Fitting: The Hough transform
Voting schemes

• Let each feature vote for all the models that are compatible with it
• Hopefully the noise features will not vote consistently for any single model
• Missing data doesn’t matter as long as there are enough features remaining to agree on a good model
Hough transform

• An early type of voting scheme
• General outline:
  • Discretize parameter space into bins
  • For each feature point in the image, put a vote in every bin in the parameter space that could have generated this point
  • Find bins that have the most votes

Parameter space representation

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

\[ y = m_0 x + b_0 \]
Parameter space representation

- What does a point \((x_0, y_0)\) in the image space map to in the Hough space?
Parameter space representation

- What does a point \((x_0, y_0)\) in the image space map to in the Hough space?
  - Answer: the solutions of \(b = -x_0m + y_0\)
  - This is a line in Hough space

![Diagram showing image space and Hough parameter space with a point mapping to a line in the Hough space](image.png)
Parameter space representation

- Where is the line that contains both $(x_0, y_0)$ and $(x_1, y_1)$?

\[ b = -x_1 m + y_1 \]
Parameter space representation

- Where is 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 \(b = -x_1m + y_1\)

Image space

Hough parameter space

\[ b = -x_0m + y_0 \]

\[ b = -x_1m + y_1 \]
Parameter space representation

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

• Problems with the (m,b) space:
  • Unbounded parameter domain
  • Vertical lines require infinite m

• Alternative: polar representation

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

Each point will add a sinusoid in the \((\theta, \rho)\) parameter space
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
    - \(\rho = x \cos \theta + y \sin \theta\)
Basic illustration

features

votes
Other shapes

Square

Circle
Several lines
A more complicated image

http://ostatic.com/files/images/ss_hough.jpg
Effect of noise

features
Effect of noise

Peak gets fuzzy and hard to locate
Effect of noise

- Number of votes for a line of 20 points with increasing noise:
Random points

Uniform noise can lead to spurious peaks in the array.
Random points

- As the level of uniform noise increases, the maximum number of votes increases too:
Dealing with 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 some points that are not exactly collinear cast votes for different buckets

• Increment neighboring bins (smoothing in accumulator array)

• Try to get rid of irrelevant features
  • Take only edge points with significant gradient magnitude
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 \((x,y)\)

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

\[ H(\theta, \rho) = H(\theta, \rho) + 1 \]

end
Hough transform for circles

- How many dimensions will the parameter space have?
- Given an oriented edge point, what are all possible bins that it can vote for?
Hough transform for circles

Image space:

- Point $(x, y)$
- Point $(x, y) + r\nabla I(x, y)$
- Point $(x, y) - r\nabla I(x, y)$

Hough parameter space:

- Lines for different $r$ values
Hough transform for circles

- Conceptually equivalent procedure: for each (x, y, r), draw the corresponding circle in the image and compute its “support”

Is this more or less efficient than voting with features?
Application in recognition

Hough circles vs. Laplacian blobs

Original images

Robustness to scale and clutter

Laplacian circles

Hough-like circles

Generalized Hough transform

• We want to find a shape defined by its boundary points and a reference point

Generalized Hough transform

- We want to find a shape defined by its boundary points and a reference point.
- For every boundary point $p$, we can compute the displacement vector $r = a - p$ as a function of gradient orientation $\theta$.

Generalized Hough transform

- For model shape: construct a table indexed by $\theta$ storing displacement vectors $r$ as function of gradient direction
- Detection: For each edge point $p$ with gradient orientation $\theta$:
  - Retrieve all $r$ indexed with $\theta$
  - For each $r(\theta)$, put a vote in the Hough space at $p + r(\theta)$
- Peak in this Hough space is reference point with most supporting edges
- Assumption: translation is the only transformation here, i.e., orientation and scale are fixed

Source: K. Grauman
Example

model shape
Example

displacement vectors for model points
Example

range of voting locations for test point
Example

range of voting locations for test point
Example

votes for points with $\theta = \uparrow$
Example

displacement vectors for model points
Example

range of voting locations for test point
Example

votes for points with $\theta = \checkmark$
Application in recognition

• 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
Application in recognition

• 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
Implicit shape models: Training

1. Build codebook of patches around extracted interest points using clustering (more on this later in the course)
Implicit shape models: Training

1. Build codebook of patches around extracted interest points using clustering
2. Map the patch around each interest point to closest codebook entry
Implicit shape models: Training

1. Build codebook of patches around extracted interest points using clustering
2. Map the patch around each interest point to closest codebook entry
3. For each codebook entry, store all positions it was found, relative to object center
Implicit shape models: Testing

1. Given test image, extract patches, match to codebook entry
2. Cast votes for possible positions of object center
3. Search for maxima in voting space
4. Extract weighted segmentation mask based on stored masks for the codebook occurrences
Implicit shape models: Details

• Supervised training
  • Need reference location and segmentation mask for each training car
• Voting space is continuous, not discrete
  • Clustering algorithm needed to find maxima
• How about dealing with scale changes?
  • Option 1: search a range of scales, as in Hough transform for circles
  • Option 2: use scale-covariant interest points
• Verification stage is very important
  • Once we have a location hypothesis, we can overlay a more detailed template over the image and compare pixel-by-pixel, transfer segmentation masks, etc.
Hough transform: Discussion

• Pros
  • Can deal with non-locality and occlusion
  • Can detect multiple instances of a model
  • Some robustness to noise: noise points unlikely to contribute consistently to any single bin

• Cons
  • Complexity of search time increases exponentially with the number of model parameters
  • Non-target shapes can produce spurious peaks in parameter space
  • It’s hard to pick a good grid size

• Hough transform vs. RANSAC