Deep Learning for Computer Vision

Feature Matching

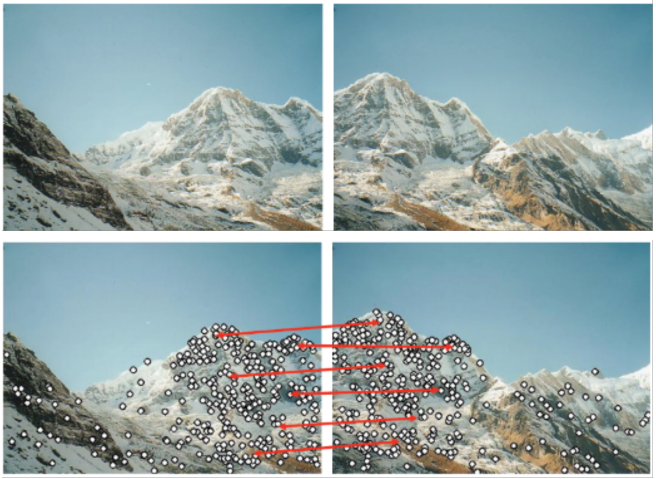
Vineeth N Balasubramanian

Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad

Acknowledgements

- Most of this lecture's slides are based on lectures of **Deep Learning for Vision** course taught by Prof Yannis Avrithis at Inria Rennes-Bretagne Atlantique

Review



How to match?

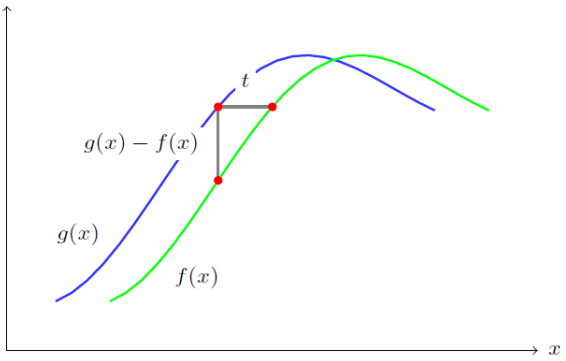
Dense Registration through Optical Flow[1]

- For each location in an image, find a displacement with respect to another reference image
- Appropriate for small displacements, e.g. stereopsis or optical flow

[1] Lucas and Kanade IJCAI 1981. An Iterative Image Registration Technique With an Application to Stereo Vision.

Dense Registration through Optical Flow[2]

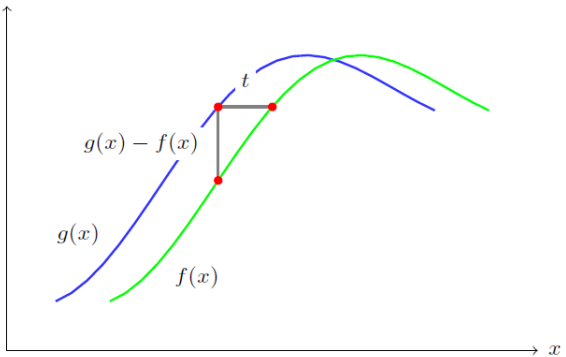
- One dimension:



[2]Lucas and Kanade IJCAI 1981. An Iterative Image Registration Technique With an Application to Stereo Vision.

Dense Registration through Optical Flow[2]

- One dimension:



Assuming $g(x) = f(x + t)$ and t is small,

$$\frac{df}{dx}(x) \approx \frac{f(x+t) - f(x)}{t} = \frac{g(x) - f(x)}{t}$$

[2]Lucas and Kanade IJCAI 1981. An Iterative Image Registration Technique With an Application to Stereo Vision.

Dense Registration through Optical Flow[2]

- Error given by:

$$E(t) = \sum_x w(x)\Big(f(x+t) - g(x)\Big)^2 \approx \sum_x w(x)\Big(f(x) + t^T \Delta f(x) - g(x)\Big)^2$$

[2] Lucas and Kanade IJCAI 1981. An Iterative Image Registration Technique With an Application to Stereo Vision.

Dense Registration through Optical Flow[2]

- Error given by:

$$E(t) = \sum_x w(x)\Big(f(x+t) - g(x)\Big)^2 \approx \sum_x w(x)\Big(f(x) + t^T \Delta f(x) - g(x)\Big)^2$$

- Error minimized when gradient vanishes

$$\frac{\partial E}{\partial t} = \sum_x w(x) 2\Delta f(x)\Big(f(x) + t^T \Delta f(x) - g(x)\Big) = 0$$

[2] Lucas and Kanade IJCAI 1981. An Iterative Image Registration Technique With an Application to Stereo Vision.

Dense Registration through Optical Flow[2]

- Error given by:

$$E(t) = \sum_x w(x)\Big(f(x+t) - g(x)\Big)^2 \approx \sum_x w(x)\Big(f(x) + t^T \Delta f(x) - g(x)\Big)^2$$

- Error minimized when gradient vanishes

$$\frac{\partial E}{\partial t} = \sum_x w(x) 2\Delta f(x)\Big(f(x) + t^T \Delta f(x) - g(x)\Big) = 0$$

- Least-squares solution *(ignoring summation and arguments for simplicity)*:

$$w\Delta f(\Delta f)^T t = w\Delta f(g - f)$$

[2] Lucas and Kanade IJCAI 1981. An Iterative Image Registration Technique With an Application to Stereo Vision.

Dense Registration through Optical Flow[2]

- Error given by:

$$E(t) = \sum_x w(x)\Big(f(x+t) - g(x)\Big)^2 \approx \sum_x w(x)\Big(f(x) + t^T \Delta f(x) - g(x)\Big)^2$$

- Error minimized when gradient vanishes

$$\frac{\partial E}{\partial t} = \sum_x w(x) 2\Delta f(x)\Big(f(x) + t^T \Delta f(x) - g(x)\Big) = 0$$

- Least-squares solution *(ignoring summation and arguments for simplicity)*:

$$w\Delta f(\Delta f)^T t = w\Delta f(g - f)$$

- 2-D equivalent: Assume an image patch defined by window w; what is the error between patch shifted by t in reference image f and patch at origin in shifted image g?

[2] Lucas and Kanade IJCAI 1981. An Iterative Image Registration Technique With an Application to Stereo Vision.

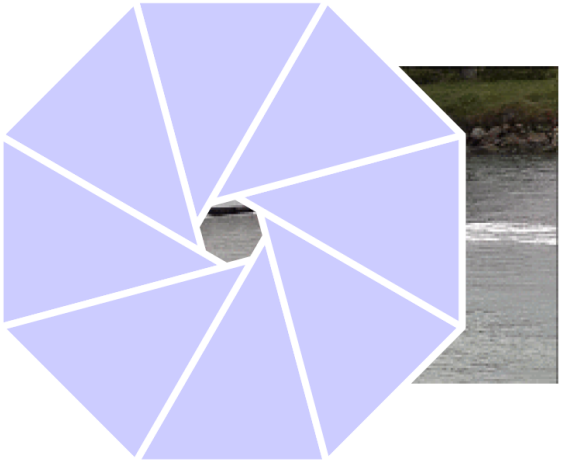
Dense Registration through Optical Flow[2]

- The Aperture Problem:

[2]Lucas and Kanade IJCAI 1981. An Iterative Image Registration Technique With an Application to Stereo Vision.

Dense Registration through Optical Flow[2]

- The Aperture Problem:



[2]Lucas and Kanade IJCAI 1981. An Iterative Image Registration Technique With an Application to Stereo Vision.

Wide Baseline Spatial Matching

- In dense registration, we started from a local "template matching" process and found an efficient solution based on a Taylor approximation
- Both make sense for small displacements
- In wide-baseline matching, every part of one image may appear anywhere in the other
- We start by pairwise matching of local descriptors without any order, and then attempt to enforce some geometric consistency according to a rigid motion model

Wide Baseline Spatial Matching

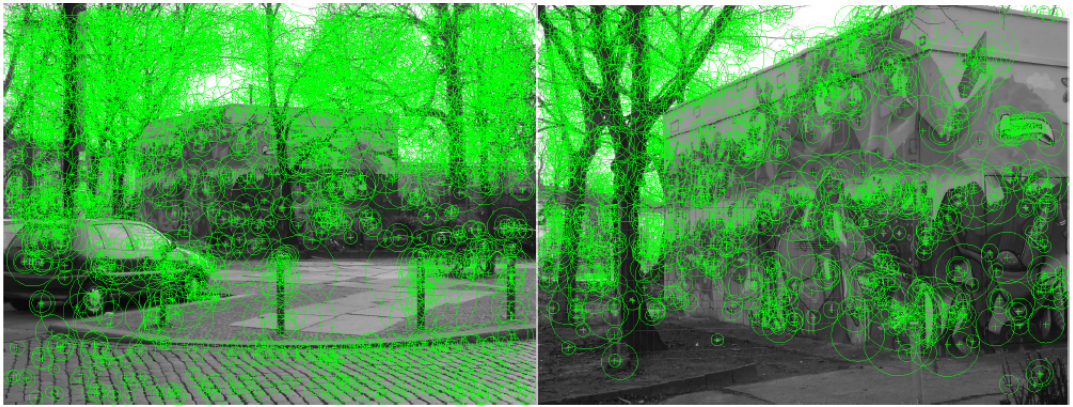
- In dense registration, we started from a local "template matching" process and found an efficient solution based on a Taylor approximation
- Both make sense for small displacements
- In wide-baseline matching, every part of one image may appear anywhere in the other
- We start by pairwise matching of local descriptors without any order, and then attempt to enforce some geometric consistency according to a rigid motion model

Wide Baseline Spatial Matching

A region in one image may appear anywhere in the other

Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

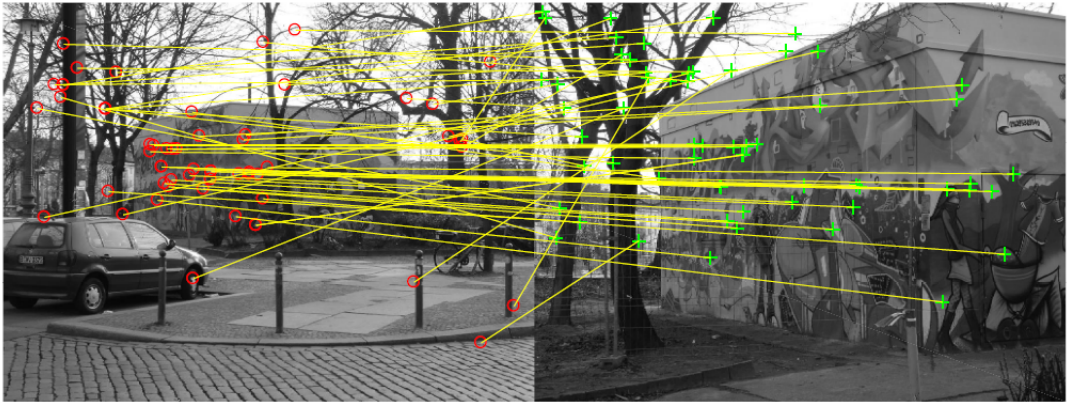
Wide Baseline Spatial Matching



Features detected independently in each image

Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

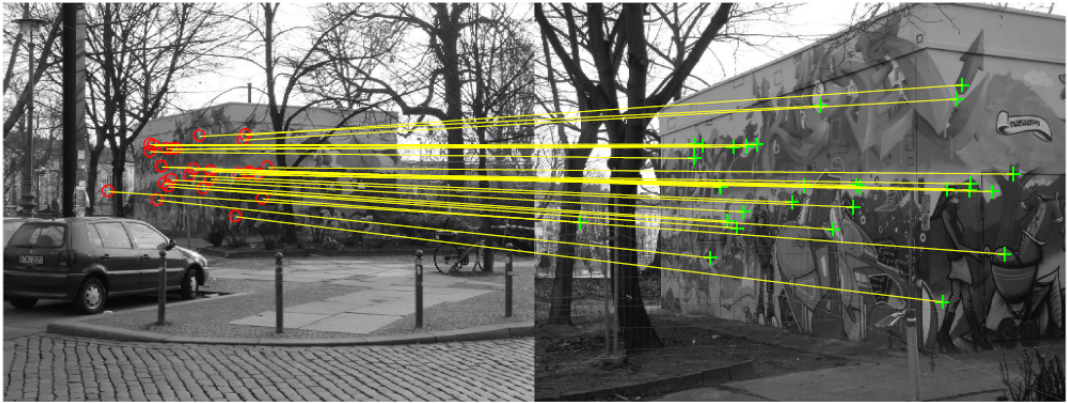
Wide Baseline Spatial Matching



Tentative correspondences by pairwise descriptor matching

Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

Wide Baseline Spatial Matching



Subset of correspondences that are 'inlier' to a rigid transformation

Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

Wide Baseline Spatial Matching

Descriptor Extraction:

For each detected feature in each image:

- Construct a local histogram of gradient orientations (HoG)
- Find one or more dominant orientations corresponding to peaks in histogram
- Resample local patch at given location, scale, and orientation
- Extract one descriptor for each dominant orientation

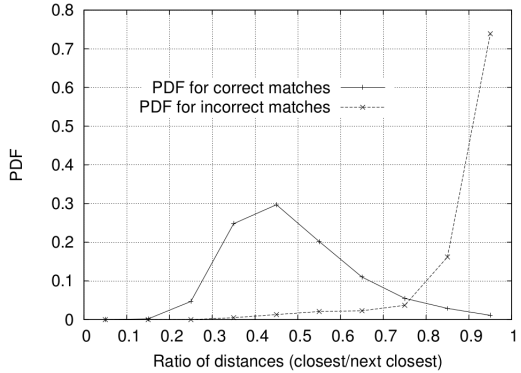
Wide Baseline Spatial Matching

Descriptor Matching:

- For each descriptor in one image, find its two nearest neighbors in the other
- If ratio of distance of first to distance of second is small, make a correspondence
- This yields a list of tentative correspondences

Wide Baseline Spatial Matching

Ratio Test:



Ratio of first to second nearest neighbour distance can determine the probability of a true correspondence

Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

Wide Baseline Spatial Matching

Why is it difficult?

- Should allow for a geometric transformation
- Fitting the model to data (correspondences) is sensitive to outliers: should find a subset of inliers first
- Finding inliers to a transformation requires finding the transformation in the first place
- Correspondences can have gross error
- Inliers are typically less than 50%

Geometric Transformations

- Two images I, I' are equal at points $\mathbf{x}, \mathbf{x}'$

$$I(\mathbf{x}) = I'(\mathbf{x}')$$

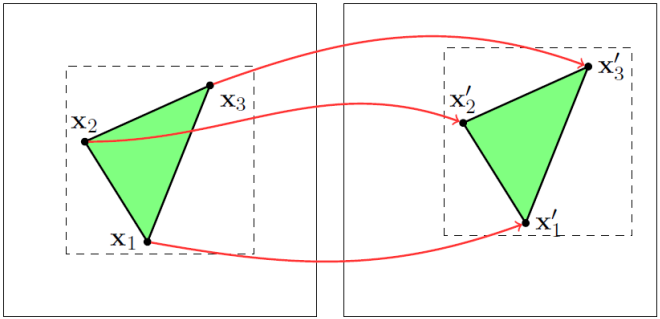
- $\mathbf{x}$ is mapped to $\mathbf{x}'$

$$\mathbf{x}' = T(\mathbf{x})$$

- T is a bijection of $\mathbb{R}^2$ to itself:

$$T : \mathbb{R}^2 \rightarrow \mathbb{R}^2$$

Geometric Transformations

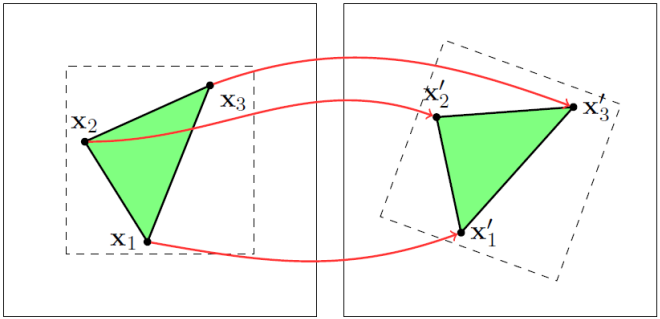


- Translation: 2 degrees of freedom

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

Geometric Transformations

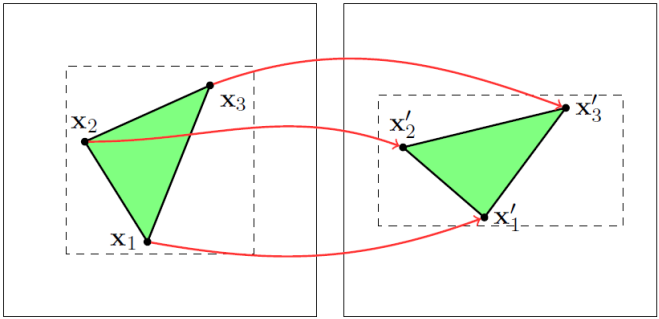


- Rotation: 1 degree of freedom

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

Geometric Transformations

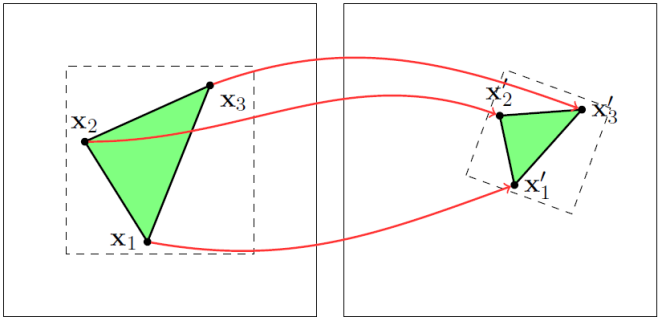


- Similarity: 4 degrees of freedom

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} r\cos\theta & -r\sin\theta & t_x \\ r\sin\theta & r\cos\theta & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

