CS 2770: Computer Vision

Fitting Models:
Hough Transform & RANSAC

Prof. Adriana Kovashka
University of Pittsburgh
February 28, 2017
Plan for today

• Last lecture: Detecting edges
• This lecture: Detecting *lines* (and other shapes)
  – Find the parameters of a line that best fits our data
  – Least squares
  – Hough transform
  – RANSAC
Characterizing edges

• An edge is a place of rapid change in the image intensity function

- image
- intensity function (along horizontal scanline)
- gradient (first derivative)

edges correspond to extrema of gradient

Adapted from L. Lazebnik
Canny edge detector

• Filter image with derivative of Gaussian
• Find magnitude and orientation of gradient
• Threshold: Determine which local maxima from filter output are actually edges

• Non-maximum suppression:
  – Thin wide “ridges” down to single pixel width

• Linking and thresholding (hysteresis):
  – Define two thresholds: low and high
  – Use the high threshold to start edge curves and the low threshold to continue them

Adapted from K. Grauman, D. Lowe, L. Fei-Fei
Thresholding gradient with a higher threshold
Related: Line detection (fitting)

• Why fit lines?
  Many objects characterized by presence of straight lines

• Why aren’t we done just by running edge detection?
Difficultly of line fitting

- **Noise** in measured edge points, orientations:
  - e.g. edges not collinear where they should be
  - how to detect true underlying parameters?

- **Extra** edge points (clutter):
  - 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?

Adapted from Kristen Grauman
Fitting other objects

- Want to associate a model with observed features
- The model could be a line, a circle, or an arbitrary shape

[Fig from Marszalek & Schmid, 2007]
Least squares line fitting

- Data: $(x_1, y_1), \ldots, (x_n, y_n)$
- Line equation: $y_i = mx_i + b$
- Find $(m, b)$ to minimize

$$E = \sum_{i=1}^{n} (mx_i + b - y_i)^2$$

where line you found tells you point is along y axis
where point really is along y axis

You want to find a single line that “explains” all of the points in your data, but data may be noisy!

$$E = \sum_{i=1}^{n} \left( \begin{bmatrix} x_i \\ 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} - y_i \right)^2 = \left\| \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} - \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \right\|^2 = \|Ap - y\|^2$$

Matlab: $p = A \backslash y$;
Outliers affect least squares fit
Outliers affect least squares fit
Outliers

- **Outliers** can hurt the quality of our parameter estimates:
  - E.g. an edge point that is noise, or doesn’t belong to the line we are fitting
- Two common ways to deal with outliers, both boil down to **voting**:
  - Hough transform
  - RANSAC
Dealing with outliers: Voting

- **Voting** is a general technique where we let the features vote for all models that are compatible with it.
  - Cycle through features, cast votes for model parameters.
  - Look for model parameters that receive a lot of votes.

- Noise & clutter features?
  - They will cast votes too, but typically their votes should be inconsistent with the majority of “good” features.
Fitting lines: Hough transform

- Given 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 some edge point lies.
2. Look for lines that get many votes.
Finding lines in an image: Hough space

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

A line in the image corresponds to a point in Hough space

\[ y = m_0 x + b_0 \]
Finding lines in an image: Hough space

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

\[ y = m_0 x + b_0 \]

- A line in the image corresponds to a point in Hough space
- 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

- To go from image space to Hough space:
  - given a pair of points \((x,y)\), find all \((m,b)\) such that \(y = mx + b\)

Adapted from Steve Seitz
Finding lines in an image: Hough space

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 $b = -x_1m + y_1$
Finding lines in an image: Hough space

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.
Finding lines in an image: Hough space

Adapted from Silvio Savarese
Parameter space representation

- Problems with the \((m,b)\) space:
  - Unbounded parameter domains
  - Vertical lines require infinite \(m\)
Parameter space representation

- Problems with the \((m, b)\) space:
  - Unbounded parameter domains
  - Vertical lines require infinite \(m\)
- Alternative: *polar representation*

\[
x \cos \theta + y \sin \theta = \rho
\]

