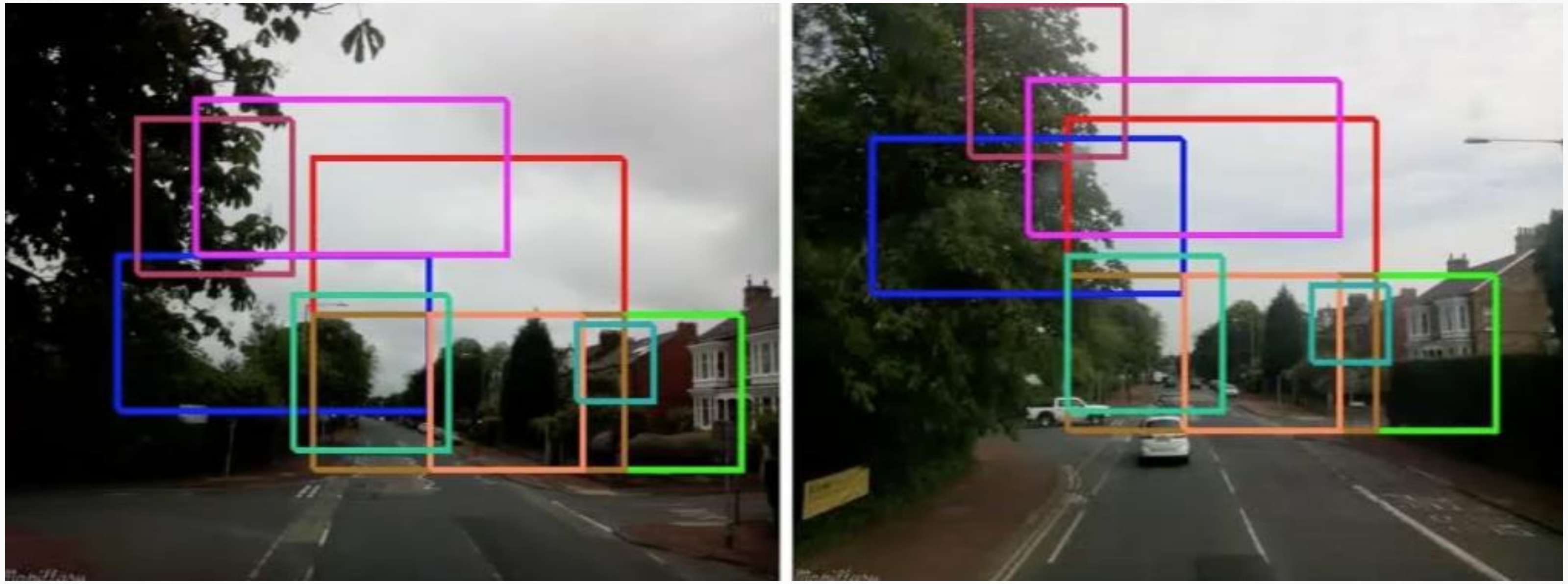
- SLAM algortihms may be divided into topological and metrical
- It would be attempting to drop the metric computations and use just the recognition of visited landmarks
  - However, metrics helps estimating loop-closures and discarging erroneous ones



Occupancy grid map, topological map, semantic map (hybrid topological-metric)

LOOP-CLOSURE

- Incorrect loop detection even harder to recover
- Bayesian filtering for loop-closure probability computation
- Images encoded according to the incremental bags of visual words scheme

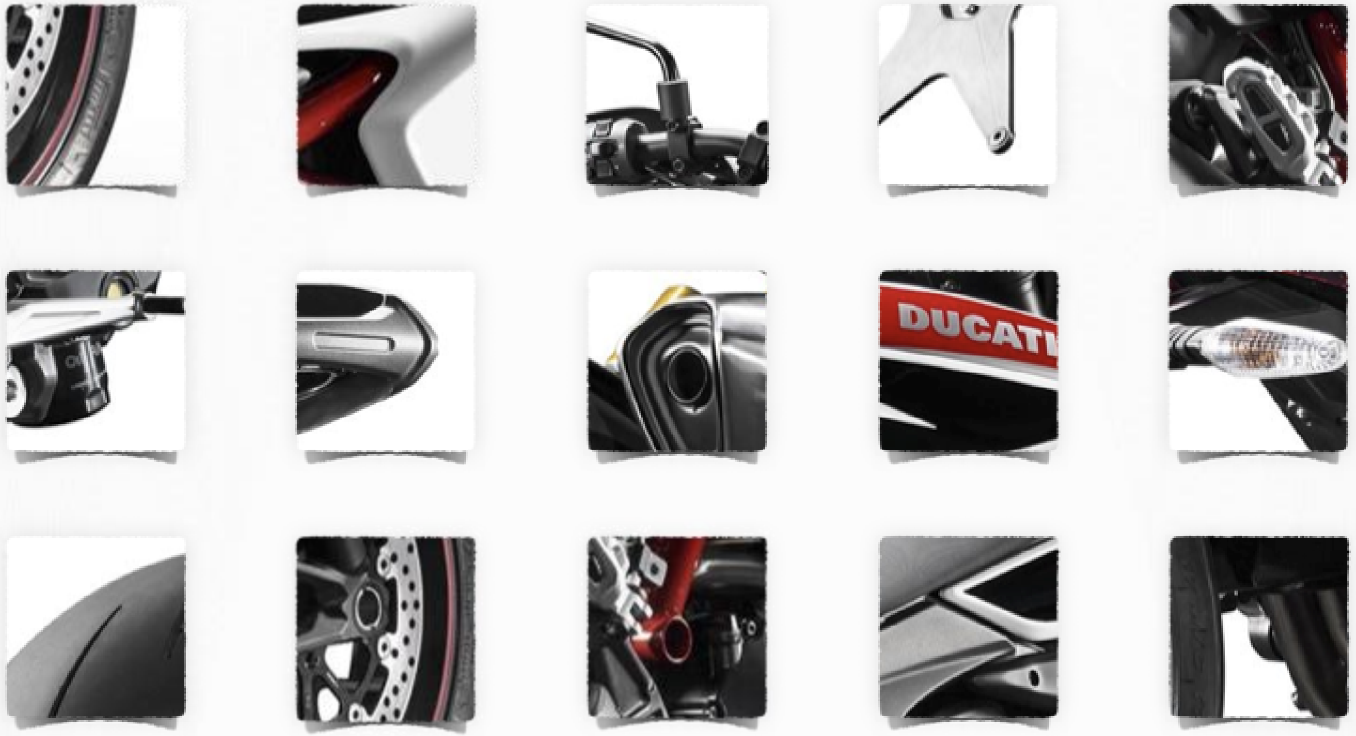


Professor Hong Zhang webdocs.cs.ualberta.ca

BAG OF WORDS

- A bag of visual words is a vector of occurrence counts of a vocabulary of local image features
- Words in images defined
  - Feature detection
  - Feature description → e.g. Sift, Surf
  - Codebook generation
- Codeword is a representative of several similar patches
  - k-means clustering over all the vectors
- Codewords are centers of the learned clusters
  - number of the clusters is the codebook size
- Each patch in an image is mapped to a certain codeword through the clustering process and the image can be represented by the histogram of the codewords

