Robotic 3D Vision

Lecture 15: 3D Object Detection 1 – Introduction, Pose Alignment and Grouping

WS 2020/21

Dr. Niclas Zeller

Artisense GmbH

What We Will Cover Today

- Introduction to 3D object detection
- Challenges in object detection
- Object detection and pose estimation with local image features
  - Affine transformation
  - Homography
- Correspondence grouping and robust alignment
  - Generalized Hough transform

Object Detection

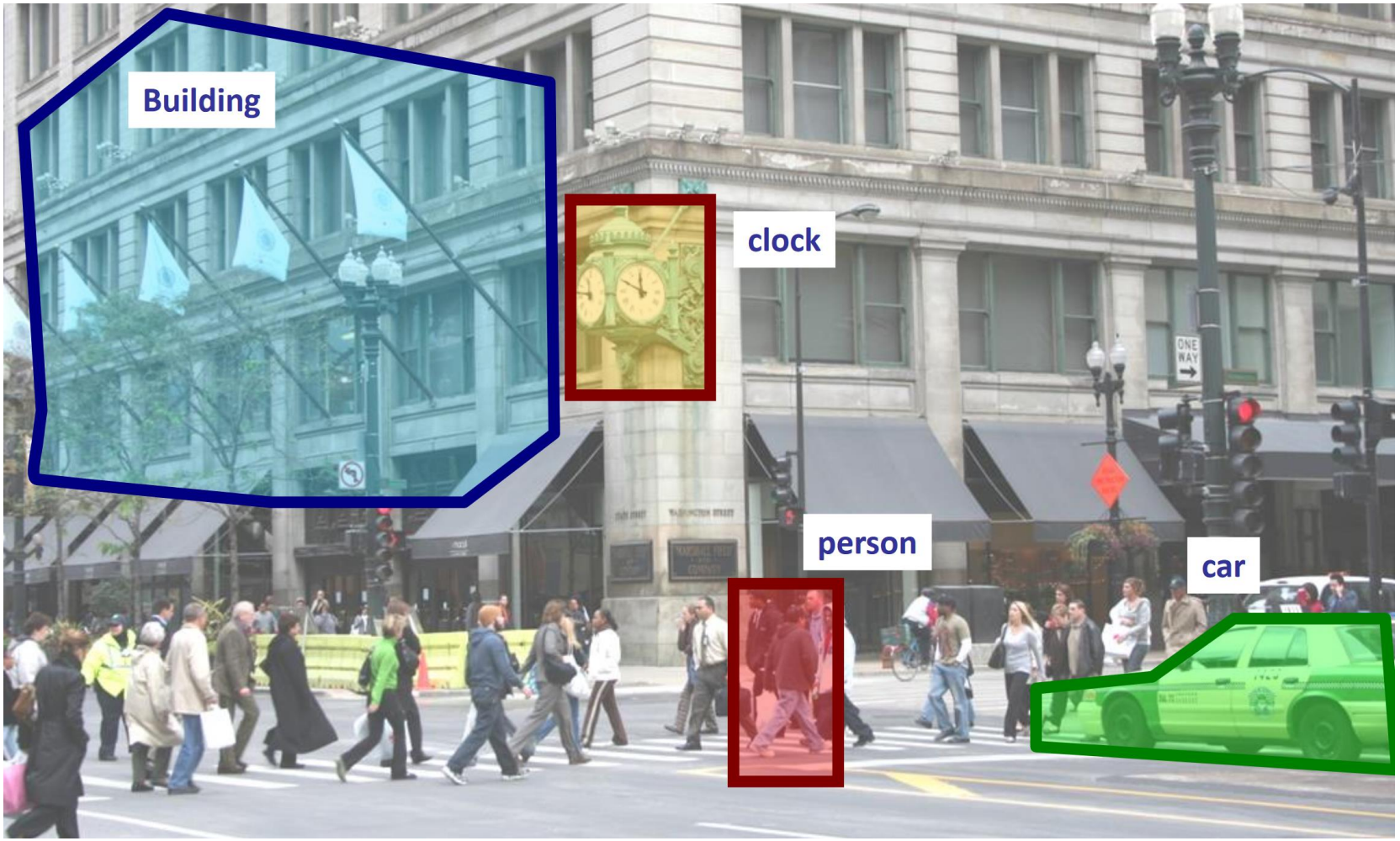
- Today research on object detection gets more and more dominated by deep learning (DL) based approaches
  - R-CNN (and its extensions), YOLO, etc.
- Though, especially in robotics applications runtime and computational complexity matters
- Hence, we will manly focused on classical object detection and alignment strategies
  - Modern approaches often also combine DL and classical alignment approaches

Object Detection

Detection: Does this image contain a car? Where is it?

Slide adapted from S. Savarese

Object Detection



Detection: Which objects does this image contain? Where are they?

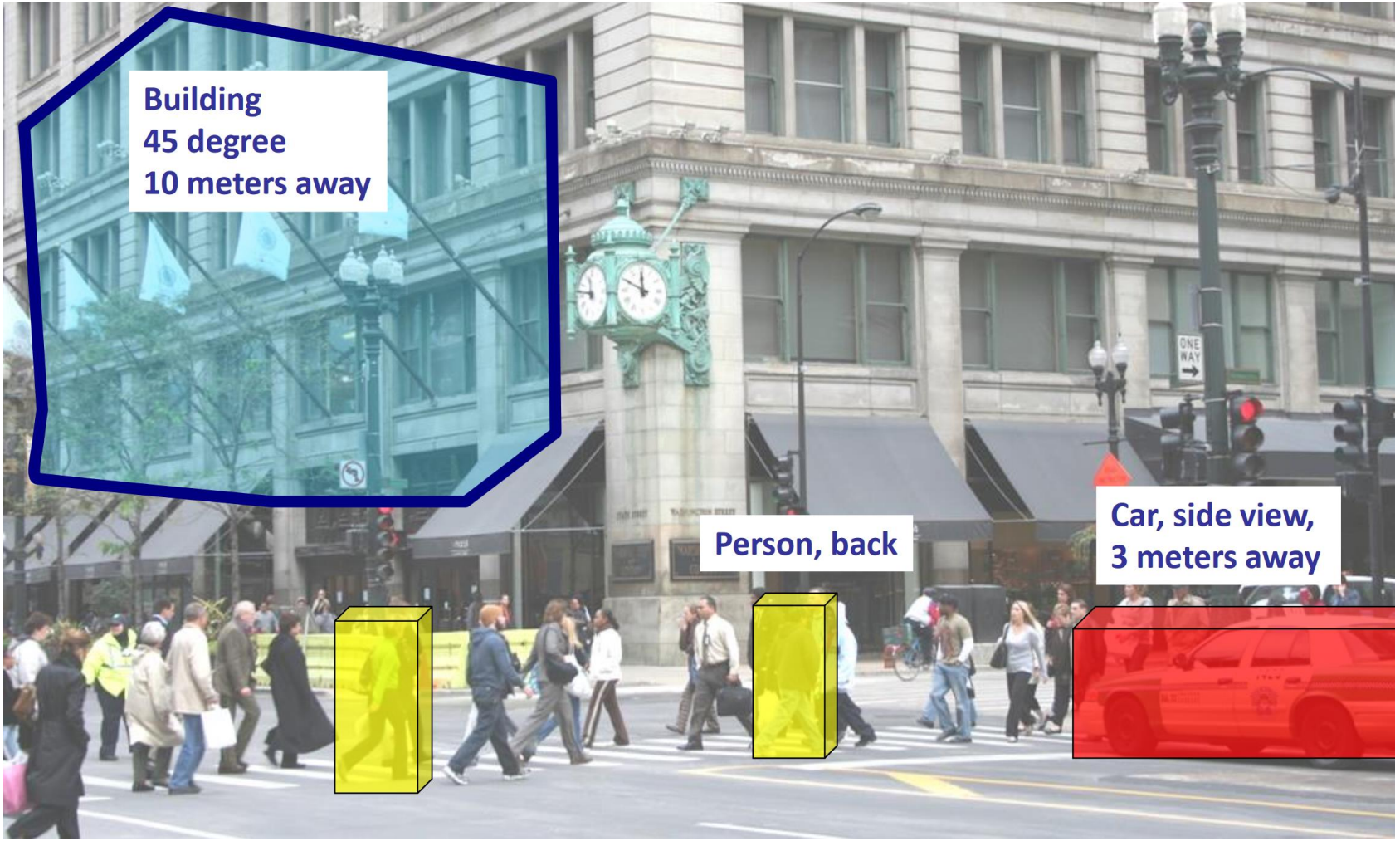
Slide adapted from S. Savarese

Object Detection

Detection: Accurate localization (instance segmentation)

Slide adapted from S. Savarese

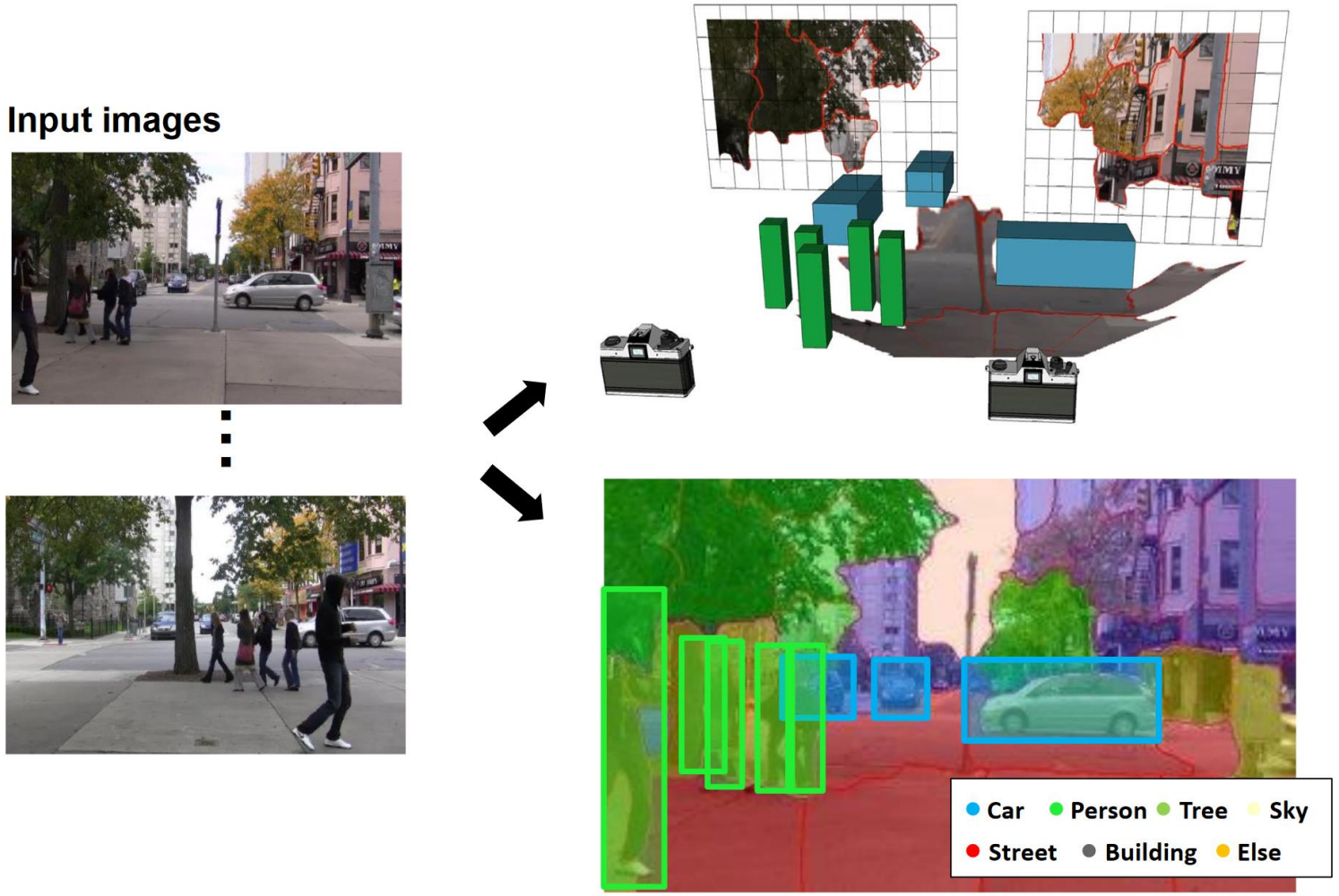
Object Detection



Detection: Where are the objects in 3D? (Position and Orientation)

