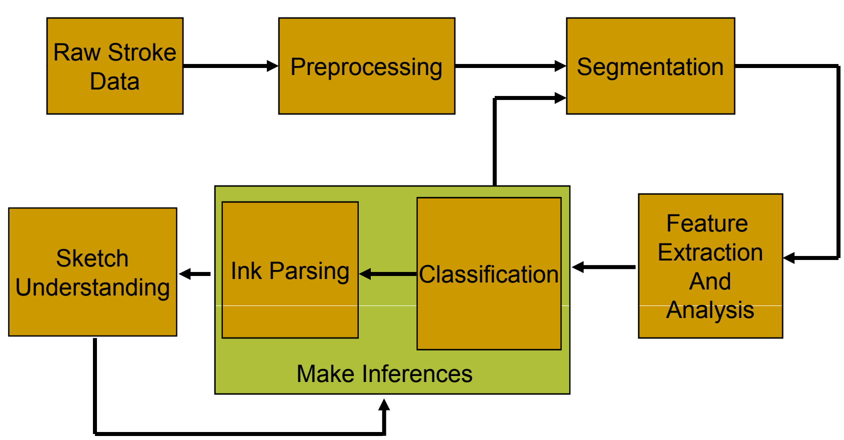
Features Extraction for Sketch-Based Recognition

Lecture #9: Feature Extraction
Joseph J. LaViola Jr.
Fall 2009

Recall Pen-Based Interface Dataflow



Feature Extraction and Analysis

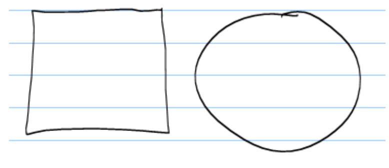
- What came first, the feature or the machine learning algorithm?
- Want to distinguish sketch components from one another
- Good features are critical
- Extract important information
  - geometrical, statistical, contextual
- Examples include
  - arc length, histograms, cusps, aspect ratio
  - self-intersections, stroke area, etc…

Finding Features

- Challenging problem
  - need fast algorithms for gathering information
  - features must be good discriminators
- Often trial and error
- Can be domain specific

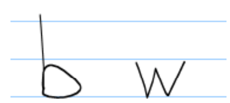
Geometric Features (1)

- Number of strokes
  - if you know how many strokes a symbol has, you can break up your recognizer into pieces (i.e., recognizer for 1 stroke symbols, recognizer for 2 stroke symbols …)
- Cusps
  - smooth vs. jagged strokes
  - distance between cusps
    - useful for when cusps are close together/far apart

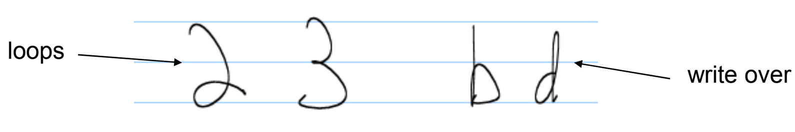


Geometric Features (2)

- Aspect ratio (width / height)
  - tall vs. flat



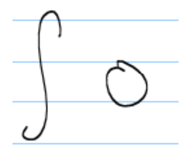
- Self Intersections
  - loops vs. no loops
  - strokes with write over
  - distance between self intersections also useful
  - use line segment intersection algorithm



Geometric Features (3)

- First and last distance
  - Strokes where first and last points are close together vs. far apart
  - simple computation – $\|p_n - p_1\|$
- Arc length
  - many different symbols have varying arc lengths
  - simple computation as well –

$$l = \sum_{i=2}^{n} \|p_i - p_{i-1}\|$$



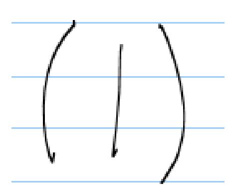
Geometric Features (4)

- Stroke area
  - area defined by the vectors created with the initial stroke point and consecutive stroke points.
  - good discriminator for straight vs. curved lines

Given $\vec{u}_i = p_{i+1} - p_1$ and $\vec{v}_i = p_{i+2} - p_1$

$$s_{area} = \sum_{i=1}^{n-2} \frac{1}{2} (\vec{u}_i \times \vec{v}_i) \cdot \mathrm{sgn}(\vec{u}_i \times \vec{v}_i)$$

where $\vec{u}_i \times \vec{v}_i$ is a scalar



Geometric Features (5)

- Fit line feature
  - sophisticated approach to finding how close a stroke is to a straight line
  - finds a least-squares approximation to a line using principal components and then uses this approximation to find the distance of the projection of the stroke points onto the approximated line
  - outputs a value in [0,1]
- What is another name for this approach?