Each point \((x, y)\) will add a sinusoid in the \((\theta, \rho)\) parameter space
Parameter space representation


Use a polar representation for the parameter space
Each line is a sinusoid in Hough parameter space

\[ x \cos \theta + y \sin \theta = \rho \]
Algorithm outline

• Initialize accumulator $H$ to all zeros
• For each edge point $(x, y)$ in the image
  For $\theta = 0$ to 180
    $\rho = x \cos \theta + y \sin \theta$
    $H(\theta, \rho) = H(\theta, \rho) + 1$
  end
end
• Find the value(s) of $(\theta^*, \rho^*)$ where $H(\theta, \rho)$ is a local maximum
• The detected line in the image is given by
  $\rho^* = x \cos \theta^* + y \sin \theta^*$
Hough transform example
Impact of noise on Hough

Image space
edge coordinates

Votes
Impact of noise on Hough Image space edge coordinates 

Votes

What difficulty does this present for an implementation?
Impact of noise on Hough

Noisy data

Need to adjust grid size or smooth
Impact of noise on Hough

Here, everything appears to be “noise”, or random edge points, but we still see peaks in the vote space.
Algorithm outline: Let’s simplify

- Initialize accumulator $H$ to all zeros
- For each edge point $(x, y)$ in the image
  - For $\theta = 0$ to $180$
    - $\rho = x \cos \theta + y \sin \theta$
    - $H(\theta, \rho) = H(\theta, \rho) + 1$
  - end
- Find the value(s) of $(\theta^*, \rho^*)$ where $H(\theta, \rho)$ is a local maximum
- The detected line in the image is given by $\rho^* = x \cos \theta^* + y \sin \theta^*$
Incorporating image gradients

- Recall: when we detect an edge point, we also know its gradient direction
- But this means that the line is uniquely determined!

- Modified Hough transform:

For each edge point \((x,y)\) in the image

\[
\theta = \text{gradient orientation at } (x,y) \\
\rho = x \cos \theta + y \sin \theta \\
H(\theta, \rho) = H(\theta, \rho) + 1
\]

end
Hough transform for circles

- A circle with radius $r$ and center $(a, b)$ can be described as:

\[
\begin{align*}
  x &= a + r \cos(\theta) \\
  y &= b + r \sin(\theta)
\end{align*}
\]
Hough transform for circles

• Circle: center \((a, b)\) and radius \(r\)

\[
(x_i - a)^2 + (y_i - b)^2 = r^2
\]

• For a fixed radius \(r\), unknown gradient direction
Hough transform for circles

- Circle: center \((a, b)\) and radius \(r\)
  \[
  (x_i - a)^2 + (y_i - b)^2 = r^2
  \]
- For a fixed radius \(r\), unknown gradient direction

Intersection: most votes for center occur here.
Hough transform for circles

- Circle: center \((a, b)\) and radius \(r\)

\[(x_i - a)^2 + (y_i - b)^2 = r^2\]

- For an unknown radius \(r\), unknown gradient direction
Hough transform for circles

- Circle: center \((a, b)\) and radius \(r\)
  \[
  (x_i - a)^2 + (y_i - b)^2 = r^2
  \]
- For an unknown radius \(r\), known gradient direction
Hough transform for circles

For every edge pixel \((x,y)\) :

For each possible radius value \(r\):

For each possible gradient direction \(\theta\):

// or use estimated gradient at \((x,y)\)

\[
x = a + r \cos(\theta)
\]

\[
y = b + r \sin(\theta)
\]

\[
a = x - r \cos(\theta) \quad \text{// column}
\]

\[
b = y - r \sin(\theta) \quad \text{// row}
\]

\[
H[a,b,r] += 1
\]

end

end

end

Modified from Kristen Grauman
Example: detecting circles with Hough

Note: a different Hough transform (with separate accumulators) was used for each circle radius (quarters vs. penny).

