EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling (2008)
C. Alvarado and M.- P. Cani (Editors)

Automatic Interpretation of Depiction Conventions in Sketched Diagrams

Kate Lockwood, Andrew Lovett, Ken Forbus, Morteza Dehghani, and Jeff Usher

Qualitative Reasoning Group, Northwestern University, Evanston, IL, United States

ABSTRACT

Diagrams are used in many educational settings to convey physical and spatial information. Sketching is used, in turn, to test students' understanding of course concepts. The availability of Tablet PCs offer an exciting opportunity to create intelligent tutoring systems which automatically provide students with feedback on sketched work, and to create systems which can capture knowledge via interaction with people. However, for such systems to provide useful and relevant feedback, the software must be able to interpret diagrams that students have drawn. Interpreting diagrams correctly requires an understanding of some basic depiction conventions common in diagrammatic representation. Here we describe how to combine general semantic information about objects in sketched diagrams with geometric information from the sketch to aid in the interpretation of regions and edges. This system is implemented as an extension to the CogSketch sketch understanding system.

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

1. Introduction

Diagrams are used throughout education to clarify physical and spatial concepts which are not easily conveyed through text alone. This is especially common in the sciences and engineering. For example, consider the figure below which shows a diagram taken from an online middle school science resource describing the layers of the Earth:

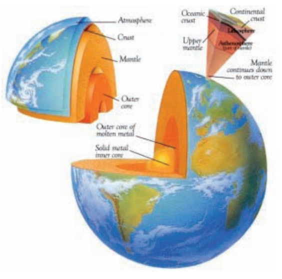

Figure 1. *A diagram from an online 6th grade earth science curriculum teaching students about the different layers of the Earth's interior.*

Much like diagrams can be used to convey information, sketching is often used to test student comprehension of spatial and physical concepts. For example, a baseline worksheet for incoming students in a geosciences class at Northwestern University included the following question: *"Draw a picture of the Earth's interior. The circle represents the Earth's surface and the dot is the very center of the Earth"* (an outline was provided, inside which the students sketched an answer). The availability of Tablet PCs creates an opportunity for creating electronic versions of assignments like these. Electronic worksheets could incorporate intelligent tutoring systems, providing students with real-time feedback on their work.

One challenge for automatically providing feedback is the huge variability in student answers to open-ended sketching questions. For example, consider Figure 2, which shows three different student answers to the geosciences worksheet question.

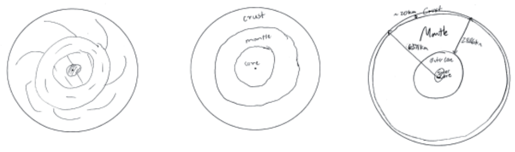

Figure 2. *Three examples of student responses to the question "Draw a picture of the Earth's interior..." taken from an introductory Geoscience course.*

As you can see, the student sketches vary greatly in the amount of detail provided and in the depiction conventions used. Being able to correctly interpret all three sketches is a huge challenge for symbol-recognition oriented sketch understanding systems. Such systems try to match user drawn ink to a fixed catalogue of known symbols. For some domains, such as electronics and UML diagrams

[AD04] [AOD02], this is a reasonable approach. But for many domains in science, engineering, and mathematics education, symbol-based aproaches are inapproriate. Certain spatial aspects of the objects depicted in Figures 1 & 2 matter, they are not simply abstract symbols. Our *nuSketch architecture* [FFU01] is designed to handle domains such as these. It is based on two insights: (1) In most human-to-human sketching, recognition is a catalyst, not a requirement. People use language to explain their sketches; we provide interface tools for providing functionally similar ways to *conceptually label* glyphs in a sketch. (2) Many of the conceptually relevant relationships in sketches are *qualitative*. For example, a student who drew the regions of the Earth's interior not quite to scale has still produced an acceptable answer, whereas leaving out a layer or putting them in the wrong order indicates a misconception that should be corrected. Our approach is to model human visual and geometric processing of the ink in a sketch, combined with formal representations of conceptual knowledge drawn from a large-scale knowledge base, to provide *open-domain* sketch understanding abilities. This is very important for building software coaches for open-ended, creative classes, such as engineering design, where the set of possible objects is extremely broad. It is also crucial for creating systems for knowledge capture, where people build knowledge bases by interacting in natural ways with intelligent software.