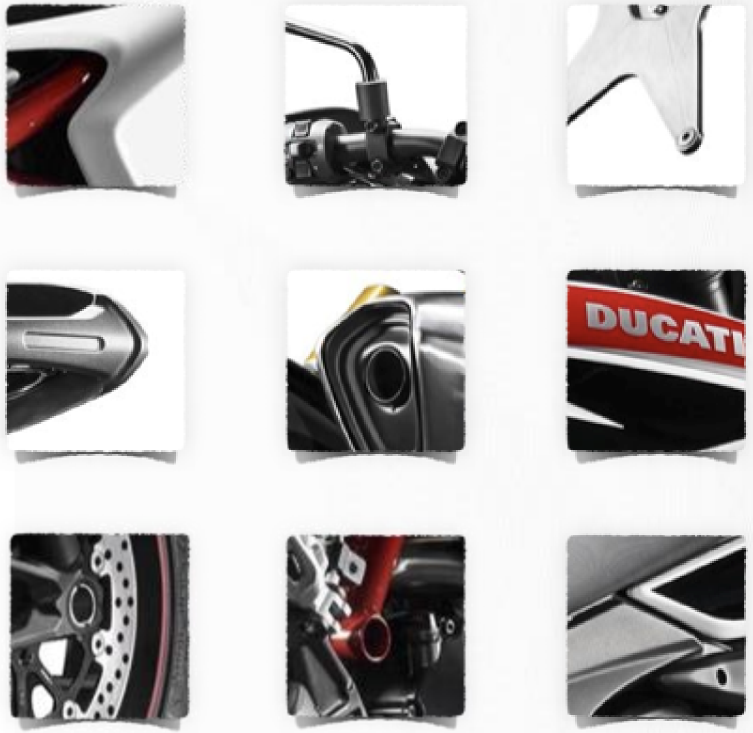
What object do these parts belong to?



Slide credit: Kris Kitani

Some local feature are very informative

An object as



a collection of local features

(bag-of-features)

- deals well with occlusion
- scale invariant
- rotation invariant

Slide credit: Kris Kitani

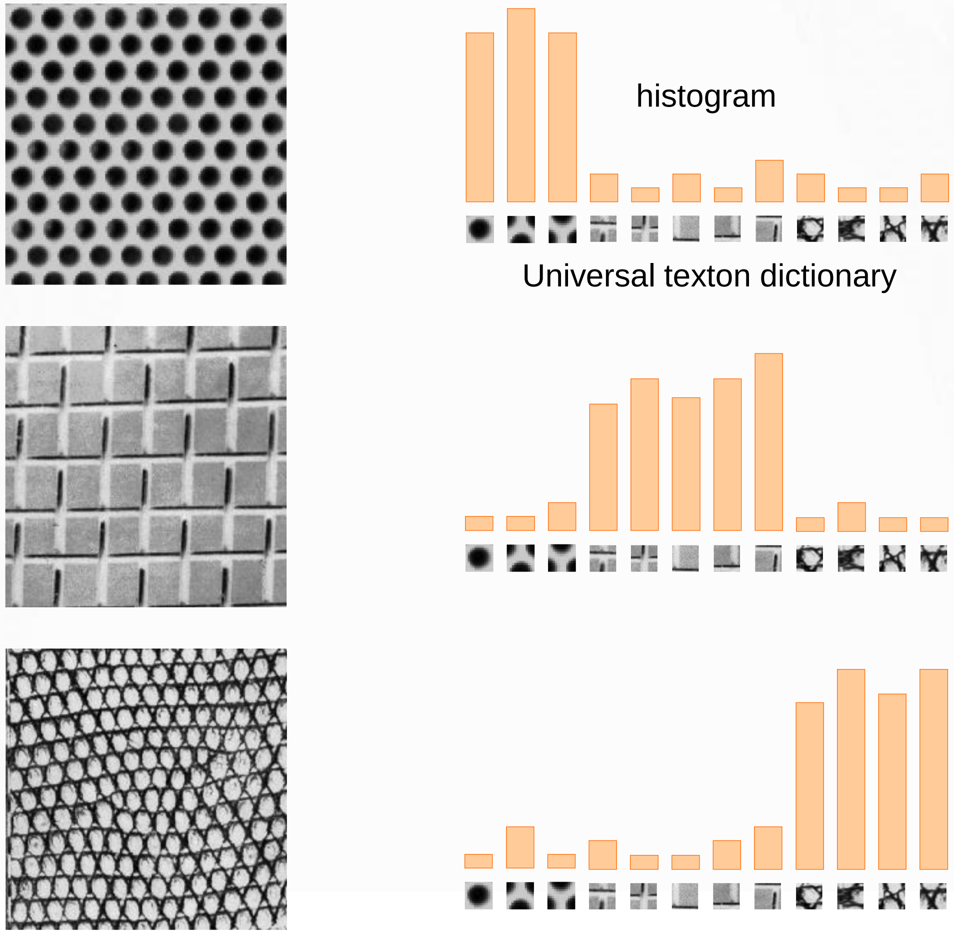
BAG-OF-FEATURES

represent a data item (document, texture, image)
as a histogram over features

an old idea

(e.g., texture recognition and information retrieval)

TEXTURE RECOGNITION



Slide credit: Kris Kitani

VECTOR SPACE MODEL

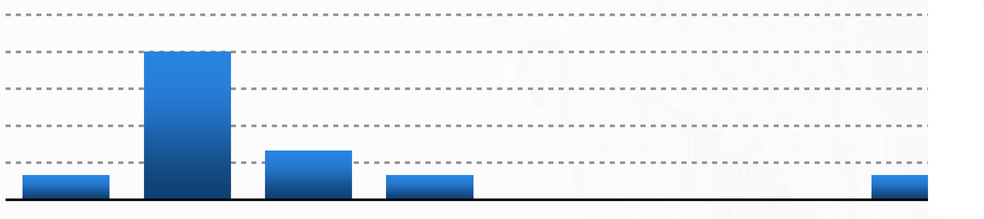
G. Salton. ‘Mathematics and Information Retrieval’ Journal of Documentation,1979

The Newspa

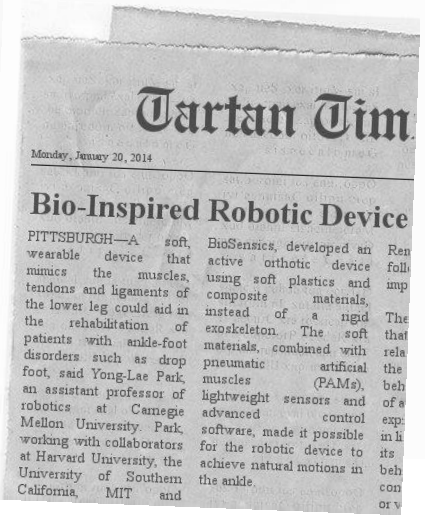
Sunday, December 22, 2013

DARPA Selects Carnegie Me

The Tartan Rescue Team from Carnegie Mellon University's National Robotics Engineering Center ranked third among teams competing in the Defense Advanced Research Projects Agency (DARPA) Robotics Challenge Trials this weekend in Homestead, Fla., and was selected by the agency as one of eight teams eligible for DARPA funding to prepare for next December's finals. The team's four-limbed CMU Highly Intelligent Mobile Platform, or CHIMP, robot scored 18 out of a possible 32 points during the two-day trials. It demonstrated its ability to perform such tasks as removing debris, cutting a hole through a wall and closing a series of valves.