Geometric Transformations

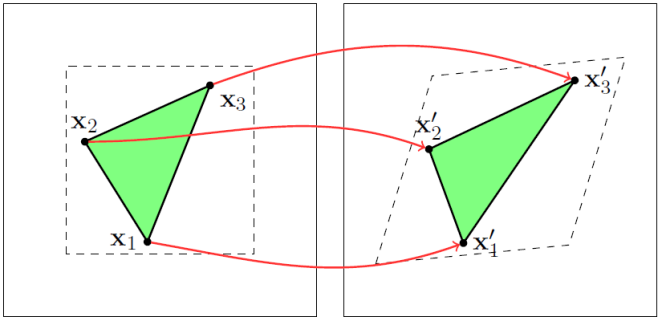


- Similarity: 4 degrees of freedom

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} r\cos\theta & -r\sin\theta & t_x \\ r\sin\theta & r\cos\theta & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

Geometric Transformations

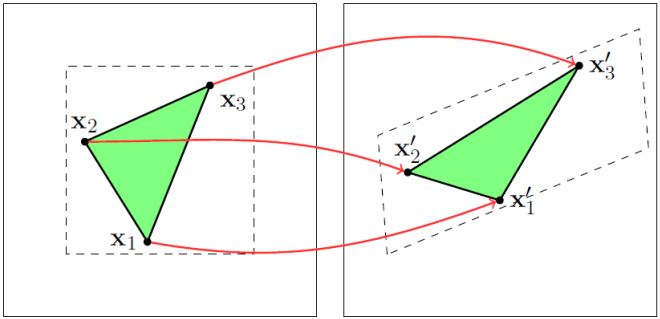


- Shear: 2 degrees of freedom

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & b_x & 0 \\ b_y & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

Geometric Transformations



- Affine: 6 degrees of freedom

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

Correspondence and Least Squares

- In all cases, the problem is transformed to a linear system (why?)

$$A\mathbf{x} = \mathbf{b}$$

where $\mathbf{x}$, $\mathbf{b}$ contain coordinates of known point correspondences from images I, I' respectively, and A contains our model parameters

Correspondence and Least Squares

- In all cases, the problem is transformed to a linear system (why?)

$$A\mathbf{x} = \mathbf{b}$$

  where $\mathbf{x}$, $\mathbf{b}$ contain coordinates of known point correspondences from images I, I' respectively, and A contains our model parameters

- We need $n = \lceil d/2 \rceil$ correspondences, where d are the degrees of freedom of our model

Correspondence and Least Squares

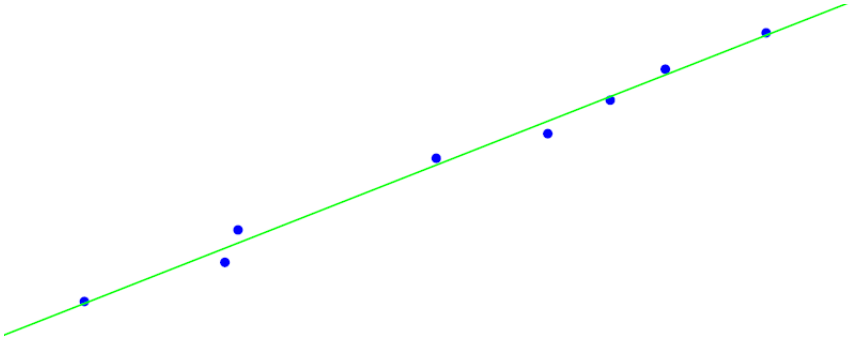
- In all cases, the problem is transformed to a linear system (why?)

$$A\mathbf{x} = \mathbf{b}$$

  where $\mathbf{x}$, $\mathbf{b}$ contain coordinates of known point correspondences from images I, I' respectively, and A contains our model parameters

- We need $n = \lceil d/2 \rceil$ correspondences, where d are the degrees of freedom of our model
- Let's take the simplest model as an example: fit a line to two points

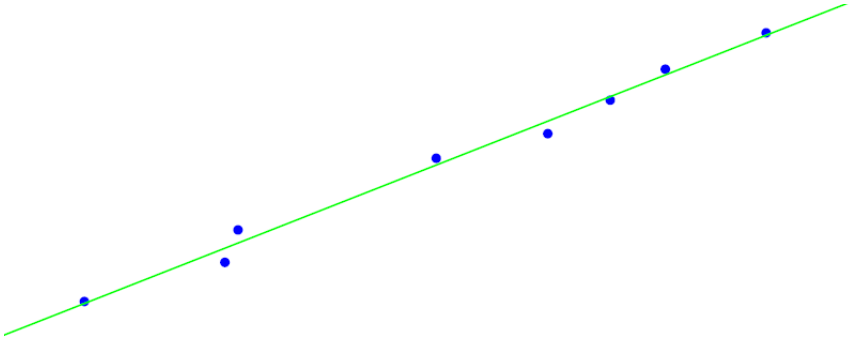
Correspondence and Least Squares



- clean data, no outliers : least squares fit ok

Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

Correspondence and Least Squares



- clean data, no outliers : least squares fit ok

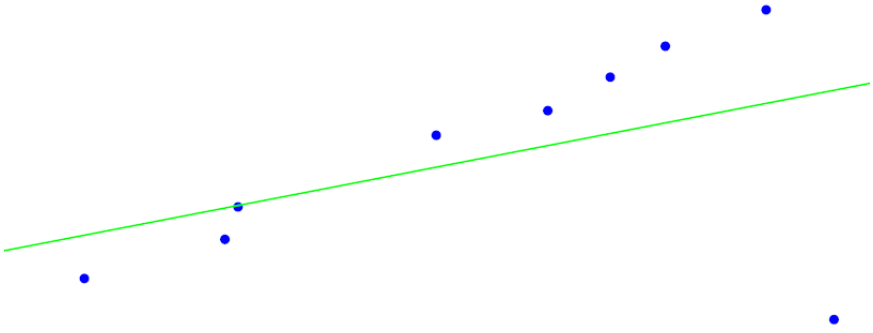
Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

Correspondence and Least Squares

- one gross outlier - least squares fit fails - what do we do?

Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

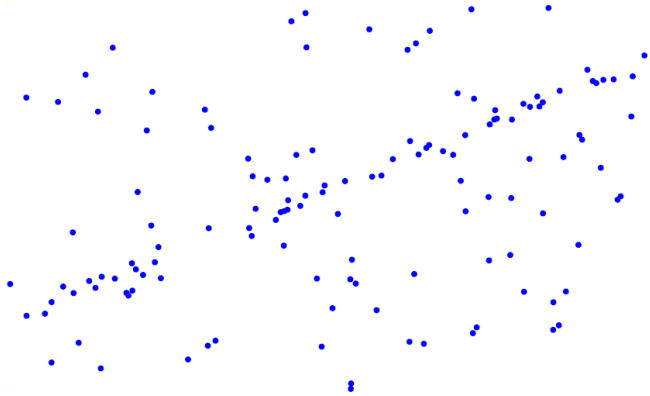
Correspondence and Least Squares



- one gross outlier - least squares fit fails - what do we do?

Credit: Yannis Avrithis, Inria Rennes-Bretagne Atlantique

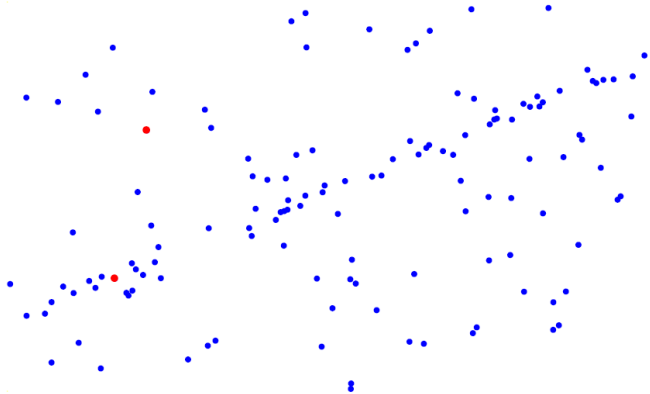
RANSAC (**RAN**dom **SA**mple **C**onsensus)[3]



- data with outliers - pick two points at random - draw line through them - set margin on either side - count inlier points

[3]Fischler and Bolles. CACM 1981. Random Sample Consensus: A Paradigm for Model Fitting With Applications to Image Analysis and Automated Cartography.

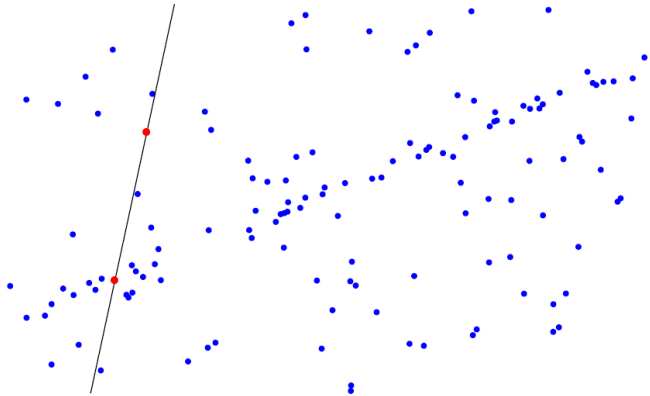
RANSAC (**RAN**dom **SA**mple **C**onsensus)[3]



- data with outliers - pick two points at random - draw line through them - set margin on either side - count inlier points

[3] Fischler and Bolles. CACM 1981. Random Sample Consensus: A Paradigm for Model Fitting With Applications to Image Analysis and Automated Cartography.

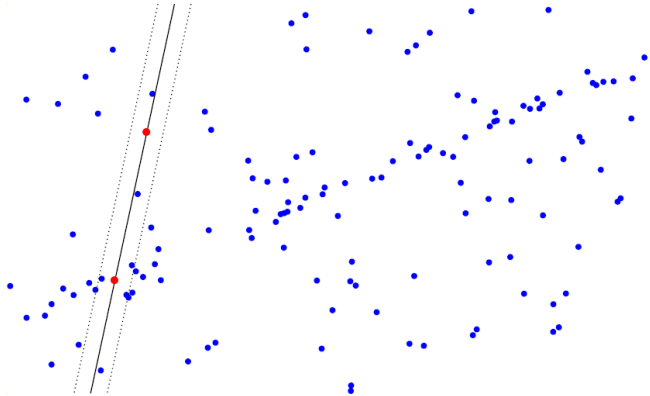
RANSAC (**RAN**dom **SA**mple **C**onsensus)[3]



- data with outliers - pick two points at random - draw line through them - set margin on either side - count inlier points

[3]Fischler and Bolles. CACM 1981. Random Sample Consensus: A Paradigm for Model Fitting With Applications to Image Analysis and Automated Cartography.

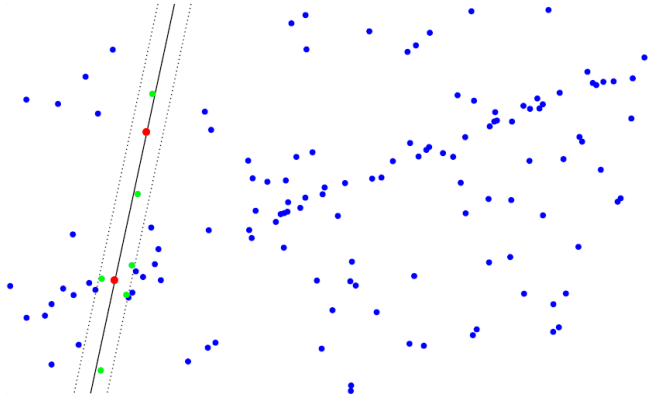
RANSAC (**RAN**dom **SA**mple **C**onsensus)[3]



- data with outliers - pick two points at random - draw line through them - set margin on either side - count inlier points

[3]Fischler and Bolles. CACM 1981. Random Sample Consensus: A Paradigm for Model Fitting With Applications to Image Analysis and Automated Cartography.

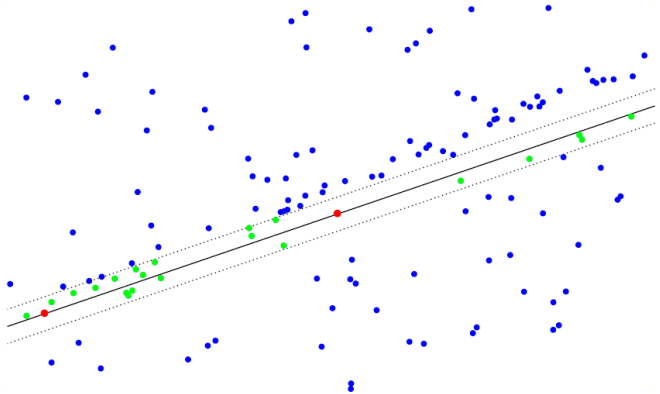
RANSAC (**RAN**dom **SA**mple **C**onsensus)[3]



- data with outliers - pick two points at random - draw line through them - set margin on either side - count inlier points

[3]Fischler and Bolles. CACM 1981. Random Sample Consensus: A Paradigm for Model Fitting With Applications to Image Analysis and Automated Cartography.

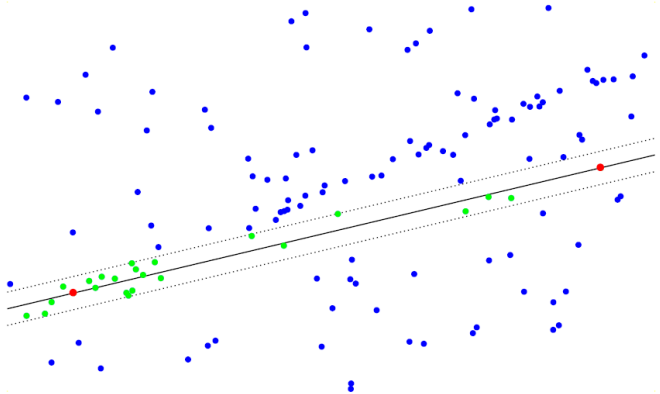
RANSAC (**RAN**dom **SA**mple **C**onsensus)[3]



- Repeat: pick two points at random, draw line through them, count inlier points at fixed distance to line, keep best hypothesis so far

[3]Fischler and Bolles. CACM 1981. Random Sample Consensus: A Paradigm for Model Fitting With Applications to Image Analysis and Automated Cartography.

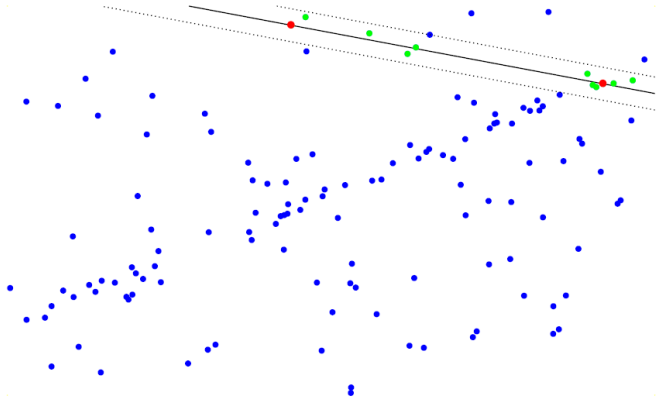
RANSAC (**RAN**dom **SA**mple **C**onsensus)[3]



- Repeat: pick two points at random, draw line through them, count inlier points at fixed distance to line, keep best hypothesis so far

[3]Fischler and Bolles. CACM 1981. Random Sample Consensus: A Paradigm for Model Fitting With Applications to Image Analysis and Automated Cartography.

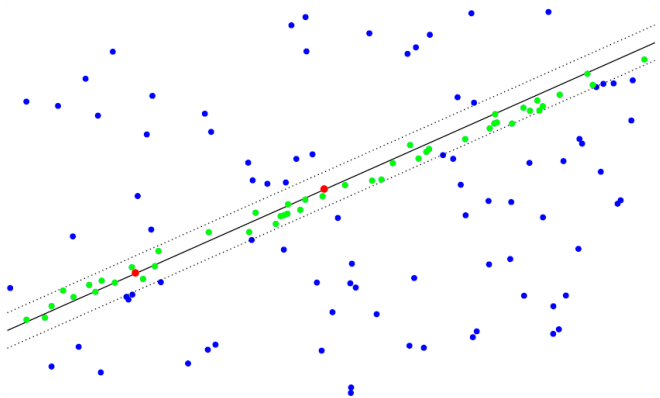
RANSAC (**RAN**dom **SA**mple **C**onsensus)[3]



- Repeat: pick two points at random, draw line through them, count inlier points at fixed distance to line, keep best hypothesis so far

[3]Fischler and Bolles. CACM 1981. Random Sample Consensus: A Paradigm for Model Fitting With Applications to Image Analysis and Automated Cartography.