Even with conceptual labelling, many interpretation problems remain. One of them is automatically parsing the sketch into different edges and regions that represent the labelled entities. For example, in Figure 2, when a student draws a circle around the middle dot and labels it "core" people viewing the sketch can infer that the entire area inclosed by the circle is meant to represent the core of the Earth, not just the edge itself. On the other hand, if the student is drawing a solar system (see Figure 3), an orbit should be interpreted as only the ink drawn by the user, not all the space enclosed by the ink. Such depiction conventions are relatively fine-grained, relying on the properties of the object as much as the global context. While orbits do not include the area inside them, a similar circle indicating a planet does, for instance.

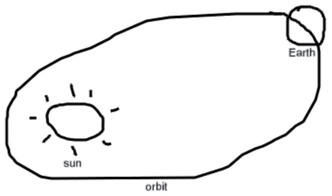

Figure 3. *Sketch of the solar system. Both the orbit and the planet are drawn with similar shaped glyphs, yet they need to be interpreted differently.*

For sketch-enabled intelligent tutoring systems to give useful feedback, they must be able to correctly segment sketches to understand student intent. Another task motivating this work is the use of sketches in multimodal knowledge capture. For example, diagrams in educational materials are accompanied by explanatory text. We are creating a system that learns from sketched diagrams plus accompanying simplified English text. Being able to correctly interpret how entities in the diagram are depicted is essential for integrating knowledge across modalities.

Our approach is to use very general conceptual information to infer what sort of geometric properties should be involved in the depiction of an object, and use visual processing on the ink to find (or construct) the appropriate spatial entities. In this paper we are specifically addressing the segmentation of entities in sketched diagrams into *regions* and *edges*. The key distinction is that regions have area while edges do not. We call this task *spatial extent identification*.

The rest of this paper describes our method for modeling this flexible interpretation of depiction conventions within the CogSketch system. First we briefly review some CogSketch basics. Next, we define the particular class of problem we are tackling, including an illustrative example. Then we describe how we use a combination of semantic information and geometric information to determine the correct interpretation for the objects in a sketch. After discussing related work, we close with some ideas for future work.

2. CogSketch

CogSketch is an open-domain sketch understanding system built on the nuSketch architecture [FFU01]. In CogSketch, each object drawn is represented by a *glyph*. A glyph contains both the actual ink drawn by the user and a *conceptual label.* The conceptual label is supplied by the user and is tied to a concept in the underlying knowledge base. Currently we are using a subset of the ResearchCyc (http://research.cyc.com/) knowledge base (including 30,000 concepts). Users can also supply a name with which to refer to the glyph. Names can be any natural language string. For example, Figure 4 shows a screenshot of a diagram drawn in CogSketch. In this diagram, the cylinder in the sketch is labeled as a `WaterTank` using the concept from ResearchCyc and is named "tank". This allows the user to refer to the tank simply as "tank". Likewise, if there were multiple tanks, they could each be given different identifying names. In CogSketch, users segment their own ink into glyphs by clicking a button at the beginning and end of drawing each glyph.

Conceptual labeling allows CogSketch to truly be domain-independent and allows us to operate in domains without clear drawing conventions. All sketch understanding work must strike a balance between constraints on the user and the depth of interpretation that is possible. While segmenting of ink into glyphs and conceptually labeling them does require more work by the user, in return they gain freedom from recognition errors and the ability to be supported by more in-depth reasoning. Aside from manual segmentation, we place no other restrictions on how users draw each glyph. For example, they can use as many strokes as they like, connected or not, and can take as long as they like. This contrasts with a common practice in multimodal interfaces of using constraints such as time-outs and pen-up events to automatically infer segmentation. For our users, who are often thinking hard about what they are drawing, time-outs and pen-up constraints are poor segmentation signals and quite annoying to them.