Kristen Grauman, images from Vivek Kwatra
Example: detecting circles with Hough

Note: a different Hough transform (with separate accumulators) was used for each circle radius (quarters vs. penny).
Example: iris detection

Gradient+threshold  Hough space  Max detections
(fixed radius)

Hemerson Pistori and Eduardo Rocha Costa
Voting: practical tips

- Minimize irrelevant tokens first (reduce noise)
- Choose a good grid / discretization
  - Too coarse: large votes obtained when too many different lines correspond to a single bucket
  - Too fine: miss lines because points that are not exactly collinear cast votes for different buckets
- Vote for neighbors, also (smoothing in accumulator array)
- Use direction of edge to reduce parameters by 1
- To read back which points voted for “winning” peaks, keep tags on the votes
Generalized Hough transform

• We want to find a template defined by its reference point (center) and several distinct types of landmark points in stable spatial configuration

Triangle, circle, diamond: some *type* of visual token, e.g. feature or edge point

Template

Adapted from Svetlana Lazebnik
Generalized Hough transform

Intuition:

Model image

Novel image

Vote space

Now suppose those colors encode gradient directions…

Adapted from Kristen Grauman
Define a model shape by its boundary points and a reference point.

**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 \( \theta \).
Detection procedure:

For each edge point:

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

Assuming translation is the only transformation here, i.e., orientation and scale are fixed.
Generalized Hough transform

- Template representation: for each type of landmark point, store all possible displacement vectors towards the center

Template

Model
Generalized Hough transform

• Detecting the template:
  • For each feature in a new image, look up that feature type in the model and vote for the possible center locations associated with that type in the model

Test image

Model
Preview: Hough for object detection

- Index displacements 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
Hough transform: pros and cons

Pros

- All points are processed independently, so can cope with occlusion, gaps
- Some robustness to noise: noise points *unlikely* to contribute *consistently* to any single bin
- Can detect multiple instances of a model in a single pass

Cons

- Complexity of search time for maxima increases exponentially with the number of model parameters
  - If 3 parameters and 10 choices for each, search is $O(10^3)$
- Non-target shapes can produce spurious peaks in parameter space
- Quantization: can be tricky to pick a good grid size

Adapted from Kristen Grauman
RANSAC

- RANdom Sample Consensus

- **Approach**: we want to avoid the impact of outliers, so let’s look for “inliers”, and use those only.

- **Intuition**: if an outlier is chosen to compute the current fit, then the resulting line won’t have much support from rest of the points.
RANSAC: General form

• RANSAC loop:

1. Randomly select a seed group of $s$ points on which to base model estimate (e.g. $s=2$ for a line)

2. Fit model to these $s$ points

3. Find inliers to this model (i.e., points whose distance from the line is less than $t$)

4. Repeat $N$ times

• Keep the model with the largest number of inliers

Adapted from Kristen Grauman and Svetlana Lazebnik
RANSAC

(RANdom SAmple Consensus):

Fischler & Bolles in ‘81.

Line fitting example

Algorithm:

1. **Sample** (randomly) the number of points required to fit the model
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence
Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

**Repeat** 1-3 until the best model is found with high confidence
RANSAC

(RANdom SAmple Consensus):

Fischler & Bolles in ‘81.

Line fitting example

Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence
RANSAC


Line fitting example

\[ N_I = 6 \]

Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (\# = 2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

**Repeat** 1-3 until the best model is found with high confidence
RANSAC

(RANdom SAmple Consensus):

Fischler & Bolles in ‘81.

Line fitting example

Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

Silvio Savarese
How to choose parameters?

• Number of samples $N$
  – Choose $N$ so that, with probability $p$, at least one random sample is free from outliers (e.g. $p=0.99$) (outlier ratio: $e$)

• Number of sampled points $s$
  – Minimum number needed to fit the model

• Distance threshold $\delta$
  – Choose $\delta$ so that a good point with noise is likely (e.g., prob=0.95) within threshold
  – E.g. for zero-mean Gaussian noise with std. dev. $\sigma$: $\delta^2 = 3.84\sigma^2$

\[
N = \log(1 - p)/\log(1 - (1 - e)^s)
\]