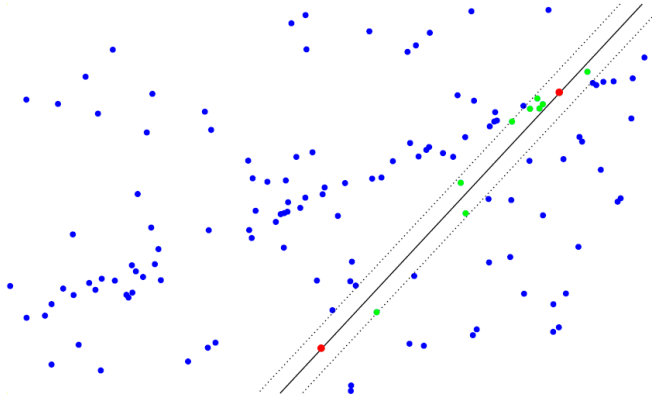
RANSAC (**RAN**dom **SA**mple **C**onsensus)[3]



- Repeat: pick two points at random, draw line through them, count inlier points at fixed distance to line, keep best hypothesis so far

[3]Fischler and Bolles. CACM 1981. Random Sample Consensus: A Paradigm for Model Fitting With Applications to Image Analysis and Automated Cartography.

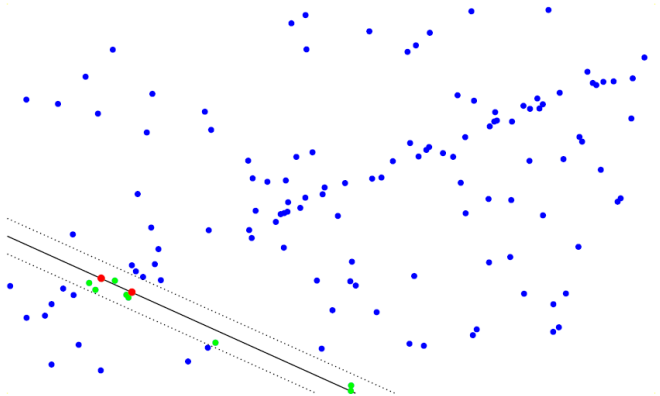
RANSAC (**RAN**dom **SA**mple **C**onsensus)[3]



- Repeat: pick two points at random, draw line through them, count inlier points at fixed distance to line, keep best hypothesis so far

[3]Fischler and Bolles. CACM 1981. Random Sample Consensus: A Paradigm for Model Fitting With Applications to Image Analysis and Automated Cartography.

RANSAC (**RAN**dom **SA**mple **C**onsensus)[3]



- Repeat: pick two points at random, draw line through them, count inlier points at fixed distance to line, keep best hypothesis so far

[3] Fischler and Bolles. CACM 1981. Random Sample Consensus: A Paradigm for Model Fitting With Applications to Image Analysis and Automated Cartography.

RANSAC

- X: data (tentative correspondences)
- n: minimum number of samples to fit a model
- $s(x;\theta)$: score of sample $\mathbf{x}$ given model parameters θ
- repeat:
  - hypothesis
    - draw n samples $H \subset X$ at random
    - fit model to H, compute parameters θ
  - verification
    - are data consistent with hypothesis? compute score $S = \sum_{\mathbf{x} \in X} s(\mathbf{x};\theta)$
    - if $S^* > S$, store solution $\theta^* := \theta$, $S^* := S$

RANSAC: Limitations

- Inlier ratio w (number of inliers in data / number of points in data) unknown
- Too expensive when minimum number of samples is large (e.g. $n > 6$) and inlier ratio is small (e.g. $w < 10\%$): 10^6 iterations for 1% probability of failure. (How?)

RANSAC: Limitations

- Inlier ratio w (number of inliers in data / number of points in data) unknown
- Too expensive when minimum number of samples is large (e.g. $n > 6$) and inlier ratio is small (e.g. $w < 10\%$): 10^6 iterations for 1% probability of failure. (How?)
  - $w^n \rightarrow$ probability that all n points are inliers

RANSAC: Limitations

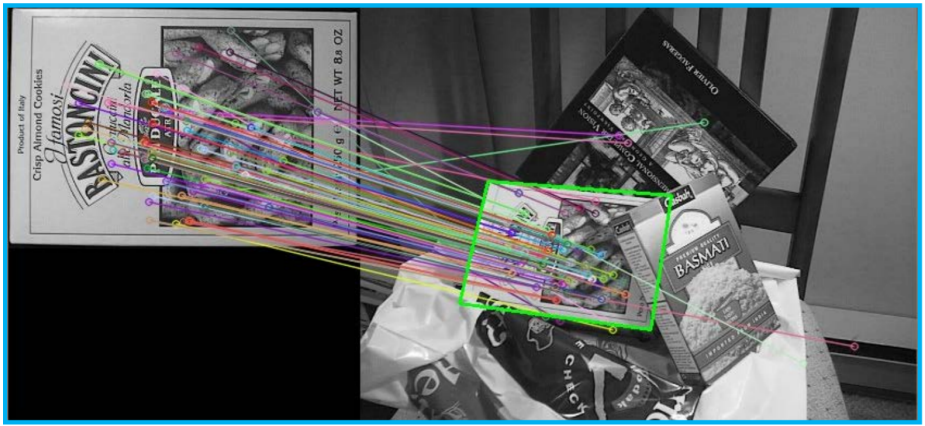
- Inlier ratio w (number of inliers in data / number of points in data) unknown
- Too expensive when minimum number of samples is large (e.g. $n > 6$) and inlier ratio is small (e.g. $w < 10\%$): 10^6 iterations for 1% probability of failure. (How?)
  - $w^n \rightarrow$ probability that all n points are inliers
  - $1 - w^n \rightarrow$ probability that at least one of n points is an outlier $\implies$ a bad model will be estimated from this point set

RANSAC: Limitations

- Inlier ratio w (number of inliers in data / number of points in data) unknown
- Too expensive when minimum number of samples is large (e.g. $n > 6$) and inlier ratio is small (e.g. $w < 10\%$): 10^6 iterations for 1% probability of failure. (How?)
  - $w^n \rightarrow$ probability that all n points are inliers
  - $1 - w^n \rightarrow$ probability that at least one of n points is an outlier $\implies$ a bad model will be estimated from this point set
  - $(1 - w^n)^k \rightarrow$ probability that algorithm never selects a set of n points which all are inliers, where $k \rightarrow$ number of iterations

RANSAC Applications

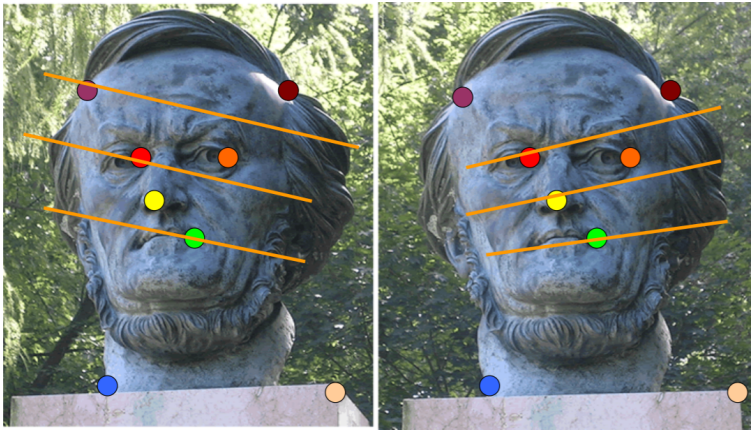
Rotation



Credit: Aaron Bobick, Washington University in St. Louis

RANSAC Applications

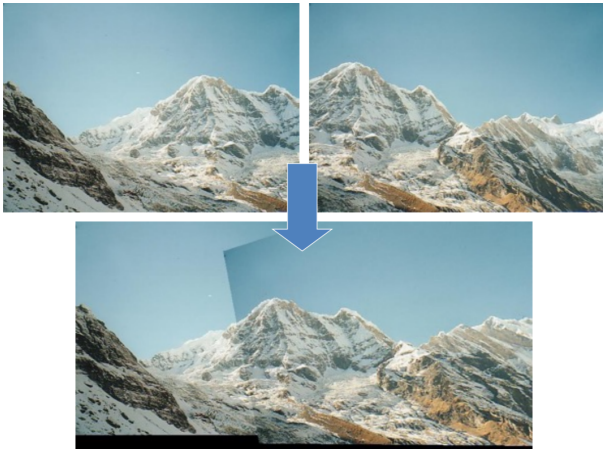
Estimating transformation matrix (also called **fundamental matrix**) relating two views



Credit: Derek Hoeim, UIUC

RANSAC Applications

Computing a **homography** (e.g., image stitching)



Credit: Ali Farhadi, Univ of Washington

Homework

Readings

- Chapter 4.3, 6.1, Szeliski, *Computer Vision: Algorithms and Applications*
- Papers on the respective slides (for more information)

References


Martin A. Fischler and Robert C. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". In: *Commun. ACM* 24.6 (June 1981), 381–395.


Bruce D. Lucas and Takeo Kanade. "An Iterative Image Registration Technique with an Application to Stereo Vision". In: *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*. IJCAI'81. Vancouver, BC, Canada: Morgan Kaufmann Publishers Inc., 1981, 674–679.


Richard Szeliski. *Computer Vision: Algorithms and Applications*. Texts in Computer Science. London: Springer-Verlag, 2011.


Avrithis, Yannis, Deep Learning for Vision (2018). URL: https://sif-dlv.github.io/ (visited on 05/21/2020).


Hoiem, Derek, CS 543 - Computer Vision (Spring 2011). URL: https://courses.engr.illinois.edu/cs543/sp2017/ (visited on 04/25/2020).

Deep Learning for Computer Vision

Hough Transform

Vineeth N Balasubramanian

Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad

Acknowledgements

- Most of this lecture's slides are based on lectures of **Deep Learning for Vision** course taught by Prof Yannis Avrithis at Inria Rennes-Bretagne Atlantique, as well as the **Computer Vision** course taught by Prof Mubarak Shah/Alper Yilmaz at the University of Central Florida

Line Fitting

- We have already seen a couple of line fitting algorithms: *Least squares fit* and *RANSAC*
- How do they perform when multiple lines are present?

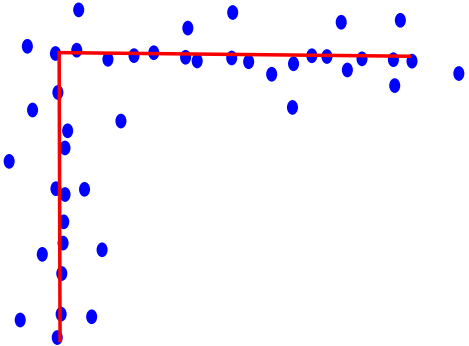


Figure 1: Example line configuration in an image.

Line Fitting: Hough Transform

- Hough, *Method and means for recognizing complex patterns*, U.S. Patent No. 3,069,654, Dec 1962
- Line equation in Cartesian co-ordinates is:
  - $y = mx + c$ $\quad$ m is slope, c is y-intercept
- Rearranging it slightly, we get:
  - $c = (-x)m + y$ $\quad$ which for a specific point (x_i, y_i) becomes $c = (-x_i)m + y_i$
- This can be thought of as the equation of line in parameter space; i.e in the (m, c) coordinate system with slope $-x_i$ and c-intercept y
- Each point in parameter space is a model

Source: Alper Yilmaz, Mubarak Shah, Fall 2011 UCF

Line Fitting: Hough Transform

- N samples needed to fit a model (2 points to fit a line)
- But even one sample brings some information
- In the space of all possible models, vote for ones that satisfy a given sample
- Collect votes for all samples, and seek for consensus

Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Hough Transform: Polar Parametrization

- The Cartesian formulation is problematic for vertical lines. Why?

Hough Transform: Polar Parametrization

- The Cartesian formulation is problematic for vertical lines. Why?
  - The slope is unbounded for vertical lines.
- Consider a polar parametrization of the line:
  - $\rho = x \cos\theta + y \sin\theta$
  - ρ is distance of line from origin and θ is angle made by normal to x-axis
  - For given line, $\rho \geq 0$ and $0 \leq \theta \leq 360^\circ$ are bounded

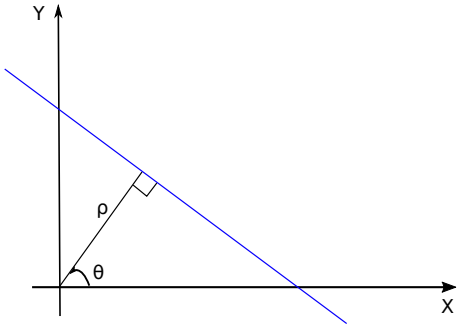


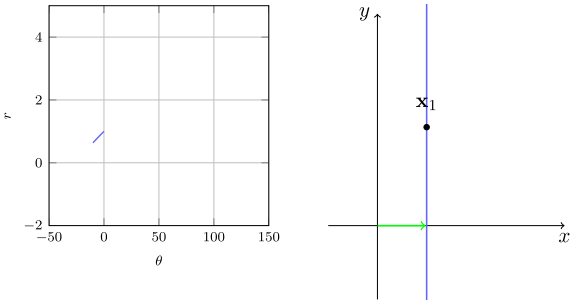
Figure 2: The ρ, θ parametrization.

Source: Alper Yilmaz, Mubarak Shah, Fall 2011 UCF

Hough Transform: Polar Parametrization

- A point (x_i, y_i) '*votes*' for many points in parameter space $\rightarrow$ **Hough Voting**
- Each line through a point (x_1, y_1) is a vote for a point in parameter space which satisfies $\rho = x_1 \cos\theta + y_1 \sin\theta$

voting in parameter space

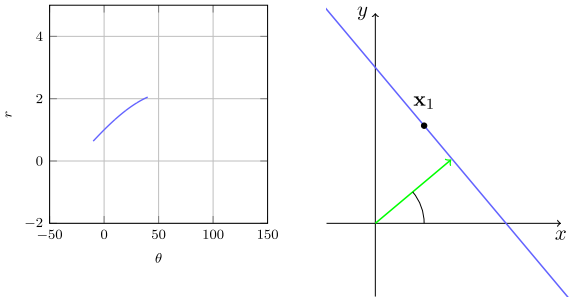


Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Hough Transform: Polar Parametrization

- A point (x_i, y_i) '*votes*' for many points in parameter space $\rightarrow$ **Hough Voting**
- Each line through a point (x_1, y_1) is a vote for a point in parameter space which satisfies $\rho = x_1 \cos\theta + y_1 \sin\theta$

voting in parameter space

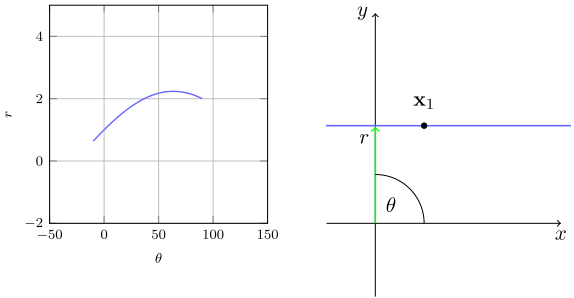


Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Hough Transform: Polar Parametrization

- A point (x_i, y_i) '*votes*' for many points in parameter space $\rightarrow$ **Hough Voting**
- Each line through a point (x_1, y_1) is a vote for a point in parameter space which satisfies $\rho = x_1 \cos\theta + y_1 \sin\theta$

voting in parameter space

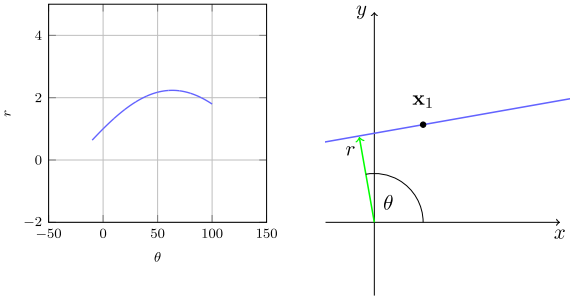


Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Hough Transform: Polar Parametrization

- A point (x_i, y_i) '*votes*' for many points in parameter space $\rightarrow$ **Hough Voting**
- Each line through a point (x_1, y_1) is a vote for a point in parameter space which satisfies $\rho = x_1 \cos\theta + y_1 \sin\theta$

voting in parameter space

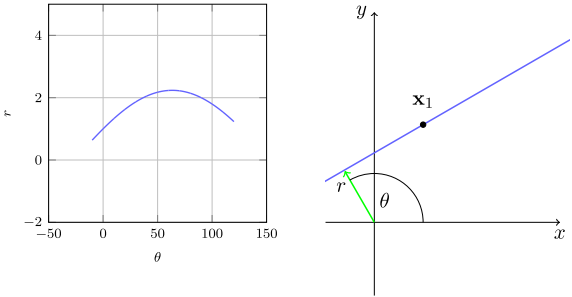


Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Hough Transform: Polar Parametrization