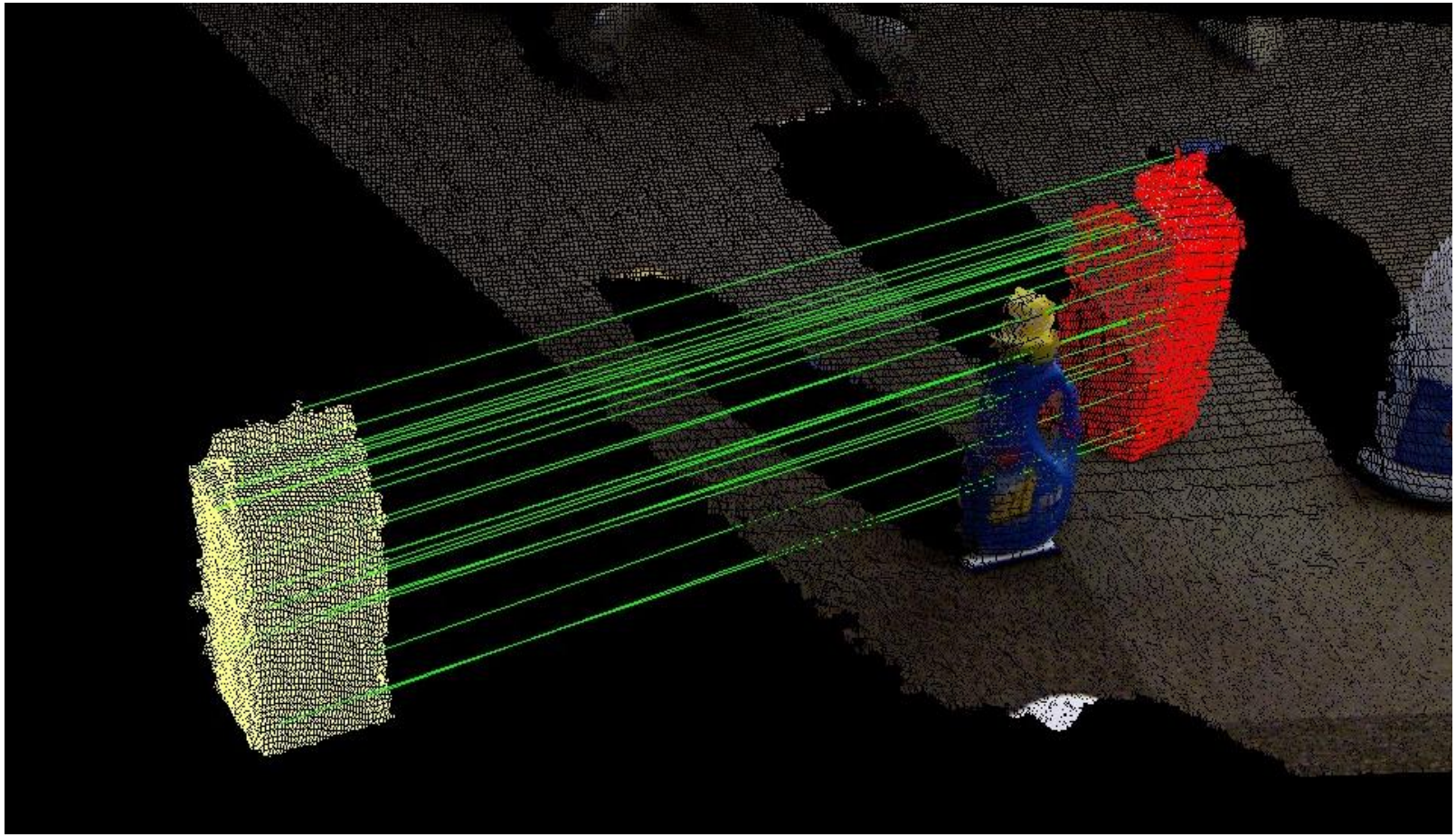
Slide adapted from S. Savarese

Joint Detection and Reconstruction



Slide adapted from S. Savarese

3D Object Detection



Detection: Where are the objects in 3D?

