EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling (2008)
Christine Alvarado and Marie-Paule Cani (Guest Editors)

# CogSketch: Open-domain sketch understanding for cognitive science research and for education

Kenneth Forbus, Jeffrey Usher, Andrew Lovett, Kate Lockwood, and Jon Wetzel

Qualitative Reasoning Group, Northwestern University

**ABSTRACT**

*In this paper, we describe CogSketch, an open-domain sketch understanding system built on the nuSketch architecture. CogSketch captures the multi-modal, unconstrained nature of sketching by focusing on reasoning over recognition. We describe this approach, as well as two application domains for CogSketch: cognitive modeling, and education.*

Categories and Subject Descriptors (according to ACM CCS): I.2.10 [Artificial Intelligence]: Vision and Scene Understanding).

## 1. Introduction

Sketching is a powerful means of reflection and communication. Whether drawing a map, the structure of a complex system, or how a process unfolds, sketching allows us to naturally externalize and communicate ideas. As a central effort in the Spatial Intelligence and Learning Center (SILC) we are creating a sketch understanding system, *CogSketch*, which is being used to explore spatial cognition and learning. CogSketch serves as a cognitive science research instrument in two ways. First, it is being used to model psychological phenomena. Second, it will be used in experiments to collect and analyze human data. These efforts are being used to improve the range of the system's spatial reasoning abilities and cognitive fidelity. They enable the third role of CogSketch: serving as a platform for sketch-based educational software. For many disciplines, learning would be enhanced by students being able to interact with software coaches and tutors via sketching. Consider the potential impact of the One Laptop Per Child project, where inexpensive robust machines for learning are widely available to learners all over the world. Our vision is that, in ten years or less, sketch-based intelligent educational software can be as widely available to students as calculators are today. This paper reports on the work in progress we are doing to make this vision a reality.

We begin by outlining open-domain sketch understanding and the nuSketch architecture. Section 3 provides an overview of how CogSketch works. Section 4 summarizes some simulation results using CogSketch, and Section 5 summarizes our current education efforts, focusing on engineering design. We close by discussing other related work and future plans.

## 2. Open-domain sketch understanding

Most sketch understanding systems treat understanding as a matter of recognizing ink, or ink plus speech, as a member of a limited number of predefined symbols (e.g., [PSC*96][AOD02]). While such systems can provide natural, valuable interfaces to existing software, our goal is fundamentally different. We want to capture the perceptual, spatial, and conceptual understanding that people bring to sketching, so that our software can participate in sketching in ways similar to people. An important insight about human-to-human sketching is that recognition is a catalyst, not a requirement. When people sketch with each other, we often use language to label the intended meaning of a piece of ink (or spaces defined implicitly by the ink). In the *nuSketch architecture* [FFU01], we have used several interface mechanics to provide the equivalent service of conceptually labeling what someone draws, as explained below. This frees us from the limitations of today's recognition technologies, enabling us to focus on visual, spatial, and conceptual understanding. The second important insight is that many of the conceptually relevant relationships in sketches are qualitative. An engineer working out an idea through sketching is focusing on the important features of their design. The step of drawing parts accurately with tolerances comes later, if the conceptual work is successful. Prior nuSketch systems have been used successfully as components of systems to plan military operations [RKF02], learn military knowledge by interacting with experts [FUC03][BBB*03], and learn to solve everyday physical reasoning problems [KFT*05][FUT05].

CogSketch is the latest system based on the nuSketch architecture. The first version is already publicly available,

although it is more AI researcher-friendly than scientist- or student-friendly. We view the three roles of CogSketch (simulation platform, experimental tool, educational platform) as synergistic. Progress in computational modeling will lead to better experimental tools, and will be a necessary part of the science base for creating software coaches and tutors for educational software. Creating educational software will, in turn, lead to new questions about how learners understand domains and how their understanding relies on visual and spatial reasoning. By tackling all three at once, working in close collaboration with psychologists, we think we can make far more progress than working on any of them in isolation.

## 3. How CogSketch Works

CogSketch combines its visual, spatial, and conceptual knowledge about the glyphs in a sketch to create a qualitative, symbolic representation of the sketch itself and what it depicts. This section outlines how.

### 3.1 Core concepts

**Glyphs.** In CogSketch, every user-drawn object in a sketch is a *glyph*. Each glyph has *ink* and *content*. The ink consists of one or more polylines, lists of points representing what the user drew. The content is a symbolic token used to represent what the glyph denotes. Visual relationships are computed over glyphs, and depending on the semantics of the sketch, can lead to inferences as to spatial relationships between the contents of those glyphs.