CogSketch computes a variety of spatial relationships automatically, including the RCC-8 qualitative topology relationships [Coh96] and connected and contained groups of glyphs (see [FFU03] for details). The digital ink itself is also available in subsequent processing, re-sampled into constant-spaced intervals from the original time-stamped pen events.

3. The Conceptual Segmentation Task

We define the task of *conceptual segmentation* to be the assignment of conceptual interpretations to regions and edges within the sketch. As noted above, conceptual labeling of ink is necessary, but not sufficient, to solving this problem. Consider the sketch in Figure 4 below showing a tank partially filled with water. We will use this example throughout this paper. This sketch consists of two glyphs: one closed polygon representing the tank, and one line representing the water. Figure 5 shows these two glyphs.

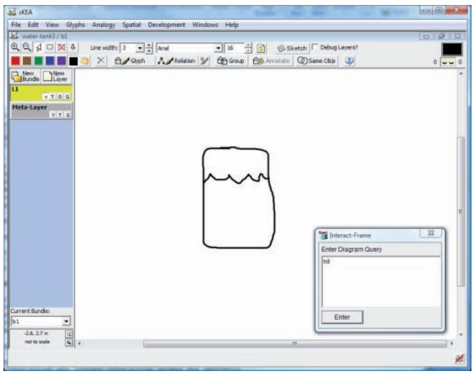
sKEA
File Edit View Glyphs Analogy Spatial Development Windows Help
Interact-Frame
Enter Diagram Query
nil
Enter

Figure 4: *A CogSketch screen shot depicting a tank partially filled with water.*

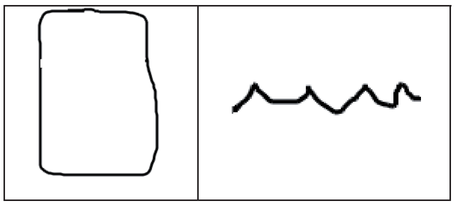

Figure 5: *The glyphs of the sketch in Figure 4.*

If we simply use the conceptual labels, the system would think that the object water in the sketch was only the edge created by the water glyph when in fact it is the area inside the tank underneath the water glyph. We could require that water be drawn using a closed polygon, but that would unnaturally constrain users, and does not scale well.

We are interested in using the fact that we know we are drawing water and what we know about how things are typically drawn – *depiction conventions* – to automatically derive the correct segmentation of the sketch. We test CogSketch's segmentations by asking it to highlight the region or edge in a sketch representing a specific entity. If the correct area is highlighted, we conclude that the system has correctly interpreted that portion of the sketch.

4. Spatial extent identification

Spatial extent identification is done through queries in CogSketch. A spatial extent query takes as input a term describing a conceptual entity referred to in the sketch, and produces as output a spatial entity representing the spatial extent of that conceptual entity. When queried from the interface, the spatial extent is highlighted. Regions are displayed as filled polygons while edges are simply highlighted lines. A step-by-step summary of the algorithm is given in Figure 6 and described further what follows.

Inputs: A sketch S and a query term Q

(1) Identify the glyph G corresponding to Q in S.

(2) Extract semantic knowledge about the query term from the KB

(3) Decide whether Q should be represented via an edge or a region, using the decision tree in Figure 8.

(4) Construct the appropriate edge or region by analyzing G and topologically related glyphs.

Figure 6. Summary of the spatial extent algorithm

The first step in spatial extent identification is to determine which glyph in the sketch corresponds to the term of the query. For example, if a user typed "water" into the diagram interaction box in Figure 4, the system would examine the glyphs in the sketch to find the one named water. In Figure 4, there is one glyph named water which is shown in Figure 7.