1	6	2	1	0	0	0	1
Tartan	robot	CHIMP	CMU	bio	soft	ankle	sensor

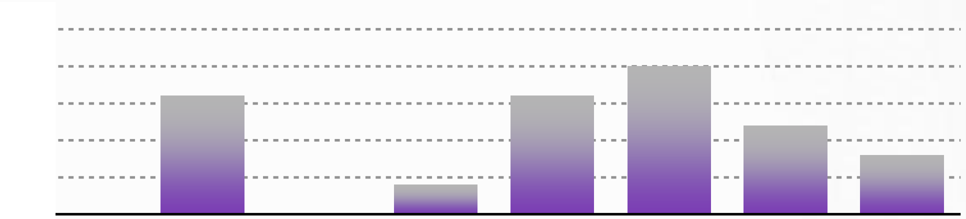


Tartan Tim

Monday, January 20, 2014

Bio-Inspired Robotic Device

PITTSBURGH—A soft, wearable device that mimics the muscles, tendons and ligaments of the lower leg could aid in the rehabilitation of patients with ankle-foot disorders such as drop foot, said Yong-Lae Park, an assistant professor of robotics at Carnegie Mellon University. Park, working with collaborators at Harvard University, the University of Southern California, MIT and BioSensics, developed an active orthotic device using soft plastics and composite materials, instead of a rigid exoskeleton. The soft materials, combined with pneumatic artificial muscles (PAMs), lightweight sensors and advanced control software, made it possible for the robotic device to achieve natural motions in the ankle.



0	4	0	1	4	5	3	2
Tartan	robot	CHIMP	CMU	bio	soft	ankle	sensor

Slide credit: Kris Kitani

A document (datapoint) is a vector of counts over each word (feature)

$$\boldsymbol{v}_d = [n(w_{1,d}) \;\; n(w_{2,d}) \;\; \cdots \;\; n(w_{T,d})]$$

just a histogram over words

$n(\cdot)$ counts the number of occurrences

What is the similarity between two documents?


The Newspa
DARPA Selects Carnegie Me
Tartan Tim
Bio-Inspired Robotic Device

Slide credit: Kris Kitani

A document (datapoint) is a vector of counts over each word (feature)

$$\boldsymbol{v}_d = [n(w_{1,d}) \quad n(w_{2,d}) \quad \cdots \quad n(w_{T,d})]$$

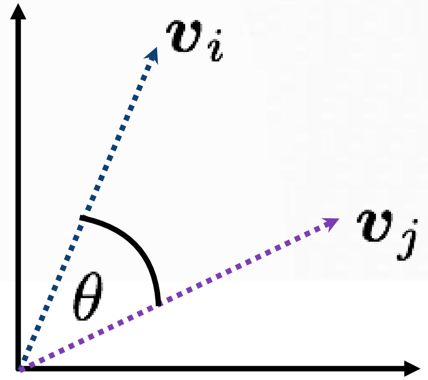
just a histogram over words

$n(\cdot)$ counts the number of occurrences

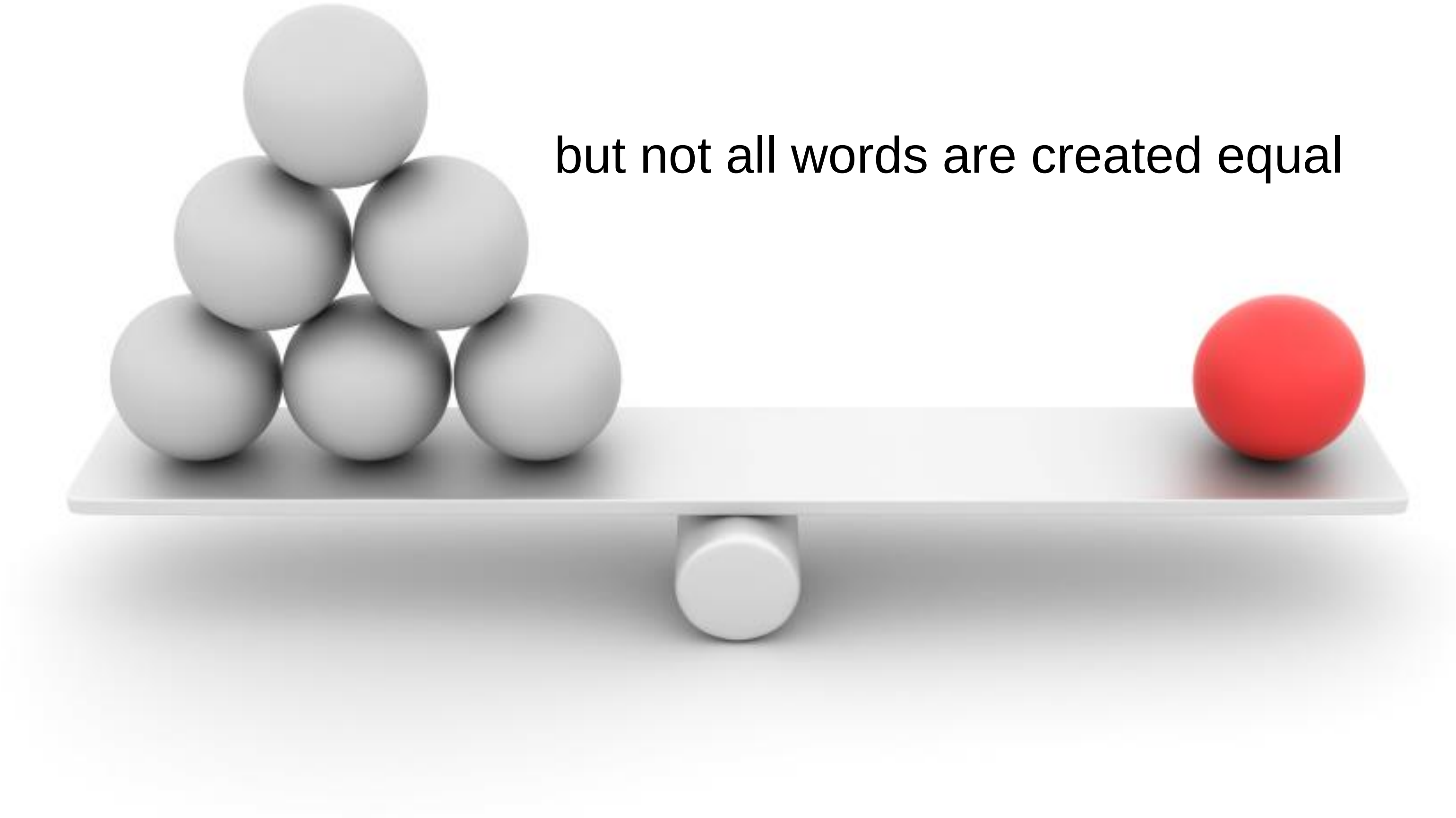
What is the similarity between two documents?