- A point (x_i, y_i) '*votes*' for many points in parameter space $\rightarrow$ **Hough Voting**
- Each line through a point (x_1, y_1) is a vote for a point in parameter space which satisfies $\rho = x_1 \cos\theta + y_1 \sin\theta$

voting in parameter space

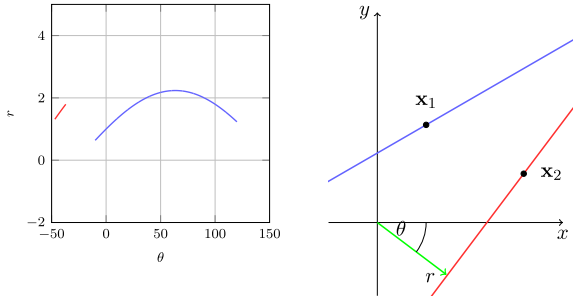


Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Hough Transform: Polar Parametrization

- Each line through a point (x_2, y_2) is a vote for a point in parameter space which satisfies $\rho = x_2 \cos\theta + y_2 \sin\theta$

voting in parameter space

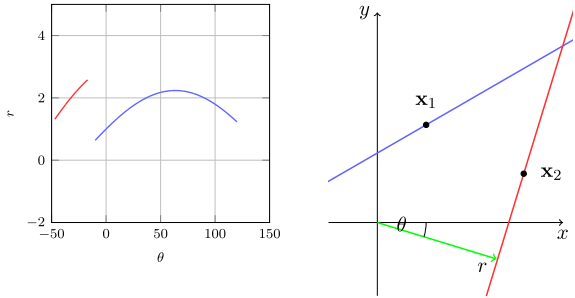


Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Hough Transform: Polar Parametrization

- Each line through a point (x_2, y_2) is a vote for a point in parameter space which satisfies $\rho = x_2 \cos\theta + y_2 \sin\theta$

voting in parameter space

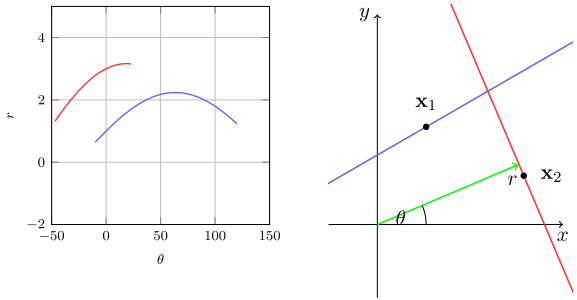


Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Hough Transform: Polar Parametrization

- Each line through a point (x_2, y_2) is a vote for a point in parameter space which satisfies $\rho = x_2 \cos\theta + y_2 \sin\theta$

voting in parameter space

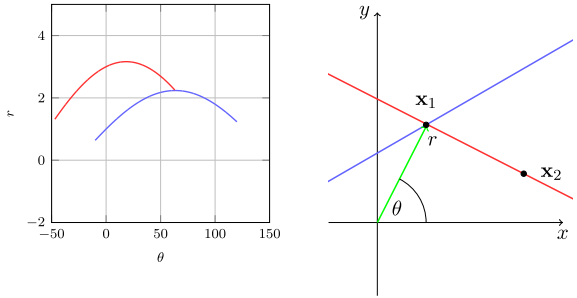


Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Hough Transform: Polar Parametrization

- Each line through a point (x_2, y_2) is a vote for a point in parameter space which satisfies $\rho = x_2 \cos\theta + y_2 \sin\theta$

voting in parameter space

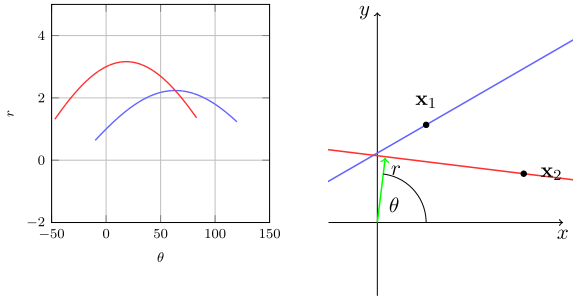


Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Hough Transform: Polar Parametrization

- Each line through a point (x_2, y_2) is a vote for a point in parameter space which satisfies $\rho = x_2 \cos\theta + y_2 \sin\theta$

voting in parameter space



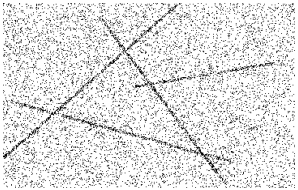
Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Hough Voting

procedure HOUGH VOTING(X, Θ)
    X: data Θ: quantized parameter $\theta_{min}, ..., \theta_{max}$
    A: accumulator array, initially zero
    for $(x, y) \in X$ **do**
        for $\theta \in \Theta$ **do**
            $\rho = x \cos\theta + y \sin\theta$
            $\triangleright$ for each set of model parameters consistent with a sample, increment A
            $A[\theta, \rho] = A[\theta, \rho] + 1$
        end for
    end for
    Non-maximum Suppression: detect local maxima in A
end procedure

Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

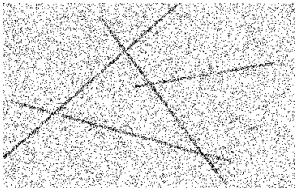
Line Detection



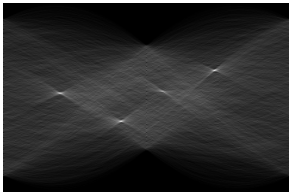
points

Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Line Detection



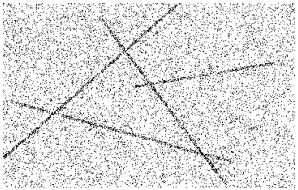
points



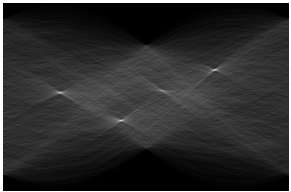
accumulator

Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Line Detection



points

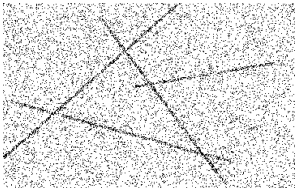


accumulator

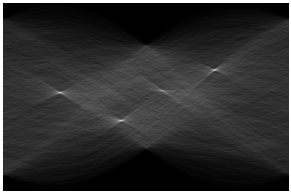
local maxima

Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

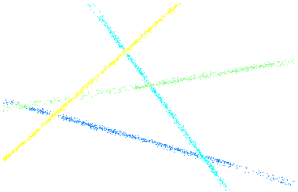
Line Detection



points



accumulator



labels

local maxima

Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Hough Transform: Finding Circles

- Circle fitting similar to line fitting
  - $(x - x_0)^2 + (y - y_0)^2 - r^2 = 0$
- What are the dimensions of accumulator A for circle fitting?

Hough Transform: Finding Circles

- Circle fitting similar to line fitting
  - $(x - x_0)^2 + (y - y_0)^2 - r^2 = 0$
- What are the dimensions of accumulator A for circle fitting?
  - 3D accumulator with dimensions x_0, y_0, r
- Fix one of the parameters (generally radius is fixed) and loop for the rest
- Increment accumulator A
- Find local maxima in A

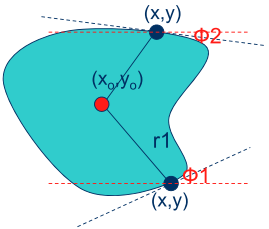
Source: Alper Yilmaz, Mubarak Shah, Fall 2011 UCF

Source: Ioannis Gkioulekas, 16-385 Computer Vision, Spring 2020, CMU

Generalized Hough Transform

- Used for shapes with no analytical expression
- Involves a training phase where R-table is computed
- Given object of interest, compute R-table as follows:
  - Compute centroid (x_c, y_c).
  - For each edge point (x_i, y_i), compute distance to centroid r_i and find edge orientation ϕ_i.
  - Construct a table of angles and r-values

Φ1	r1, r2, r3 …
Φ2	r14, r21, r23 …
Φ3	r41, r42, r33 …
Φ4	r10, r12, r13 …



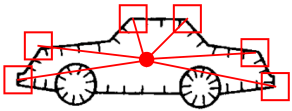
Source: Alper Yilmaz, Mubarak Shah, Fall 2011 UCF

Generalized Hough Transform

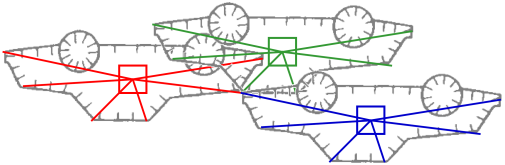
1: **procedure** GENERALIZED HOUGH TRANSFORM($X, R, A[x_c, y_c]$)
2: X: data R: R-table $A[x_c, y_c]$: Accumulator array with quantization $x_{c_{\min}}, \cdots, x_{c_{\max}}$ and $y_{c_{\min}}, \cdots, y_{c_{\max}}$
3: A: accumulator array, initially zero
4: **for** $(x, y) \in X$ **do**
5: **for** each $(r_i, \phi_i) \in R$ **do**
6: $x_c = x + r_i \cos \phi_i$
7: $y_c = y + r_i \sin \phi_i$
8: $A[x_c, y_c] = A[x_c, y_c] + 1$
9: **end for**
10: **end for**
11: Non-maximum Suppression: detect local maxima in A
12: **end procedure**

Generalized Hough Transform: Example

- **Build model:** Record coordinates relative to reference point
- **Test phase:** Each point votes for all possible coordinates of reference point



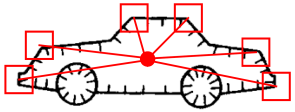
Model image



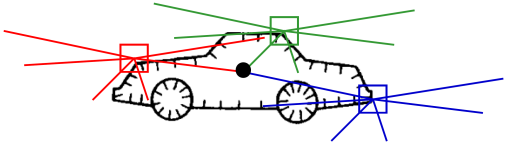
Test image

Generalized Hough Transform: Example

- **Build model:** Record coordinates relative to reference point
- **Test phase:** Each point votes for all possible coordinates of reference point

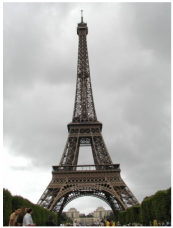


Model image



Test image

Generalized Hough Transform: Example

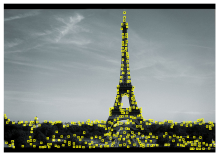


Model image

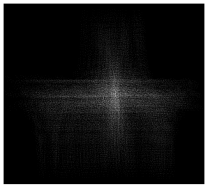
Test image

Generalized Hough Transform: Example

Model image points



Test image points



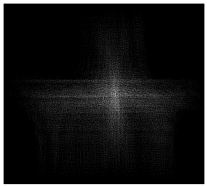
Accumulator

Local Maxima

Generalized Hough Transform: Example

Model image points

Test image with location



Accumulator

Local Maxima

Hough Transform

- Can be used for detection of:
  - shapes
  - objects, including multiple instances
- Advantages
  - Deals with occlusion well
  - Robust to noise
- Disadvantages
  - Can be computationally expensive
  - Setting parameters is not easy

Source: Ioannis Gkioulekas, 16-385 Computer Vision, Spring 2020, CMU

Homework

Readings

- Chapter 4.3, Szeliski, *Computer Vision: Algorithms and Applications*

Questions

- How would you use Hough transform to detect ellipses, squares and rectangles?
- Your friend working in a diagnostics startup asks you how to use Hough transform to automatically count Red Blood Cells in a blood sample. What would you advise your friend?

References

Richard O. Duda and Peter E. Hart. "Use of the Hough Transformation to Detect Lines and Curves in Pictures". In: *Commun. ACM* 15.1 (Jan. 1972), 11–15.

John Illingworth and Josef Kittler. "A survey of the Hough transform". In: *Computer vision, graphics, and image processing* 44.1 (1988), pp. 87–116.

Richard Szeliski. *Computer Vision: Algorithms and Applications*. Texts in Computer Science. London: Springer-Verlag, 2011.

Deep Learning for Computer Vision

From Points to Images: Bag-of-Words and VLAD Representations

Vineeth N Balasubramanian

Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad

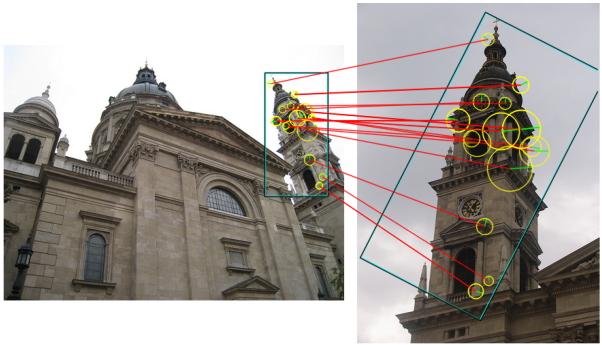
Acknowledgements

- Most of this lecture's slides are based on lectures of **Deep Learning for Vision** course taught by Prof Yannis Avrithis at Inria Rennes-Bretagne Atlantique

Review

So far:

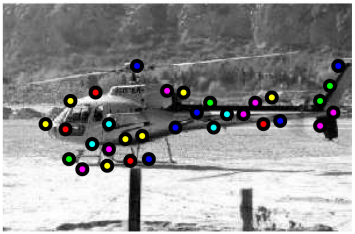
- Descriptors for matching features between different views of the same scene/object
  - Used in image stitching, image retrieval, etc.

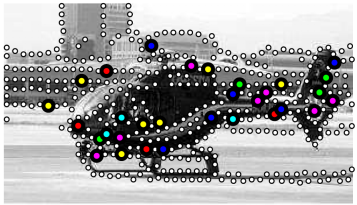


Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Review

- Can the same descriptors be used for matching different instances of the object?
  - Used in image classification, detection, etc.

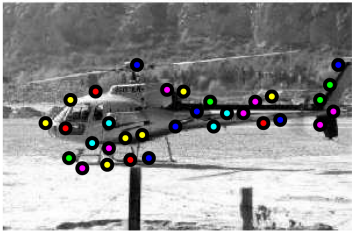


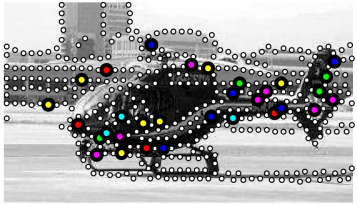


Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Review

- Can the same descriptors be used for matching different instances of the object?
  - Used in image classification, detection, etc.





Rigid transformations may not work here. Can we discard geometry for now, and bring it back in other ways?

Source: Yannis Avrithis, Deep Learning for Vision, Spring 2019 SIF

Our First Attempt: Bag-of-Words (BoW)

Samples

Our First Attempt: Bag-of-Words (BoW)

Samples

Form Vocabulary

Our First Attempt: Bag-of-Words (BoW)

Samples

Form Vocabulary

Histogram

Our First Attempt: Bag-of-Words (BoW)

1) John likes to watch movies. Mary likes movies too.

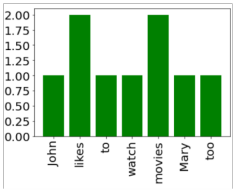
2) Mary also likes to watch football games..

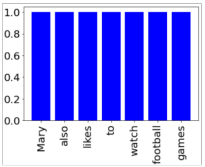
Samples

"John","likes","to","watch","movies","Mary","likes","movies","too"

"Mary","also","likes","to","watch","football","games"

Form Vocabulary





Histogram

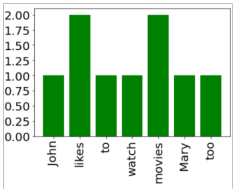
Our First Attempt: Bag-of-Words (BoW)

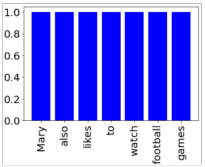
1) John likes to watch movies. Mary likes movies too.

2) Mary also likes to watch football games..

"John","likes","to","watch","movies","Mary","likes","movies","too"

"Mary","also","likes","to","watch","football","games"

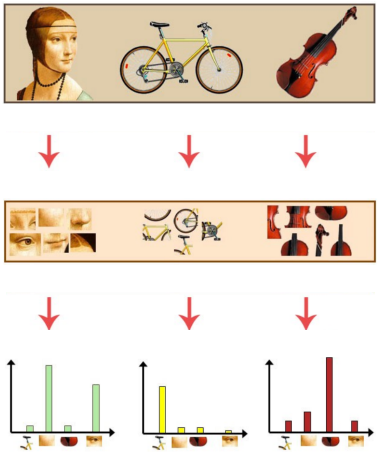




Samples

Form Vocabulary

Histogram



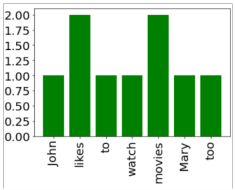
Our First Attempt: Bag-of-Words (BoW)

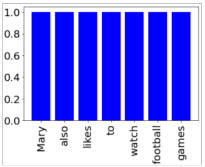
1) John likes to watch movies. Mary likes movies too.

2) Mary also likes to watch football games..

Samples

Uses of Bag-of-Words?