Figure 7. *The glyph in the sketch in Figure 4 that is named water. Note that the water is drawn as one line that spans the tank depicted in the sketch.*

4.1 Using semantic information for depiction reasoning

Once the appropriate glyph is identified, we access the conceptual label(s) provided by the user. In our example, the glyph being considered is labeled with the concept `Water` from the ResearchCyc KB. Knowing what the glyph represents helps us figure out how to interpret the diagram correctly. For example, ResearchCyc has 335 facts about water. This includes information about its role in the ResearchCyc ontology and, especially important for our purposes, some linguistic knowledge about the term.

Figure 8 below shows the decision tree used to identify whether an entity should be depicted via an edge or a region, using both the conceptual label and the ink of the glyph depicting it. The first level choices are made according to whether the entity belongs to: (1) a mass noun or entity that subclasses from the Cyc concept `TangibleStuffCompositionType` (2) an entity that subclasses from `Path-Spatial` (3) or a physical object.

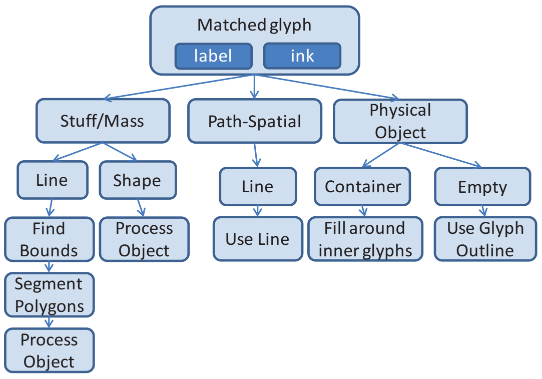

Figure 8. *Decision tree for spatial extent identification*

For example, a concept might contain information that, linguistically, the word referring to it is a mass noun or a count noun. Mass nouns refer to entities that can be viewed as spatially flexible pieces of stuff, such as liquids and powders, whose boundaries are highly constrained by containment relationships. Returning to our example from Figure 4, the concept `Water` is linguistically a mass noun, and consequently the system infers that a region is required to depict it. While this decision tree covers a broad range of useful cases, we do not expect that it is complete with respect to the set of conventions people commonly use, an issue we return to later.

4.2 Inferring the geometry of depiction

Once the system has inferred the conceptual category for a glyph, it attempts to find or construct the appropriate geometric entity. For the water/tank example (an instance of the stuff/mass path through Figure 8) it starts by classifying the geometric properties of the ink for the glyph, determining if it is a line or a polygon. For example, the glyph representing water in Figure 7 is a line, not a polygon. Since the depiction of water requires a region, the system has more work to do. (A user could have drawn the water by tracing out a region inside the tank, in which case the system would be satisfied with the glyph itself as the geometric entity.)

The next step is to determine if there are other glyphs which can help constrain the extent of the object. In this example, the tank glyph constrains the extent. We find such glyphs by looking for RCC8 relationships, i.e., glyphs for which the water is either TPP or NTPP (i.e., Tangential Proper Part or Non-Tangential Proper Part). When these relationships hold between the tank glyph and the water glyph, we then do a follow-up check to see if the water intersects (within a threshold) both sides of the tank.

Once we have both glyphs (the water and the tank) we need to find the region representing the part of the tank where the water is found. This is accomplished by combining the ink from the two glyphs and segmenting it into edges and *edge cycles*.

Edges are identified by segmenting the ink at places where one line intersects another, or where there is a clear corner along a line. Edge cycles are identified by finding minimal closed cycles among the edges. In the current example, CogSketch identifies two edge cycles, one representing the area in the tank above the water and the other representing the area in the tank below the water.

For stuff/mass nouns, the system assumes the user has drawn the uppermost edge of the object, and that the object descends from there to fill the container below it. Thus, in the current example, the system looks for a cycle such that glyph for water overlaps with the top of the cycle, while the rest of the cycle is made up of points from the tank glyph. If an appropriate cycle is found, it is identified as the region that the user is looking for, and it is then converted to a polygon and processed like a physical object.