Fit Line Feature Implementation

Input: A set of stroke points P.
Output: A distance measure
$\text{FitLine}(P)$

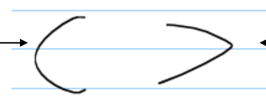
$x_1 \leftarrow \sum_{i=1}^{n} X(P_i)$
$y_1 \leftarrow \sum_{i=1}^{n} Y(P_i)$
$x_2 \leftarrow \sum_{i=1}^{n} X(P_i)^2$
$y_2 \leftarrow \sum_{i=1}^{n} Y(P_i)^2$
$xy_1 \leftarrow \sum_{i=1}^{n} X(P_i)Y(P_i)$
$x_3 \leftarrow x_2 - x_1^2/n$
$y_3 \leftarrow y_2 - y_1^2/n$
$xy_2 \leftarrow xy_1 - (x_1 y_1)/n$
$rad \leftarrow \sqrt{(x_3 - y_3)^2 + 4xy_2^2}$
$error \leftarrow (x_3 + y_3 - rad)/2$
$rms \leftarrow \sqrt{error/n}$
if $x_3 > y_3$
  $a \leftarrow -2xy_2$
  $b \leftarrow x_3 - y_3 + rad$
else if $x_3 < y_3$
  $a \leftarrow y_3 - x_3 + rad$
  $b \leftarrow -2xy_2$
else
  if $xy_2 = 0$
    $a \leftarrow b \leftarrow c \leftarrow 0$
    $error \leftarrow +\infty$
  else
    $a \leftarrow 1$
    $b \leftarrow -1$
$mag \leftarrow \sqrt{a^2 + b^2}$
$c \leftarrow \frac{(-ax_1 - by_1)/n}{mag}$
$a \leftarrow \frac{a}{mag}$
$b \leftarrow \frac{b}{mag}$
$min_1 \leftarrow +\infty$
$max_1 \leftarrow -\infty$
for i=1 **to** n
  $err \leftarrow aX(P_i) + bY(P_i) + c$
  $pX \leftarrow X(P_i) - a \cdot err$
  $pY \leftarrow Y(P_i) - b \cdot err$
  $ploc \leftarrow -b \cdot pX + b \cdot pY$
  $min_1 \leftarrow \min(min_1, ploc)$
  $max_1 \leftarrow \max(max_1, ploc)$
return $\frac{100 \cdot rms}{max - min}$

Fall 2009 CAP 6105 – Pen-Based User Interfaces ©Joseph J. LaViola Jr.

Statistical Features (1)

- Side ratios
  - first and last point of strokes have variable locations with respect to the bounding box
  - Approach
    - take the x coordinates of the first and last point of a stroke
    - subtract them from the left side of the symbol's bounding box (i.e., the bounding box's leftmost x value)
    - divide by the bounding box width.

1st and last point on right side of bbox →

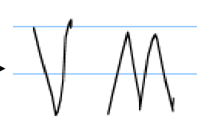


← 1st and last point on left side of bbox

Statistical Features (2)

- Top and Bottom ratios
  - similar to side ratios except we are dealing with *y* coordinate
  - approach
    - take y coordinate of the first and last point of a stroke
    - subtract from the top of the symbol's bounding box (i.e., the bounding box's topmost y value)
    - these values are divided by the bounding box height.

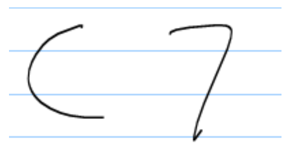
1st and last points close to top of bbox →



← 1st and last points are at the bottom of bbox

Statistical Features (3)

- Point Histogram
  - distribution of point locations in stroke bounding box
  - discrimination where point concentrations are high
  - approach
    - break up box into *n x m* grid
    - Count number of points in each sub box
    - divide by total number of points



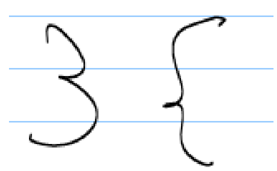
Statistical Features (4)

- Angle Histogram
  - similar to point histogram except dealing with angles
  - Approach

$$\text{Given } \vec{\mathrm{v}}_{\mathrm{j}} = p_i - p_{i-1} \text{ for } 2 \le i \le n \text{ and } \vec{x} = (1,0)$$

