Hough Transform

- It is a technique used to isolate curves of a given shape in an image.
- In its classical formulation it requires the curve to be specified in some parametric form (usually lines, circles or ellipses).
- Also... it can be generalized to arbitrary curved shapes.
- Advantage: robust to “gaps” in the object.
- Disadvantages:
  - parametric description of the shape
  - depending on the number of parameters might become slow.
Hough Transform for lines

- We want to detect points lying on a straight line
- Start with the equation of a straight line in parametric form:

\[ x \cos \phi + y \sin \phi = r \]

where \( r \) is the length of a normal to the line from the origin and \( \phi \) is its angle with the x-axis
In case you don’t trust the equation…

\[ r = \rho \cos(\alpha) = \rho \cos(\vartheta - \phi) \]

\[ r = \rho \cos \vartheta \cos \phi + \rho \sin \vartheta \sin \phi \]

\[ r = x \cos \phi + y \sin \phi \]
Given a point \( x_i, y_i \) on this line we have:

\[
x_i \cos \phi + y_i \sin \phi = r
\]

where \( r \) and \( \phi \) are constant

Consider now \( r \) and \( \phi \) variable \((r_i, \phi_i)\) and \( x_i y_i \) constant \((x, y)\), the equation describes all possible lines passing through the point, these lines are described by:

\[
r_i = x \cos \phi_i + y \sin \phi_i \quad \phi \in [0, 2\pi]
\]

The Hough Transform of the point is the plot of this equation on the \((r_i, \phi_i)\) space (Hough Space)

we get a sinusoidal curve
Hough Transform of one point
Hough Transform

• Define the Hough Space: range for $r$ (example: 0-255) and the angular resolution of the sampling of $\phi$ (example: 6 degrees)
• This gives a Hough Space (HS) of 256x60 points
• Each point in the image “vote” for a set of lines passing through it; all these votes are accumulated in the HS
• For each $(x,y)$ increment all accumulator cells $(r,\phi)$ which satisfy the equation:

$$r = x \cos \phi + y \sin \phi$$

• At the end we scan the accumulator searching for cells which have high count: they correspond to lines for which there are many points in the image plane
Example: start with one point (pixel)
Example: add another pixel
Example: now we have three pixels
Example: the last one…
Finally: search intersections in the HT

$p_1, p_2$ and $p_3$ intersect at the orientation of $45^\circ$

$p_3$ and $p_4$ intersect at $90^\circ$, $p_4$ intersects the other points around $60^\circ$
Improvements:

- Usually start with the output of an edge detector, to reduce the number of points in the image
- Search along directions perpendicular to the image gradient
Hough Example

extract 5 strongest lines

houghdemo.m from http://homepages.cae.wisc.edu/~ece533/matlab/index.html
Circular Hough Transform

- Detect circles in the image
- Parametric equation of a circle:
  \[(x - a)^2 + (y - b)^2 = r^2\]
  where \((a, b)\) are the coordinates of the center of the circle and \(r\) its radius
- \(a, b\) and \(r\) define the parameter space, the accumulator is three-dimensional

\[
\begin{align*}
(a_1, b_1, r_1) \\
(a_2, b_2, r_2) \\
(a_3, b_3, r_3) \\
\ldots
\end{align*}
\]
Suppose we don’t have a simple analytical equation for the object
Instead we use a LUT defining the relationship between the coordinates of the point, its orientation and the Hough parameters

Build the LUT:
1. Select an arbitrary reference point \((x_{\text{ref}}, y_{\text{ref}})\)
2. For all points of the boundary \((x_i, y_i)\)
   - draw a line from \((x_i, y_i)\) to \((x_{\text{ref}}, y_{\text{ref}})\)
   - Measure \((\beta_i, r_i)\)
   - Compute the orientation of the boundary \(\Omega_i\)
   - Add \((\beta_i, r_i)\) to a table indexed by \(\Omega_i\)

\[
\begin{align*}
x_{\text{ref}} &= x_i + r_i \cos \beta_i \\
y_{\text{ref}} &= y_i + r_i \sin \beta_i
\end{align*}
\]
• Probably there will be more than one occurrence of a particular orientation…

\[
\begin{align*}
x_{\text{ref}} &= x_i + r_i \cos \beta_i \\
y_{\text{ref}} &= y_i + r_i \sin \beta_i
\end{align*}
\]

| \( \Omega_0 \) | … |
| \( \Omega_1 \) | … |
| … | … |
| \( \Omega_i \) | \( (r_1, \beta_1), (r_2, \beta_2) \) … |
| … | … |