CogSketch relies on the user to segment their ink into glyphs. They click a button to start drawing a glyph, and click the same button again when the glyph is done. All the ink drawn in between is considered to be part of the glyph. This avoids problems with existing automatic segmentation algorithms, such as time-outs or pen-up events, which our users find problematic. The ink in a glyph can be one stroke or many, connected or disconnected, and even intertwined with the ink of another glyph. This maximizes expressiveness for our users.

CogSketch also relies on the user to indicate the type of the content of the glyph, in terms of concepts in the underlying knowledge base. This is one form of *conceptual labeling*. They select a glyph and use an interface to select one or more concepts from CogSketch's knowledge base. Currently our KB contents are derived from OpenCyc, which contains over 58,000 concepts, plus our own extensions for qualitative and analogical reasoning.

Given the large number of concepts that can be used, many of which have no common agreed-upon visual representation, it is hard to imagine how any purely recognition-based approach could work. Creating natural interfaces for conceptual labeling is one of the important design tradeoffs in nuSketch systems. We have used three techniques, providing different degrees of simplicity and breadth of coverage. (1) *Concept lists* focus on a small subset of the KB contents relevant to a particular application (see Figure 1). The user sees an English string for the concept, and can only pick from that list. This method is designed for student worksheets and some kinds of psychology experiments. (2) *Direct access* lets the user type in any concept name from the KB, using string completion to help. Direct access maximizes flexibility, at the cost of forcing users to understand relevant portions of the KB. This method is fine for cognitive modelers and AI researchers. (3) *Glyph bars* provide pictorial labels for KB concepts, and compositional widgets to let users express a large number of concepts without knowing anything about the underlying KB. For example, nuSketch Battlespace [FUC03] enabled military personnel to select over 800 distinct concepts quickly with little training.

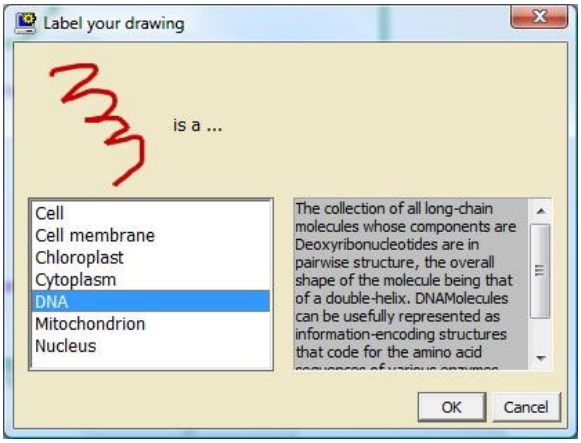


**Figure 1.** *Using a concept list to label glyphs representing parts of a cell.*

CogSketch also includes a mode where the user may type in an unrestricted natural language string as a conceptual label. This is included for gathering data in psychology experiments, and for experimentation with a future natural-language based concept picker.

**Layers, bundles, and the metalayer.** People annotating physical maps or drawings sometimes use acetate sheets as overlays, so they can draw over something without changing it. Most graphics programs exploit this metaphorically, by providing layers as part of their interface. CogSketch also provides layers. Normal layers allow inking, and a special bitmap layer allows users to specify a bitmap that can be drawn over by other layers. Layers share the same coordinate system, but many default CogSketch operations are only done between glyphs on the same layer. Each layer has a *genre* and *pose*, which help CogSketch construct appropriate spatial relationships for the contents from visual relationships between the glyphs. For example, in the abstract genre, the visual relationships between the glyphs (left/right, above/below) provides no information about spatial relationships between their contents (e.g., electronic components in a schematic, elements of a UML diagram). For the physical and geospatial genres, the relationship between visual and spatial relationships also depends on the pose. For example, if glyph A is above glyph B and the genre is physical and the pose is side-view, then A is above B. But

if the genre is geospatial and the pose is top-view, then A is north of B.

Complex sketches often consist of multiple subsketches. For example, in describing a building, one might have a subsketch that shows how it looks from the street, another subsketch representing its floor plan, and a third subsketch that is a schematic of part of its electrical system. Subsketches in CogSketch are represented by *bundles* (so called because each is a collection of layers). A sketch must have at least one bundle, but conceptually there is no upper limit as to how many it can have.

