

Ink Preprocessing and Preparation

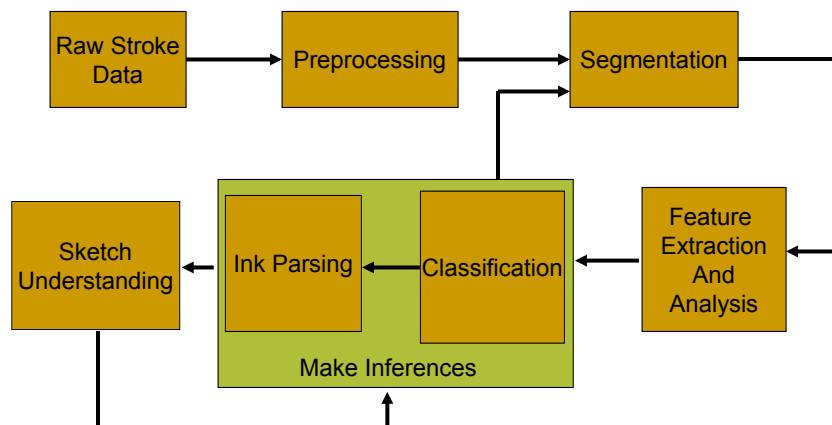
Lecture #5: Preparing Ink
Joseph J. LaViola Jr.
Fall 2012

Fall 2012

CAP 6105 – Pen-Based User Interfaces

©Joseph J. LaViola Jr.

Recall Pen-Based Interface Dataflow



Fall 2012

CAP 6105 – Pen-Based User Interfaces

©Joseph J. LaViola Jr.

Representing Data

- Points and strokes

$$s = p_1 p_2 \dots p_n$$

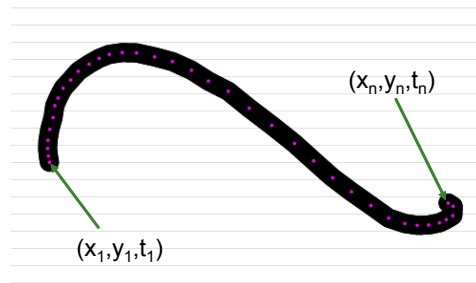
where

$$p_i = (x_i, y_i, t_i), 1 \leq i \leq n$$

$$S = s_1 s_2 \dots s_m$$

- Image

- pixel matrix
- not as popular



Fall 2012

CAP 6105 – Pen-Based User Interfaces

©Joseph J. LaViola Jr.

Preprocessing

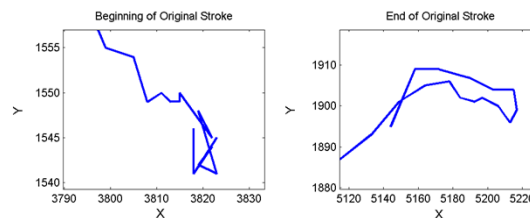
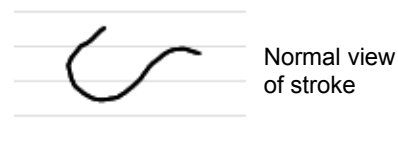
- Often required to clean raw data

- Stroke Invariance

- scale
- position
- orientation
- slant/skew
- order/direction

- Filtering and Smoothing

- Dehooking



Zoomed in view of stroke showing unwanted cusps and self-intersections

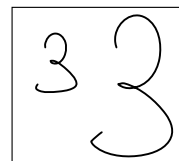
Fall 2012

CAP 6105 – Pen-Based User Interfaces

©Joseph J. LaViola Jr.