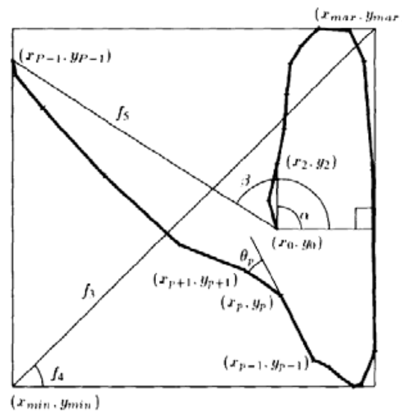
$$\alpha_j = \arccos\left(\vec{x} \cdot \frac{\vec{\mathrm{v}}_{\mathrm{j}}}{\left\| \vec{\mathrm{v}}_{\mathrm{j}} \right\|} \right)$$

- put angles into bins of *n* degrees



The Rubine Feature Set (Rubine 1991)

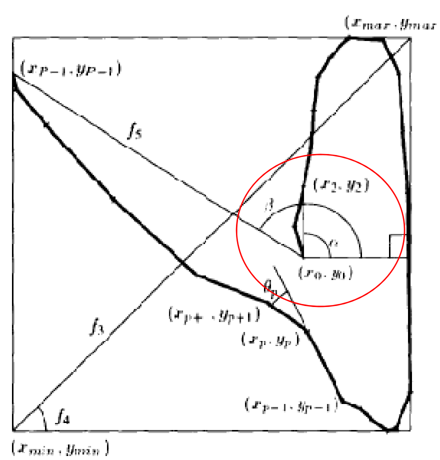
- Part of Rubine's gesture recognition system
  - we will see this next class
- Stroke
  - *P* = total number of points
  - *p* = middle point
  - first point (x_0, y_0, t_0)
  - last point $(x_{P-1}, y_{P-1}, t_{P-1})$
  - compute x_{min}, y_{min}, x_{max}, y_{max}



Feature f_1

- Cosine of starting angle

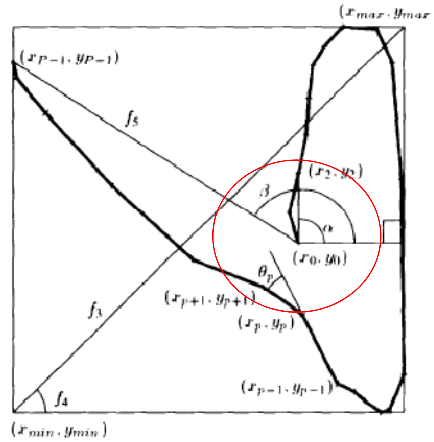
$$f_1 = \cos(\alpha) = \frac{(x_2 - x_0)}{\sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2}}$$



Feature f_2

- Sine of starting angle

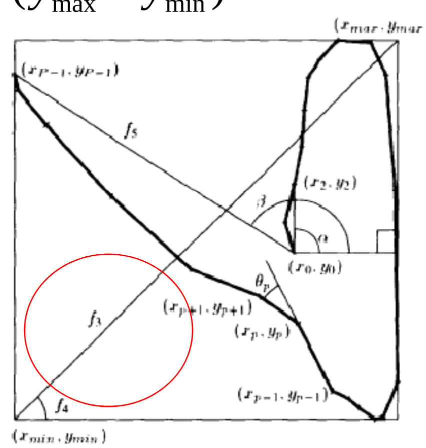
$$f_2 = \sin(\alpha) = \frac{(y_2 - y_0)}{\sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2}}$$



Feature f_3

$$f_3 = \sqrt{(x_{max} - x_{min})^2 + (y_{max} - y_{min})^2}$$

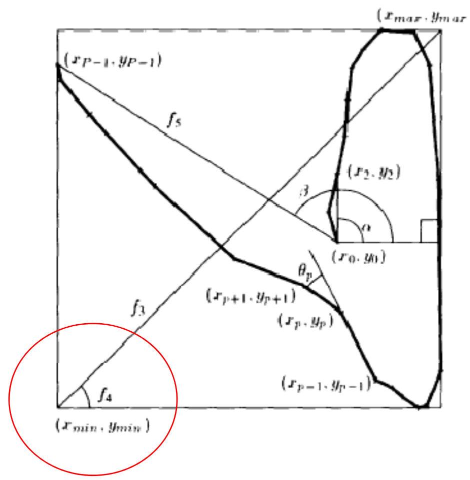
- Length of diagonal of bounding box (gives an idea of the size of the bounding box)



Feature f_4

- Angle of diagonal
- gives an idea of the shape of the bounding box (long, tall, square)