It is important to be able to express relationships between bundles. This is done on the *metalayer*, a special

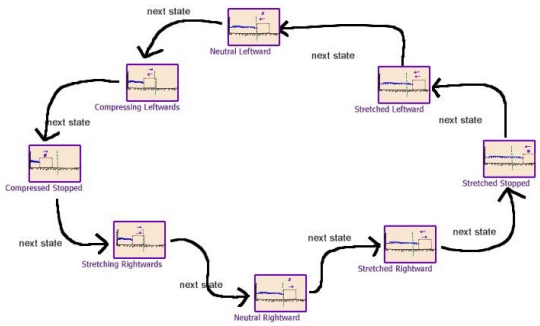


**Figure 2:** *The metalayer can be used to describe complex processes, such as this sequence of states in a spring-block oscillator. Each step of this sequence is a bundle.*

interface where each bundle is treated as if it were a glyph. Arrows can be drawn between these glyphs to describe relationships between bundles. This can be used to describe sequences of states in a complex behavior (e.g., Figure 2), and to represent distinct possible outcomes via *comic graphs* [FUC03].

### 3.2 Visual Processing

**Ink Processing.** CogSketch automatically computes a number of qualitative visual relations and attributes for glyphs in a layer. These represent the general visual features of the sketch, and so they do not make use of any task- or domain-specific knowledge about the objects being sketched. For example, a glyph's size is based on the area of its bounding box. A symbolic description of size, ranging from tiny to huge, is computed by comparing this area to the overall size of the sketch.

CogSketch computes the RCC-8 qualitative relations [Coh96] that describe all possible topological relations between two-dimensional shapes (e.g., *disconnected*, *edge-connected*, *partially-overlapping*). RCC-8 relations are used to guide the generation of other spatial relations. These include positional relations (e.g., above/below, left/right) and containment.

Positional relations are only computed by default between adjacent glyphs, the intuition being that the network of visual relationships we compute should respect the neighborhood structure of the sketch. CogSketch calculates adjacency via Voronoi diagrams [EM98]. There are conditions for which Voronoi adjacency may not accurately reflect psychological judgments of locality, but empirically it has been sufficient for our purposes.

CogSketch also uses RCC-8 relations to identify two types of glyph groups in a sketch: *connected glyph groups* and *contained glyph groups*. A connected glyph group consists of a set of glyphs whose ink strokes intersect. A contained glyph group consists of a single container glyph and the set of glyphs fully contained within it.

**Interactions with Conceptual Knowledge.** Conceptual labelling of glyphs represents one kind of link between the visual and conceptual that CogSketch supports. There are three others: *relation glyphs*, *visual/conceptual relations*, and *annotation glyphs*. We discuss each in turn.

Relation glyphs express a binary relationship between other glyphs. They are created by pressing a different button to indicate that one is drawing a relation glyph. The same interface mechanics are used for labelling relation glyphs, except that the choices are limited to binary relations instead of concepts. If the ink in a relation glyph can be interpreted by CogSketch as an arrow, it looks for the closest glyphs to the head and tail whose contents satisfy the argument type constraints of the relation. The choice CogSketch makes can be overrriden by the user if necessary.

Often the visual relationships between glyphs suggest conceptual relationships that might hold between the objects that they depict. For example, two glyphs that visually touch might suggest that the objects involved are touching, or are connected, hinged, etc., if the objects themselves are rigid physical objects. CogSketch can, on demand, create a list of potential conceptual relationships for pairs of entities that are suggested by the visual relationships between their glyphs and the concepts that they represent. Users can choose which relationship holds, as a way of further informing CogSketch as to their intended meaning.

People use a variety of annotations to highlight particular properties of a situation and to provide domain-specific inputs. For example, when reasoning about a lever, the distance from the fulcrum to the load is very important. Annotation glyphs provide a means of expressing this information. Like other types of glyphs, there is a button which indicates when one is drawing an annotation glyph and when it is finished. The conceptual label for annotation glyphs is tightly constrained, e.g., marking distances, angles, and arrows for forces and torques. The ink of an annotation glyph is interpreted depending on the type of glyph. For example, the position of the head of a force arrow indicates where the force is being applied, and the orientation of the arrow's shaft indicates the direction in which the force is applied.

**Shape Decomposition & Comparison.** CogSketch can also analyze an individual glyph's form. This is accomplished by automatically segmenting a glyph into

edges and computing a set of qualitative relations between the edges. These include relations describing relative length, relative orientation (*parallel* or *perpendicular*), and characteristics of the corners between edges (*convex* or *concave*). An edge-focused representation can stand on its own. Alternatively, it can be used to compare two glyphs, determine the corresponding edges in the glyphs, and identify *shape relations* between glyphs. The simplest shape relation is same-shape, which indicates that two glyphs are both the same shape (e.g., they are both rectangles). Other shape relations describe transformations between shapes, such as rotations and reflections.