Scale Invariance

- Why? – want to ensure stroke has a canonical representation so its *size* makes no difference in recognition
- Approach
 - define constant width or height
 - scale stroke maintaining aspect ratio
 - choose constant width or height based on stroke

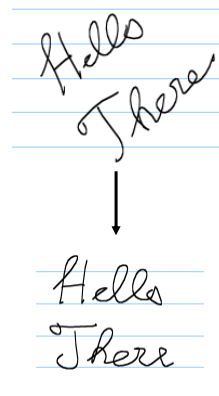


Translation Invariance

- Why? – want to ensure stroke has canonical representation so its *position* makes no difference in recognition
- Approach
 - translate stroke to origin
 - use stroke bounding box
 - possible translation points
 - top left point
 - center point

Rotation Invariance

- Primarily used when for handwriting (sometimes for shapes)
- Why? – want to remove baseline drift which could affect recognition
- Baseline drift – deviation between baseline and horizontal axis
- Difficult problem to deal with
 - ambiguous baseline locations
- One approach (Guerfali and Plamondon 1993)
 - uses center of mass of word regions
 - least squares for baseline construction



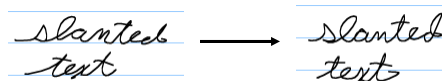
Fall 2012

CAP 6105 – Pen-Based User Interfaces

©Joseph J. LaViola Jr.

Slant/Skew Invariance

- Important in handwriting recognition
- Handwriting slant – deviation between the principal axis of strokes and vertical axis
 - Often referred to as *deskewing process*
- Why? – can be important for segmentation
- Difficult problem – very subjective
- One approach (Guerfali and Plamondon 1993)
 - zone extraction
 - observation windows
 - local and global slants



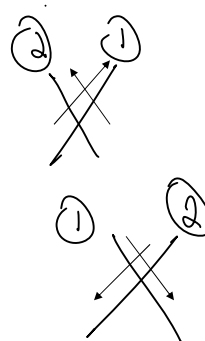
Fall 2012

CAP 6105 – Pen-Based User Interfaces

©Joseph J. LaViola Jr.

Stroke Direction and Ordering Invariance

- Can be large variation in ways a symbol is drawn
 - order of strokes
 - direction of strokes
- Possible approach is to model each possible combination
 - combinatorially expensive
 - could hurt recognition accuracy
- Want to assign canonical ordering and direction
 - see Matsakis (1999)



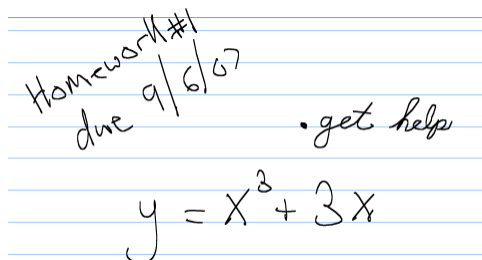
Fall 2012

CAP 6105 – Pen-Based User Interfaces

©Joseph J. LaViola Jr.

Stroke Invariance Summary

- Want to have canonical representation
- Makes calculating features easier
- Makes recognition easier



Fall 2012

CAP 6105 – Pen-Based User Interfaces

©Joseph J. LaViola Jr.

Resampling

- Why? – sometimes we want to have all strokes have the same number of points
 - helps deal with some recognition algorithms
- Approach
 - linear interpolation between points

Filtering and Smoothing

- Remove duplicate points
- Remove unwanted cusps and self-intersections
- Thinning – reduce points
- Dot reduction – reduce dots to single point
- Stroke connection- deal with extraneous pen lifts (e.g., stroke segmentation)

Gaussian Smoothing

$$p_i^{filt} = \sum_{j=-3\sigma}^{3\sigma} w_j p_{j+i}$$

σ is a scaling parameter

Should try to maintain cusps when filtering

$$w_j = \frac{e^{-\frac{j^2}{2\sigma^2}}}{\sum_{k=-3\sigma}^{3\sigma} e^{-\frac{k^2}{2\sigma^2}}}$$

Fall 2012

CAP 6105 – Pen-Based User Interfaces

©Joseph J. LaViola Jr.

A Filtering Algorithm

Input: Stroke s_i and a self-intersection threshold α .

