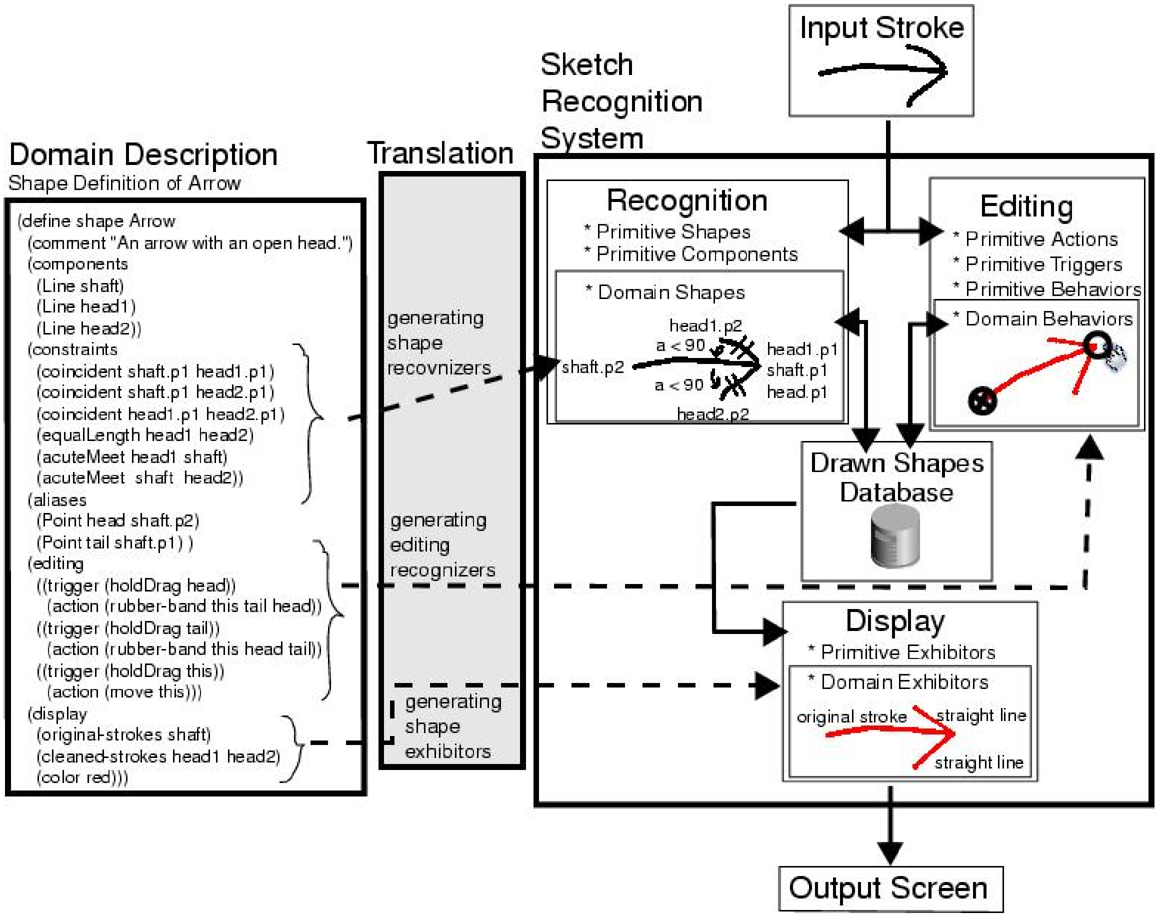
Testing Shape Descriptions by Automatically Translating them for Use in Sketch Recognition

Tracy Hammond and Randall Davis

What: We present a translator which takes symbolic descriptions of how shapes are drawn, displayed, and edited in a domain and automatically translates them into shape recognizers, editing recognizers, and shape exhibitors for use in our domain independent sketch recognition system. The implementation of this translator and domain independent sketch recognition system serves to show both that such a framework is feasible and that LADDER [1] is an acceptable language for describing domain information. The aim of our implementation is its simplicity in design.