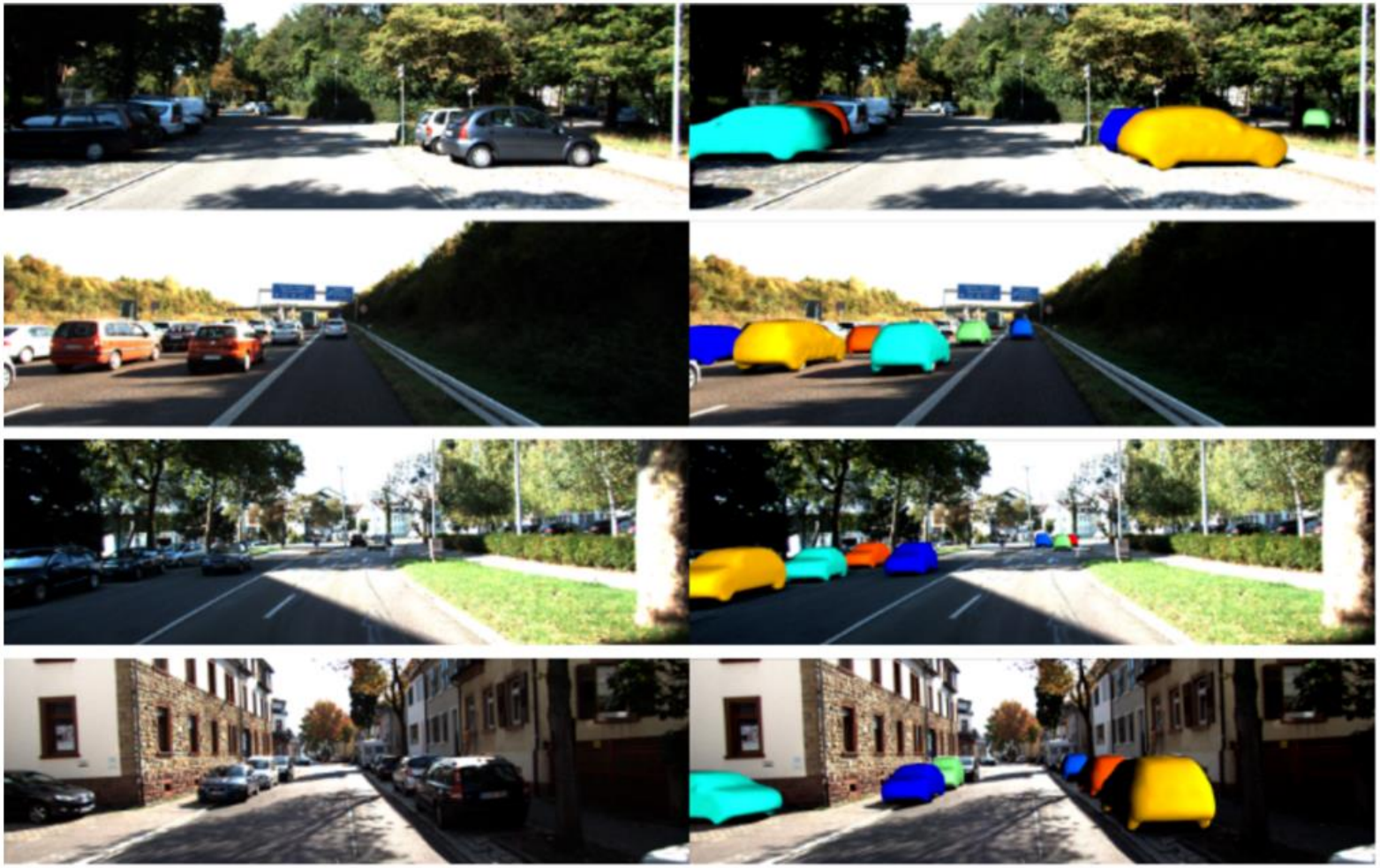
Image from pointclouds.org

3D Object Detection for Robotic Grasping

Papazov et al., IJRR 2012

https://www.youtube.com/watch?v=qlt1os_WJRs

3D Object Detection for Autonomous Driving



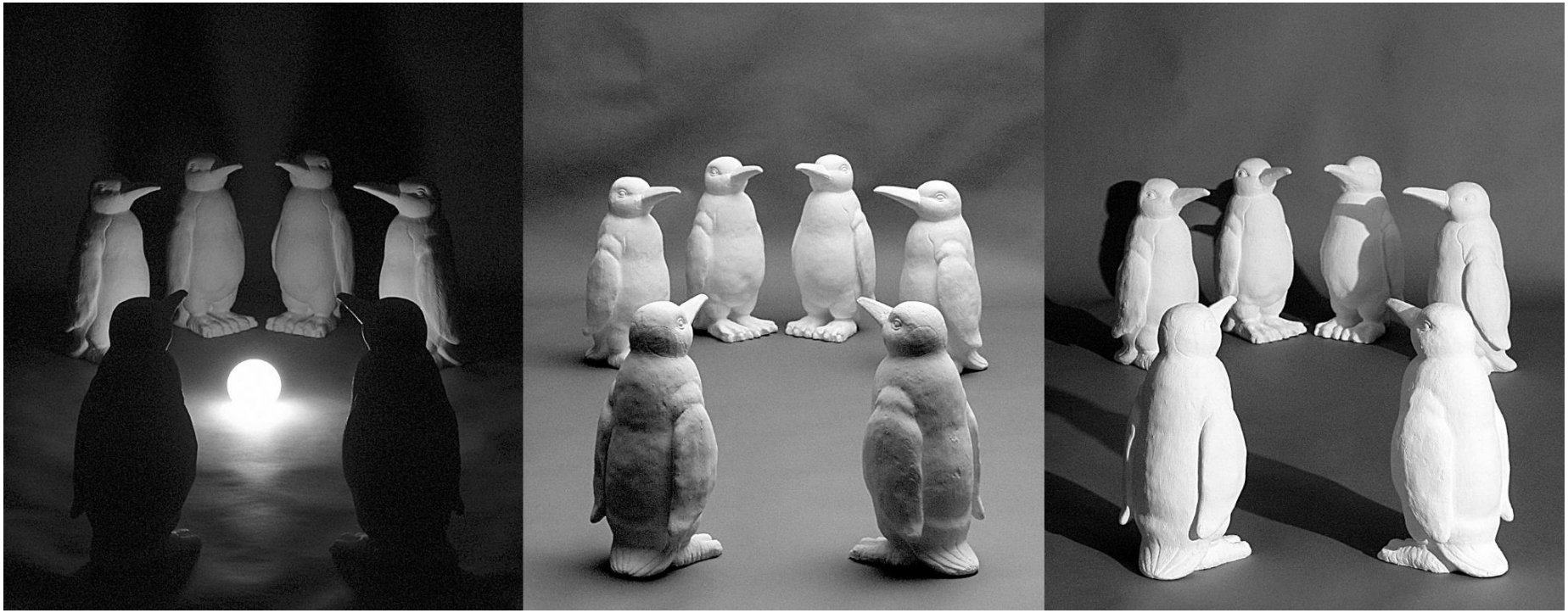
Wang et al. DirectShape, ICRA 2020

Challenges in Object Detection

View-point variation

Slide adapted from F. Li, A. Torralba

Challenges in Object Detection



Illumination variation

Image credit: J. Koenderink

Challenges in Object Detection

Scale

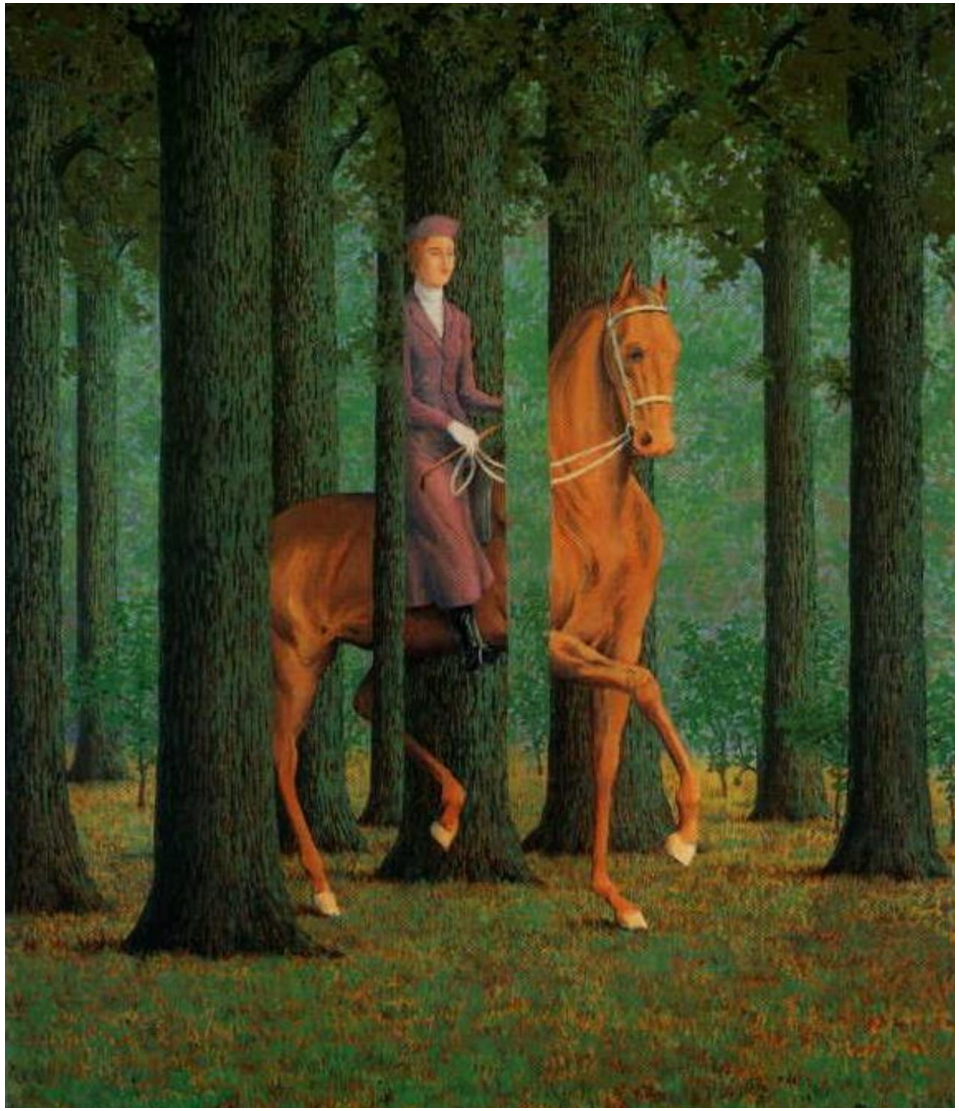
Slide adapted from F. Li, A. Torralba

Challenges in Object Detection

Deformation

Slide adapted from S. Savarese

Challenges in Object Detection



Occlusions

Slide adapted from F. Li and A. Torralba

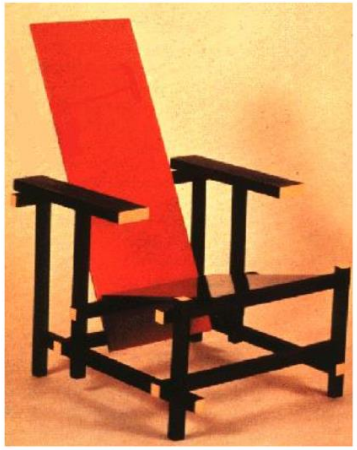
Challenges in Object Detection

Background clutter

Image: Kilmeny Niland, 1995
Slide adapted from S. Savarese

Challenges in Object Detection



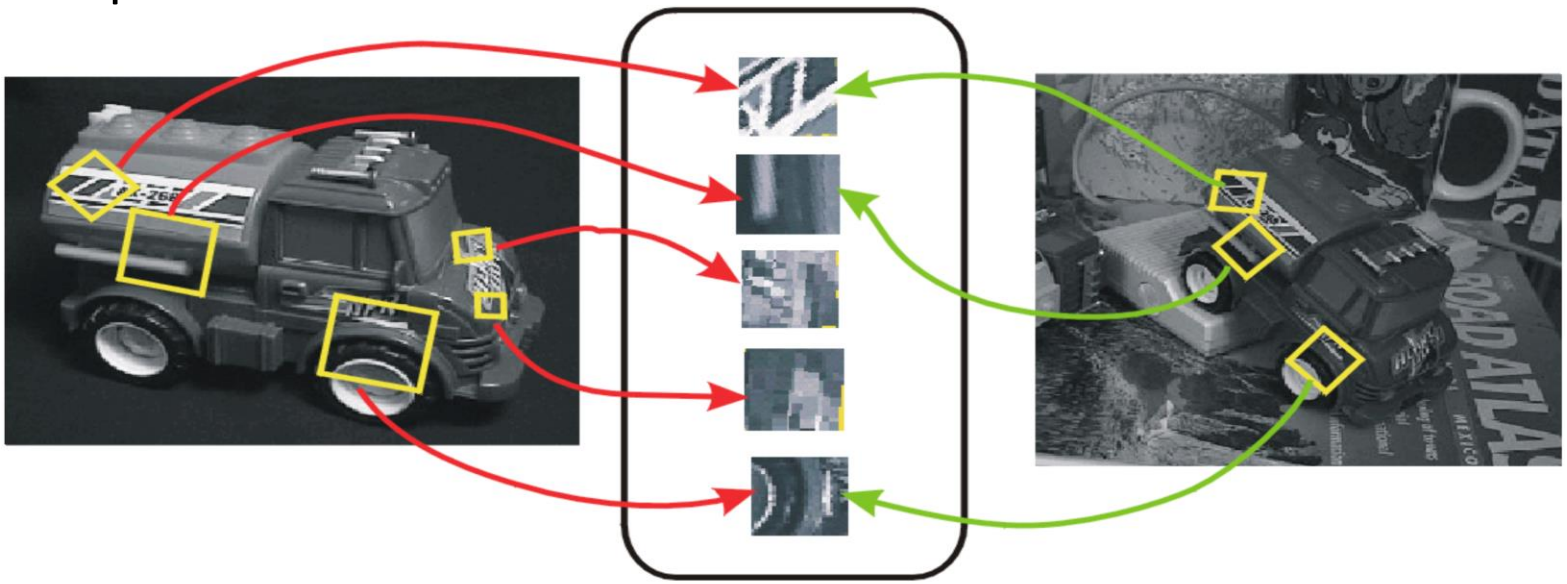



Intra-class variation vs. specific object detection

Slide adapted from F. Li and A. Torralba

Object Detection with Local Features

- Can we make use of local features to detect a certain object in the scene?
  - Detect and match a set of local keypoints between model and scene (image)
  - Object detection is supposed to be invariant to different view points



Keypoints e.g. SIFT

Image from D. Lowe

Object Detection with Local Features

- Which transformations can we estimate, if we have only given 2D views on an object with 2D image locations of keypoints?
  - Affine transformations
  - Projective transformations (homography)

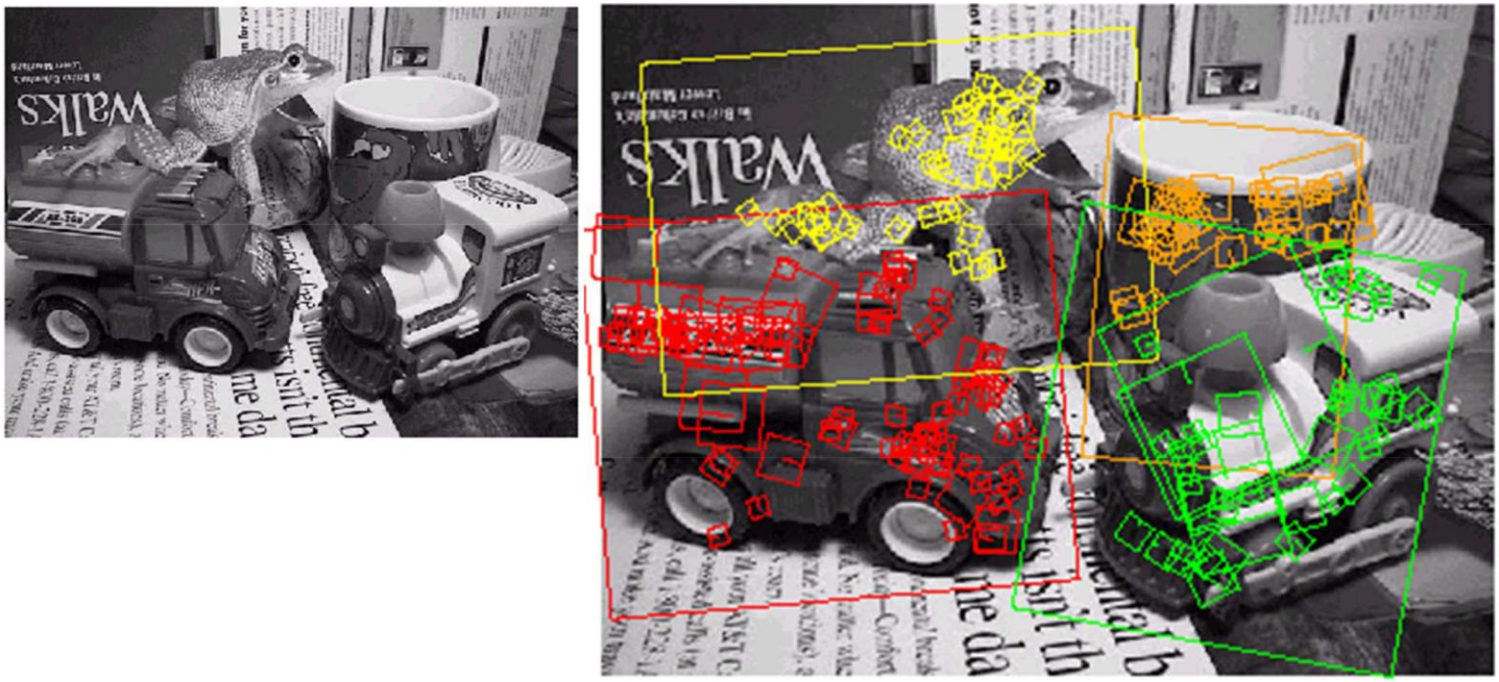
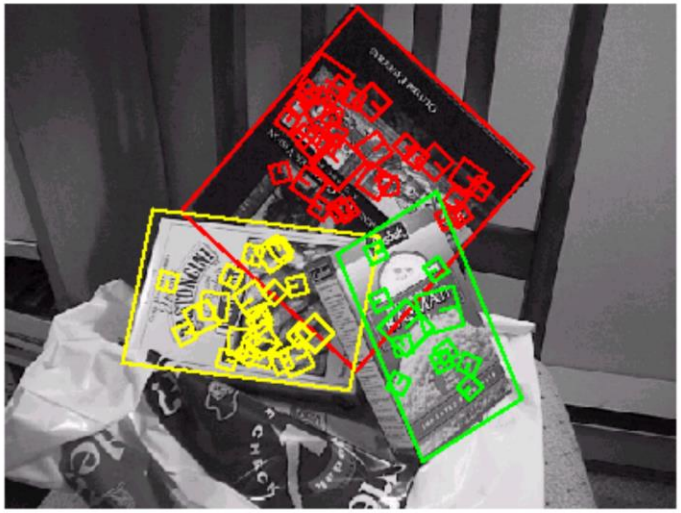


