Basics of Sketch Recognition

15-397
Pen Based Computing
Fall 2008
Sketch Recognition vs Voice Recognition

- Many similarities with voice recognition
  - Noisy environments
  - Real time recognition or delayed recognition
  - Word recognition or concept recognition
  - Stroke recognition or sketch recognition
  - Good recognition requires domain understanding
Terminology

Stroke is an Array of points collected between pen down and pen up events, time stamped.

Sketch is an Informal, messy diagram consisting of a collection of strokes.
Basic Shapes

• Easy to recognize regardless of start and end points
• Better recognition if completed using a single stroke
• Use direction, curvature and speed to recognize the edge cases. Eg: Minimum speed, maximal curvature
• Bunch of stuff from SDK to process strokes
Tablet SDK
## Tablet SDK stuff

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rectangle GetBoundingBox()</td>
<td>Gets the rectangle frame surrounding the stroke in <strong>ink</strong> coordinates</td>
</tr>
<tr>
<td>Point[] GetPoints</td>
<td>Gets the set of points making up the stroke</td>
</tr>
<tr>
<td>Move(float xoffset, float yoffset)</td>
<td>Moves a stroke by a particular offset</td>
</tr>
<tr>
<td>Scale(float xscale, floatyscale)</td>
<td>Scales the stroke in 2 dimensions by a pair of factors</td>
</tr>
<tr>
<td>Transform(Matrix m)</td>
<td>Transforms a stroke (scales, rotates, moves a stroke)</td>
</tr>
<tr>
<td>float NearestPoint(Point referencePoint, out distance)</td>
<td>Returns the nearest interpolated stroke index on the stroke to a reference point. Also returns the shortest distance to the point on the stroke</td>
</tr>
<tr>
<td>bool HitTest(Point centerPoint, float radius)</td>
<td>Returns true if the stroke falls within or intersects with a given circle specified by the center and radius</td>
</tr>
<tr>
<td>SetPoint(int index, Point p)</td>
<td>Sets a specific point in the stroke</td>
</tr>
<tr>
<td>float[] FindIntersections(Strokes strokes)</td>
<td>Finds the collection of indexes in the stroke that intersects all the strokes that are passed. If the parameter is null, it looks for all intersections in the Ink</td>
</tr>
</tbody>
</table>

Tablet SDK – Ink Recognition

- Ink is a first class data type
  - Ink remains as ink
- Gesture based command recognizers
  - Ink strokes forming pre-defined shapes
  - Table 7-1 – Page 356
- Recognizer Architecture in SDK
  - Built-in Language recognizers (text)
  - Support for building custom recognizers
Text Recognition Results

- Always not perfect
- Results provided with a confidence level
- Provide UI support to correct
- Segmentation of ink together
- Chinese, Korean, Japanese
  - Word is a set of discrete symbols
  - Exponentially large set of possible combinations
  - Provide alternatives to each symbol (not word)
  - Many challenges
Text versus Object Recognition

- Most recognizers on Tablet SDK are ink to text recognizers
- Object recognition can be aided by context
- A context - relevant, application-specific information provided to a recognizer to improve recognition accuracy
  - Factoids in SDK
    - Number
    - Email
    - Postal code
    - Telephone
  - Microsoft.Ink.Factoid
    - theRecognizerContext.Factoid = Factoid.PostalCode;

- But, ink strokes can also be custom recognized (with special recognizers)
  - Application commands
  - Mathematical formulas
  - Musical notes etc..
Gesture Recognition

- Gesture is one or more ink strokes map into a specific command
  - See page 356

Example:
- iov = new InkCollector(player.Handle, true);
- iov.CollectionMode = CollectionMode.GestureOnly;
- iov.SetGestureStatus(ApplicationGesture.Square, true);
- iov.SetGestureStatus(ApplicationGesture.Circle, true);
Real time and Delayed Recognition

- Synchronous recognition
  - Thread requesting recognition results blocks ink collection until computation is complete

- Asynchronous recognition
  - Thread requesting recognition result is allowed to continue
  - Object recognized at the end
Recognition Techniques