Why: To date, sketch recognition systems have been domain-specific, with the recognition details of the domain hard-coded into the system. Developing such a sketch interface is a substantial effort. We propose that rather than build a separate recognition system for each domain, we instead build a single domain independent recognition system that can be customized for multiple domains. To build a sketch recognition system for a new domain, the developer would need only to write a domain description, describing how domain shapes are drawn, displayed and edited, for use with our domain independent sketch recognition system. This domain description would then be translated into shape recognizers, editing recognizers, and shape exhibitors for use in a domain independent sketch recognition system. The inspiration from such a framework stems from the speech recognition domain which has been using this approach with some success [2].



```
(define shape Arrow
  (comment "An arrow with an open head.")
  (components
    (Line shaft)
    (Line head1)
    (Line head2))
  (constraints
    (coincident shaft.p1 head1.p1)
    (coincident shaft.p1 head2.p1)
    (coincident head1.p1 head2.p1)
    (equalLength head1 head2)
    (acuteMeet head1 shaft)
    (acuteMeet  shaft  head2))
  (aliases
    (Point head shaft.p2)
    (Point tail shaft.p1) )
  (editing
    ((trigger (holdDrag head))
      (action (rubber-band this tail head))
    ((trigger (holdDrag tail))
      (action (rubber-band this head tail))
    ((trigger (holdDrag this))
      (action (move this)))
  (display
    (original-strokes shaft)
    (cleaned-strokes head1 head2)
    (color red)))
```

Figure 1: Framework Overview showing LADDER Domain Description, Translator, and Domain Independent Sketch Recognition System.

How: Our goal is to make development of a sketch recognition system easier, by enabling domain experts (rather than programmers) to describe the shapes to be recognized. Figure 1 gives an overview of our framework consisting of: 1) a sketch description language, LADDER, 2) a translator that converts a domain description into components for use by the domain independent sketch recognition system, and 3) a domain independent sketch recognition system that uses the newly generated components to recognize, edit, and display shapes in the domain.

To create the domain specific sketch recognition system, the developer writes a LADDER domain description consisting of multiple shape definitions. The left box of Figure 1 gives an example of an Arrow shape definition. The components and the constraints define how the shape is drawn and are translated into shape recognizers. The display section specifies how the shape is to be displayed when recognized and is translated into shape exhibitors. The editing section specifies the editing behaviors that can be performed on the recognized shape and are translated into editing recognizers. LADDER supplies a number of predefined shapes, constraints, display methods, and editing behaviors. These predefined elements are hard-coded into the domain independent recognition system, allowing it to recognize, display, and edit these predefined shapes (line, ellipse, curve, arc, point, etc.).

The translation process (the middle box of Figure 1) parses the shape definitions and generates code specifying how to recognize shapes and editing triggers as well as how to display the shapes once they are recognized and what action to perform once an editing trigger occurs. The translator creates additional domain shape recognizers, domain editing recognizers, and domain shape exhibitors for use by the domain independent sketch recognition system.

The third box of Figure 1 shows the domain independent sketch recognition system. Recognition is carried out as a series of bottom up opportunistic data driven triggers where the critical event is drawing or editing. When a stroke is drawn, the system first examines the drawn shapes database and checks if the gesture drawn is an editing trigger for any shape. If the stroke is not an editing gesture, it must be a drawing gesture, and thus the system preprocesses the stroke into a collection of primitives. The primitives are added to the drawn shapes database, and the recognition module examines the drawn shapes database to attempt to build higher order shapes. The display module then displays the viewable shapes as defined by the domain description or as per their primitives and displayed onto the screen.

Progress: We have created 1) a sketch description language, LADDER, 2) a translator that converts a domain description into components for use by the domain independent sketch recognition system, and 3) a domain independent sketch recognition system that uses the newly generated components to recognize, edit, and display shapes in the domain.

Future: We are trying to make LADDER as intuitive as possible. However, shape definitions can often be difficult to describe textually as it is much more intuitive to draw a shape rather than type out a verbal description. Thus we would like to build a user interface to aid sketch interface designers in writing and debugging descriptions.

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

References:

[1] Tracy Hammond and Randall Davis. Ladder: A language to describe drawing, display, and editing in sketch recognition. *Proceedings of the 2003 Internaltional Joint Conference on Artificial Intelligence (IJCAI)*, 2003.

[2] Zue and Glass. Conversational interfaces: Advances and challenges. *Proc IEEE*, pages 1166–1180, 2000.