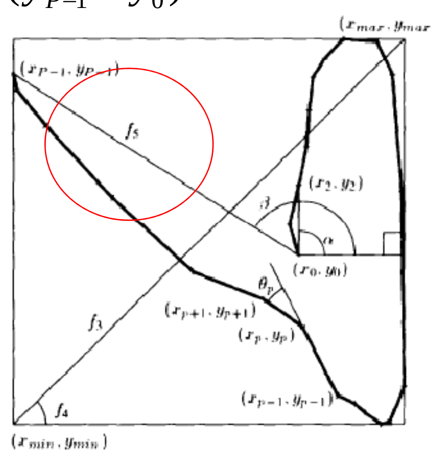
$$f_4 = \arctan\left(\frac{y_{\max} - y_{\min}}{x_{\max} - x_{\min}}\right)$$



Feature f_5

$$f_5 = \sqrt{(x_{P-1} - x_0)^2 + (y_{P-1} - y_0)^2}$$

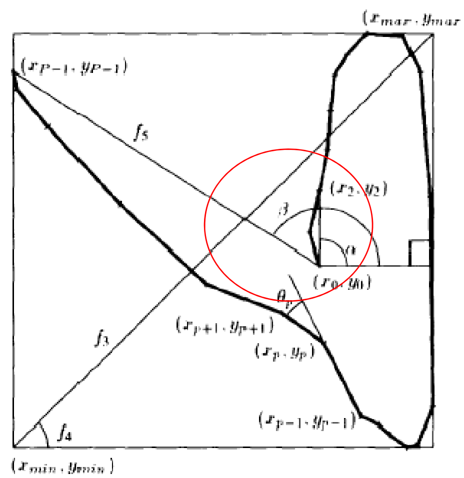
- Distance from start to end of stroke



Feature f_6

- Cosine of ending angle

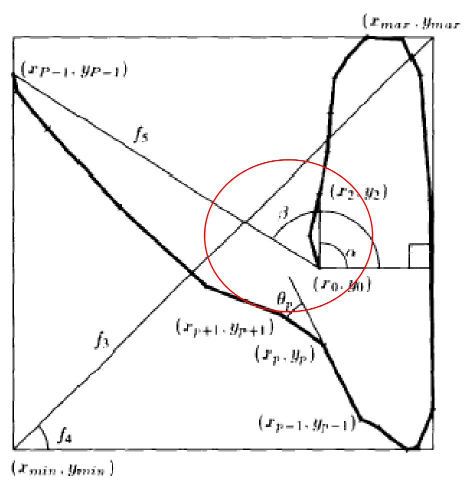
$$f_6 = \cos(\beta) = \frac{(x_{P-1} - x_0)}{f_5}$$



Feature f_7

- Sine of ending angle

$$f_7 = \sin(\beta) = \frac{(x_{P-1} - x_0)}{f_5}$$



More Definitions (before we continue)

$$\text{Let } \Delta x_p = x_{p+1} - x_p \text{ and } \Delta y_p = y_{p+1} - y_p$$

$$\text{Let } \theta_p = \arctan \frac{\Delta x_p \Delta y_{p-1} - \Delta x_{p-1} \Delta y_p}{\Delta x_p \Delta x_{p-1} + \Delta y_p \Delta y_{p-1}}$$

Directional angle

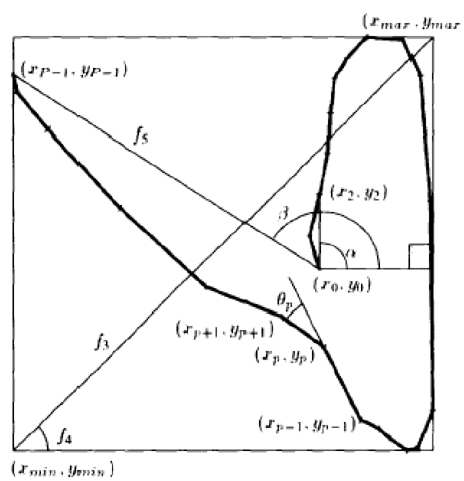
$$\text{Let } \Delta t_p = t_{p+1} - t_p$$