Image from D. Lowe

2D Affine Transformations

- 2D affine transformations approximate perspective projection of planar objects



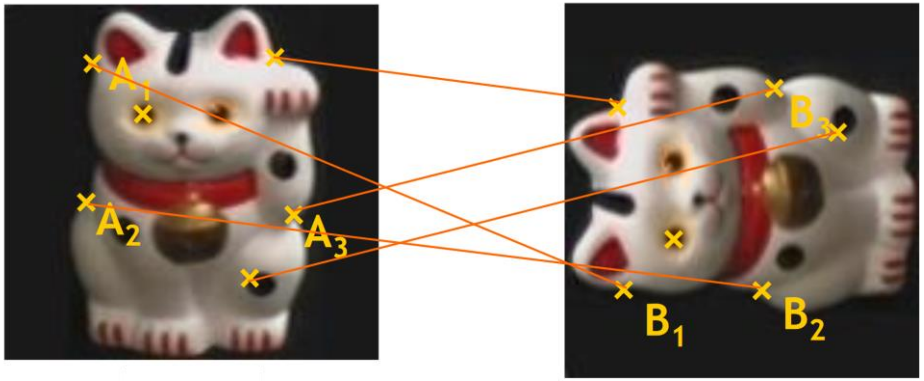
- Can work well for (almost) planar objects and (almost) orthographic camera

Image from D. Lowe

2D Affine Transformations

- 2D affine transformations approximate perspective projection of planar objects

$$\overline{\mathbf{y}}' = \begin{pmatrix} m_{11} & m_{12} & t_1 \\ m_{21} & m_{22} & t_2 \\ 0 & 0 & 1 \end{pmatrix} \overline{\mathbf{y}}$$



$\mathbf{y}$ $\mathbf{y}'$

- Parallel lines remain parallel

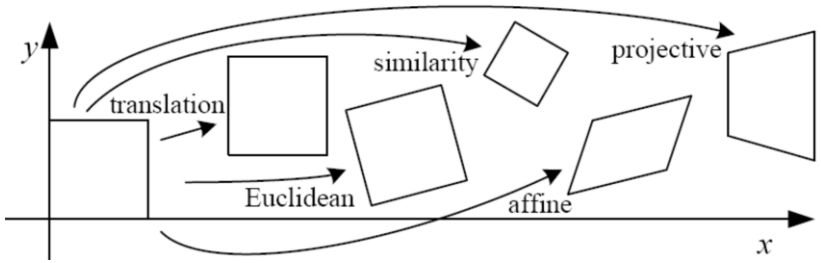


Image from A. Efros

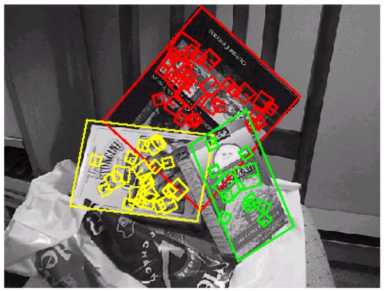


Image from D. Lowe

2D Affine Transformations

- Which basic transformations can we represent with affine transformations?

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

2D rotation

$$\overline{\mathbf{y}}' = \begin{pmatrix} 1 & 0 & t_1 \\ 0 & 1 & t_2 \\ 0 & 0 & 1 \end{pmatrix} \overline{\mathbf{y}}$$

2D translation

$$\overline{\mathbf{y}}' = \begin{pmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \overline{\mathbf{y}}$$

2D scaling

$$\overline{\mathbf{y}}' = \begin{pmatrix} 1 & sh_1 & 0 \\ sh_2 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \overline{\mathbf{y}}$$

2D shearing

Estimating 2D Affine Transformations

- Write constraints on affine transformation from multiple 2D point correspondences as

$$\begin{pmatrix} \vdots \\ x_i' \\ y_i' \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} m_{11} \\ m_{12} \\ m_{21} \\ m_{22} \\ t_1 \\ t_2 \end{pmatrix}$$

$$\mathbf{b} \qquad \mathbf{A} \qquad \boldsymbol{\theta}$$

- Linear least squares estimation

$$\boldsymbol{\theta} = \left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{b}$$

Projective Transformation/Homography

- Affine transformation does not consider the perspective distortion of a (pinhole) camera

$$\tilde{\mathbf{y}}' = \tilde{\omega}'\bar{\mathbf{y}}' = \tilde{\omega}'(z_c')^{-1}\mathbf{x}' = \omega\mathbf{x}'$$
$$\mathbf{x}' = z_c\boldsymbol{R}\bar{\mathbf{y}} + \boldsymbol{t}$$
$$= z_c \begin{pmatrix} r_{11} & r_{12} & r_{13} + t_x \\ r_{21} & r_{22} & r_{23} + t_y \\ r_{31} & r_{32} & r_{33} + t_z \end{pmatrix} \bar{\mathbf{y}}$$

$$\tilde{\mathbf{y}}' = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \bar{\mathbf{y}}$$

  - due to scale ambiguity we can set $h_{33} = 1$
  - Homography holds also for pixel coordinates $\bar{\mathbf{y}}^p$
  - images of **points on a 3D plane** taken from different views are related by a homography

Projective Transformation/Homography

- Under a pinhole projection model, images of points on a 3D plane taken from different views are related by a homography

$$\widetilde{\mathbf{y}}' = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \overline{\mathbf{y}}$$

$$\mathbf{H}$$

due to scale ambiguity we can set $h_{33} = 1$

- Parallel lines in 3D do not remain parallel in the image
- Straight lines are preserved
- Rectangle maps to quadrilateral

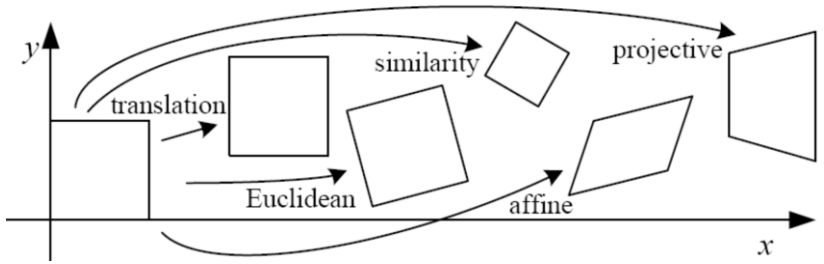
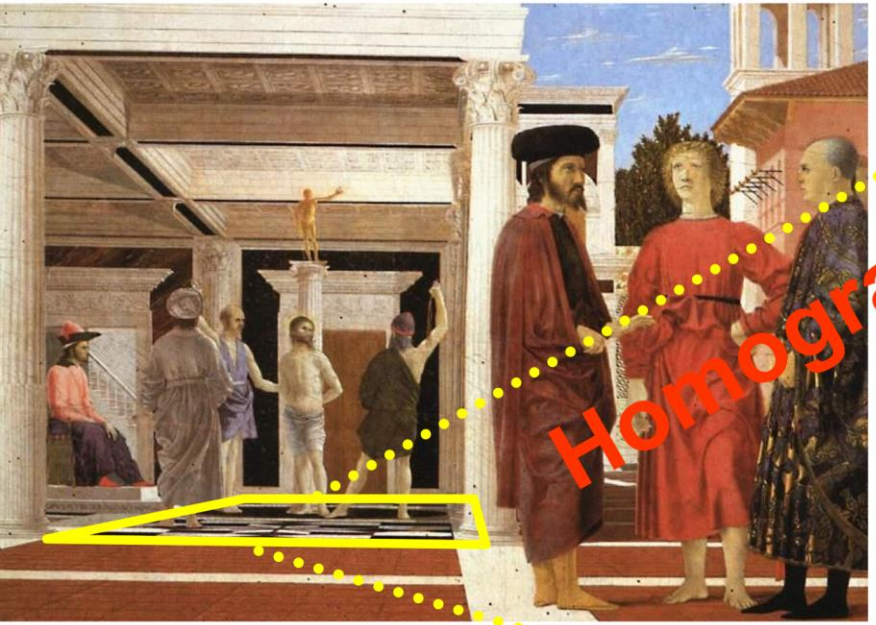
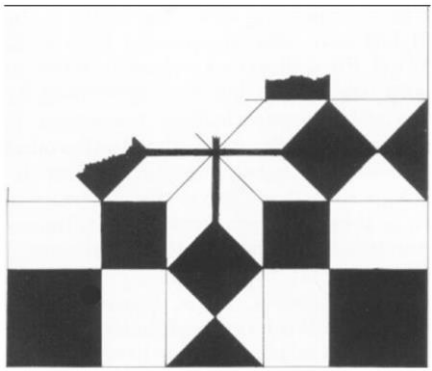


Image from A. Efros

Homography Example

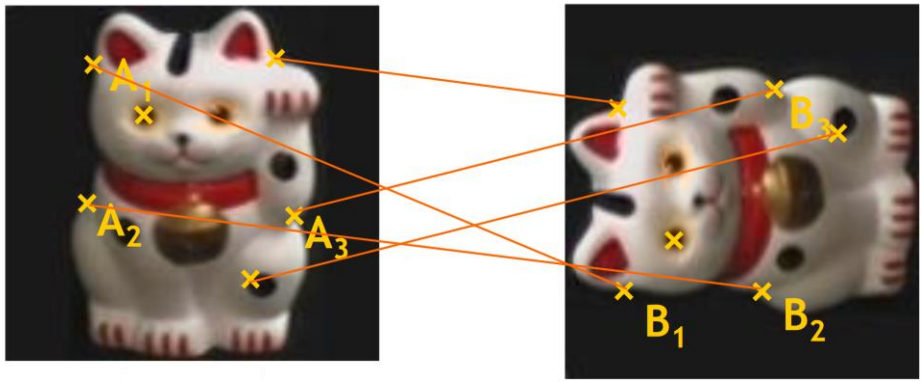




Manual reconstruction by Martin Kemp, The Science of Art

Image from A. Criminisi

Estimating Homographies



- Each 2D point correspondence provides the constraints

$$\widetilde{\mathbf{y}}' = \mathbf{H}\overline{\mathbf{y}} \qquad \overline{\mathbf{y}}' = \frac{1}{\widetilde{w}'}\widetilde{\mathbf{y}}'$$