It is not always clear when a user wants to focus on the relations between glyphs and when the user wants to focus on the individual edges within a glyph. Therefore, shape decomposition is turned off by default. However, the user can specific that he or she wants to focus on edges within a glyph by creating a special sketch type, the *Perceptual Sketchpad.* In addition, several of the cognitive simulations described in Section 4 make use of edge-focused representations.

### 3.3 Analogical Comparison

Analogical mapping is built into CogSketch. Any two layers or bundles can be compared. Comparisons are carried out using the structure-mapping engine (SME) [FFG86][FO90]. SME is based on Gentner's [Gen83] structure-mapping theory of analogy. In structure-mapping, analogy and similarity are defined in terms of a structural alignment process operating over structured, relational representations. SME takes as input two cases, a *base* and a *target*. It produces as output between one and three *mappings* describing the comparison between base and target. Each mapping consists of: (1) *correspondences* between elements in the base and elements in the target; (2) a *structural evaluation score*, a numerical characterization of how similar the base and target are; and (3) *candidate inferences*, conjectures about the target made by projecting partially-mapped base structures. Candidate inferences can be run in both directions (from the base to the target and from the target to the base) and can be used to describe differences between the two cases. Thus, they complement the correspondences, which describe commonalities between the cases. There is considerable psychological evidence supporting structure-mapping theory as a model of analogy and comparison in humans, and SME has been used to successfully model a variety of psychological phenomena. This is important for CogSketch, since we want to maximize cognitive fidelity. While obviously important for cognitive simulation, we believe this will also be crucial for education applications: If two sketches look alike to the student, they should look alike to the software, and vice-versa, modulo expertise effects.

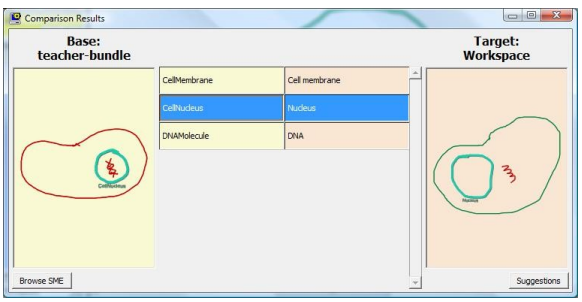


**Figure 3.** *The analogy results dialog, showing the corresponding glyphs in two sketches of a cell.*

**Comparing Sketches with SME.** CogSketch provides a simple interface for comparing two layers or two bundles using SME. Once the comparison has been completed, CogSketch presents a dialog displaying the mapping (see Figure 3). The results dialog shows which glyphs have been placed in correspondence and how much each correspondence contributes to the overall strength of the mapping (the structural evaluation score). Technically-minded users can choose to view more detailed mapping results in a web browser

### 3.4 Interface Design

The three intended roles for CogSketch (cognitive simulation platform, experimental data collection/analysis tool, eduational software platform) have extremely different interface requirements, beyond just the choice of mechanic for conceptual labelling. However, since most of our work so far has focused on cognitive simulation, that is what the current interface is optimized for. The software is architected to support alternate interfaces, which are being designed and developed as part of our ongoing collaborations with psychologists, learning scientists, and instructors. Our goal is to create authoring environments that enable others to quickly customize elements from a library of interfaces for their own needs, but that is at least two years away at this point.

## 4. CogSketch as Cognitive Simulation

A number of psychological studies and intelligence tests involve presenting people with a visual scene and asking them to make a choice, e.g., picking out the best image to complete a visual pattern, or choosing a label to describe what is being shown. Because CogSketch automatically generates qualitative spatial representations of sketches, which we believe are similar to those computed by humans, it can be used in cognitive simulations of these types of tasks. It can be used in conjunction with a reasoning engine, such as SME, to model a task from end to end, that is, from encoding of the visual stimuli to reasoning over the stimuli and selecting the best response. The model of encoding—CogSketch—and the model of reasoning constrain each other, in that CogSketch must generate representations that the reasoner can use. In this way, we can use CogSketch to enhance our understanding of each of the processes being modeled.

Next we describe two cognitive simulations which have been performed using CogSketch: geometric analogies and spatial language learning. Then we summarize other simulation efforts in progress.