Histogram

Our First Attempt: Bag-of-Words (BoW)

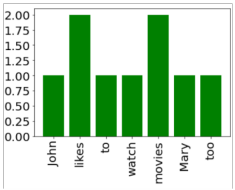
1) John likes to watch movies. Mary likes movies too.

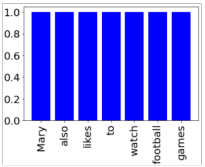
2) Mary also likes to watch football games..

Samples

Uses of Bag-of-Words?

- Retrieval
- Classification





Histogram

BoW for Retrieval[1]

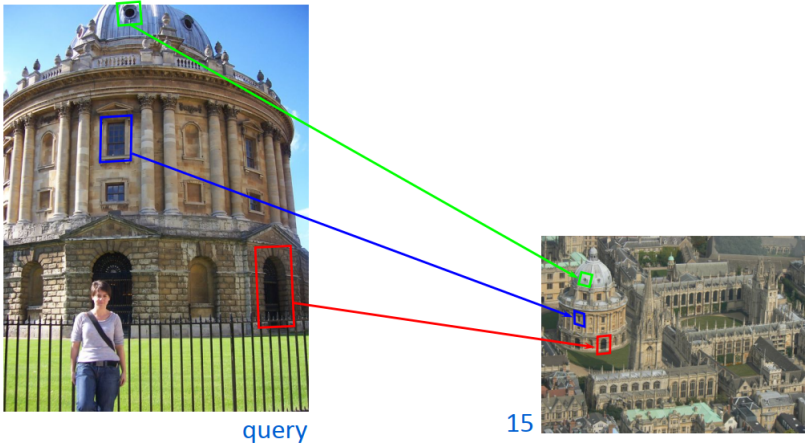
query

15

Query vs dataset image

[1] Sivic and Zisserman, Video Google: A Text Retrieval Approach to Object Matching in videos, ICCV 2003

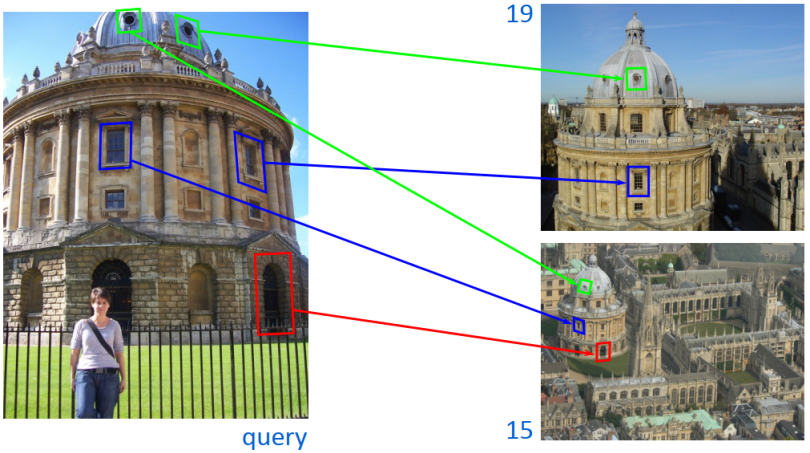
BoW for Retrieval[1]



Pairwise descriptor matching

[1]Sivic and Zisserman, Video Google: A Text Retrieval Approach to Object Matching in videos, ICCV 2003

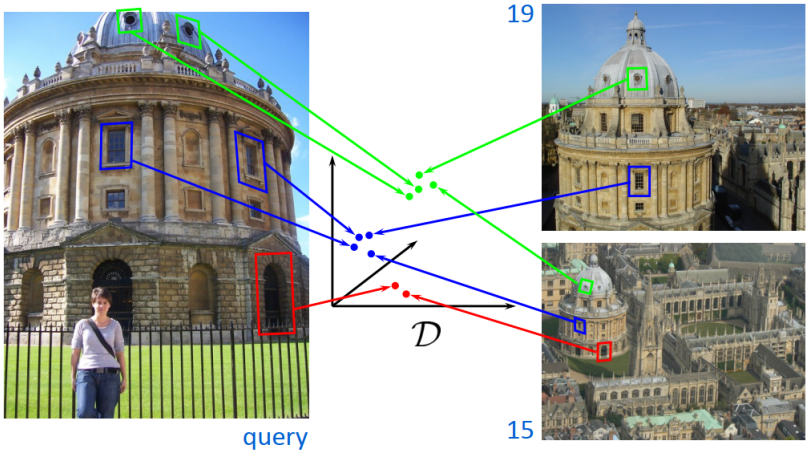
BoW for Retrieval[1]



Pairwise descriptor matching for every dataset image

[1]Sivic and Zisserman, Video Google: A Text Retrieval Approach to Object Matching in videos, ICCV 2003

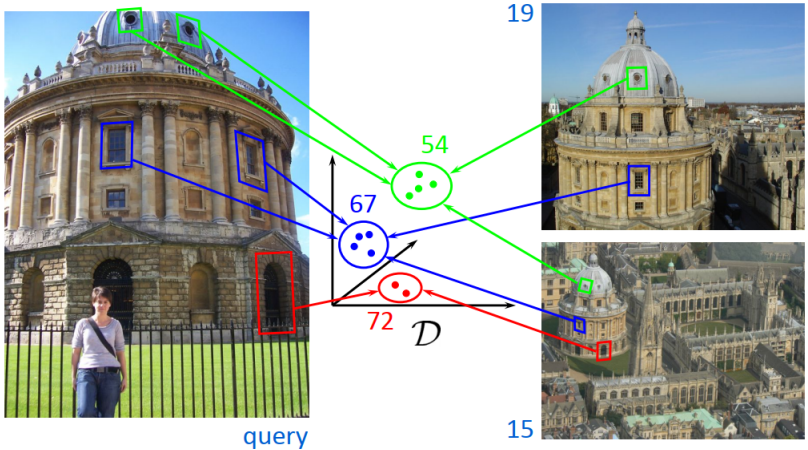
BoW for Retrieval[1]



Similar descriptors should all be nearby in the descriptor space

[1] Sivic and Zisserman, Video Google: A Text Retrieval Approach to Object Matching in videos, ICCV 2003

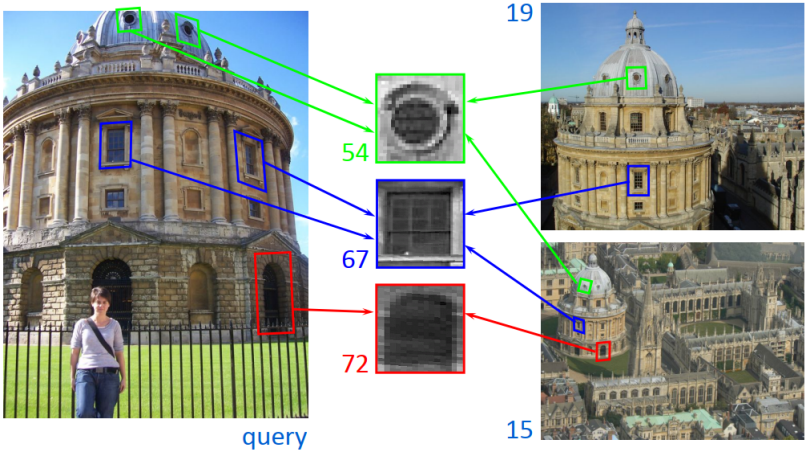
BoW for Retrieval[1]



Quantize them into visual words

[1] Sivic and Zisserman, Video Google: A Text Retrieval Approach to Object Matching in videos, ICCV 2003

BoW for Retrieval[1]



Now visual words act as a proxy. No pairwise matching needed

[1] Sivic and Zisserman, Video Google: A Text Retrieval Approach to Object Matching in videos, ICCV 2003

BoW for Retrieval

- Each image is represented by a vector $\mathbf{z} \in \mathbb{R}^k$, where k is size of codebook
  - Each element $z_i = w_i n_i$ where w_i is a fixed weight per visual word and n_i number of occurrences of this word in the image

BoW for Retrieval

- Each image is represented by a vector $\mathbf{z} \in \mathbb{R}^k$, where k is size of codebook
  - Each element $z_i = w_i n_i$ where w_i is a fixed weight per visual word and n_i number of occurrences of this word in the image
- Given a set of n images represented by matrix $Z \in \mathbb{R}^{k \times n}$ (each image as a column) and query image $\mathbf{q}$, we need a vector of similarities:

$$s = S_{\text{BoW}}(Z, \mathbf{q}) := Z^\top \mathbf{q}$$

and then sort s by descending order

- Note: With L_2-normalization, this is equivalent to measuring Euclidean distance: for vectors $\mathbf{z}$ and $\mathbf{q}$, $||\mathbf{z} - \mathbf{q}||^2 = 2(1 - \mathbf{z}^T\mathbf{q})$

BoW for Retrieval

- Each image is represented by a vector $\mathbf{z} \in \mathbb{R}^k$, where k is size of codebook
  - Each element $z_i = w_i n_i$ where w_i is a fixed weight per visual word and n_i number of occurrences of this word in the image
- Given a set of n images represented by matrix $Z \in \mathbb{R}^{k \times n}$ (each image as a column) and query image $\mathbf{q}$, we need a vector of similarities:

$$s = S_{\text{BoW}}(Z, \mathbf{q}) := Z^\top \mathbf{q}$$

  and then sort s by descending order
  - Note: With L_2-normalization, this is equivalent to measuring Euclidean distance: for vectors $\mathbf{z}$ and $\mathbf{q}$, $||\mathbf{z} - \mathbf{q}||^2 = 2(1 - \mathbf{z}^T\mathbf{q})$
- When $k \gg p$, where p is the number of features per image on average, Z and $\mathbf{q}$ are sparse

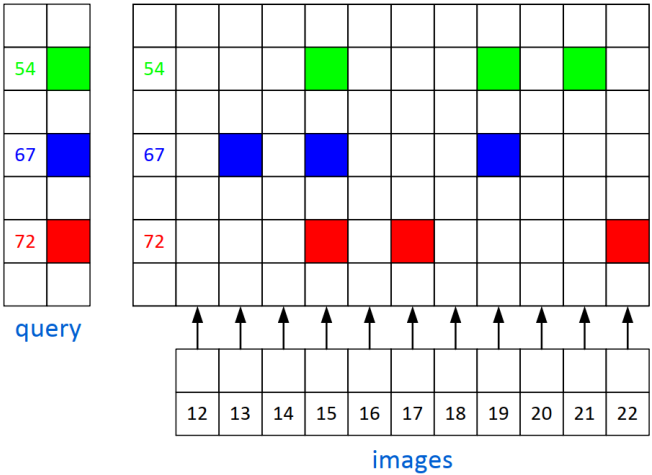
BoW for Retrieval

- Each image is represented by a vector $\mathbf{z} \in \mathbb{R}^k$, where k is size of codebook
  - Each element $z_i = w_i n_i$ where w_i is a fixed weight per visual word and n_i number of occurrences of this word in the image
- Given a set of n images represented by matrix $Z \in \mathbb{R}^{k \times n}$ (each image as a column) and query image $\mathbf{q}$, we need a vector of similarities:

$$s = S_{\text{BoW}}(Z, \mathbf{q}) := Z^\top \mathbf{q}$$

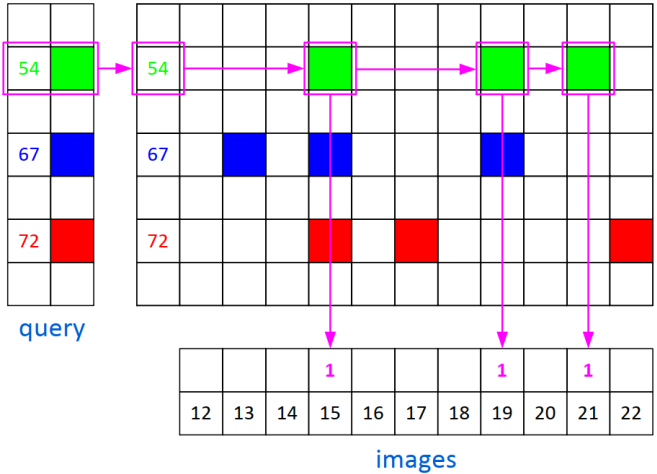
  and then sort s by descending order
  - Note: With L_2-normalization, this is equivalent to measuring Euclidean distance: for vectors $\mathbf{z}$ and $\mathbf{q}$, $||\mathbf{z} - \mathbf{q}||^2 = 2(1 - \mathbf{z}^T\mathbf{q})$
- When $k \gg p$, where p is the number of features per image on average, Z and $\mathbf{q}$ are sparse
- Rather than check whether a word is contained in an image, check which images contain a given word

BoW for Retrieval: Inverted File Index[2]



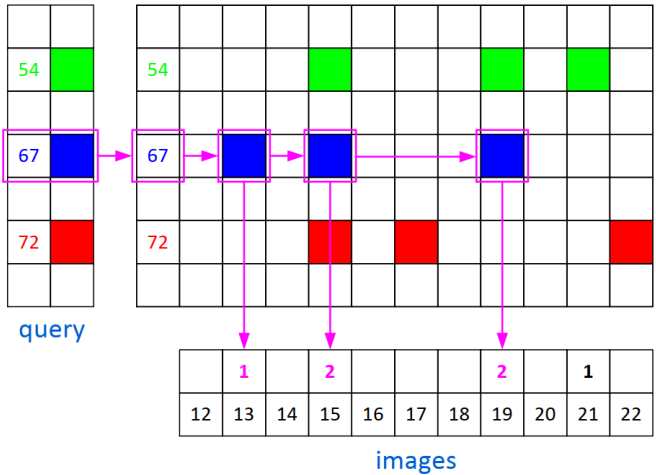
[2]Sivic and Zisserman, Video Google: A Text Retrieval Approach to Object Matching in videos, ICCV 2003

BoW for Retrieval: Inverted File Index[2]



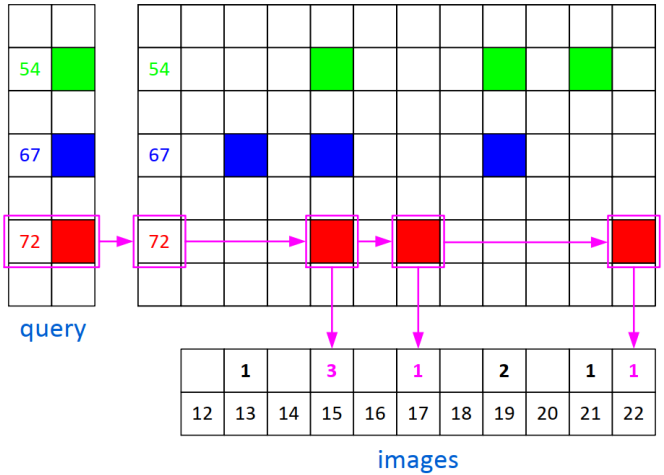
[2]Sivic and Zisserman, Video Google: A Text Retrieval Approach to Object Matching in videos, ICCV 2003

BoW for Retrieval: Inverted File Index[2]



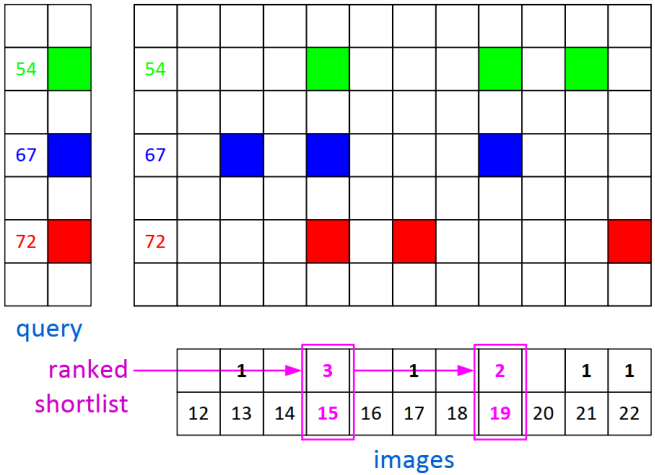
[2]Sivic and Zisserman, Video Google: A Text Retrieval Approach to Object Matching in videos, ICCV 2003

BoW for Retrieval: Inverted File Index[2]



[2]Sivic and Zisserman, Video Google: A Text Retrieval Approach to Object Matching in videos, ICCV 2003

BoW for Retrieval: Inverted File Index[2]



[2]Sivic and Zisserman, Video Google: A Text Retrieval Approach to Object Matching in videos, ICCV 2003

BoW for Classification

How?

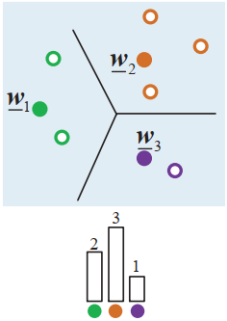
BoW for Classification

- Each image represented by $\mathbf{z} \in \mathbb{R}^k$; each element z_i the number of occurrences of visual word c_i in the image.