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + 1}$$

$$y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + 1}$$

Estimating Homographies

- Constraints can be written as

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + 1}$$

$$x'h_{31}x + x'h_{32}y + x' = h_{11}x + h_{12}y + h_{13}$$

$$x' = -x'h_{31}x - x'h_{32}y + h_{11}x + h_{12}y + h_{13}$$

$$y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + 1}$$

$$y'h_{31}x + y'h_{32}y + y' = h_{21}x + h_{22}y + h_{23}$$

$$y' = -y'h_{31}x - y'h_{32}y + h_{21}x + h_{22}y + h_{23}$$

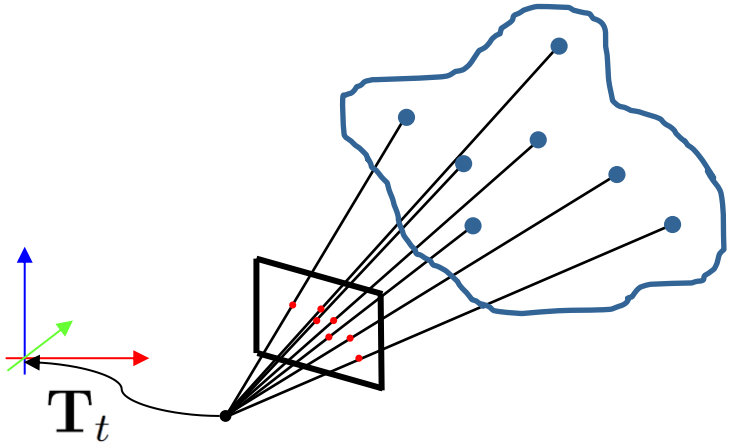
Estimating Homography

- Leads to a set of linear equations

$$\begin{pmatrix} \vdots \\ x_i' \\ y_i' \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_i & y_i & 1 & 0 & 0 & 0 & -x_i' x_i & -x_i' y_i \\ 0 & 0 & 0 & x_i & y_i & 1 & -y_i' x_i & -y_i' y_i \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{pmatrix}$$

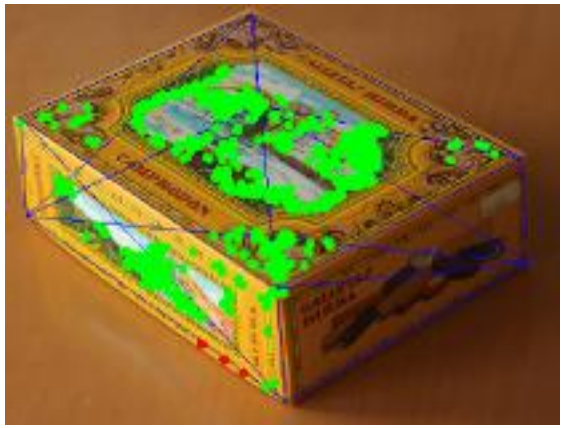
Monocular 3D Object Pose Estimation

- If we have a 3D model of keypoints on the object available, we can use PnP algorithms (see Lec. 6) to determine 3D rotation and translation of the object from 2D-to-3D keypoint matches



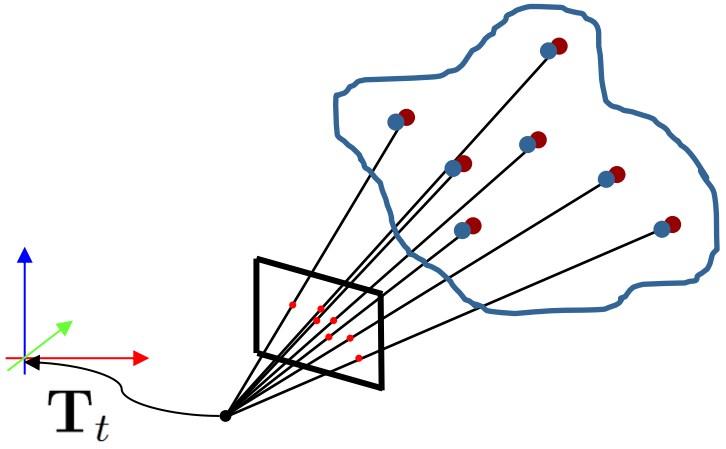
- How do we get the 3D model?

- Example: Render textured CAD model from different viewpoints to extract descriptors
and generate keypoint database with 3D coordinates in object coordinate frame

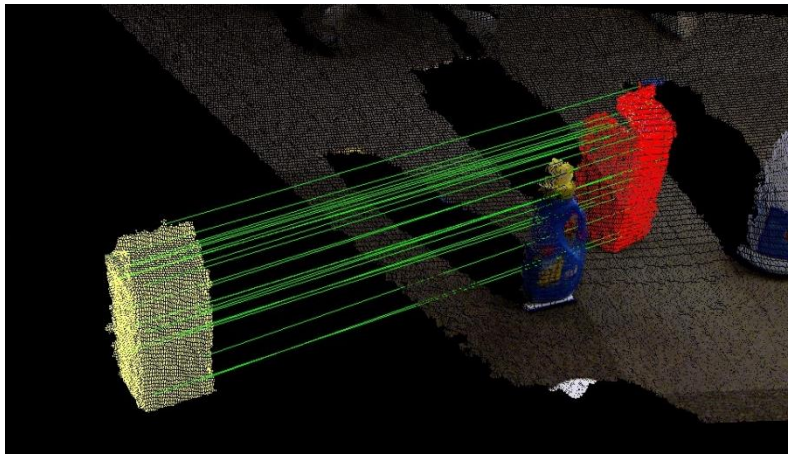


3D Object Pose Estimation in RGB-D Images

- With RGB-D images, we can also perform 3D-to-3D alignment of matched keypoints between model and image



- Alternatively to 2D image points in RGB images, 3D shape keypoints and global shape descriptors of object segments have been proposed that can be extracted from the depth images



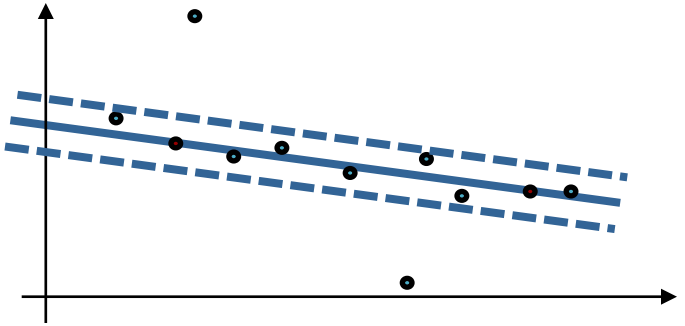
Correspondence Grouping and Robust Alignment

- Methods described so far assume perfect correspondences between model and object
- If keypoint matches are erroneous, direct least squares fitting will fail
- If multiple objects are present in a scene, we need a process to group correspondences of each single object before alignment
- However, methods still work for simple tasks like single object picking in controlled environment

Correspondence Grouping and Robust Alignment

- How can we group correspondences of multiple objects?
  - Approach 1: RANSAC (see Lec. 7)
  - Approach 2: Hough Transform (this lecture)

- RANSAC correspondence grouping
  - Sample minimal set of matches to perform alignment and determine LS fit to best inlier set
  - Remove inliers and fit next object
  - Requires high number of iteration, since the outlier ratio (per object) is quite high

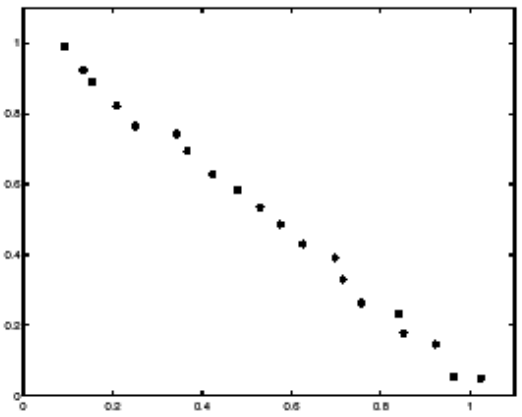


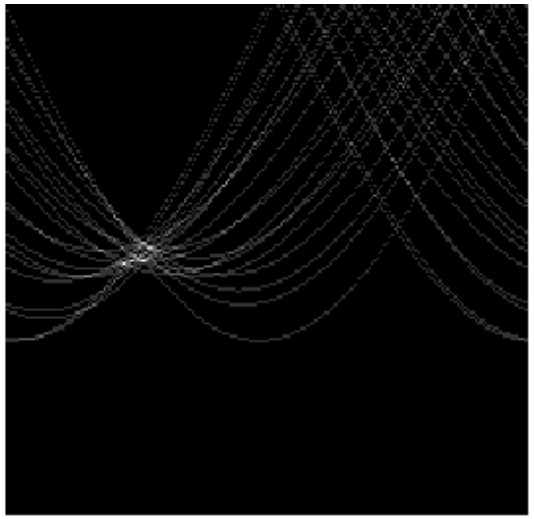
Target object

Image from Rabin et al. 2010

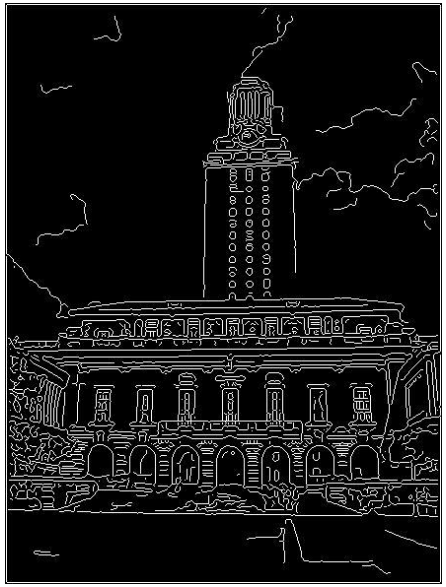
Correspondence Grouping and Robust Alignment

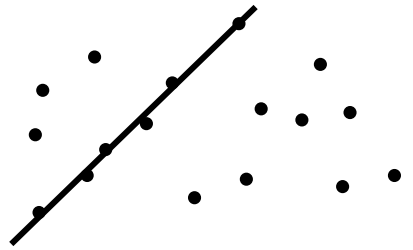
- Hough Transform
  - Each minimal set of matches needed for alignment votes in pose parameter space (using a discretization/histogram)
  - Object poses correspond to maxima in pose parameter histogram with sufficient number of votes





Example: Line Fitting





- Extra edge points (clutter), multiple models:
  - Which points go with which line, if any?
- Only some parts of each line detected, and some parts are missing:
  - How to find a line that bridges missing evidence?
- Noise in measured edge points, orientations:
  - How to detect true underlying parameters?