### 4.1 Geometric Analogy

One basic measure of human intelligence is the ability to answer analogy problems, i.e., problems of the form A : B :: C : ? ("A is to B, as C is to...?"). In his seminal work, Evans [Eva68] wrote a program to solve geometric analogy problems of this form (see Figure 4). In an effort to better understand how people solve problems of this form, we built a model which combined automatic sketch representation with analogic reasoning [TLF*05].

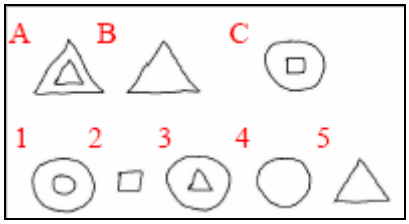


**Figure 4.** *Sample geometric analogy problem.*

The key insight of our model is that these problems can be solved through a process of *two-stage structure mapping*. In the first stage, the model compares image A to image B to compute Δ(A,B), a representation of the differences between images A and B. Δs are based on candidate inferences, part of the mapping produced by SME when it compares two cases. Similarly, the model compares image C to each of the five possible answers to produce a Δ(C,x) for each answer.

In the second stage, the Δs produced by SME in the first stage are fed back through SME for a second comparison. In this stage, Δ(A,B) is compared to the Δ's for each of the five possible answers. At this stage, the measure of interest is SME's structural evaluation score, a measure of the similarity between the representations being compared. The goal is to pick out the answer x for which Δ(A,B) is most similar to Δ(C,x). In other words, this is the answer such that the differences between image A and image B are most similar to the differences between image C and this answer.

Our model successfully solved all 20 of the problems used by Evans to evaluate his original system. Evans' original goal was to show that the problems could be solved by a machine. Our model goes further, by showing that models of component cognitive processes can be combined into a model of a larger-scale cognitive task.

### 4.2 Spatial Language Learning

Spatial relationships play an important role in many reasoning tasks, such as navigation and solving physics/engineering problems. Because space is such an important component of so many tasks, humans have developed specialized language for describing spatial relationships (i.e. prepositions such as *in* and *on*). Ideally, intelligent systems would be able to understand and use spatial language in their interactions with human users, particularly when doing visual-spatial tasks.

We are using CogSketch along with SEQL [KFG*00], an analogical generalization algorithm, to simulate spatial preposition learning. Given a set of cases represented as sets of predicate calculus facts, SEQL divides them into generalizations (categories) based on similarity using structure mapping. The output of SEQL is a set of generalizations, each consisting of a list of facts. Each fact has an associated probability based on the number of cases it appears in [HF05]. Cases for this experiment are sketches (e.g., Figure 5) that are recreations of stimuli used in psychology experiments with human participants. Each case contains both geometric information from the ink in the sketches and conceptual knowledge about the objects in them. We use SEQL to learn the categories for spatial prepositions. Further, by examining the facts associated with each category/generalization we can determine which facts were key to the formation of the generalizations. Preliminary results for this work were reported in [LFH*06]. The model can distinguish between *in*, *on*, *above*, *below*, and *left* after being trained on only 10 sketches per preposition, which is several orders of magnitude fewer training examples than prior cognitive models.

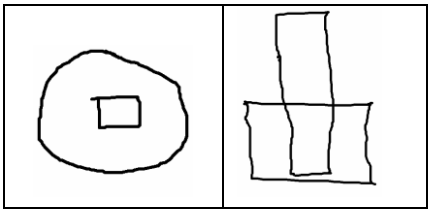

**Figure 5.** *Sample stimuli for the spatial language category learning experiments, both showing examples of "in" relationships.*

### 4.3 Other Simulations

Most of the built-in visual relationship computations in CogSketch focus on relationships between glyphs. In doing the simulations above, we found it was important to incorporate visual analysis of the ink within glyphs. For example, to recognize that two glyphs are instances of the same shape, or that one glyph is a transformation of another (via rotation, reflection, or scaling), requires being able to segment ink within a glyph into pieces and describe their relationships in ways that support matching. Thus, our work on cognitive simulations has driven the development of techniques for edge segmentation and representation, as described in Section 3.2. These techniques, in turn, have led to the ability to simulate a broader range of visual phenomena. For example, in one study [LFU07], CogSketch and SME were used to solve problems from the Raven's Progressive Matrices, a visual pattern completion task which has been used to evaluate general intelligence. The model scores as well as adult Americans on the two sections of the test (out of five) it is currently capable of taking. In another study [LLF08], CogSketch and SEQL are being used to simulate human responses on an Oddity Task, a task in which individuals are shown an array of images and asked to pick the image which does not belong.