Use any distance you want but the cosine distance is fast.

$$d(\boldsymbol{v}_i, \boldsymbol{v}_j) = \cos\theta = \frac{\boldsymbol{v}_i \cdot \boldsymbol{v}_j}{\|\boldsymbol{v}_i\| \|\boldsymbol{v}_j\|}$$



Slide credit: Kris Kitani



but not all words are created equal

Term Frequency - Inverse Document Frequency

TF-IDF

$$\boldsymbol{v}_d = [n(w_{1,d}) \;\; n(w_{2,d}) \;\; \cdots \;\; n(w_{T,d})]$$

weigh each word by a heuristic

$$\boldsymbol{v}_d = [n(w_{1,d})\alpha_1 \;\; n(w_{2,d})\alpha_2 \;\; \cdots \;\; n(w_{T,d})\alpha_T]$$

$$n(w_{i,d})\alpha_i = \underbrace{n(w_{i,d})}_{\text{term frequency}} \overbrace{\log\left\{\frac{D}{\sum_{d'} \mathbf{1}[w_i \in d']}\right\}}^{\text{inverse document frequency}}$$

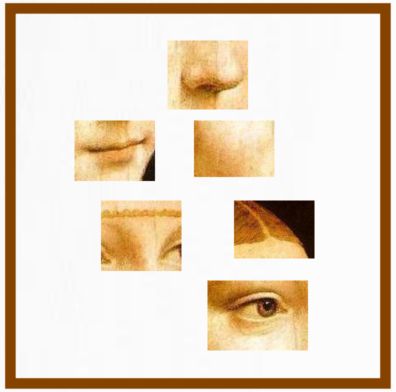
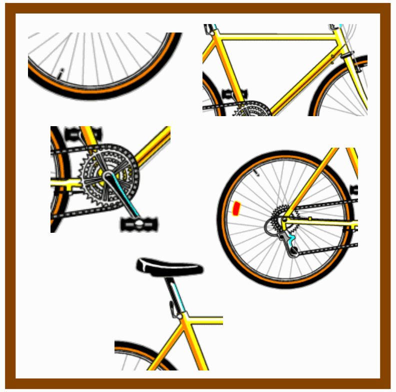
(down-weights **common** terms)

Slide credit: Kris Kitani

STANDARD BOW PIPELINE
(FOR IMAGE CLASSIFICATION)

Dictionary Learning:
Learn Visual Words using clustering

1. extract features (e.g., SIFT) from images


Slide credit: Kris Kitani

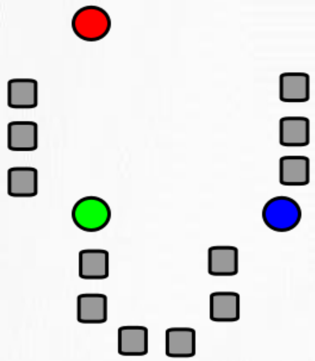
Dictionary Learning:

Learn Visual Words using clustering

2. Learn visual dictionary (e.g., K-means clustering)

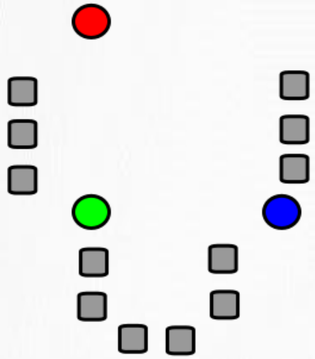
Slide credit: Kris Kitani

K-MEANS CLUSTERING

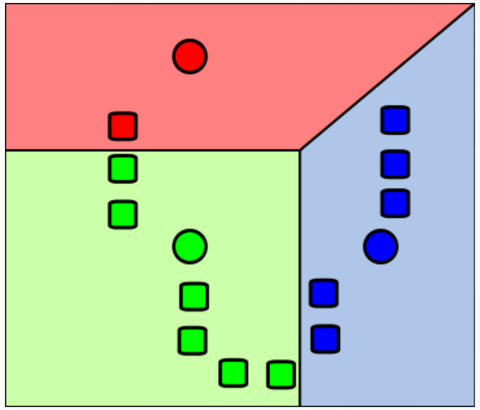


1. Select initial
centroids at random

Slide credit: Kris Kitani

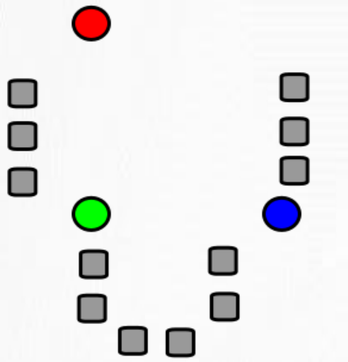


1. Select initial centroids at random

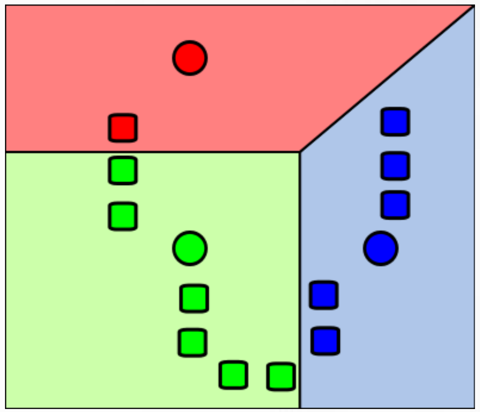


2. Assign each object to the cluster with the nearest centroid.

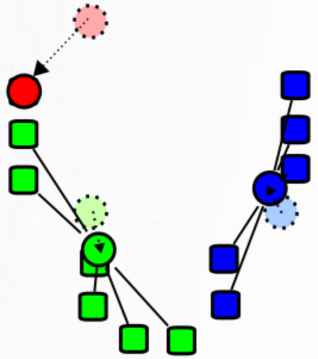
Slide credit: Kris Kitani



1. Select initial centroids at random

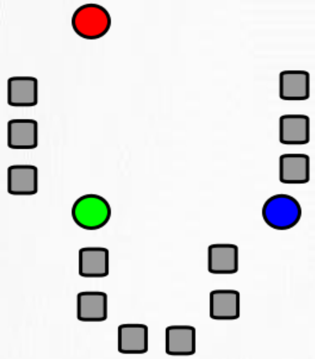


2. Assign each object to the cluster with the nearest centroid.

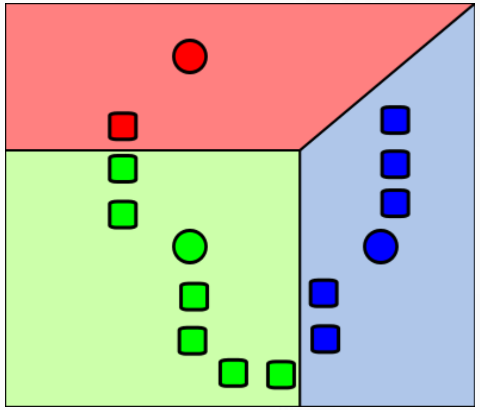


3. Compute each centroid as the mean of the objects assigned to it (go to 2)

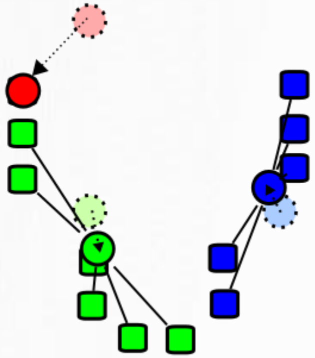
Slide credit: Kris Kitani



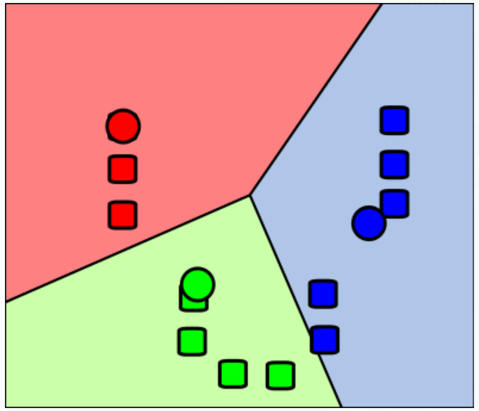
1. Select initial centroids at random



2. Assign each object to the cluster with the nearest centroid.

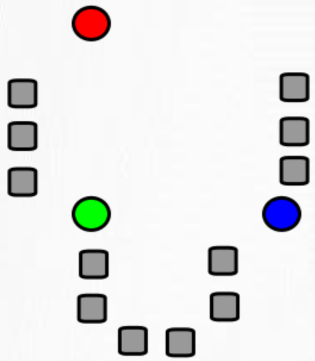


3. Compute each centroid as the mean of the objects assigned to it (go to 2)

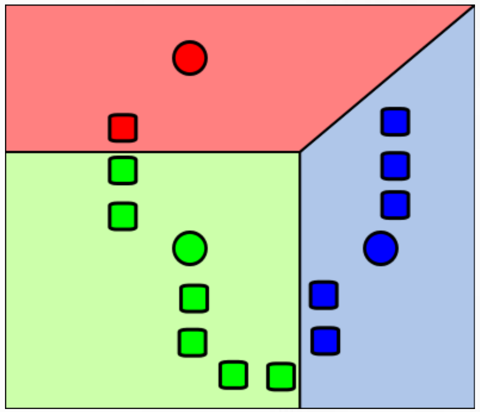


2. Assign each object to the cluster with the nearest centroid.

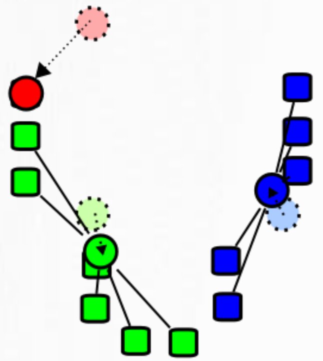
Slide credit: Kris Kitani



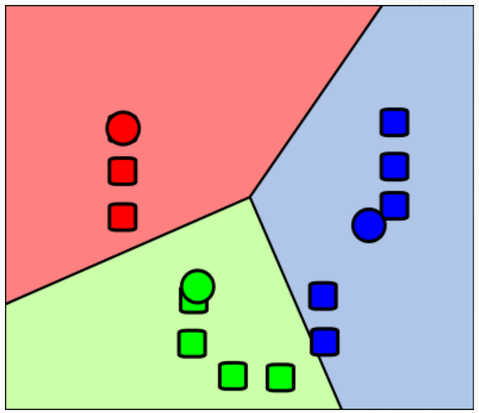
1. Select initial centroids at random



2. Assign each object to the cluster with the nearest centroid.



3. Compute each centroid as the mean of the objects assigned to it (go to 2)



2. Assign each object to the cluster with the nearest centroid.

Slide credit: Kris Kitani

Repeat previous 2 steps until no change

K-MEANS CLUSTERING

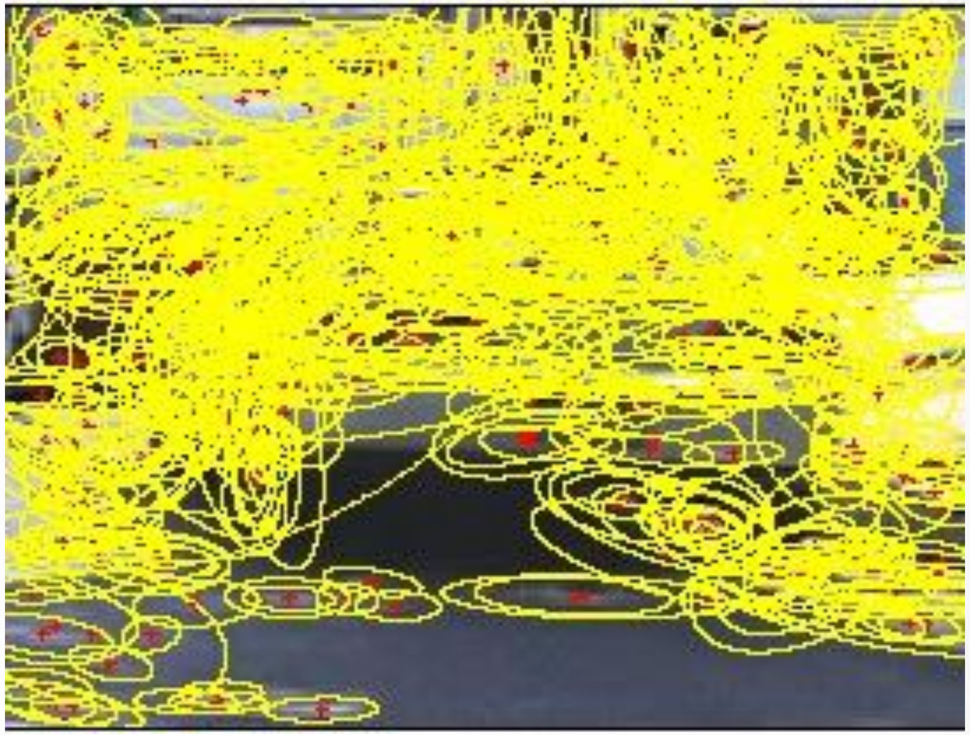
Given k:

1. Select initial centroids at random.
2. Assign each object to the cluster with the nearest centroid.
3. Compute each centroid as the mean of the objects assigned to it.
4. Repeat previous 2 steps until no change.