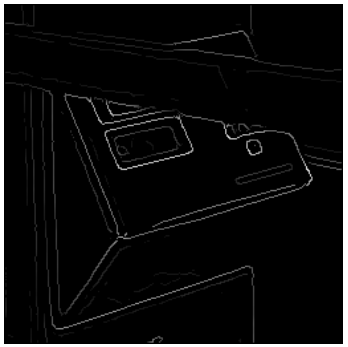
Slide adapted from K. Grauman

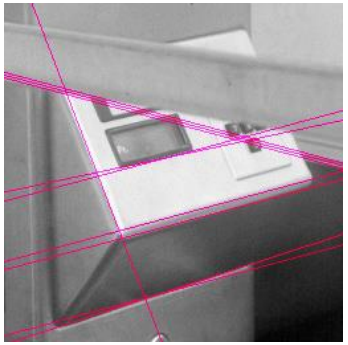
Fitting Lines with the Hough Transform

- Given all points that belong to a line, what is the line?
- How many lines are there?
- Which points belong to which lines?

- **Hough Transform** is a voting technique that can be used to answer all of these questions.

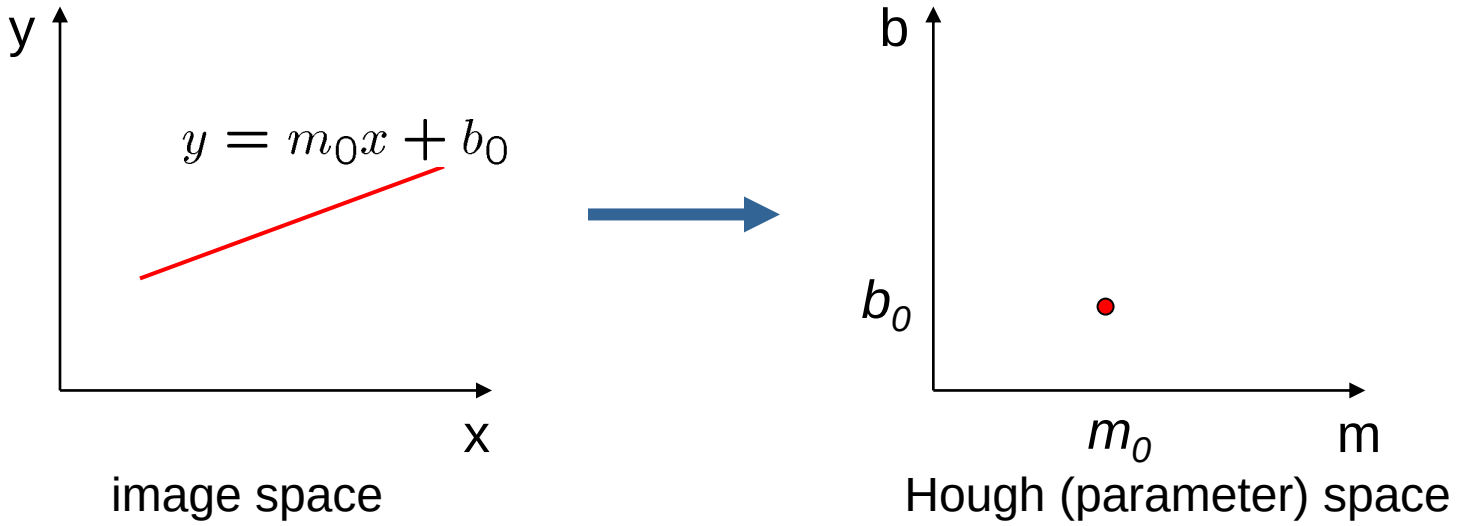
- Main idea:
  - 1. Record vote for each possible line on which an each edge point lies
  - 2. Look for lines that get many votes





Slide adapted from K. Grauman

Fitting Lines with the Hough Transform

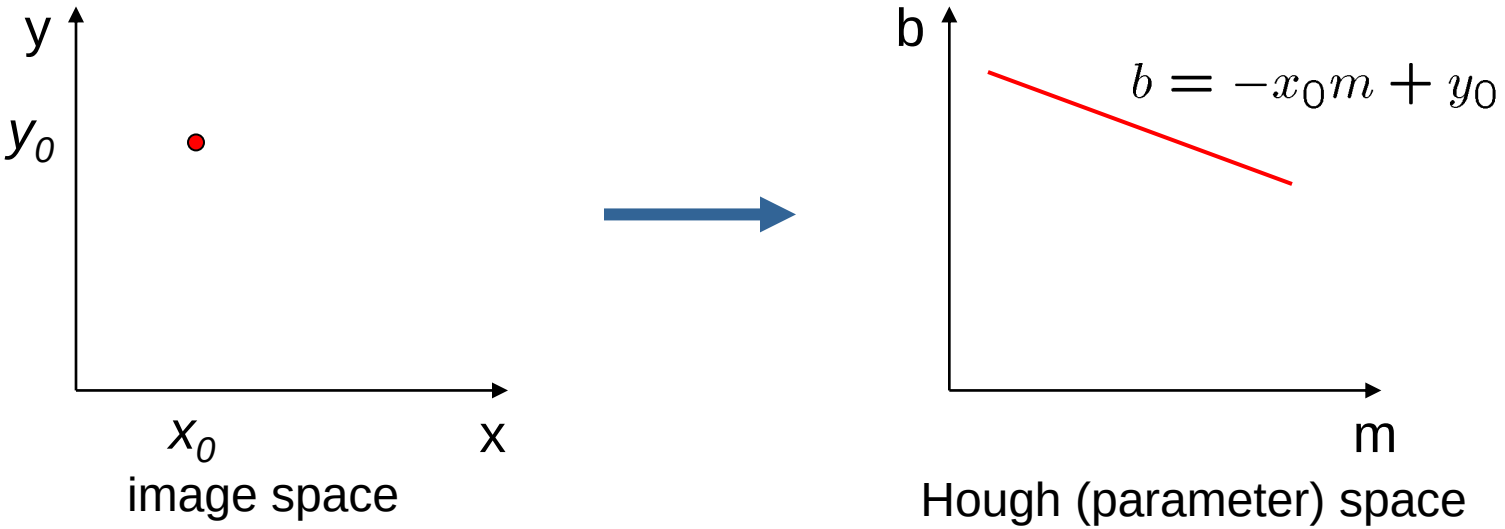


Connection between image (x,y) and Hough (m,b) spaces

- A line in the image corresponds to a point in Hough space
- To go from image space to Hough space:
  - given a set of points (x,y), find all (m,b) such that y = mx + b

Slide adapted from S. Seitz

Fitting Lines with the Hough Transform

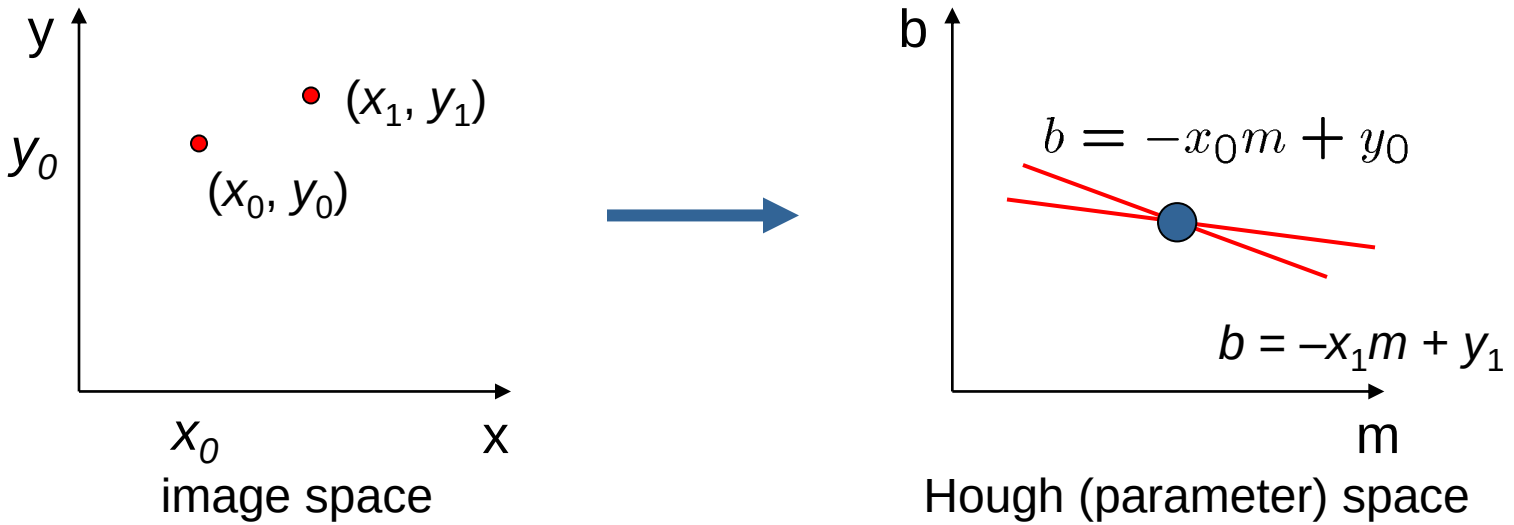


Connection between image (x,y) and Hough (m,b) spaces

- A line in the image corresponds to a point in Hough space
- To go from image space to Hough space:
  - given a set of points (x,y), find all (m,b) such that y = mx + b
- What does a point (x_0, y_0) in the image space map to?
  - Answer: the solutions of $b = -x_0 m + y_0$
  - this is a line in Hough space

Slide adapted from S. Seitz

Fitting Lines with the Hough Transform

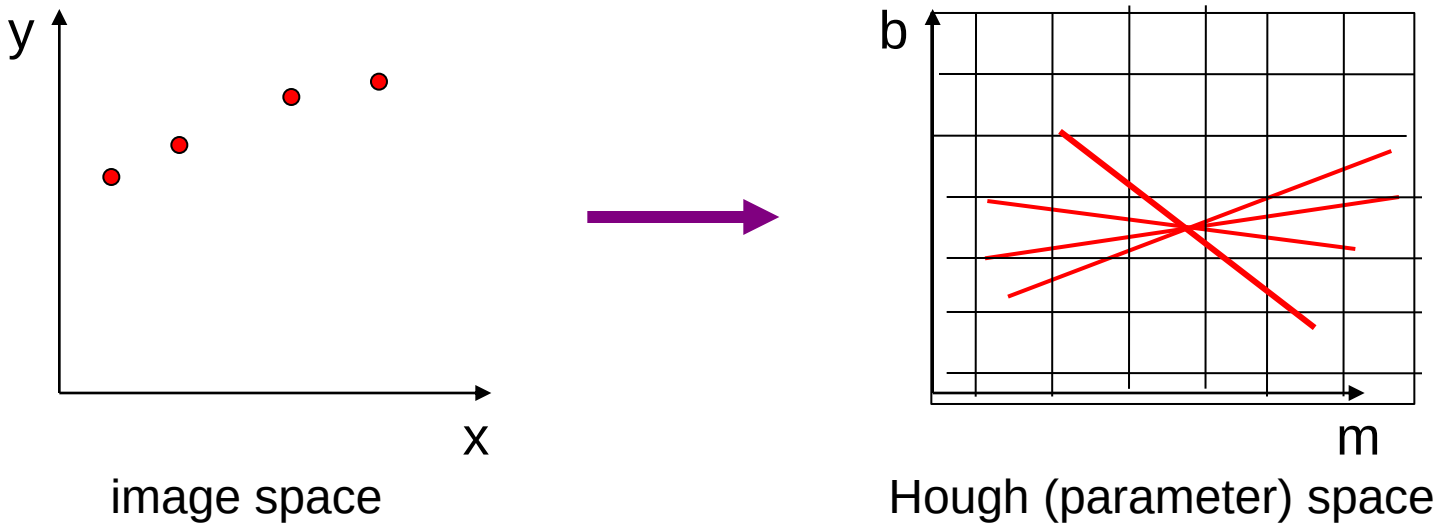


What are the line parameters for the line that contains both (x_0, y_0) and (x_1, y_1)?