R-table
• Once we have the R-table for the object, we can perform the Hough Transform of the image
• For each point in the image \((x_i, y_i)\), we compute the point \((x_{ref}, y_{ref})\) from:
\[
\begin{align*}
x_{ref} &= x_i + r_i \cos \beta_i \\
y_{ref} &= y_i + r_i \sin \beta_i
\end{align*}
\]
• where \((r_i, \beta_i)\) are derived from the R-table, starting from the orientation of the point \(\Omega_i\)
• We accumulate the Hough Space in \((x_{ref}, y_{ref})\):
\[
A(x_{ref}, y_{ref})++
\]
• Finally search for local maxima in \(A\) to identify the center(s) of the object(s)
• It is easy to extend the search for different object orientations $\varphi$ and scales $S$:

\[
\begin{align*}
  x_{\text{ref}} &= x_i + S \cdot r_i \cos(\beta_i + \varphi) \\
  y_{\text{ref}} &= y_i + S \cdot r_i \sin(\beta_i + \varphi)
\end{align*}
\]

• In this case we explore and accumulate a four-dimensional space:

\[
A(x_{\text{ref}}, y_{\text{ref}}, \varphi, S)++
\]

SINA 08/09
Measures of similarity

• Another approach to detect some previously defined object within a given image is to perform template matching.
• A template is a sub-image representing the “ideal” pattern that is sought in the image.
• Involves the translation of the template to every possible position of the image, and the evaluation of a the level of match between the template and the image in that position.
• Global versus local template.
• For each position of the template in the image we evaluate a similarity measure between the template and the image
• Possible measures are summation difference or cross-correlation
• These measures are not only used for template matching but also to estimate the level of similarity between two signals

1. **Euclidean distance**
• Estimate the similarity between \( g \) and \( t \) in \( m,n \):

\[
E(m,n) = \sqrt{\sum_i \sum_j [g(i,j) - t(i-m, j-n)]^2}
\]

The sum is computed for all points for which \((i-m, j-n)\) is a valid coordinate of \( t \), in other words for all points in the template image
• To find the best match we look for the smallest value in \( E(m,n) \)
• From an intuitive point of view we are computing the distance between two \(H\times W\) dimensional points, where \(H, W\) is the size of the template.

• Since we are not interested in the exact value of the distance (we search for the minimum in \(E\)) we can avoid computing the square root (Sum of Squared Differences):

\[
SSD(m, n) = E^2(m, n) = \sum_i \sum_j [g(i, j) - t(i - m, j - n)]^2
\]

• Or:

\[
S(m, n) = \sum_i \sum_j |g(i, j) - t(i - m, j - n)|
\]
\[ E^2(m, n) = \sum_i \sum_j \left[ g(i, j)^2 + t(i - m, j - n)^2 - 2g(i, j)t(i - m, j - n) \right] \]

2. Cross-correlation

\[ R(m, n) = \sum_i \sum_j \left[ g(i, j)t(i - m, j - n) \right] \]

\( t \) and \( g \) are similar when \( R(i,j) \) is large

Problems with cross-correlation:
- False matches if the image energy \( \sum \sum g(i, j)^2 \) varies with the position
- Values of \( R \) depends on the size of the template
- Not invariant to illumination change

SINA 08/09
Consider the correlation with a constant pattern, of gray value $v$

\[
\begin{array}{ccc}
  a & b & c \\
  d & e & f \\
  g & h & i \\
\end{array}
\quad
\begin{array}{ccc}
  v & v & v \\
  v & v & v \\
  v & v & v \\
\end{array}
\]

\[R = v^*(a+b+c+...+i)\]
Now consider the correlation with a constant image twice as bright

\[
\begin{array}{ccc}
  a & b & c \\
  d & e & f \\
  g & h & i \\
\end{array}
\quad \begin{array}{ccc}
  2v & 2v & 2v \\
  2v & 2v & 2v \\
  2v & 2v & 2v \\
\end{array}
\]

\[
R=2\times v\times (a+b+c+\ldots+i)>v\times (a+b+c+\ldots+i)
\]