BoW for Classification

- Each image represented by $\mathbf{z} \in \mathbb{R}^k$; each element z_i the number of occurrences of visual word c_i in the image.
- **Naive Bayes:** Chose maximum posterior probability of class $\mathbf{C}$ given image $\mathbf{z}$ assuming features are independent $\rightarrow$ linear classifier with parameters estimated by visual word statistics on training set
- **Support Vector Machine (SVM):** Images $\mathbf{z}_1, \mathbf{z}_2$ compared using a kernel function $\phi(\cdot)$; if $\phi(\mathbf{z}_1)^T \phi(\mathbf{z}_2) = \mathbf{z}_1^T \mathbf{z}_2$, this is again a linear classifier

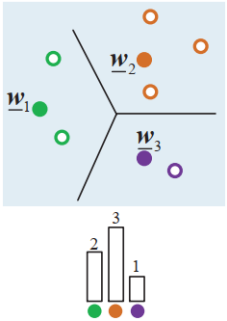
Extension of BoW: Vector of Locally Aggregated Descriptors (VLAD)



(BoW)

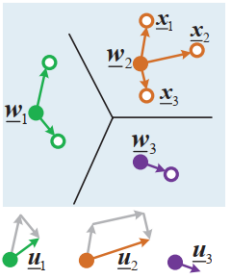
- Yields a scalar frequency
- Limited information

Extension of BoW: Vector of Locally Aggregated Descriptors (VLAD)



(BoW)

- Yields a scalar frequency
- Limited information

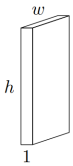


(VLAD)

- Yields a vector per visual word
- Comparatively more information, resulting in better discrimination by classifier

Credit: Li Liu

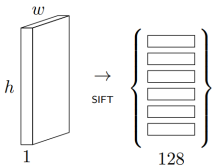
BoW Vs VLAD



- 3-channel RGB input $\rightarrow$ 1-channel gray-scale

Credit: Yannis A

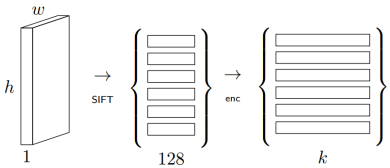
BoW Vs VLAD



- 3-channel RGB input $\rightarrow$ 1-channel gray-scale
- Set of $\sim$ 1000 features $\times$ 128-dim SIFT descriptors

Credit: Yannis A

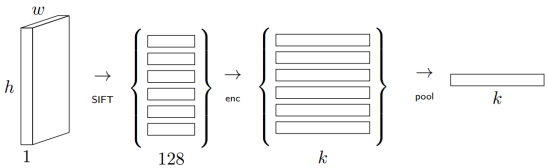
BoW Vs VLAD



- 3-channel RGB input $\rightarrow$ 1-channel gray-scale
- Set of $\sim$ 1000 features $\times$ 128-dim SIFT descriptors
- Element-wise encoding (hard assignment) on $k \sim 100$ visual words

Credit: Yannis A

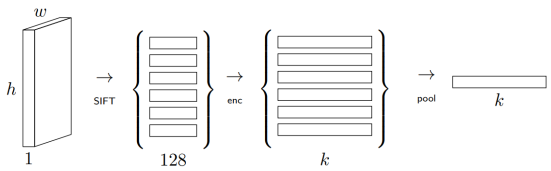
BoW Vs VLAD



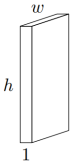
- 3-channel RGB input $\rightarrow$ 1-channel gray-scale
- Set of $\sim$ 1000 features $\times$ 128-dim SIFT descriptors
- Element-wise encoding (hard assignment) on $k \sim 100$ visual words
- Global sum pooling, L_2 normalization.

Credit: Yannis A

BoW Vs VLAD



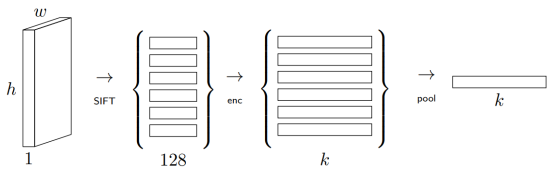
- 3-channel RGB input $\rightarrow$ 1-channel gray-scale
- Set of $\sim$ 1000 features $\times$ 128-dim SIFT descriptors
- Element-wise encoding (hard assignment) on $k \sim 100$ visual words
- Global sum pooling, L_2 normalization.



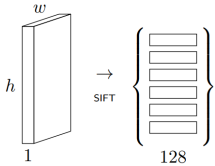
- 3-channel RGB input $\rightarrow$ 1-channel gray-scale

Credit: Yannis A

BoW Vs VLAD



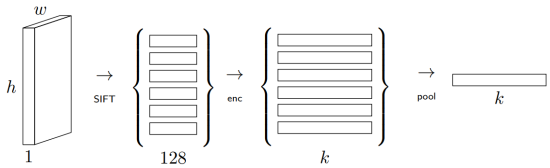
- 3-channel RGB input $\rightarrow$ 1-channel gray-scale
- Set of $\sim$ 1000 features $\times$ 128-dim SIFT descriptors
- Element-wise encoding (hard assignment) on $k \sim 100$ visual words
- Global sum pooling, L_2 normalization.



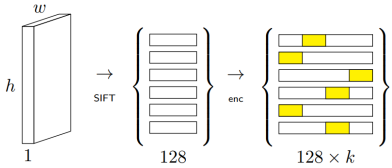
- 3-channel RGB input $\rightarrow$ 1-channel gray-scale
- Set of $\sim$ 1000 features $\times$ 128-dim SIFT descriptors

Credit: Yannis A

BoW Vs VLAD



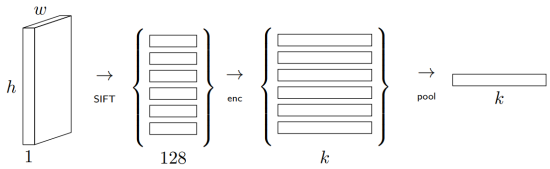
- 3-channel RGB input $\rightarrow$ 1-channel gray-scale
- Set of $\sim$ 1000 features $\times$ 128-dim SIFT descriptors
- Element-wise encoding (hard assignment) on $k \sim 100$ visual words
- Global sum pooling, L_2 normalization.



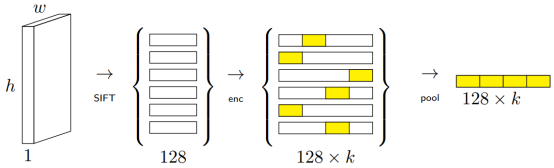
- 3-channel RGB input $\rightarrow$ 1-channel gray-scale
- Set of $\sim$ 1000 features $\times$ 128-dim SIFT descriptors
- Element-wise encoding (hard assignment) on $k \sim 100$ visual words. Yields a residual vector rather than a scalar vote

Credit: Yannis A

BoW Vs VLAD



- 3-channel RGB input $\rightarrow$ 1-channel gray-scale
- Set of $\sim$ 1000 features $\times$ 128-dim SIFT descriptors
- Element-wise encoding (hard assignment) on $k \sim 100$ visual words
- Global sum pooling, L_2 normalization.



- 3-channel RGB input $\rightarrow$ 1-channel gray-scale
- Set of $\sim$ 1000 features $\times$ 128-dim SIFT descriptors
- Element-wise encoding (hard assignment) on $k \sim 100$ visual words. Yields a residual vector rather than a scalar vote
- Global sum pooling, L_2 normalization.

Credit: Yannis A

Homework

Readings

- Chapter 14.3, 14.4, Szeliski, *Computer Vision: Algorithms and Applications*

Questions

- What is the connection of BoW to the k-means clustering algorithm?
- How would you now use k-means variants such as hierarchical k-means and approximate k-means to extend BoW? (*Hint: Look up vocabulary trees!*)

References

J Sivic and A Zisserman. "Video Google: A text retrieval approach to object matching in videos". In: ICCV. 2003.

Gabriella Csurka et al. "Visual categorization with bags of keypoints". In: *Workshop on statistical learning in computer vision, ECCV*. Vol. 1. 1-22. 2004, pp. 1–2.

Richard Szeliski. *Computer Vision: Algorithms and Applications*. Texts in Computer Science. London: Springer-Verlag, 2011.

Deep Learning for Computer Vision

Image Descriptor Matching

Vineeth N Balasubramanian

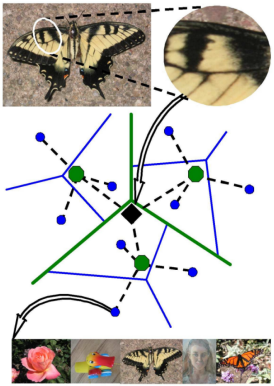
Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad

Acknowledgements

- Most of this lecture's slides are based on lectures of **Deep Learning for Vision** course taught by Prof Yannis Avrithis at Inria Rennes-Bretagne Atlantique

Review[1]

Hierarchical k-means and BoW:



- Apply hierarchical k-means and build a fine partition tree
- Descriptors descend from root to leaves by finding nearest node at each level
- Image represented by $x_i = w_i n_i$ as in BoW
- Dataset searched by inverted files at leaves
- No principled way of defining w_i across levels
- Distortion minimized only locally; points can get assigned to leaves that are not globally nearest

[1] Nister and Stewenius, Scalable Recognition With a Vocabulary Tree, CVPR 2006

Image Descriptor Matching: Options So Far

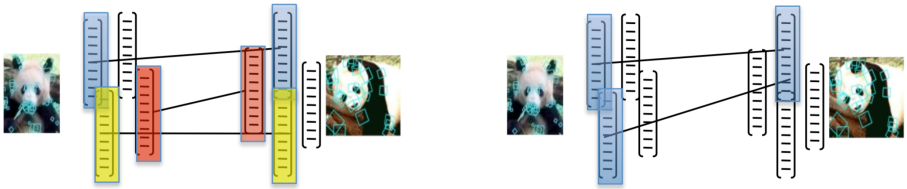
Nearest Neighbor Matching:

- Use each feature in a set to independently index into second set. Any problems you see?

Image Descriptor Matching: Options So Far

Nearest Neighbor Matching:

- Use each feature in a set to independently index into second set. Any problems you see?
- Ignores possibly useful information of co-occurrence $\Longrightarrow$ fails to distinguish between instances where an object has varying numbers of similar features since multiple features may be matched to a single feature in the other set



Source: Alberto Del Bimbo, UNIFI, Italy

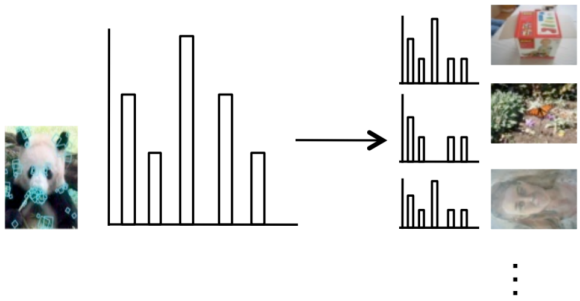
Image Descriptor Matching: Options So Far

Bag-of-Words Matching: Any glaring limitation?

Image Descriptor Matching: Options So Far

Bag-of-Words Matching: Any glaring limitation?

- Can only compare entire images to one another and does not allow partial matchings
- This implies an **all-all** matching; it is often preferable to have a **one-one** matching instead



Source: Alberto Del Bimbo, UNIFI, Italy

Generalizing Descriptor Matching using Kernels[2]

- Consider an image described by a set of n descriptors (features) $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, each of d dimensions

[2] Tolias et al, To Aggregate or Not to aggregate: Selective Match Kernels for Image Search, CVPR 2013

Generalizing Descriptor Matching using Kernels[2]

- Consider an image described by a set of n descriptors (features) $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, each of d dimensions
  - Descriptors typically quantized using k-means clustering
  - Quantizer $q : R^d \rightarrow C \subset R^d$ maps each descriptor to a representative descriptor, a.k.a **visual word**
  - $C = \{\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_k\}$ is a codebook consisting of k visual words.

[2] Tolias et al, To Aggregate or Not to aggregate: Selective Match Kernels for Image Search, CVPR 2013

Generalizing Descriptor Matching using Kernels[2]

- Consider an image described by a set of n descriptors (features) $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, each of d dimensions
  - Descriptors typically quantized using k-means clustering
  - Quantizer $q : R^d \rightarrow C \subset R^d$ maps each descriptor to a representative descriptor, a.k.a **visual word**
  - $C = \{\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_k\}$ is a codebook consisting of k visual words.
- To compare two image representations X and Y, let us define a general family of matching kernels:

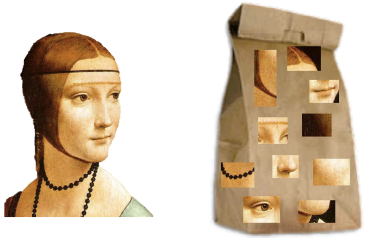
$$K(X, Y) = \gamma(X)\gamma(Y) \sum_{\mathbf{c} \in C} M(X_c, Y_c) \tag{1}$$

where $X_c = \{\mathbf{x} \in X : q(\mathbf{x}) = \mathbf{c}\}$ is the set of descriptors assigned to the same visual word, M is a within-cell matching function, and γ is a normalization function

[2] Tolias et al, To Aggregate or Not to aggregate: Selective Match Kernels for Image Search, CVPR 2013

Bag of Words Matching[3]

Recall: BoW model characterizes an image solely by visual words



- Cosine similarity in BoW model can be defined by defining M as:

$$M(X_c, Y_c) = \sum_{\mathbf{x} \in X_c} \sum_{\mathbf{y} \in Y_c} 1$$

Image Credit: Fei-Fei, Fergus and Torralba, Recognizing and Learning Object Categories, CVPR 2007 Tutorial

[3] Jegou et al, Aggregating local descriptors into a compact image representation, CVPR 2010

Hamming Embedding for Matching[4]

- In addition to being quantized, each descriptor $\mathbf{x}$ is binarized as $\mathbf{b}_x$.
- Score is computed between all pairs of descriptors assigned to the same visual word as:

$$M(X_c, Y_c) = \sum_{\mathbf{x} \in X_c} \sum_{\mathbf{y} \in Y_c} 1[h(\mathbf{b}_x, \mathbf{b}_y) \leq \tau]$$

where $h(\cdot, \cdot)$ is the Hamming distance between two binary vectors, and τ is a threshold to count matched pairs

[4] Jegou et al, Aggregating local descriptors into a compact image representation, CVPR 2010

VLAD Matching[5]

- **Recall:** For each visual word, VLAD performs *pooling* by constructing a vector representing the sum of residuals as:

$$V(X_c) = \sum_{\mathbf{x} \in X_c} r(\mathbf{x}) \qquad \text{where} \qquad r(\mathbf{x}) = \mathbf{x} - q(\mathbf{x})$$

[5] Jegou et al, Aggregating local descriptors into a compact image representation, CVPR 2010

VLAD Matching[5]

- **Recall:** For each visual word, VLAD performs *pooling* by constructing a vector representing the sum of residuals as:

$$V(X_c) = \sum_{\mathbf{x} \in X_c} r(\mathbf{x}) \qquad \text{where} \qquad r(\mathbf{x}) = \mathbf{x} - q(\mathbf{x})$$

- A $d \times k$ vector is constructed for an image X as follows:

$$V(X) = (V(X_{c_1}), V(X_{c_2}), V(X_{c_3}), .., V(X_{c_k}))$$

- Matching kernel now defined as:

$$M(X_c, Y_c) = V(X_c)^T V(Y_c) = \sum_{\mathbf{x} \in X_c} \sum_{\mathbf{y} \in Y_c} r(\mathbf{x})^T r(\mathbf{y})$$

[5] Jegou et al, Aggregating local descriptors into a compact image representation, CVPR 2010

Aggregated Selective Match Kernel (ASMK)[6]

- ASMK is a combination of two borrowed ideas:
  - Non-linear selective function (from Hamming Embedding)
  - Pooling Residuals (from VLAD)
- Matching kernel M is expressed as:

$$M(X_c, Y_c) = \sigma_\alpha(\hat{V}(X_c)^T \hat{V}(Y_c))$$

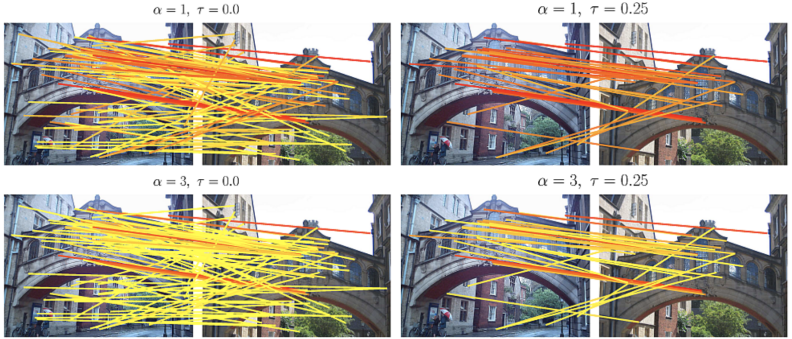
where σ_α is a non-linear function given by:

$$\sigma_\alpha(u) = \begin{cases} \text{sign}(u)|u|^\alpha & \text{if } u > \tau \\ 0 & \text{otherwise} \end{cases}$$

and $\hat{V}(X_c) = V(X_c)/||V(X_c)||$ and $V(X_c)$ is VLAD representation discussed earlier

[6] Tolias et al, To Aggregate or Not to aggregate: Selective Match Kernels for Image Search, CVPR 2013

Aggregated Selective Match Kernel (ASMK)



- ASMK matching with different values of distance threshold and selectivity parameter
- Yellow corresponds to 0 similarity and red to maximum similarity per image pair, as defined by selective function
- Larger selectivity drastically down-weights false correspondences
- This replaces hard thresholding in the Hamming Embedding method

Aggregated Selective Match Kernel (ASMK)

ASMK Example: Each visual word is drawn with a different color

Efficient Match Kernels[7]

- Instead of threshold-based matching functions (as used in HE), we can use a continuous function $\kappa(\mathbf{x}, \mathbf{y})$ and avoid using computationally intensive codebooks:

$$K(X, Y) = \gamma(X)\gamma(Y) \sum_{\mathbf{x} \in X} \sum_{\mathbf{y} \in Y} \kappa(\mathbf{x}, \mathbf{y})$$

- Such a function $K(X, Y)$ can be decomposed into an inner product of $\Phi(X)$ and $\Phi(Y)$
- To do that, we learn a low-dimensional feature map ϕ such that $\kappa(x, y) = \phi(x)^T \phi(y)$ and:

$$K(X, Y) = \left(\gamma(X) \sum_{\mathbf{x} \in X} \phi(x) \right)^T \left(\gamma(Y) \sum_{\mathbf{y} \in Y} \phi(y) \right) = \Phi(X)^T \Phi(Y)$$

[7] Bo and Sminchisescu, Efficient Match Kernels between Sets of Features for Visual Recognition, NeurIPS 2009

Homework

Readings

- Tolias *et al.* To Aggregate or Not to aggregate: Selective Match Kernels for Image Search. CVPR 2013
- Chapter 14.4, Szeliski, *Computer Vision: Algorithms and Applications*

References

Liefeng Bo and Cristian Sminchisescu. "Efficient Match Kernels between Sets of Features for Visual Recognition". In: *Proceedings of the 22nd International Conference on Neural Information Processing Systems*. NIPS'09. Vancouver, British Columbia, Canada: Curran Associates Inc., 2009, 135–143.