Physical objects (the third path in Figure 8) are checked to see if they contain other glyphs (containment is one of the spatial relationships computed automatically by CogSketch). If the glyph has other objects inside of it, the algorithm as currently implemented assumes that the correct segmentation for the glyph is the space around the inner objects. This is the correct interpretation for situations like the layers of the earth, or bubbles in soda. Figure 9 shows the results of the query "mantle" in a sketch of the layers of the Earth.

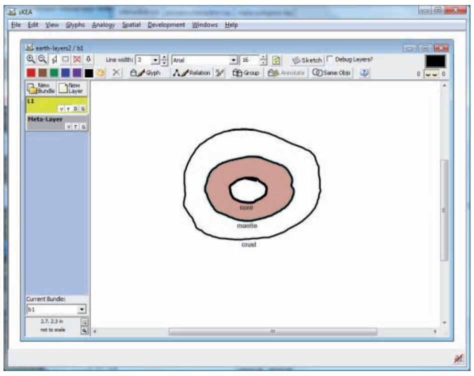

Figure 9. *The results from the user query "mantle" in a sketch of the layers of the Earth.*

If a physical object has no interior glyphs, the whole area of the polygon is considered the correct depiction and it is highlighted in the diagram. Figure 10 shows the results of our system on the water and tank example when queried for "water". In the series of figures that follows we will show the performance of our system on a series of diagrams that highlight some other properties of our algorithm.

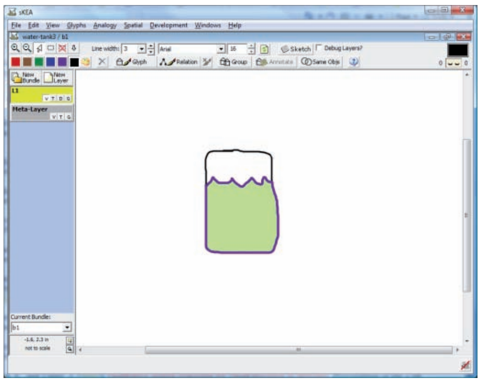

Figure 10. *A screen shot of CogSketch showing the results from the user query "water". The shaded are represents the region that the system infers is water.*

This approach easily extends to other, more complex situations. In Figure 11 the sketch is composed of four glyphs: tank1 (the tank on the left), a pipe, tank2 (the tank on the right), and one glyph representing the water. Since our algorithm for locating cycles of edges is flexible enough to find cycles over multiple glyphs, the two tank problem is easily handled.

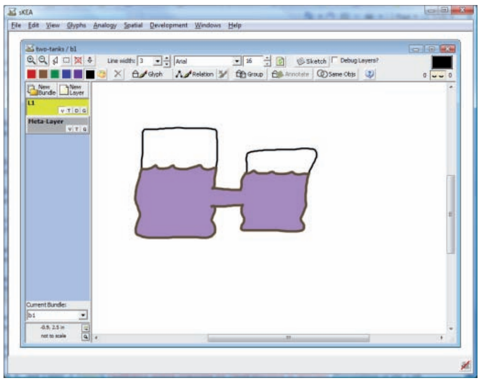
sKEA
File Edit View Glyphs Analogy Spatial Development Windows Help
two-tanks / b1
Current Bundle:
b1

Figure 11. *Another result for the query "water". In this case the sketch contains four separate glyphs {tank1, tank2, pipe, water}.*

We are also able to handle situations where there are several glyphs that are conceptually labeled as mass nouns, even if they are drawn similarly. In Figure 12 the sketch is a tank with both oil and water in it. When queried for "oil" our system is able to easily identify the extend of the area representing oil. Situations like this would be particularly tricky for template based systems since both oil and water are drawn with similar glyphs. Also, while the wavy line is typical of a convention used to indicate liquid in a sketch, it is by no means a standardized symbol.