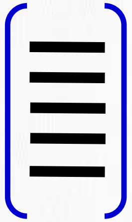
Slide credit: Kris Kitani

What kinds of features can we extract?

- Regular grid
  - Vogel & Schiele, 2003
  - Fei-Fei & Perona, 2005
- Interest point detector
  - Csurka et al. 2004
  - Fei-Fei & Perona, 2005
  - Sivic et al. 2005
- Other methods
  - Random sampling (Vidal-Naquet & Ullman, 2002)
  - Segmentation-based patches (Barnard et al. 2003)

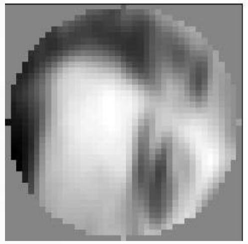


Slide credit: Kris Kitani

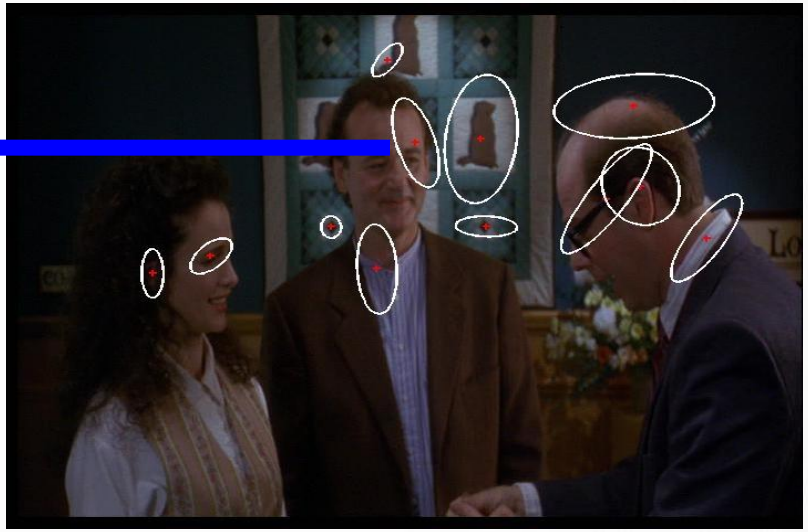


Compute SIFT descriptor

[Lowe'99]



Normalize patch



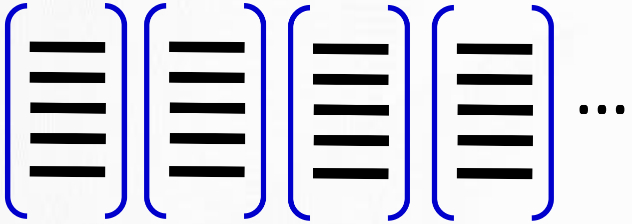
Detect patches

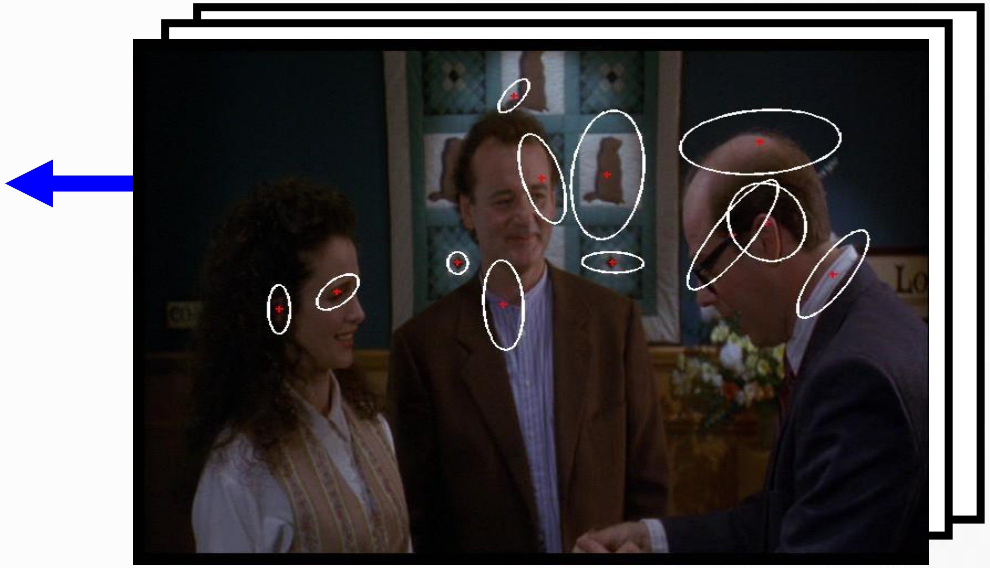
[Mikojaczyk and Schmid '02]

[Mata, Chum, Urban & Pajdla, '02]

[Sivic & Zisserman, '03]