- Define a visual language
  - Use a shape library
  - Use probabilities to reduce ambiguity
  - Bayes theorem

- Recognition can be done
  - After sketching – image recognition
  - During sketching – stroke recognition
Image recognition

- Pattern Recognition is done
  - Prior knowledge
  - Statistical Analysis

- Pattern Recognition System
  - Collect, extract(features), Analyze

- Techniques
  - discriminant analysis
  - feature extraction
  - error estimation
  - cluster analysis
  - grammatical inference and parsing
Recognizer Architecture
Basic Architecture of a Recognizer

Sketch Panel

Stroke listener
Stroke Classifier

Application

Recognized Figure

strokes
Possible Class Designs

- **SketchPanel**
  - Gathers stroke data
  - Displays raw strokes as they are drawn
  - Has methods for adding and removing Stroke data listeners

- **StrokeData**
  - This class holds and computes stroke related information such as points in the stroke, pen speed, curvature
  - The constructor takes an array of points
  - SketchPanel creates this object after each mouse up event
Possible Classes

- **StrokeClassifier**
  - Constructor takes a StrokeData object
  - Has a method `int classify()`
  - This method can returns an int (or any other type) indicating the type of the approximation generated by the classifier

- **Basic Shapes in the classifier**
  - Line
  - Polygon
  - Rectangle
  - Circle
  - Ellipse
  - Path etc..
Rule Based Recognizers (MIT)

- Constrained rule based sketches
- Language for describing shapes

```
(define shape OpenArrow
  (description "An arrow with an open head")
  (components
   (Line shaft)
   (Line head1)
   (Line head2))
  (constraints
   (coincident shaft.pl head1.pl)
   (coincident shaft.pl head2.pl)
   (coincident head1.pl head2.pl)
   (equal-length head1 head2)
   (acute-meet head1 shaft)
   (acute-meet shaft head2))
  (aliases
   (Point head shaft.pl)
   (Point tail shaft.pl))
  (editing
   (trigger (click_hold_drag shaft))
   (action
     (translate this)
     (set-cursor DRAG)
     (show-handle MOVE tail头)))
   (trigger (click_hold_drag head))
   (action
     (rubber-band this head tail)
     (show-handle MOVE head)
     (set-cursor DRAG)))
   (trigger (click_hold_drag tail))
   (action
     (rubber-band this tail head)
     (show-handle MOVE tail)
     (set-cursor DRAG)))
  (display (original-strokes)))
```

Figure 1: An open arrow.

Figure Courtesy MIT ASSIST
Research in Recognizers
Research in Sketch Recognition

- **Assist – MIT/CSAIL**
  - allowed people to sketch user interfaces and provided a perspective on the importance of sketches in early design.

- **SILK – CMU**
  - an interactive technique known as mediation is used to correct recognition errors.

- **Burlap – Mankoff & Hudson**
  - a sketch-based tool for web designers, takes an alternative approach by minimizing the amount of recognition needed while still retaining sketchy interaction.
Recognizer Correction Strategies

- Mediation
  - Correcting recognition errors at the time of interaction

Figure 3. Burlap is a rougher version of SILK that demonstrates how mediators can be used to interactively deal with recognition errors.

- Human intervention after design
MIT’s Assist Project

- Home of the Physics Illustrator
- Limit the domain to improve recognition
- Interpret strokes while being drawn
- Use contextual knowledge to resolve ambiguities
- Move from **low level – geometric** to **high level – domain specific**
MIT’s Java Toolkit

- Given a freehand stroke, generate a recognized sketch
Exercise
Characterization of Sketch Recognition

- **Segmentation**
  - Task of grouping strokes

- **Classification**
  - Determining which object each set of strokes constitute

- **Labeling**
  - Assigning labels to the recognized figure
    - Head, torso etc.. In stick figure
Improving Accuracy

- View sketching as an incremental process
- Capture individuals preferred stroke order during drawing
Recognition Complexity

- Treating sketches as images
  - Exponential time complexity
- M object classes, each object model with k components
  - Recognizing an object with n strokes
  - $m^* (n \choose k) * k!$
- Exponential time and space requirements
- Polynomial time desirable