The current algorithm for physical objects has been sufficient for all of the diagrams that we have considered in this paper, however, when a glyph is a container it isn't always the case that you want just the space around the interior glyphs. For example, consider a glass of water with a straw in it. When you are determining the spatial extent of the water, it actually also covers the area occupied by the straw. We are extending the spatial extent algorithm to account for situations like this by further examining the objects in container/contained groups. This is another example where we will need to combine conceptual information from the KB with spatial information from the ink to identify the correct spatial extent of entities.

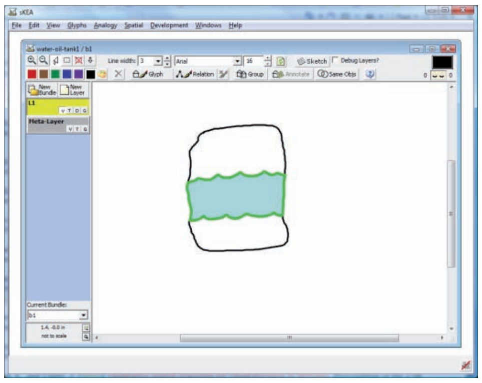
sKEA
File Edit View Glyphs Analogy Spatial Development Windows Help
water-oil-tank1 / b1
Current Bundle:
b1

Figure 12. *In this example, the system is able to easily discriminate between the region representing "oil" and that representing "water" using the same techhniques.*

The processing for an entity that has been determined to be an instance of a Path-Spatial proceeds much like the processing of a mass noun, by first checking to see how the object is drawn in the sketch. Consider again the solar system/orbit example from Figure 3. In this case, the system checks to see if the path is represented by a single line, like the orbit in the sketch. This suggests that the points on the line make up the conceptual entity. The other option, of course, is that a path is depicted by mulitple lines or polygons such as a drawing of a railroad track or road. This condition is not currently being handled by our system, but is in the process of being added.

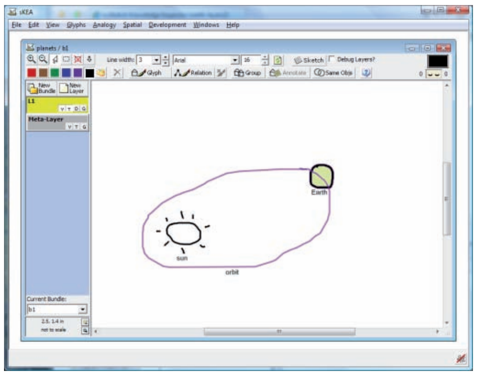
sKEA
File Edit View Glyphs Analogy Spatial Development Windows Help
planets / b1
Earth
sun
orbit
Current Bundle:
b1

Figure 13. *A screenshot showing the spatial extent identified for "orbit" and "Earth" in a simplified drawing of the solar system. Even though both objects are drawn similarly, conceptual information provides cues on their different interpretations.*

4.3 Compound Queries

Often the parts of a diagram that need to be referred to are far more complex than just "water". For example, when doing problems in physics or chemistry, it may be useful to be able to refer to the water in one part of the apparatus only. Our system also handles queries of the form *<object> <relation> <object>*. Information about relations from ResearchCyc is used to understand the semantics of such queries. Figure 14 illustrates the result for the query "water in tank1". The analysis is essentially that of Figure 11, with the additional specification of "in tank1" leading to the intersection of the water polygon and the tank1 polygon.