- It is the intersection of the lines $b = -x_0m + y_0$ and $b = -x_1m + y_1$

Slide adapted from K. Grauman

Fitting Lines with the Hough Transform

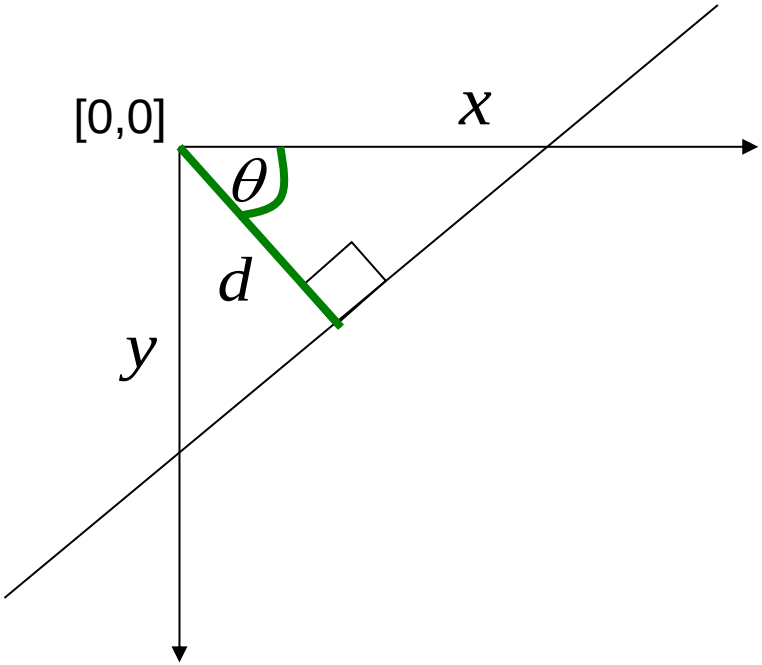


How can we use this to find the most likely parameters (m,b) for the most prominent line in the image space?

- Let each edge point in image space *vote* for a set of possible parameters in Hough space
- Accumulate votes in discrete set of bins; parameters with the most votes indicate line in image space

Slide adapted from K. Grauman

Polar Line Representation



d : perpendicular distance from line to origin

θ : angle between the perpendicular and the x-axis

$$x\cos\theta - y\sin\theta = d$$

- Issues with usual (*m,b*) parameter space: can take on infinite values, undefined for vertical lines.
- Use polar representation of lines
- Point in image space → sinusoid segment in Hough space

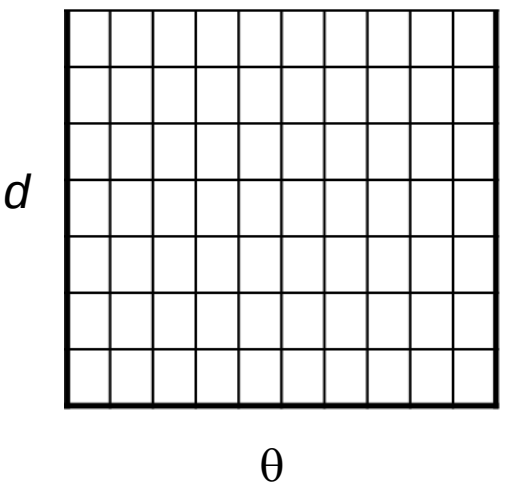
Slide adapted from K. Grauman

Hough Transform Algorithm (for Lines)

Using the polar parameterization:

$$x\cos\theta - y\sin\theta = d$$

H: accumulator array (votes)



Basic Hough transform algorithm

1. Initialize H[d, θ]=0
2. for each edge point I[x,y] in the image

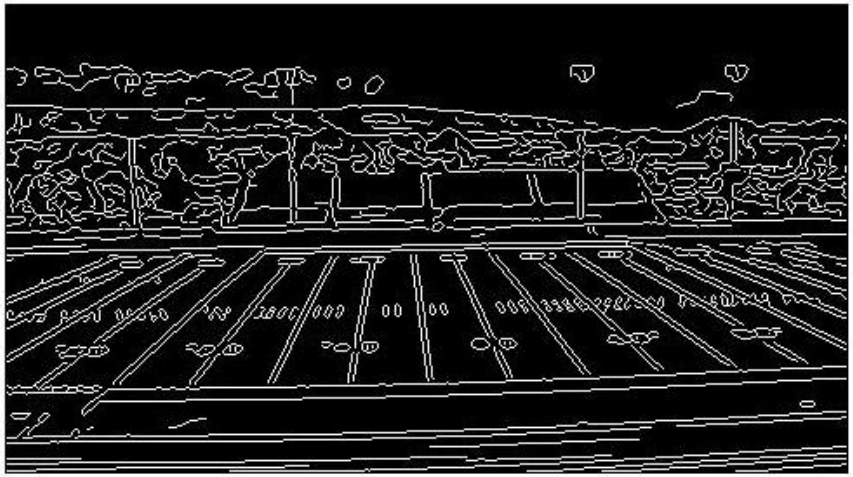
   for θ = [θ_{min} to θ_{max}] // some quantization

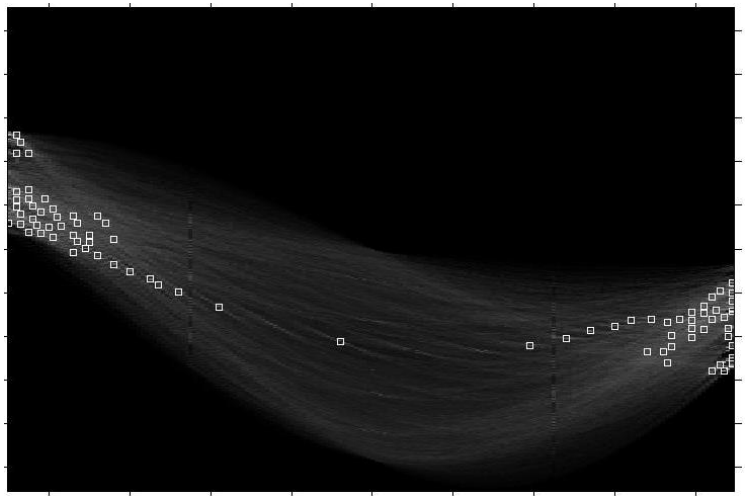
   $d = x\cos\theta - y\sin\theta$

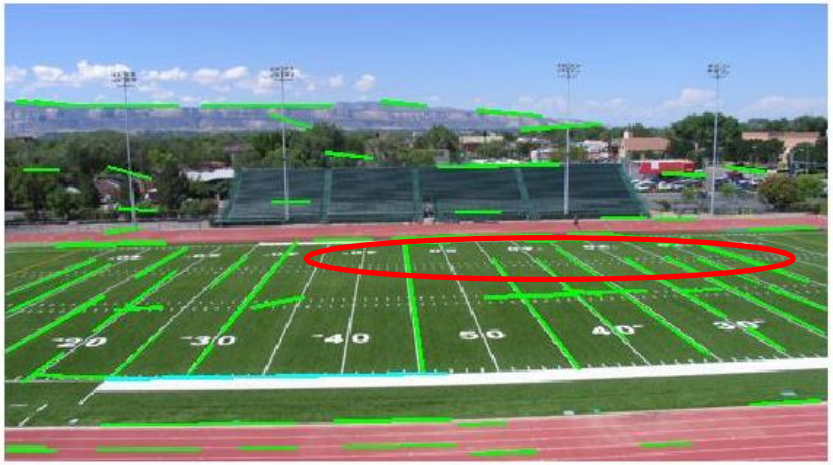
   H[d, θ] += 1
3. Find the value(s) of (d, θ) where H[d, θ] is maximum
4. The detected line in the image is given by $d = x\cos\theta - y\sin\theta$

Slide adapted from K. Grauman

Fitting Lines with the Hough Transform







Showing longest segments found

Slide adapted from K. Grauman

Impact of Noise on the Hough Transform

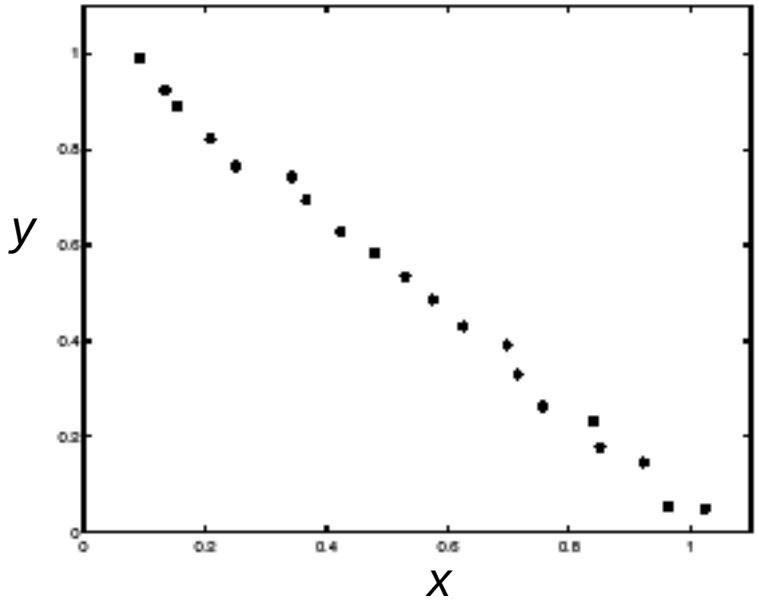
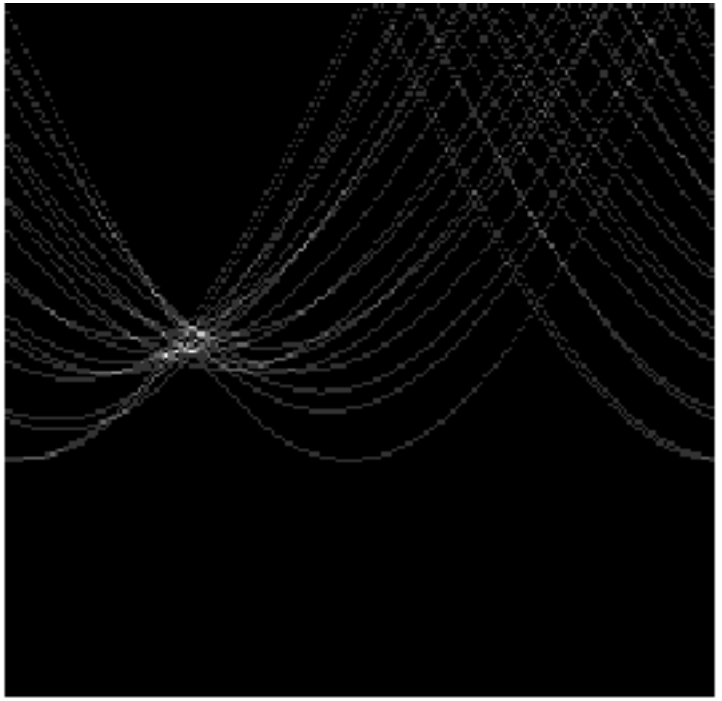