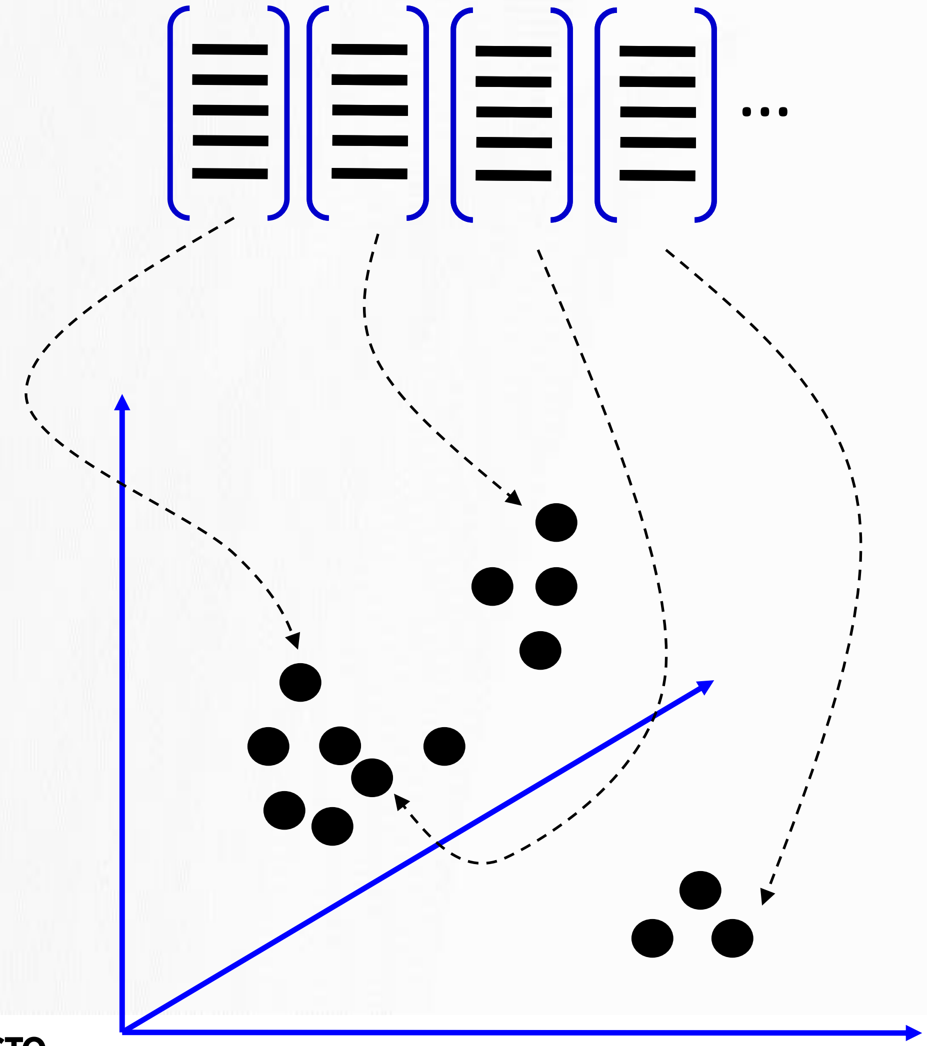
Slide credit: Kris Kitani



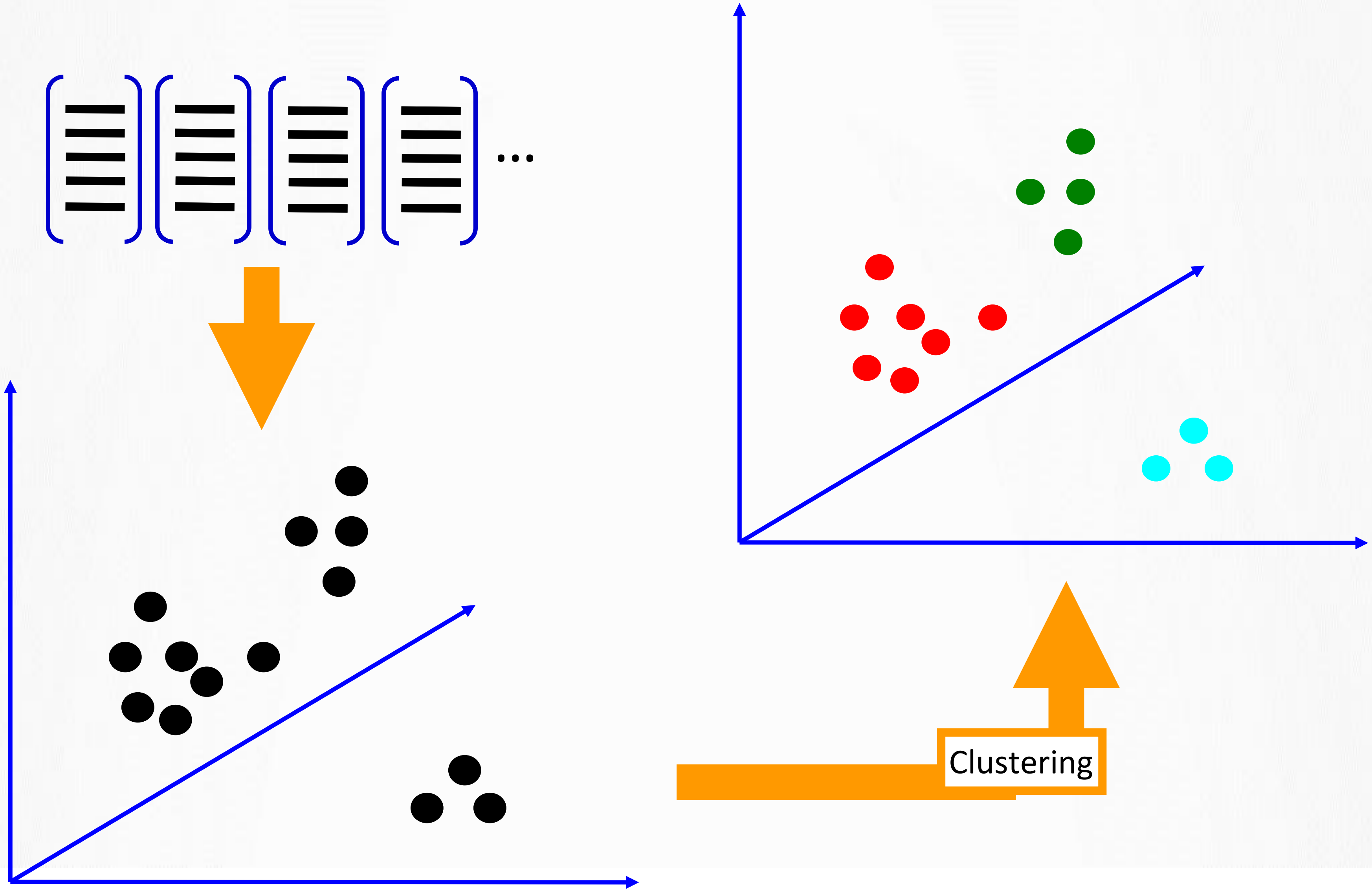


Slide credit: Kris Kitani

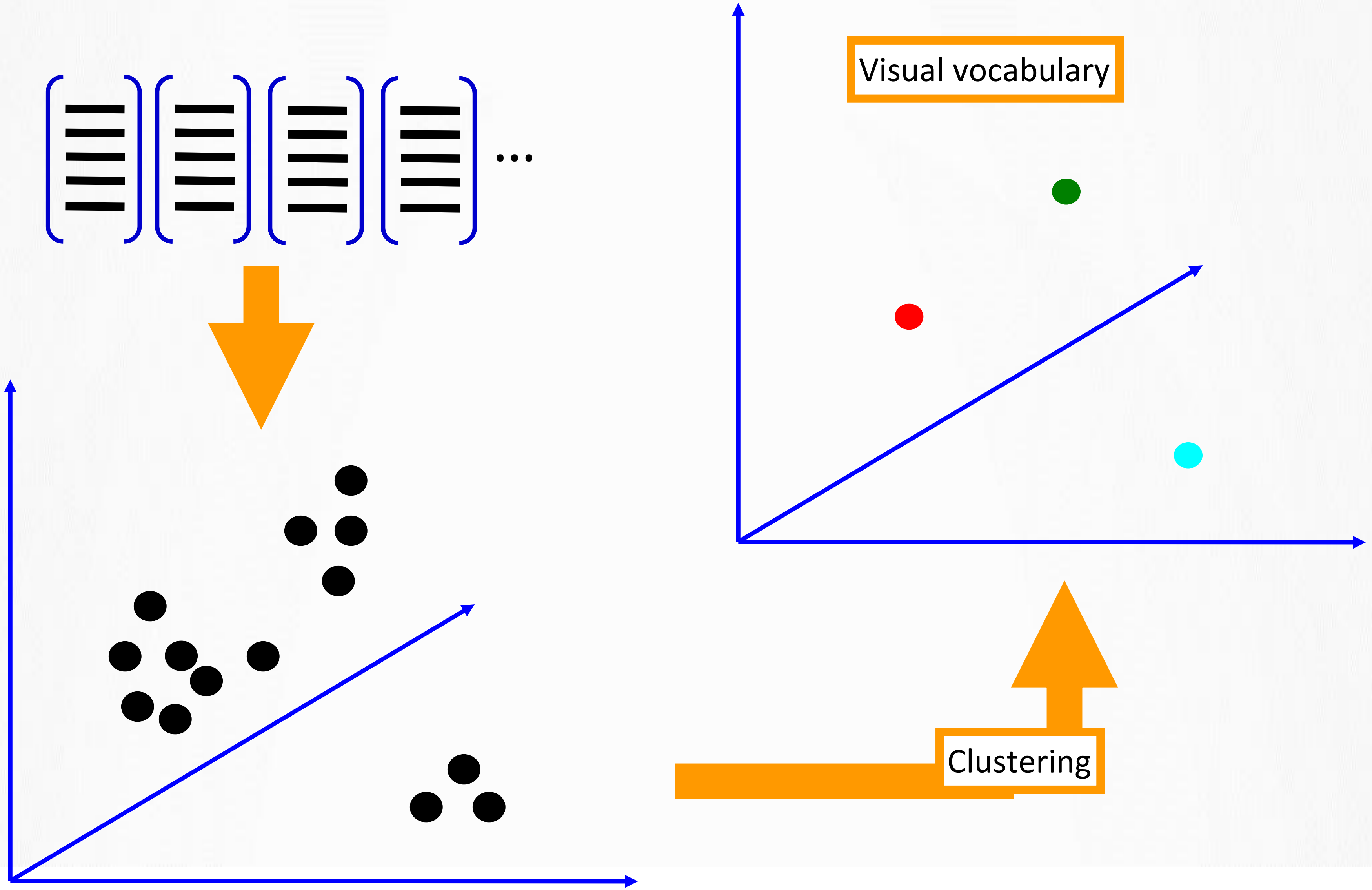
How do we learn the dictionary?



Slide credit: Kris Kitani



Slide credit: Kris Kitani



HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Slide credit: Kris Kitani

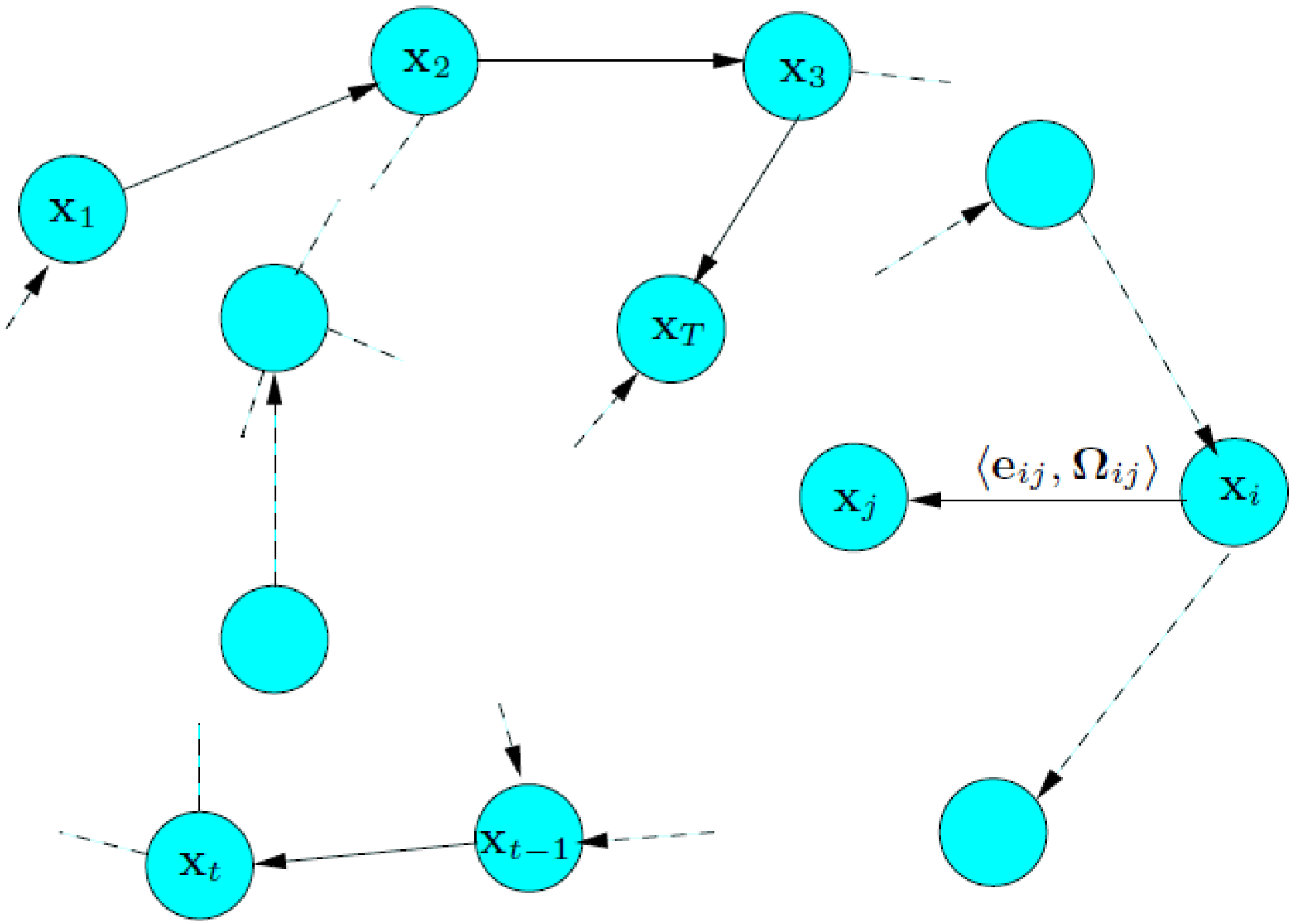
BAG-OF-WORDS IN SLAM

- Image is a (histogram) bag of words
- Image is presented by a vector, typically 1000 – 4000 dimensional
- Doesn't contain any spatial information, no order of features
- While operating, extract features, compute histograms and frequencies => compare bag to the bags in the database => distance matrix

- Pose graph optimization

POSE GRAPH OPTIMIZATION

- Every node corresponds to a robot pose
- Nearby poses are connected by edges
- When loop-closure is detected, an edge is added



Grisetti et al. (2010). A Tutorial on Graph-Based SLAM, IEEE ITS Magazine

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI


60˚ 10 1.2 N, 24˚ 57 18 E