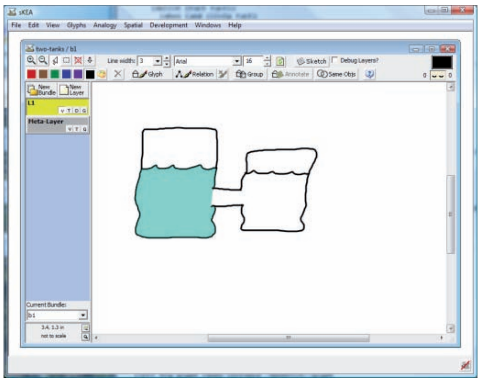

Figure 14. *Screenshot showing the result of the query "water in tank1"*

5. Related Work

The division of scene elements into edges and regions in sketches was explored in the Mapsee program of Reiter and Mackworth [RM89]. They proposed a logical framework for depiction that formalized the mapping between images and scenes of simple maps containing roads, rivers, shores (represented as edges in the images) and water and land (represented by regions in the images). They identified a set of six visual relations ({tee, chi, bounds, closed, interior, and exterior}) and provided axioms and constraints which combined these visual primitives and mapped them to the scene elements (roads, rivers, etc). Like Mapsee, we are concerned with modeling how conceptual entities are depicted. However, Mapsee was designed for one domain, maps, and its axioms map visual elements directly to interpretations in that domain. By contrast, our model works through an intermediate distinction – regions versus edges – and performs reasoning over a large-scale, off-the-shelf knowledge base to identify depiction constraints. Their task was fundamentally one of image interpretation, recognizing unlabelled lines as map elements, whereas our task starts with conceptually labeled ink.

Alvarado and colleagues [AD04] [AOD02] describe a multi-domain sketch recognition engine. Their systems use a hierarchical shape description language where low level shape description (circles, arrows, etc) are defined once in a domain-independent fashion. Then a separate set of rules ties a given shape to a domain specific interpretation (e.g. an arrow represents a child link in a family tree diagram). This approach works well in a very tightly constrained domain with a small number of differentiated symbols (family trees, circuit diagrams, etc) however, it does not work as well in the more open-domain, unconstrained types of sketches that we are concerned with.

Futrelle has explored parsing graphs from scientific papers [Fut90][FKA*92]. His Diagram Understanding System uses a Context-based Constraint Grammar to describe the parts of a diagram and a set of Generalized Equivalence Relations (GERs) like *near* and *parallel* to describe the relationships between objects in the diagram. This works well for domains where diagrams are uniform and easily described in terms of a grammar, such as x,y plots and finite state diagrams. However, having to define a new grammar for each type of diagram does not scale well to the open-domain diagram understanding problem.

Kara and Stahovich's SimuSketch [KS04] takes a two-stage approach to recognition in sketched diagrams containing arrows. Other ink in the sketches is grouped and segmented based on clustering around the head and tail of recognized arrows. The clusters of ink are then matched against 24x24 templates for recognition. This is a unique and interesting approach to segmentation which gets around many cumbersome algorithms such as time outs or requiring single strokes. SimuSketch is embedded in Matlab's SimuLink system, to use ink recognition to set up engineering simulations.

Approaches like those outlined above that rely on low-level shape recognition along with domain-specific rules for interpretation represent a complementary approach to ours. A hybrid system, which combines low-level recognition for common elements (e.g., arrows) and a more generative interpretation process might be useful in many tasks. For example, in a physics system, it might be useful to automatically recognize arrows and interpret them as forces while leaving the types of objects that those forces can act on unconstrained given the wide variety of physical objects in the world.

Saund and colleagues [SMF*02][Sau02] have also worked on intelligently segmenting sketches. Like us, they do not work on recognizing objects per se, focusing instead on identifying visually natural decompositions of ink. This information can be used to make intelligent decisions about which parts of a sketch a user is trying to select or edit.

Anderson and Armen's DiaSketch [AA02] is interested in inter-diagrammatic reasoning – learning from multiple diagrams of the same information. They focus on sketching as a way of interacting with a more precisely defined diagram (such as one that was scanned in).

We believe that recognition is not very important for the sketch understanding tasks we are focused on. Unlike sketches in engineering design, where later versions will need to be imported to a formal CAD system, sketches produced for student assesment are meant to be short lived. Also, while the amount of detail can vary greatly, much of it is superfluous to the pedagogical goals of the assignment and thus not important for interpreting student understanding. Similarly, for knowledge capture, where new concepts are being continually introduced, only allowing a pre-identified set of objects to be drawn is simply not appropriate, unless the system is limited to a narrow domain.