Image space
edge coordinates



Votes

Slide adapted from K. Grauman

Impact of Noise on the Hough Transform

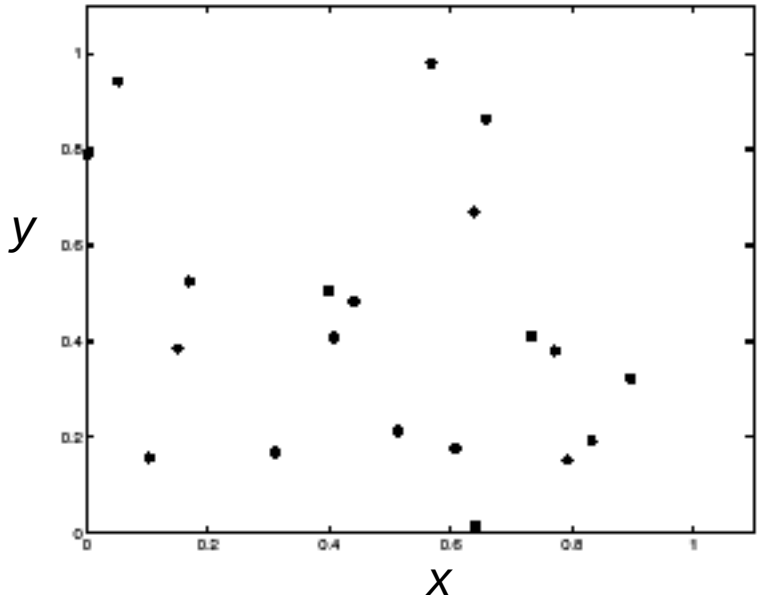
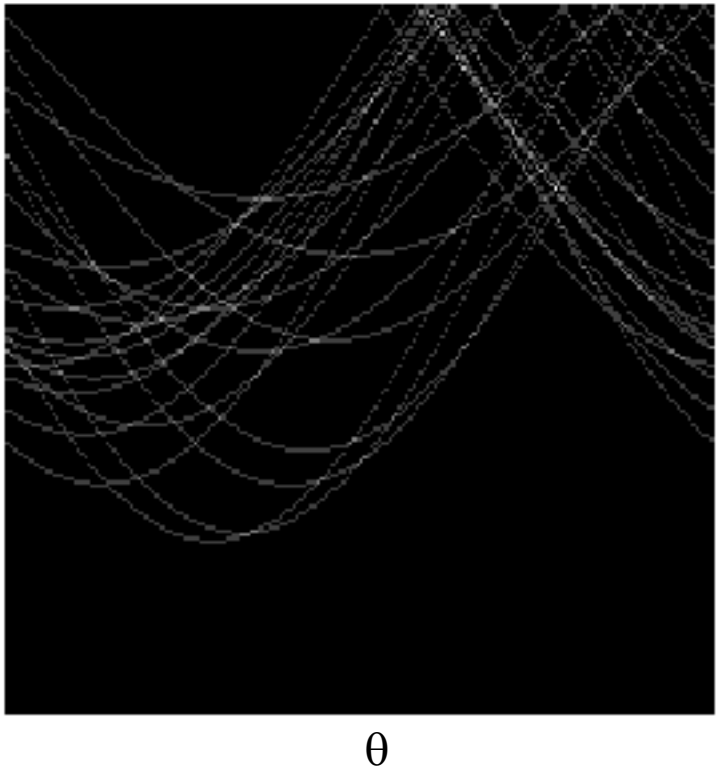


Image space
edge coordinates



Votes

Slide adapted from K. Grauman

Extensions

Extension 1: Use the image gradient

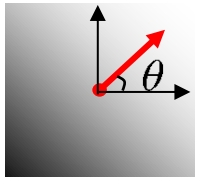
1. same
2. for each edge point I[x,y] in the image

   θ = gradient angle at (x,y)

   $d = x\cos\theta - y\sin\theta$

   H[d, θ] += 1
3. same
4. same

(Reduces degrees of freedom)



$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

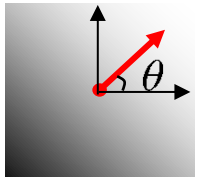
$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

Slide adapted from K. Grauman

Extensions

Extension 1

- Use the image gradient



$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

Extension 2

- Give more votes for stronger edges (use magnitude of gradient)

Extension 3

- Change the sampling of (d, θ) to give more/less resolution

Extension 4

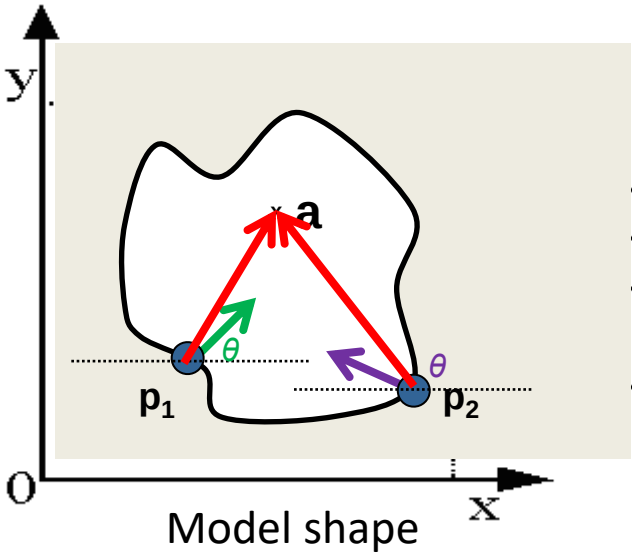
- The same procedure can be used with circles, squares, or any other analytically defined shape …

Can we generalize the Hough transform to any arbitrary shape?

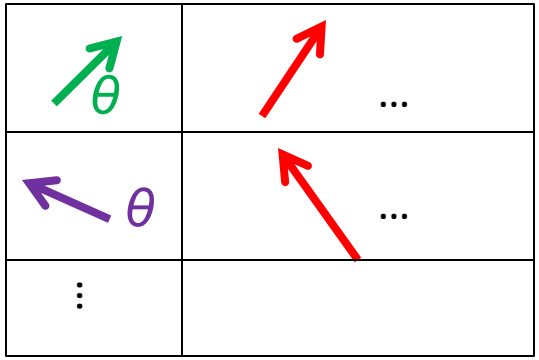
Slide adapted from K. Grauman

Generalized Hough Transform

- Define a model shape by its boundary (edge) points and a reference point



Model shape



Offline procedure:

At each boundary point, compute displacement vector:

$\mathbf{r} = \mathbf{a} - \mathbf{p}_i$

Store these vectors in a table indexed by gradient orientation θ

(Dana H. Ballard, Generalizing the Hough Transform to Detect Arbitrary Shapes, 1980)

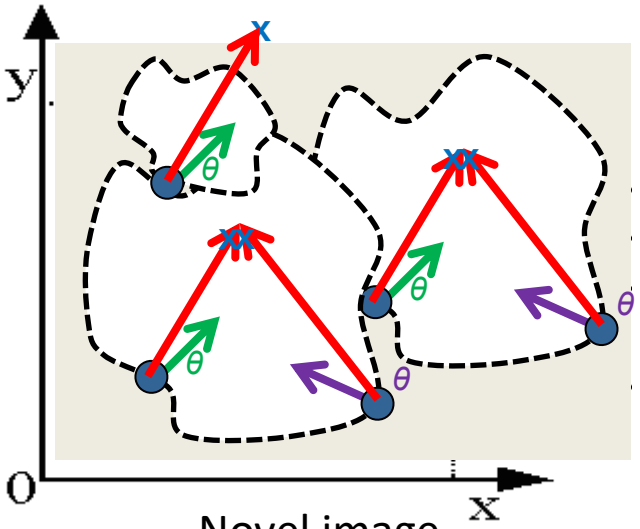
Slide adapted from K. Grauman

Generalized Hough Transform

Detection procedure:

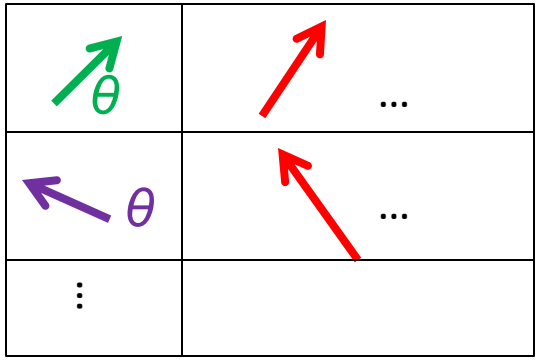
For each edge point:

- Use its gradient orientation to index into stored table
- Use retrieved **r** vectors to vote for reference point



Novel image

Model shape



[Dana H. Ballard, Generalizing the Hough Transform to Detect Arbitrary Shapes, 1980]

Slide adapted from K. Grauman

Generalized Hough Transform

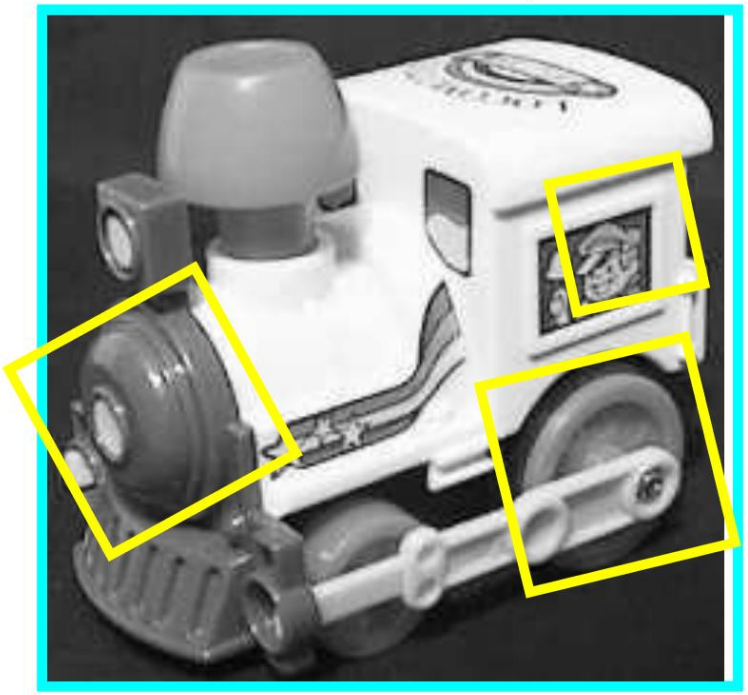
- Instead of indexing displacements by gradient orientation, index by “visual codeword”

B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model, ECCV Workshop on Statistical Learning in Computer Vision 2004

Slide adapted from K. Grauman

Hough Voting: 2D-to-2D Matching

- Oriented local 2D keypoint matches cast votes for affine transformations (f.e. 2D translation, scale & 2D rotation)



Slide adapted from S. Lazebnik

Lessons Learned Today

- Object detection is about localization and recognition of objects in images

- 3D object detection:
  - pose estimation of specific objects
  - From 2D-to-2D keypoint correspondences to an object model we can estimate affine and projective transformations
  - If we have 3D position of keypoints in a model available, we can apply PnP algorithms to estimate 6-DoF pose

- Generalized Hough transform as alternative to RANSAC for correspondence grouping and robust alignment

Thanks for your attention!

Slides Information

- These slides have been initially created by Jörg Stückler as part of the lecture “Robotic 3D Vision” in winter term 2017/18 at Technical University of Munich.
- The slides have been revised by myself (Niclas Zeller) for the same lecture held in winter term 2020/21
- Acknowledgement of all people that contributed images or video material has been tried (please kindly inform me if such an acknowledgement is missing so it can be added).