The model performs as well as most human subjects and shows an error pattern similar to humans. Additionally, an ablation experiment (on the model) suggests why certain kinds of problems are hard for people.

## 5. CogSketch as educational software platform

The simplest educational application we are exploring are electronic worksheets. Paper-based worksheets are a staple in many classrooms. For example, a student might be asked to draw the structure of something they learned about in class (e.g., the structure of a cell). When the answers can be expressed via qualitative spatial relationships, CogSketch can potentially be used to provide tutoring.

CogSketch includes a prototype worksheet interface. By editing specialized files, curricula can be specified in terms of problems grouped by chapters. For each problem, the worksheet author must provide a problem description, a worked solution, a concept list with English equivalents for conceptual labelling, and a specification of which relationship(s) are important. The student interface displays the problem description and enables students to create a sketch, limited to the list of concepts provided for conceptual labelling. When finished drawing, the student can click the Tutor button to get suggestions. The tutor uses SME to find an analogy between the student sketch and the solution sketch. Differences between the sketches that are related to the important relationships are used to generate suggestions. The current worksheet interface is intended for development purposes, since it exposes the solution sketch when showing the analogy, so it is not ready for classroom use. It is included to support experimentation, and garner feedback that will be used to create a student-friendly version.

A more complex education application we are investigating is engineering design education. An important skill for engineering students to learn is how to communicate. At Northwestern University, 1st and 2nd year engineering students take Engineering Design and Communications, which teaches both skills in an integrated manner. Students working in teams of three or four tackle problems for real clients. Examples include patients at the Rehabilitation Institute who need new tools to help them achieve everyday tasks, like chopping vegetables or trimming their nails, despite physical handicaps. Students build prototypes of their designs to explore particular issues, with regular feedback from potential users. Conversations with instructors revealed that one significant problem they had was helping students learn to use their sketches to communicate ideas, both within the team and to clients. We are creating a CogSketch-based system, the *Design Buddy*, to tackle this problem.

The Design Buddy is intended to be a "crash test dummy" for students to use to practice how to explain a design using a sketch. They will sketch their ideas, explaining the parts, what they are made of, their intended behaviors, and the intended functional roles of the parts. The system will reason through the possible behaviors of the parts itself, based on its understanding of qualitative mechanics, materials, and everyday actions. It will compare its predictions of behaviors with the student's intended behaviors, and ask the student questions about discrepancies. These may include the student not mentioning some critical aspect of the behavior in their explanation, or predicting a behavior that the system does not think is possible. This is a very demanding task, for several reasons: (1) The qualitative mechanics reasoning must be very general and robust. The design projects change constantly, and a wide range of problems arise. (2) The interface must be both sufficiently natural to not be a distraction, and must help the student learn to explain things in terms that practicing engineers would use. (3) The coaching software must have enough strategies to provide students with effective help in learning how to think about their particular design and the design process itself. Currently we are building up the necessary qualitative mechanics reasoning in CogSketch, using ideas from [Nie89][Kim93]. We plan to run initial pull-out experiments with students during the 2008-2009 academic year.

## 6. Related Work

We are inspired in part by Saund and Mahoney's perceptual organization approach to sketching [SMF02], which shows how human-like understanding of ink can lead to more natural editing interactions. We differ from them in our emphasis on adding conceptual understanding into the software as well, and working closely with psychologists to calibrate our visual processing with human data as much as possible. The SketchIt system of Stahovich [SDS00] shares our concern with carrying out qualitative mechanics analyses of sketched devices, but requires users to hand-segment surfaces. That may be reasonable for a professional design tool, but for education we must do this automatically. Newton's Pen [LSP*07], was a physics tutoring system built into a pentop computer. The system provided feedback for students as they were sketching free body diagrams and writing out equations to solve a problem. However, because of the limited processing power of the computer, the researchers were forced to use a simplified sketch understanding system which required users to sketch the parts of their free body diagrams in a particular order.

The Electronic Cocktail Napkin [GD96] was an earlier sketch understanding system meant to facilitate design. Like our system, it was able to decompose glyphs into their component edges. However, it was focused more on learning to recognize the objects represented by glyphs and less on using edge representations to determine how different glyphs' shapes relate to each other.