6. Conclusions and Future Work

We have described how to use a combination of semantic and geometric information to identify one type of depiction convention in sketched diagrams, whether something should be represented as a region or an edge. Our interpretation process closely couples semantic and geometric information to reason about depiction conventions and to use those conventions to segment the sketch into meaningful regions and edges.

As noted earlier, creating a platform for sketch-enabled educational software and creating systems for multimodal knowledge capture are the motivation for this work. We are currently constructing prototypes of both types of systems. Experience with these prototypes will further refine the algorithms used here. For example, one aspect of knowledge capture is learning how people in a community depict different kinds of entities. An important intermediate goal is to be able to automatically expand the decision tree of Figure 8 via learned knowledge.

We are also interested in studying depiction conventions which are widely used, but not domain or situation dependent. For example, call-outs and cut-aways are two conventions that are used across disciplines which have important implications for how diagrams (and the spatial relations in them) should be interpreted. CogSketch is free and available online.[1] (The online version comes bundled with OpenCyc, as opposed to ResearchCyc which was used for this work because it currently contains more natural language knowledge). As more people download and use CogSketch, we are hoping to amass a large library of sketches. This library will enable us to more thoroughly survey the conventions used in sketched diagrams. It will also provide a corpus of labeled sketches that we hope will be useful to us and to others in the sketch understanding community.

7. Acknowledgements

This work was supported by a grant from the Office of Naval Research and by the National Science Foundation under Grant No. SBE-0541957, the Spatial Intelligence and Learning Center. The authors would also like to thank Benjamin Jee and Brad Sageman for providing the student sketches of Geoscience phenomena and Victor Lee for sharing his database of textbook examples.

[1] http://spatiallearning.org/projects/cogsketch_index.html

References

[AA02] ANDERSON, M. and ARMEN, C.: DiaSketches. *Int. Symp. On Smart Graphics.* Hawthorne, NY. (June 2002)

[AD04] ALVARADO, C., DAVIS, R.: SketchREAD: A Multi-domain Sketch Recognition Engine. *Proceedings of the 17th annual ACM symposium on user interface software and technology* (2004).

[AOD02] ALVARADO, C., OLTMANS, M., DAVIS, R.: A Framework for Multi-Domain Sketch Recognition. *Proceedings of AAAI Spring Symposium on Sketch Understanding*, (2002).

[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.

[FFU01] FORBUS, K., FERGUSON, R., USHER, J.: Towards a Computational Model of Sketching. *Intelligent User Interfaces (IUI)* (January, 2001).

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

[Fut90] FUTRELLE, R.: Strategies for Diagram Understanding: Generalized Equivalence, Spatial/Object Pyramids and Animate Vision. *In 10th ICPR,*(1990) pp. 403-408.

[FKA*92] FUTRELLE, R., KAKADIARIS, I., ALEXANDER, J., CARRIERO, C., and NIKOLAKIS, N.: Understanding Diagrams in Technical Documents. *IEEE Computer*, 25(7), 75-78 (1992).

[KS04] KARA, L. and STAHOVICH, T.: Hierarchical Parsing and Recognition of Hand-Sketched Diagrams. *UIST '04.* Santa Fe, New Mexico.

[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). Hanover, NH.

[RM89] REITER, R., MACKWORTH, A.K..: A Logical Framework for Depiction and Image Interpretation. *Artificial Intelligence* (1989), 41, 125-155.

[SMF*02] SAUND, E., MAHONEY, J., FLEET, D., LARNER, D., and LANK, E.: Perceptual Organization as a Foundation for Intelligent Sketch Editing. In *Proc AAAI Spring Symposium on Sketch Understanding* (2002).

[Sau03] SAUND, E.: Finding Perceptually Closed Paths in Sketches and Drawings. *IEEE Transaction on Pattern Analysis nad Machine Intelligence.*(2003), 25:4, 475-491.