We get higher correlation, with the same template…
• Solution: normalize the intensity
  – subtract the mean of both signals
  – divide by std. deviation

\[
\hat{g} = \frac{g - \bar{g}}{\sqrt{\sum (g - \bar{g})^2}}, \quad \hat{t} = \frac{t - \bar{t}}{\sqrt{\sum (t - \bar{t})^2}}
\]

The normalized cross-correlation is defined as:

\[
NCC(m, n) = \frac{\sum_{i,j} \left[ g(i, j) - \bar{g}_{m,n} \right] \left[ t(i-m, j-n) - \bar{t} \right]}{\sqrt{\sum_{i,j} \left[ g - \bar{g}_{m,n} \right]^2} \sum_{i,j} \left[ t(i-m, j-n) - \bar{t} \right]^2} \in [-1, 1]
\]

\(\bar{t}\) mean value of \(t\), \(\bar{g}_{m,n}\) mean value of \(g\) in the region under \(t\)
Problems with template matching

• The template represents the object as we expect to find it in the image
• The object can indeed be scaled or rotated
• This technique requires a separate template for each scale and orientation
• Template matching become thus too expensive, especially for large templates
Local template matching

• A possible solution is to reduce the size of the templates, and detect *salient* features in the image that characterize the object we are interested in
• Extract a set of local features that are invariant to translation, rotation and scale…
• Perform matching only on these local features
• We then analyze the spatial relationships between those features

See for example:
Corner detector (Harris and Stephens, 1988)
SIFT (Lowe, 1999)
Example:
Application:
How do we build a panorama?

• We need to match (align) images
Matching with Features

• Detect feature points in both images
• Find corresponding pairs
Matching with Features

• Detect feature points in both images
• Find corresponding pairs
• Use these pairs to align images
Gaussian Pyramid
(Burt and Adelson 1983)

• Image pyramid is a collection of representations of an image
• Typically each layer of the pyramid is half the width and half the height of the previous layer
• If we stack the layers on top of each other, we get a pyramid
Idea: Represent NxN image as a "pyramid" of 1x1, 2x2, 4x4, ..., 2^k x 2^k images (assuming N=2^k)
• In a Gaussian pyramid, each layer is smoothed by a symmetric Gaussian kernel, and resampled to get the next layer.
• The smallest image is the most heavily smoothed.
GAUSSIAN PYRAMID

\[ g_0 = \text{IMAGE} \]

\[ g_L = \text{REDUCE} \left[ g_{L-1} \right] \]

\[ g_l(i, j) = \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m, n) g_{l-1}(2i + m, 2j + n). \]
Applications (1)

- Search over scale – objects can be represented as small image patterns; if we want to search across different scales we can search across the layers of the gaussian pyramid; bigger objects will be found in the coarser scale layers, smaller objects will be found in the finer scales
Applications (2)

• Spatial search: often we have a point in one image and want to find the same point in another image (example: stereo vision). This can be achieved more efficiently if we first start to search the object in the coarser layers, and then refine the match by searching in the finer layers (coarse-to-fine matching)
Applications (3)

- Feature tracking: features (e.g. edges) found at coarse levels are associated with high-contrast image events (low contrast patches are easily lost during consequent smoothing); at fine scales there are probably many more features with lower contrast. A common strategy for improving a set of features obtained at a fine scale is to track features to coarser scales and accept only the fine scale features that are identifiable at coarser scales (feature tracking)
Gabor filters

• Suppose we want to analyze the spatial frequency content of an image
• Fourier transform is a way to do this, the problem with this approach is that Fourier coefficients depend on the entire image
• In this way we loose spatial information
• Gabor filters allow to do this; they have stronger response at points in an image where there are components that locally have a particular spatial frequency and orientation
• Gabor filters have impulse response defined by a harmonic function multiplied by a Gaussian function.