Several other research groups share our interest in multimodal sketch understanding [AD04][LHK*02]. For example, Adler and Davis' modifications to the ASSIST system [AD04] allowed users to sketch a physical system while verbally describing it. A speech recognition system parsed the description and used the information to refine the sketch (for example, positioning objects such that they are equally spaced in a row). This type of system requires that the designer specify the meaning of the words that are

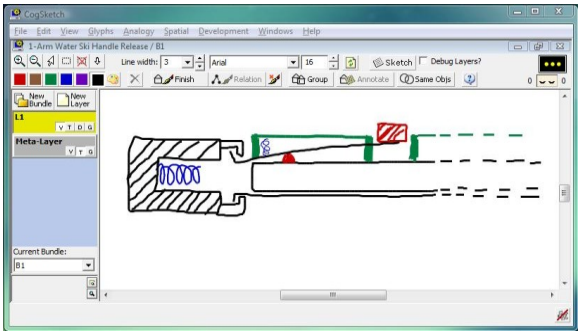

**Figure 6.** *A sketch of a student design for a spring-loaded cap release for a one-handed water-ski handle. Design Buddy will provide suggestions to help the sketch match the student's expressed intent for the design.*

of interest for sketching, thus limiting the system's breadth. In contrast, our system ties in conceptual labels to a large, preexisting knowledge base.

## 7. Discussion and Future Work

CogSketch is an ambitious project, and clearly we are far from achieving our vision. Nevertheless, we are encouraged by our results so far. As Section 4 indicates, we have already successfully simulated a number of psychological findings, which lays some of the groundwork for our education work in progress.

In addition to the simulation studies and education projects outlined above, we are currently working with our collaborators on using CogSketch to gather data from participants in experiments. In addition to being immediately useful for those experiments, this experience will help us better understand how to create easy-to-use sketch-based interfaces for everyday people. The same technology we develop for automatic data scoring in experiments will very likely be adaptable to assessing student performance in educational settings. This is another potential source of synergy that we hope to exploit.

In addition to making the current version of CogSketch publicly available, we are committed to providing updates at least yearly (and perhaps more often, depending on when new features become stable). We are eager for feedback that helps us make CogSketch more usable by the research community and by educators. To support AI researchers, for example, CogSketch has an API that provides access to all of its visual processing and reasoning capabilities. Community feedback will help guide us in CogSketch's future development, so we can realize our vision and help make sketch-based intelligent systems commonplace in education.

## 8. Acknowledgements

This work was supported by NSF SLC Grant SBE-0541957, the Spatial Intelligence and Learning Center (SILC), and by a grant from the Office of Naval Research.

## References

[AD04] ADLER A., DAVIS R.: Speech and sketching for multimodal design. In *Proceedings of the 9th International Conference on Intelligent User Interfaces* (2004).

[AOD02] ALVARADO C., OLTMANS M., DAVIS R.: A framework for multi-domain sketch recognition. In *Proceedings of AAAI Spring Symposium on Sketch Understanding* (2002).

[BBB*03] BARKER K., BLYTHE J., BORCHARDT G., CHAUDHRI V., CLARK P., COHEN P., FITZGERALD J., FORBUS K., GIL Y., KATZ B., KIM J., KING G., MISHRA S., MORRISON C., MURRAY K., OTSTOTT C., PORTER B., SCHRAG R., URIBE T., USHER J., YEH P.: A knowledge acquisition tool for course of action analysis. In *Proceedings of the Innovative Applications of Artificial Intelligence Conference* (2003).

[Coh96] COHN A.: Calculi for qualitative spatial reasoning. In *Artificial Intelligence and Symbolic Mathematical Computation*, LNCS 1138, eds: J Calmet, J A Campbell, J Pfalzgraph, Springer Verlag, (1996) 124-143.

[EM98] EDWARDS G. , MOULIN, B.: Toward the simulation of spatial mental images using the Voronoi model. In P. Olivier and K. P. Gapp (Eds.), *Representation and Processing of Spatial Expressions*. LEA Press (1998).

[Eva68] EVANS T.: A program for the solution of geometric-analogy intelligence test questions. In *Semantic Information Processing*, ed: M Minsky, MIT Press (1968).

[FFG86] FALKENHAINER B., FORBUS K., GENTNER D.: The structure-mapping engine. In *Proceedings of the Fifth National Conference on Artificial Intelligence* (1986), pp. 272-277.

[FO90] FORBUS K., OBLINGER D.: Making SME greedy and pragmatic. In *Proceedings of the 12th Annual Meeting of the Cognitive Science Society* (1990).

[FFU01] FORBUS K., FERGUSON R., USHER, J.: Towards a computational model of sketching. In *Intelligent User Interfaces (IUI)* (January, 2001).

[FTU03] FORBUS K., TOMAI E., USHER, J.: Qualitative spatial reasoning for visual grouping in sketches. In *Proceedings of the 17th International Workshop on Qualitative Reasoning* (August, 2003).