Output: A filtered list of points

```

FILTERSTROKE( $s_i, \alpha$ )
(1)  $P \leftarrow Points(s_i)$ 
(2)  $cur_{pt} \leftarrow P_1$ 
(3) for  $i = 2$  to  $n$ 
(4)   if  $cur_{pt} = P_i$ 
(5)      $BadPts \leftarrow P_i$ 
(6)   else
(7)      $cur_{pt} = P_i$ 
(8)    $RemovePointsFromPointList(BadPts, P)$ 
(9)    $SelfInts \leftarrow SelfIntersectionLocations(P)$ 
(10)   $prev \leftarrow -1$ 
(11)  for  $i = 1$  to  $\|P\|$ 
(12)    if  $prev \neq -1$  and  $SelfInts_i - prev > \alpha$ 
(13)      for  $j = prev$  to  $SelfInts_i$ 
(14)         $BadPts \leftarrow P_j$ 
(15)       $prev \leftarrow SelfInts_i$ 
(16)   $RemovePointsFromPointList(BadPts, P)$ 
(17)  return  $P$ 

```

Fall 2012

CAP 6105 – Pen-Based User Interfaces

©Joseph J. LaViola Jr.

Dehooking

- Want to eliminate hooks that can occur at the end of strokes (sometimes at the beginning)
- Hooks come from
 - inaccuracies in pen-down detection
 - rapid and erratic stylus motion
- Hooks vary depending on user and on stroke

A Dehooking Algorithm

Input: Stroke s_i , minimum and maximum hook threshold $hook_{min}$ and $hook_{max}$, and a dehooking distance threshold ϵ_{hook} .

Output: A dehooked list of points

DEHOOK($s_i, hook_{min}, hook_{max}, \epsilon_{hook}$)

```

(1)   $P \leftarrow Points(s_i)$ 
(2)   $maxdist \leftarrow 0$ 
(3)  for  $i = 2$  to  $\min(hook_{min}, P_n - hook_{max})$ 
(4)     $dist \leftarrow \|P_i - P_1\|$ 
(5)    if  $dist > \epsilon_{hook}$ 
(6)      break
(7)    if  $dist \geq maxdist$ 
(8)       $maxdist = dist$ 
(9)    else
(10)     for  $j = 1$  to  $i$ 
(11)        $BadPts \leftarrow P_j$ 
(12)     break
(13)   $maxdist \leftarrow 0$ 

```


Dehooking Algorithm Cont'd

```

(14)   for  $i = P_{n-1}$  down to  $\max(\text{hook}_{max}, P_n - \text{hook}_{min})$ 
(15)      $dist \leftarrow \|P_n - P_i\|$ 
(16)     if  $dist > \epsilon_{hook}$ 
(17)       break
(18)     if  $dist \geq \text{maxdist}$ 
(19)        $\text{maxdist} = dist$ 
(20)     else
(21)       for  $j = n$  down to  $i$ 
(22)          $BadPts \leftarrow P_j$ 
(23)       break
(24)    $RemovePointsFromPointList(BadPts, P)$ 
(25)   return  $P$ 

```

Next Class – Discussion

- Readings
 - Wolin, A., Eoff, B., and Hammond, T. ShortStraw: A Simple and Effective Corner Finder for Polylines. Eurographics 5th Annual Workshop on Sketch-Based Interfaces and Modeling, Annecy, France, June, 2008, pp. 33-40.
 - Xiong, Y. and LaViola, J. “Revisiting ShortStraw – Improving Corner Finding in Sketch-Based Interfaces”, Proceedings of the Sixth Eurographics/ACM Symposium on Sketch-Based Interfaces and Modeling 2009, 101-108, August 2009.
 - Herold, J. and Stahovich, T. SpeedSeg: A Technique for Segmenting Pen Strokes Using Pen Speed Computers and Graphics, Volume 35, Issue 2, 2011, pp. 250-264