• Also used as models of the receptive fields of simple cells.

\[
g(x, y) = e^{-\frac{x'^2 + y'^2}{2\sigma^2}} \cos(2\pi \frac{x'}{\lambda} + \psi)
\]

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

\[
y' = -x \sin \theta + y \cos \theta
\]

- \(\sigma\) – width of the Gaussian
- \(\gamma\) – shape of the Gaussian
- \(\theta\) – orientation
- \(\lambda\) – wavelength
- \(\psi\) – phase
Figure 9.12 The image on the top shows a detail from an image of a zebra, chosen because it has stripes at somewhat different scales and orientations. This has been convolved with the kernel in the center, which is a Gabor filter kernel. The image at the bottom shows the absolute value of the result; notice that the response is large when the spatial frequency of the bars roughly matches that windowed by the Gaussian in the Gabor filter kernel (i.e., the stripes in the kernel are about as wide as, and at about the same orientation as, the three stripes in the kernel). When the stripes are larger or smaller, the response falls off; thus, the filter is performing a kind of local spatial frequency analysis. This filter is one of a quadrature pair (it is the symmetric component). The response of the anti-symmetric component is similarly frequency selective. The two responses can be seen as the two components of the (complex valued) local Fourier transform, so that magnitude and phase information can be extracted from them.

from: Forsyth and Ponce
Visual attention

• Only a small fraction of the information registered by the visual system at any given time reaches levels of processing that directly influence behavior.
• Visual attention controls access to this privileged level and ensures that the selected information is relevant to behavioral priorities and objectives.

• A *spotlight* that enhances important information.
• Can diverge from the direction of the gaze.
Bottom-Up & Top-Down cues

SINA 08/09
Example: Itti’s model (1998)

- Visual attention model, inspired by the behavior and neuronal architecture of the early primate visual system
- “Feature integration theory” to explain human visual search strategies

- Visual input is decomposed in a set of feature maps
- Different spatial locations compete for saliency within each map
- All maps converge into a “master saliency map” which codes local saliency over the entire visual scene
- This map has internal dynamics which generate attentional shifts
1. Create nine spatial scales using Gaussian pyramids (low-pass and subsample), scales from 0 to 8 (1:1… 1:256)

2. Features are computed by a set of linear “center-surround” operations, implemented as the difference between fine and coarser scales:

this produces multi-scale feature extraction, including different ratios between the center and surround regions (in the paper 6 different ratios are used)
• Compute:
\[ I = \frac{(r + g + b)}{3} \]
\[ R = r - \frac{(g + b)}{2}; G = g - \frac{(r + b)}{2} \]
\[ B = b - \frac{(r + g)}{2}; Y = \frac{(r + g)}{2} - \left| r - g \right| \left| 2 - b \right| \]

• And center-surround differences at different scales \((s, c)\):
\[ I(c, s) = \left| I(c) - I(s) \right| \]
\[ RG(c, s) = \left| (R(c) - G(c)) - (G(s) - R(s)) \right| \]
\[ BY(c, s) = \left| (B(c) - Y(c)) - (Y(s) - B(s)) \right| \]

• Local orientation information is obtained from \( I \) using Gabor filters at different scale and orientation \((0, 45, 90, 135)\):
\[ O(c, s, \theta) = \left| O(c, \theta) - O(s, \theta) \right| \]
• Difference are taken between fine and coarse scales, if the center is a pixel at scale $c$, the surround is the corresponding pixel at scale $s = c + \delta$, where:

\[
\begin{align*}
c & \in \{2, 3, 4\} \\
\delta & \in \{3, 4\} \\
\Rightarrow & \text{six combinations}
\end{align*}
\]

• Total of 42 maps: 6 for intensity, 12 for color and 24 for orientation
$N()$ operator, promote maps in which a small number of strong peaks of activity is present, suppress maps with a large number of comparable responses:

- normalize the map to a fixed range $[0..M]$

- compute the average of $m$ of all other local maxima

- multiply the map by $(M-m)^2$, boost maps with small number of strong peaks
Input image

Linear filtering

- colors
- intensity
- orientations

Center-surround differences and normalization

- Feature maps (12 maps)
- (6 maps)
- (24 maps)

Across-scale combinations and normalization

- Conspicuity maps

Linear combinations

Saliency map

Winner-take-all

Inhibition of return

Attended location

SINA 08/09
Example 2 (Orabona et al. 2005)

Input image

Saliency Map

Edges

Color quantization

R+G-
G+R-
B+Y-

SINA 08/09