H. Jégou et al. "Aggregating local descriptors into a compact image representation". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2010, pp. 3304–3311.

Richard Szeliski. *Computer Vision: Algorithms and Applications*. Texts in Computer Science. London: Springer-Verlag, 2011.

G. Tolias, Y. Avrithis, and H. Jégou. "To Aggregate or Not to aggregate: Selective Match Kernels for Image Search". In: *2013 IEEE International Conference on Computer Vision*. 2013, pp. 1401–1408.

Deep Learning for Computer Vision

Pyramid Matching

Vineeth N Balasubramanian

Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad

Acknowledgements

- Most of this lecture's slides are based on lectures of **Deep Learning for Vision** course taught by Prof Yannis Avrithis at Inria Rennes-Bretagne Atlantique

Recall: Descriptor Matching

- Given two images with descriptors $X, Y \subset \mathbb{R}^d$, $X_c = \{\mathbf{x} \in X : q(\mathbf{x}) = \mathbf{c}\}$ where q maps vector $\mathbf{x}$ to its nearest centroid, bag-of-words similarity on C is given by:

$$s_{BoW}(X, Y) \propto \sum_{\mathbf{c} \in C} w_c |X_c||Y_C| = \sum_{\mathbf{c} \in C} w_c \sum_{\mathbf{x} \in X_c} \sum_{\mathbf{y} \in Y_c} 1 \tag{1}$$

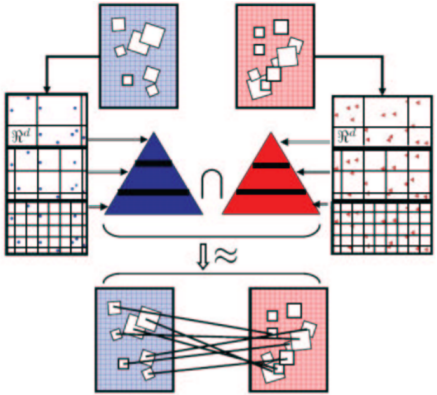
- More general form:

$$K(X, Y) := \gamma(X)\gamma(Y) \sum_{\mathbf{c} \in C} w_c M(X_c, Y_c)$$

where M is a within-cell matching function, and $\gamma(X)$ serves for normalization

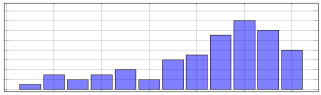
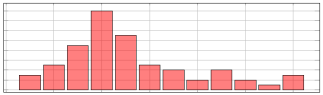
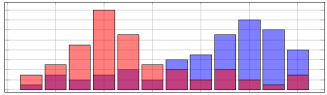
Going Beyond Single-level Matching: Pyramid Match Kernel (PMK)[1]

- **Pyramid matching:** an efficient method that maps unordered feature sets to multi-resolution histograms
- Computes a weighted histogram intersection to find implicit correspondences based on finest resolution histogram cell where a matched pair first appears
- Approximates similarity measured by optimal correspondences between feature sets of unequal cardinality



[1]Grauman and Darrell, The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features, IEEE ICCV 2005, Vol 2, pp. 1458–1465

Histogram Intersection[2]

- Given two histograms $\mathbf{x}, \mathbf{y}$ of b bins each, their **histogram intersection** is:

$$\kappa_{HI}(\mathbf{x}, \mathbf{y}) = \sum_{i-1}^{b} \min(x_i, y_i)$$

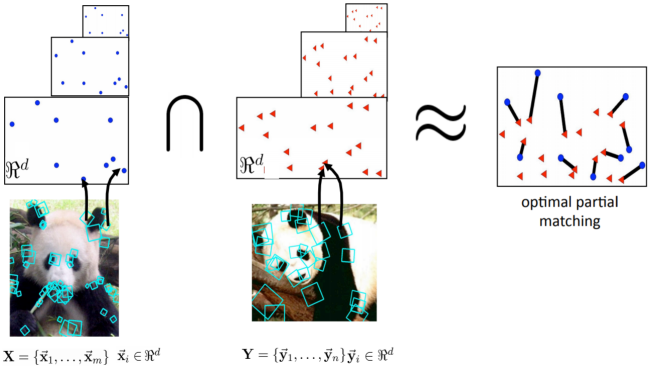
- This is related to L_1 distance as:

$$||\mathbf{x} - \mathbf{y}||_1 = ||\mathbf{x}||_1 + ||\mathbf{y}||_1 - 2\kappa_{HI}(\mathbf{x}, \mathbf{y})$$

[2] Swain and Ballard, Color Indexing, IJCV 1991, pp 11–32

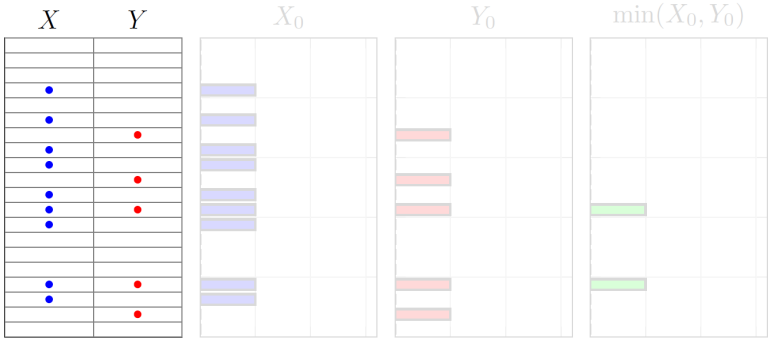
Pyramid Match Kernel (PMK)[3]

Weighted sum of histogram intersections at different levels of two images approximates their optimal pairwise matching



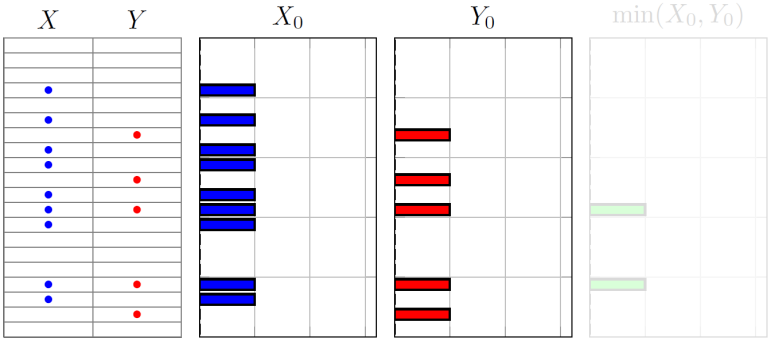
[3] Grauman and Darrell, The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features, IEEE ICCV 2005, Vol 2, pp. 1458–1465

Pyramid Match Kernel: Method



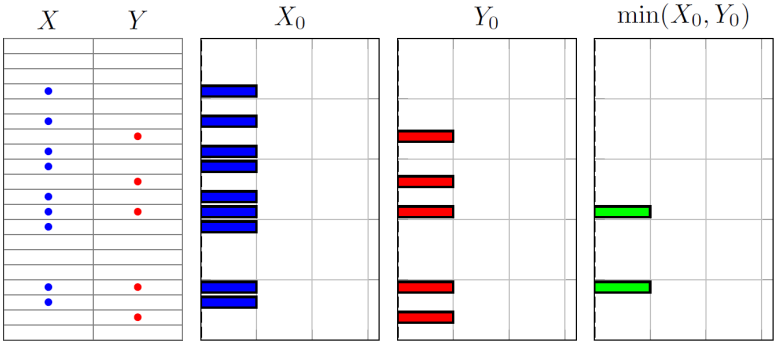
- 1-D point sets X, Y on grid of size 1

Pyramid Match Kernel: Method



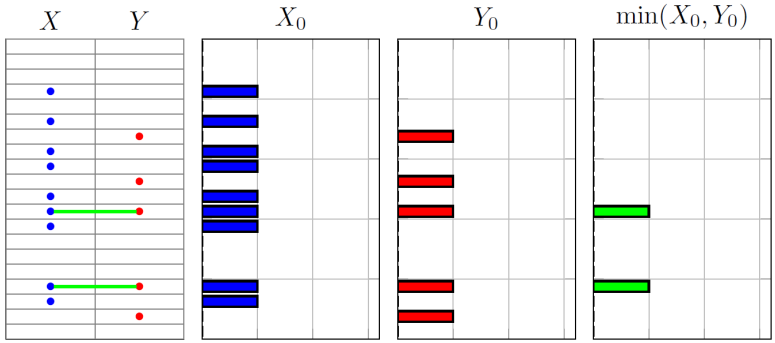
- 1-D point sets X, Y on grid of size 1 - level 0 histograms

Pyramid Match Kernel: Method



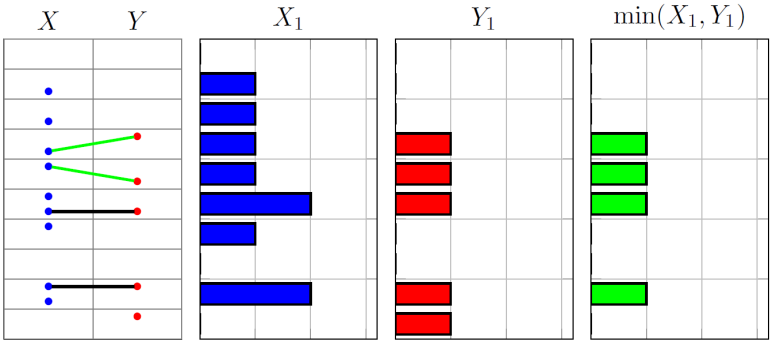
- 1-D point sets X, Y on grid of size 1 - level 0 histograms - intersection

Pyramid Match Kernel: Method



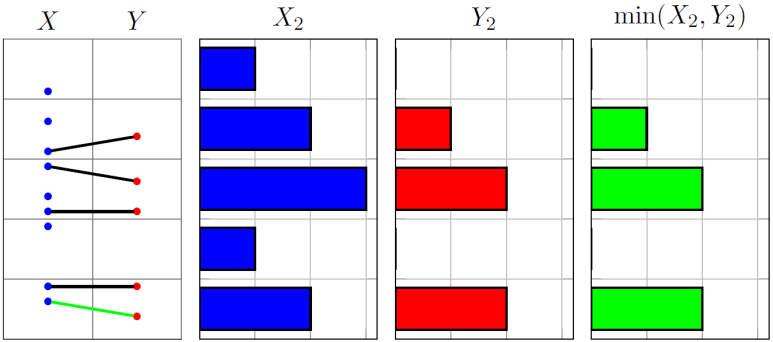
- 1-D point sets X, Y on grid of size 1 - level 0 histograms - intersection
- 2 matches weighted by 1
- Total similarity score: $2 \times 1 = 2$

Pyramid Match Kernel: Method



- 1-D point sets X, Y on grid of size 2 - level 1 histograms - intersection
- (2 matches weighted by 1) + (2 weighted by $\frac{1}{2}$)
- Total similarity score: $2 \times 1 + 2 \times \frac{1}{2} = 3$

Pyramid Match Kernel: Method



- 1-D point sets X, Y on grid of size 4 - level 2 histograms - intersection
- (2 matches weighted by 1) + (2 weighted by $\frac{1}{2}$) + (1 weighted by $\frac{1}{4}$)
- Total similarity score: $2 \times 1 + 2 \times \frac{1}{2} + 1 \times \frac{1}{4} = \mathbf{3.25}$

Pyramid Match Kernel

- Given a set $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\} \subset \mathbb{R}^d$, where distances of elements range in $[1, D]$

Pyramid Match Kernel

- Given a set $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\} \subset \mathbb{R}^d$, where distances of elements range in $[1, D]$
- Let X_i be a histogram of X in $\mathbb{R}^d$ on a regular grid of side length 2^i
  - i ranges from -1 (a base case where there is no intersection, 0 (where each bin has atmost one element), and so on to $L = \lceil \log_2 D \rceil$, where all of X is contained in a single bin

Pyramid Match Kernel

- Given a set $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\} \subset \mathbb{R}^d$, where distances of elements range in $[1, D]$
- Let X_i be a histogram of X in $\mathbb{R}^d$ on a regular grid of side length 2^i
  - i ranges from -1 (a base case where there is no intersection, 0 (where each bin has atmost one element), and so on to $L = \lceil \log_2 D \rceil$, where all of X is contained in a single bin
- Given two images with descriptors $X, Y \subset \mathbb{R}^d$, their **pyramid match** is:

$$
\begin{aligned}
K_\Delta(X, Y) &= \gamma(X)\gamma(Y) \sum_{i=0}^{L} \frac{1}{2^i} \Bigg(\underbrace{\kappa_{HI}(X_i, Y_i)}_{\text{Matches at this level}} - \underbrace{\kappa_{HI}(X_{i-1}, Y_{i-1})}_{\text{Matches at previous level}} \Bigg) \\
&= \gamma(X)\gamma(Y) \left(\frac{1}{2^L} \kappa_{HI}(X_L, Y_L) + \sum_{i=0}^{L-1} \frac{1}{2^{i+1}} \kappa_{HI}(X_i, Y_i) \right)
\end{aligned}
\tag{2}
$$

where $\gamma(X)$ serves for normalization

Pyramid Match Kernel

- Given a set $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\} \subset \mathbb{R}^d$, where distances of elements range in $[1, D]$
- Let X_i be a histogram of X in $\mathbb{R}^d$ on a regular grid of side length 2^i
  - i ranges from -1 (a base case where there is no intersection, 0 (where each bin has atmost one element), and so on to $L = \lceil \log_2 D \rceil$, where all of X is contained in a single bin
- Given two images with descriptors $X, Y \subset \mathbb{R}^d$, their **pyramid match** is:

$$K_\Delta(X, Y) = \gamma(X)\gamma(Y) \sum_{i=0}^{L} \frac{1}{2^i} \Bigg(\underbrace{\kappa_{HI}(X_i, Y_i)}_{\text{Matches at this level}} - \underbrace{\kappa_{HI}(X_{i-1}, Y_{i-1})}_{\text{Matches at previous level}} \Bigg) \tag{2}$$

$$= \gamma(X)\gamma(Y) \left(\frac{1}{2^L} \kappa_{HI}(X_L, Y_L) + \sum_{i=0}^{L-1} \frac{1}{2^{i+1}} \kappa_H(X_i, Y_i) \right)$$

where $\gamma(X)$ serves for normalization

Counts number of new pairs matched

PMK is a Positive Definite Kernel

- K_Δ can be written as a weighted sum of κ_{HI} terms, with non-negative coefficients
- κ_{HI} can be written as a sum of min terms
- min can be written as a dot product:

x	$\phi(x)$							
3	1	1	1	0	0	0	0	0
5	1	1	1	1	1	0	0	0
$\min(x, y) = 3$	1	1	1	0	0	0	0	0

- Therefore, so can K_Δ

PMK as an Embedding[4]

- There is an explicit embedding for κ_{HI}, therefore also for K_Δ. What could it be?