[FUC03] FORBUS K., USHER J., CHAPMAN, V.: Sketching for military courses of action diagrams. In *Proceedings of IUI'03* (January, 2003).

[FUT05] FORBUS K., USHER J., TOMAI E.: Analogical learning of visual/conceptual relationships in sketches. In *Proceedings of AAAI-05* (2005).

[Gen83] GENTNER D.: Structure-mapping: A theoretical framework for analogy. *Cognitive Science, 7* (1983), 155-170.

[GD96] GROSS M. D., DO E.: Demonstrating the electronic cocktail napkin: A paper-like interface for early design. *ACM Conference on Human Factors (CHI '96)* (1996).

[HF05] HALSTEAD D., FORBUS K.: Transforming between propositions and features: Bridging the gap. In *Proceedings of AAA-05* (2005).

[Kim93] KIM H.: Qualitative reasoning about fluids and mechanics. Ph.D. dissertation and ILS Technical Report, Northwestern University (1993).

[KFT*05] KLENK M., FORBUS K., TOMAI E., Kim H., KYCKELHAHN, B.: Solving everyday physical reasoning problems by analogy using sketches. In *Proceedings of 20th National Conference on Artificial Intelligence (AAAI-05)* (2005).

[KFG*00] KUEHNE S., FORBUS K., GENTNER D., QUINN B.: SEQL: Category learning as progressive abstraction using structure mapping. In *Proceedings of the 22nd Annual Meeting of the Cognitive Science Society* (2000).

[LFH*06] LOCKWOOD K., FORBUS K., HALSTEAD D., USHER J.: Automatic categorization of spatial prepositions. In *Proceedings of the 28th Annual Conference of the Cognitive Science Society* (2006).

[LHK*02] LANDAY J. A., HONG J., KLEMMER S., LIN J., NEWMAN, M.: Informal PUIs: No recognition required. In *Proceedings of AAAI Spring Symposium on Sketch Understanding* (2002).

[LDF06] LOVETT A., DEHGHANI M., and FORBUS K.: Efficient learning of qualitative descriptions for sketch recognition. In *Proceedings of the 20th International Workshop on Qualitative Reasoning (QR'06)* (2006).

[LFU07] LOVETT A., FORBUS K., USHER J.: Analogy with qualitative spatial representations can simulate solving Raven's Progressive Matrices. In *Proceedings of the 29th Annual Meeting of the Cognitive Science Society* (2007).

[LLF08] LOVETT A., LOCKWOOD K., FORBUS K.: A computational model of the visual oddity task. In *Proceedings of the 30th Annual Meeting of the Cognitive Science Society* (2008).

[LSP*07] LEE W., DE SILVA R., PETERSON E. J., CALFEE R.C., STAHOVICH, T. F.: Newton's Pen – A pen-based tutoring system for statics. In *Proceedings of the Eurographics Workshop on Sketch-Based Interfaces and Modeling* (2007).

[Nie89] NIELSEN P.E.: A qualitative approach to rigid body mechanics. (Tech. Rep. No. UIUCDCS-R-88-1469; UILU-ENG-88-1775). Urbana, Illinois: University of Illinois at Urbana-Champaign, Department of Computer Science (1988).

[PSC*96] PITTMAN J., SMITH I., COHEN P., OVIATT S., YANG, T.: Quickset: a multimodal interface for military simulations. In *Proceedings of the Sixth Conference on Computer-Generated Forces and Behavioral Representation* (1996), pp. 217-24.

[RKF02] RASCH R., KOTT A., FORBUS K.: AI on the Battlefield: An experimental exploration. In *Proceedings of the 14th Innovative Applications of Artificial Intelligence Conference* (2002).

[RM89] REITER R., MACKWORTH A.K..: A logical framework for depiction and image interpretation. *Artificial Intelligence 41* (1989), 125-155.

[SMF02] SAUND E., MAHONEY J., FLEET D., LARNER D., LANK, E.: Perceptual oganization as a foundation for intelligent sketch editing. In *Proceedings of AAAI Spring Symposium on Sketch Understanding* (2002), pp. 118-125.

[SDS00] STAHOVICH T., DAVIS R., SHROBE H.: Qualitative rigid-body mechanics. *Artificial Intelligence 119*, 1-2 (2000), 19-60.

[TLF*05] TOMAI E., LOVETT A., FORBUS K., USHER J.: A structure mapping model for solving geometric analogy problems. In *Proceedings of the 27th Annual Conference of the Cognitive Science Society* (2005).