Efficient search space exploration for sketch recognition

Tevfik Metin Sezgin and Randall Davis

What: Sketch recognition is a natural form of interaction that has been receiving increasing attention in the research community [4], [1], [6]. One of the challenges in sketch recognition is efficient recognition. We propose to create efficient recognition by treating sketching as a dynamic process and by deriving recognition strategies from the object models specified in the language proposed in [3].

Why: In online sketching, strokes are put on the sketching surface one at a time. We propose a sketch recognition strategy that takes this incremental nature of the sketching process into account. Current approaches to online sketch recognition apply object recognition methods developed for static images to sketches that are formed in a dynamic, incremental fashion. The incremental nature of sketching necessitates a strategy for assigning image features to model features[1] that keeps the number of ambiguous interpretations minimum, or alternatively, the branching factor of the corresponding interpretation tree low[2]. Otherwise, creating all plausible interpretations that can be generated at a given time may result in too many partial interpretations.

How: The approach we take is to analyze object models and derive a recognition strategy specifying the actions to take under different drawing scenarios. The actions that a recognizer can take may include:

- Checking if certain constraints are satisfied between image features
- Creating a partial interpretation by assigning an image feature to a model feature
- Delaying the partial interpretation creation until more strokes are drawn by the user

The kinds of analysis include inspection of object descriptions, as well as simulations with hand drawn instances of the object. We conducted a simulation with a hand drawn example to serve as a proof of concept experiment. Our goal was to check if there were ways of filling in image features that minimized the number of partial interpretations created by the time the object is completely drawn. The recognition strategy used in our experiment is a variant of the interpretation tree algorithm. After each stroke is drawn, for each type of object to be recognized:

- If there are no partial interpretations from the object class we are trying to recognize, and if the geometric primitive derived from the latest stroke fits into a model feature without violating any constraints, create a new partial interpretation with that primitive assigned to the model feature.
- If there are existing partial interpretations that can be extended with the latest primitive without violating any constraints, these interpretations are cloned and extended.

We implemented a stick-figure recognizer using this strategy. We recorded raw strokes for a stick-figure (total of 6 strokes) and added the strokes to the sketching surface in different drawing orders, to simulate different ways in which an object can be drawn. Fig. 1 shows the costs for different orders, sorted in an ascending order.

Observations As seen in the figure, the cost (i.e., the number of instantiated IT nodes) will vary depending on the drawing order. An example illustrates why this happens: For a stick figure, if we start with two touching lines, they could potentially be pairs of arms, legs, or the body along with any of the other limbs. On the other hand, if we have an oval touching a line, these strokes can only be the head and the body. In one case there are multiple ways in which two strokes can be labeled, while in the other the labeling is unique.

In the light of the above observation, an efficient recognition strategy would be to *delay the assignment of the image features to the model features until the user draws something that can be interpreted as the head*. This approach is also in tune with the concept of *key* or *anchor* components used in the object recognition literature where the goal is to identify unique components or features of the object model that are robust under noise, occlusion, and viewpoint variations and initiate recognition from these components. We believe techniques from decision theoretic, non-myopic approaches to search, POMDPs and planning will be useful in deriving such strategies.

[1] In the computer vision literature, model feature refers to a component of a particular object model. For example, a plus sign has two model features corresponding to the intersecting horizontal and vertical lines. Primitives derived from the user drawn strokes are named image features and they are detected using the toolkit described in [5].

[2] Details of the IT approach for object recognition are found in [2].

```
(define Stick-Figure
 (components
   (Ellipse head)
   (Line body)
   (Line leftArm)
   (Line rightArm)
   (Line leftLeg)
   (Line rightLeg))
 (constraints
   (meets leftArm rightArm leftArm.p1 rightArm.p1)
   (meets leftLeg rightLeg leftLeg.p1 rightLeg.p1)
   (left leftArm     rightArm)
   (left leftLeg     rightLeg)
   (touches leftArm  body leftArm.p1)
   (touches rightArm body rightArm.p1)
   (meets leftLeg    body leftLeg.p1 body.p2)
   (meets rightLeg   body rightLeg.p1 body.p2)
   (touches body     head body.p1)))
```

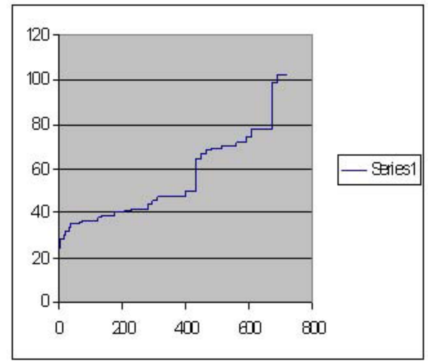

Figure 1: Stick-Figure, object model, and the number of partial interpretations generated during recognition of stick-figures drawn with different stroke orderings, sorted in ascending order. The y-axis shows the cost.

Progress and Future: We are currently working on extending this analysis for object constraints. Above, we illustrated that building partial interpretations starting with the key components yields fewer partial interpretations. Similarly we believe that satisfaction of a constraint or group of constrains can be indicative of an unambiguous partial interpretation. An analysis similar to the one applied to object components can be done for object constraints. This analysis would yield groups of constraints that, when satisfied, signal a good time to instantiate partial interpretations.

Research Support: This work is supported in part by the MIT/Microsoft iCampus initiative and in part by MIT's Project Oxygen.

References:

[1] Christine Alvarado, Mike Oltmans, and Randall Davis. A framework for multi-domain sketch recognition. *AAAI Spring Symposium: Sketch Understanding*, 2002.

[2] W. Eric L. Grimson. The combinatorics of heuristic search termination for object recognition in cluttered environments. *IEEE Pattern Analysis and Machine Intelligence, 13(9), 920–935, 1991*, 1991.

[3] Tracy Hammond. A domain description language for sketch recognition. *Proceedings of 2002 SOW*, 2002.

[4] James A. Landay and Brad A. Myers. Sketching interfaces: Toward more human interface design. *IEEE Computer, vol. 34, no. 3, March 2001, pp. 56-64.*, 2001.

[5] Tevfik Metin Sezgin, Thomas Stahovich, and Randall Davis. Sketch based interfaces:early processing for sketch understanding. *Proceedings of PUI-2001, November 2001*, 2001.

[6] James Landay Thomas Stahovich and Chairs. Randall Davis. A framework for multi-domain sketch recognition. *AAAI Spring Symposium: Sketch Understanding March 25-27, Stanford CA*, 2002.