[4] Indyk and Thaper, Fast Image Retrieval via Embeddings, WSCTV. 2003

PMK as an Embedding[4]

- There is an explicit embedding for κ_{HI}, therefore also for K_Δ. What could it be?
- If $|X| \leq |Y|$ and $\pi : X \to Y$ is one-to-one, then $K_\Delta(X, Y)$ approximates the optimal pairwise matching:

$$\max_{\pi} \sum_{\mathbf{x} \in X} ||\mathbf{x} - \pi(\mathbf{x})||_1^{-1}$$

[4] Indyk and Thaper, Fast Image Retrieval via Embeddings, WSCTV. 2003

PMK as an Embedding[4]

- There is an explicit embedding for κ_{HI}, therefore also for K_Δ. What could it be?
- If $|X| \leq |Y|$ and $\pi : X \to Y$ is one-to-one, then $K_\Delta(X, Y)$ approximates the optimal pairwise matching:

$$\max_{\pi} \sum_{\mathbf{x} \in X} ||\mathbf{x} - \pi(\mathbf{x})||_1^{-1}$$

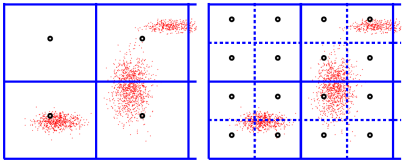
- This is similar to the **Earth mover's distance**:

$$\min_{\pi} \sum_{\mathbf{x} \in X} ||\mathbf{x} - \pi(\mathbf{x})||_1$$

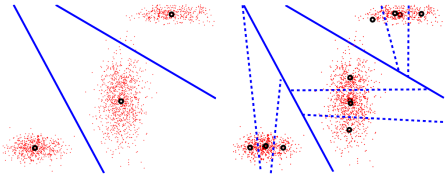
- But PMK is a *similarity* measure; it allows partial matching and does not penalize clutter, except for the normalization

[4] Indyk and Thaper, Fast Image Retrieval via Embeddings, WSCTV. 2003

PMK and Vocabulary Tree[5]



Uniform bins

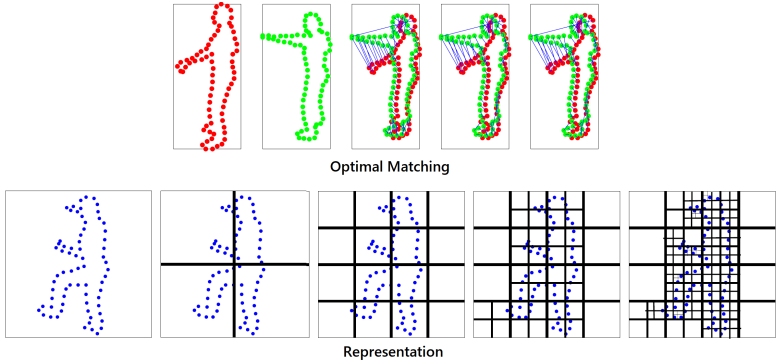


Vocabulary-guided bins

- Replace regular grid with hierarchical vocabulary cells
- Compared to vocabulary tree, there is a principle in assigning cell weights
- Still, its approximation quality can suffer at high dimensions

[5] Grauman and Darrell, Approximate Correspondences in High Dimensions, NeurIPS 2007

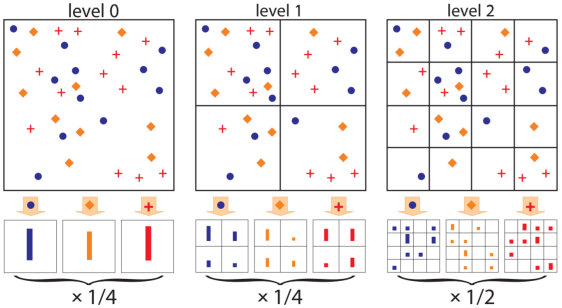
PMK and Spatial Matching[6]



- Same idea, applied to image 2-D coordinate space for spatial matching
- Matching cost is only based on point coordinates; No appearance

[6] Grauman and Darrell, Fast Contour Matching Using Approximate Earth Mover's Distance, CVPR 2004

Spatial Pyramid Matching (SPM)[7]

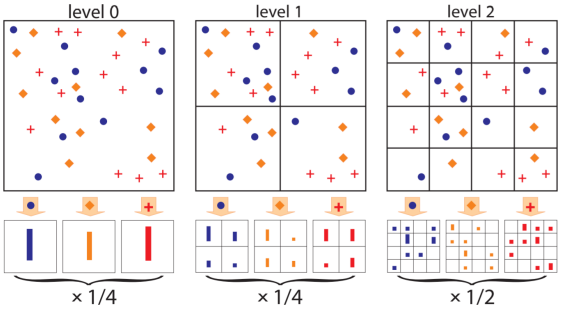


- If $X^{(j)}, Y^{(j)}$ are the feature coordinates of images X, Y with descriptors assigned to visual word j,

$$K_{SP}(X, Y) = \sum_{j=1}^{k} K_{\Delta}(X^{(j)}, Y^{(j)})$$

[7] Lazebnik et al, Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories, CVPR 2006

Spatial Pyramid Matching (SPM)[7]



- Coupled with BoW, it is a set of joint appearance-geometry histograms
- Robust to deformation but not invariant to transformations; Applied for global scene classification

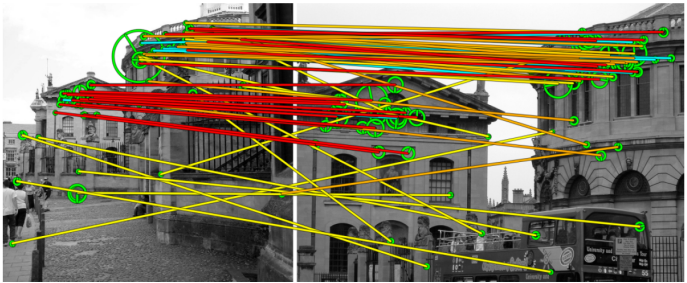
[7] Lazebnik et al, Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories, CVPR 2006.

Hough Pyramid Matching (HPM)[8]

Fast Pyramid Matching

[8] Tolias and Avrithis, Speeded-up, relaxed spatial matching, ICCV 2011

Hough Pyramid Matching (HPM)[8]



Hough Pyramid Matching

- Work with a single set of correspondences instead of two sets of features
- Determine a transformation hypothesis by a pair of features and then use histograms to collect votes in the transformation space

[8] Tolias and Avrithis, Speeded-up, relaxed spatial matching, ICCV 2011

Hough Pyramid Matching

- A **local feature** p in image P has position $\mathbf{t}(p)$, scale $s(p)$ and orientation $\theta(p)$ given by matrix $R(p) \in \mathbb{R}^{2\times 2}$:

$$F(p) = \begin{pmatrix} s(p)R(p) & \mathbf{t}(p) \\ \mathbf{0}^\intercal & 1 \end{pmatrix}$$

Hough Pyramid Matching

- A **local feature** p in image P has position $\mathbf{t}(p)$, scale $s(p)$ and orientation $\theta(p)$ given by matrix $R(p) \in \mathbb{R}^{2\times 2}$:

$$F(p) = \begin{pmatrix} s(p)R(p) & \mathbf{t}(p) \\ \mathbf{0}^\intercal & 1 \end{pmatrix}$$

- A **correspondence** $c = (p, q)$ is a pair of features $p \in P$, $q \in Q$ of two images P, Q and determines relative similarity transformation from p to q:

$$F(c) = F(q)F(p)^{-1} = \begin{pmatrix} s(c)R(c) & \mathbf{t}(c) \\ \mathbf{0}^\intercal & 1 \end{pmatrix}$$

with translation $\mathbf{t}(c) = \mathbf{t}(q) - s(c)R(c)\mathbf{t}(p)$, relative scale $s(c) = s(q)/s(p)$ and rotation $R(c) = R(q)R(p)^{-1}$ or $\theta(c) = \theta(q) - \theta(p)$

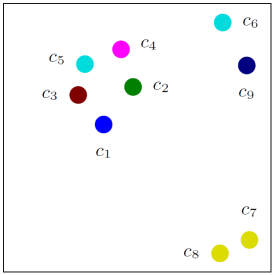
Hough Pyramid Matching

- The 4-DoF relative transformation represented by 4-D vector:

$$f(c) = (\mathbf{t}(c), s(c), \theta(c))$$

- To enforce one-to-one mapping, two correspondences $c = (p, q)$ and $c^{'} = (p^{'}, q^{'})$ are said to be **conflicting** if they refer to the same feature on either image, i.e., $p = p^{'}$ or $q = q^{'}$

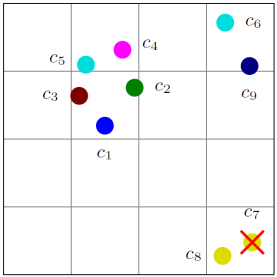
Hough Pyramid Matching



	p	q	similarity score
c_1			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_1)$
c_2			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_2)$
c_3			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_3)$
c_4			$(1+\frac{1}{2}3+\frac{1}{4}2)w(c_4)$
c_5			$(1+\frac{1}{2}3+\frac{1}{4}2)w(c_5)$
c_6			0
c_7			0
c_8			$(\frac{1}{4}6)w(c_8)$
c_9			$(\frac{1}{4}6)w(c_9)$

- Correspondence c weighted by $w(c)$, based e.g. on visual word

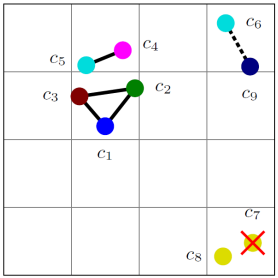
Hough Pyramid Matching



	p	q	similarity score
c_1			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_1)$
c_2			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_2)$
c_3			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_3)$
c_4			$(1+\frac{1}{2}3+\frac{1}{4}2)w(c_4)$
c_5			$(1+\frac{1}{2}3+\frac{1}{4}2)w(c_5)$
c_6			0
c_7			0
c_8			$(\frac{1}{4}6)w(c_8)$
c_9			$(\frac{1}{4}6)w(c_9)$

- Correspondence c weighted by $w(c)$, based e.g. on visual word
- Conflicting correspondences in same bin are **erased**

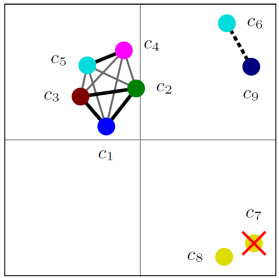
Hough Pyramid Matching



	p	q	similarity score
c_1			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_1)$
c_2			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_2)$
c_3			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_3)$
c_4			$(1+\frac{1}{2}3+\frac{1}{4}2)w(c_4)$
c_5			$(1+\frac{1}{2}3+\frac{1}{4}2)w(c_5)$
c_6			0
c_7			0
c_8			$(\frac{1}{4}6)w(c_8)$
c_9			$(\frac{1}{4}6)w(c_9)$

- Correspondence c weighted by $w(c)$, based e.g. on visual word
- Conflicting correspondences in same bin are **erased**
- In bin b with n_b correspondences, each correspondence groups with $[n_b - 1]_+$ others
- Level 0 Weight 1

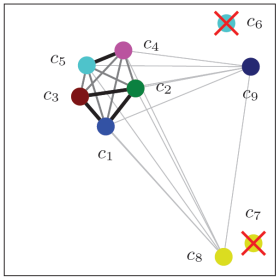
Hough Pyramid Matching



	p	q	similarity score
c_1			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_1)$
c_2			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_2)$
c_3			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_3)$
c_4			$(1+\frac{1}{2}3+\frac{1}{4}2)w(c_4)$
c_5			$(1+\frac{1}{2}3+\frac{1}{4}2)w(c_5)$
c_6			0
c_7			0
c_8			$(\frac{1}{4}6)w(c_8)$
c_9			$(\frac{1}{4}6)w(c_9)$

- Correspondence c weighted by $w(c)$, based e.g. on visual word
- Conflicting correspondences in same bin are **erased**
- In bin b with n_b correspondences, each correspondence groups with $[n_b - 1]_+$ others
- Level 1 Weight $\frac{1}{2}$

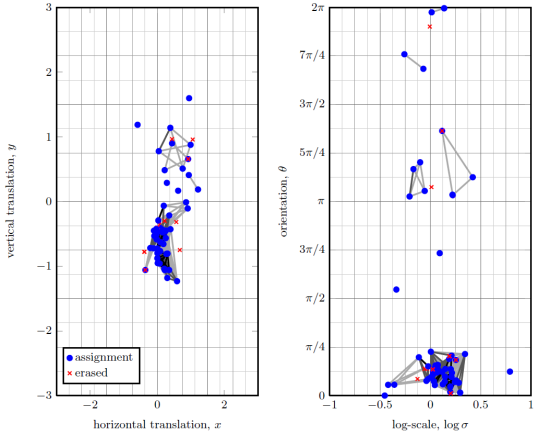
Hough Pyramid Matching



	p	q	similarity score
c_1			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_1)$
c_2			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_2)$
c_3			$(2+\frac{1}{2}2+\frac{1}{4}2)w(c_3)$
c_4			$(1+\frac{1}{2}3+\frac{1}{4}2)w(c_4)$
c_5			$(1+\frac{1}{2}3+\frac{1}{4}2)w(c_5)$
c_6			0
c_7			0
c_8			$(\frac{1}{4}6)w(c_8)$
c_9			$(\frac{1}{4}6)w(c_9)$

- Correspondence c weighted by $w(c)$, based e.g. on visual word
- Conflicting correspondences in same bin are **erased**
- In bin b with n_b correspondences, each correspondence groups with $[n_b - 1]_+$ others
- Level 2 Weight $\frac{1}{4}$

Hough Pyramid Matching



- *Mode Seeking:* We are looking for regions where density is maximized in transformation space

Hough Pyramid Matching

- Linear in number of correspondences; no need to count inliers
- Robust to deformations and multiple matching surfaces, invariant to transformations
- Only applies to same instance matching

Homework

Readings

- Chapter 16.1.4, Forsyth and Ponce, *Computer Vision: A Modern Approach (2nd ed.)*

Other Resources

- Pyramid Match Kernel project page

References


Michael J. Swain and Dana H. Ballard. "Color Indexing". In: *Int. J. Comput. Vision* 7.1 (Nov. 1991), 11–32.


Piotr Indyk and Nitin Thaper. "Fast Image Retrieval via Embeddings". In: *WSCTV*. 2003.


Kristen Grauman and Trevor Darrell. "Fast Contour Matching Using Approximate Earth Mover's Distance". In: *IEEE CVPR*. Oct. 2004, pp. I–220.


K. Grauman and T. Darrell. "The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features". In: *IEEE ICCV*. Nov. 2005, Vol 2, pp. 1458–1465.


Svetlana Lazebnik, Cordelia Schmid, and J. Ponce. "Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories". In: *IEEE CVPR*. Vol. 2. Feb. 2006, pp. 2169 –2178.


Kristen Grauman and Trevor Darrell. "Approximate Correspondences in High Dimensions". In: *Neural Information Processing Systems*. 2007.


Giorgos Tolias and Yannis Avrithis. "Speeded-up, relaxed spatial matching". In: *IEEE ICCV*. 2011, pp. 1653–1660.


David Forsyth and Jean Ponce. *Computer Vision: A Modern Approach*. 2 edition. Boston: Pearson Education India, 2015.

Deep Learning for Computer Vision

Transitioning From Traditional Vision to Deep Learning

Vineeth N Balasubramanian

Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad

Summarizing Topics So Far

Fundamental Operations

- **Convolution** is a unique operation
  - linear, shift-invariant
  - *Useful properties:* Commutative, Associative, Distributive (over addition)
- Forms basis of image operations and even modern-day neural networks working on images

Summarizing Topics So Far

Fundamental Operations

- **Convolution** is a unique operation
  - linear, shift-invariant
  - *Useful properties:* Commutative, Associative, Distributive (over addition)
- Forms basis of image operations and even modern-day neural networks working on images

Common Pipeline in Traditional Vision Tasks

- Extract corners or patches in images $\rightarrow$ Extract descriptors
- Use banks of filters, such as Steerable filters or Gabor filters
- Use descriptors for tasks such as retrieval, matching or classification

Traditional Vision: High-level Abstractions

Image-Level Understanding

- Going from low-level image understanding to aggregation of descriptors
- Banks of filters capture responses at different scales and orientations
- Histograms can be viewed as "*encoding*" and "*pooling*"
- Similarities to the human visual system

Traditional Vision: High-level Abstractions

Image-Level Understanding

- Going from low-level image understanding to aggregation of descriptors
- Banks of filters capture responses at different scales and orientations
- Histograms can be viewed as "*encoding*" and "*pooling*"
- Similarities to the human visual system

Local Features/Understanding

- Not all spatial regions important, depends on task (stereopsis, motion estimation, instance recognition compared to class recognition)
- Encoding makes features sparse
  - Many words in BoW have zero count
- Operators that detect local features can be viewed as "*convolution*" followed by some kind of "*competition*"

Traditional Vision: High-level Abstractions

Representing Images/Regions as Descriptors

- Learn descriptors/representations such that dot product is good enough for matching
- Some invariance to geometric transformations, designed or learned in certain cases

Traditional Vision: High-level Abstractions

Representing Images/Regions as Descriptors

- Learn descriptors/representations such that dot product is good enough for matching
- Some invariance to geometric transformations, designed or learned in certain cases

Moving on to Deep Learning...

Although not by design, Deep Learning seems to build on some of the above principles, but in a learnable manner...we will see soon