Explanation in Szeliski 6.1.4

<table>
<thead>
<tr>
<th>$s$</th>
<th>5%</th>
<th>10%</th>
<th>20%</th>
<th>25%</th>
<th>30%</th>
<th>40%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>11</td>
<td>17</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>4</td>
<td>7</td>
<td>9</td>
<td>11</td>
<td>19</td>
<td>35</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>13</td>
<td>17</td>
<td>34</td>
<td>72</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>6</td>
<td>12</td>
<td>17</td>
<td>26</td>
<td>57</td>
<td>146</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>7</td>
<td>16</td>
<td>24</td>
<td>37</td>
<td>97</td>
<td>293</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>8</td>
<td>20</td>
<td>33</td>
<td>54</td>
<td>163</td>
<td>588</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>9</td>
<td>26</td>
<td>44</td>
<td>78</td>
<td>272</td>
<td>1177</td>
</tr>
</tbody>
</table>
RANSAC pros and cons

• Pros
  • Applicable to many different problems
  • Often works well in practice

• Cons
  • Lots of parameters to tune (see previous slide)
  • Doesn’t work well for low inlier ratios (too many iterations, or can fail completely)

• Common applications
  • Image stitching, relating two views
  • Spatial verification

Adapted from Svetlana Lazebnik
Application: Keypoint match for search

1. Find a set of distinctive keypoints
2. Define a region around each keypoint (window)
3. Compute a local descriptor from the region
4. Match descriptors

\[ d(f_A, f_B) < T \]

Adapted from K. Grauman, B. Leibe
Example: solving for translation

Given matched points in \{A\} and \{B\}, estimate the translation of the object

\[
\begin{bmatrix}
    x_i^B \\
    y_i^B
\end{bmatrix} = \begin{bmatrix}
    x_i^A \\
    y_i^A
\end{bmatrix} + \begin{bmatrix}
    t_x \\
    t_y
\end{bmatrix}
\]
Example: solving for translation

### Least squares solution

1. Write down objective function in form $Ax=b$
2. Solve e.g. using pseudo-inverse

\[
\begin{bmatrix}
    x_i^B \\
    y_i^B
\end{bmatrix}
= 
\begin{bmatrix}
    x_i^A \\
    y_i^A
\end{bmatrix} + 
\begin{bmatrix}
    t_x \\
    t_y
\end{bmatrix}
\]

\[
\begin{bmatrix}
    1 & 0 \\
    0 & 1 \\
    \vdots & \vdots \\
    1 & 0 \\
    0 & 1
\end{bmatrix}
\begin{bmatrix}
    t_x \\
    t_y
\end{bmatrix}
= 
\begin{bmatrix}
    x_1^B - x_1^A \\
    y_1^B - y_1^A \\
    \vdots \\
    x_n^B - x_n^A \\
    y_n^B - y_n^A
\end{bmatrix}
\]

Adapted from Derek Hoiem
Example: solving for translation

Problem: outliers

RANSAC solution
1. Sample a set of matching points (1 pair)
2. Solve for transformation parameters
3. Repeat N times

\[
\begin{bmatrix}
\chi_i^B \\
y_i^B
\end{bmatrix} = \begin{bmatrix}
\chi_i^A \\
y_i^A
\end{bmatrix} + \begin{bmatrix}
t_x \\
t_y
\end{bmatrix}
\]

Adapted from Derek Hoiem
Example: solving for translation

Problem: outliers, multiple objects

Hough transform solution

1. Initialize a grid of parameter values
2. Each matched pair casts a vote for consistent values
3. Find the parameters with the most votes

\[
\begin{bmatrix}
\chi_i^B \\
y_i^B
\end{bmatrix} = \begin{bmatrix}
\chi_i^A \\
y_i^A
\end{bmatrix} + \begin{bmatrix}
t_x \\
t_y
\end{bmatrix}
\]
Fitting and Matching: Summary

- **Fitting** problems require finding any supporting evidence for a model, even within clutter and missing features.

- **Voting and inlier** approaches, such as the **Hough transform and RANSAC**, make it possible to find likely model parameters without searching all combinations of features, and with decreased impact from outliers.

- Can use these approaches to compute robust feature alignment/matching, and to match object templates.

Adapted from Kristen Grauman and Derek Hoiem