Time delta

Feature f_8

- Total stroke length

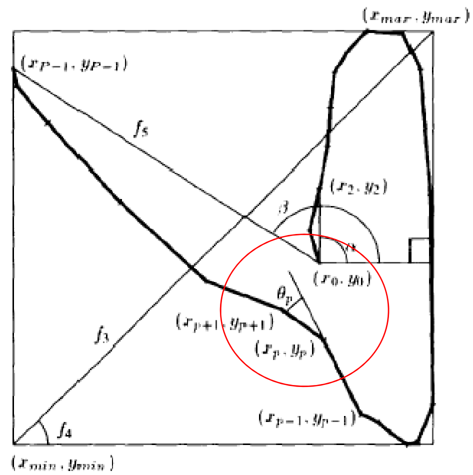
$$f_8 = \sum_{p=0}^{P-2} \sqrt{\Delta x_p^2 + \Delta y_p^2}$$



Feature f_9

- Total rotation (from start to end point)
- (not the same as β-α – think of spirals)

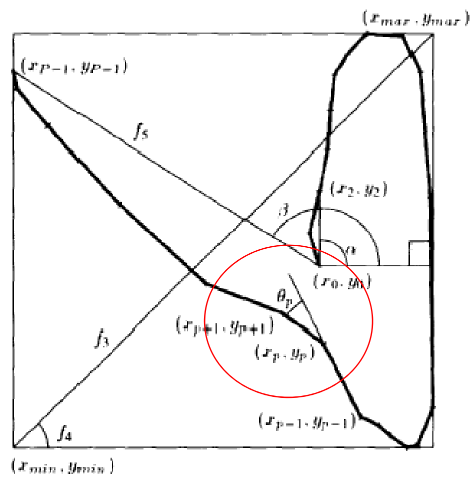
$$f_9 = \sum_{p=1}^{P-2} \theta_p$$



Feature f_{10}

- Absolute rotation
- How much does it move around

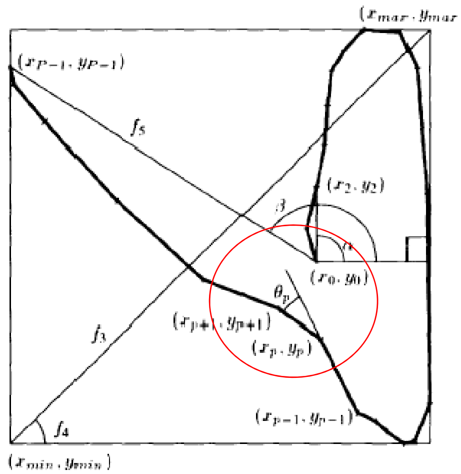
$$f_{10} = \sum_{p=1}^{P-2} |\theta_p|$$



Feature f_{11}

- Rotation squared
- How smooth are the turns?
- Measure of sharpness

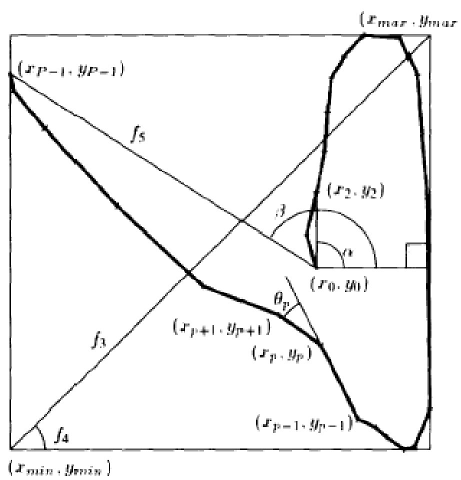
$$f_{11} = \sum_{p=1}^{P-2} \theta_p^2$$



Feature f_{12}

- The maximum speed reached (squared)

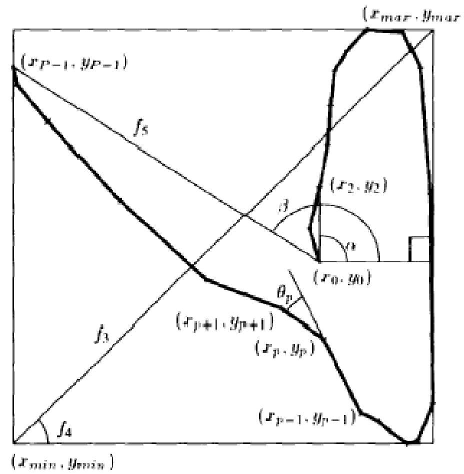
$$f_{12} = \max_{p=0}^{P-2} \frac{\Delta x_p^2 + \Delta y_p^2}{\Delta t_p^2}$$



Feature f_{13}

- Total time of stroke

$$f_{13} = t_{P-1} - t_0$$



Next Class

- Start discussing machine learning algorithms
  - linear classifiers (e.g., Rubine)
  - template matching
  - SVM
  - AdaBoost
  - etc…