AI Magazine Volume 25 Number 3 (2004) (© AAAI)

Qualitative Spatial Reasoning about Sketch Maps

Kenneth D. Forbus, Jeffrey Usher, and Vernell Chapman

■ Sketch maps are an important spatial representation used in many geospatial-reasoning tasks. This article describes techniques we have developed that enable software to perform human-like reasoning about sketch maps. We illustrate the utility of these techniques in the context of nuSketch Battlespace, a research system that has been successfully used in a variety of experiments. After an overview of the nuSketch approach and nuSketch Battlespace, we outline the representations of glyphs and sketches and the nuSketch spatial reasoning architecture. We describe the use of qualitative topology and Voronoi diagrams to construct spatial representations, and explain how these facilities are combined with analogical reasoning to provide a simple form of enemy intent hypothesis generation.

Maps are a ubiquitous tool for human geospatial reasoning. Computer support for geospatial reasoning often takes the form of *geographic information systems* (GIS)—sophisticated systems that combine computational geometry with database techniques to provide powerful abilities to manipulate and visualize vast quantities of digital terrain data. GISs are the computer-aided design (CAD) software of geospatial tasks. However, it is well known that in engineering, CAD software is not terribly useful for the early stages of design (conceptual design) where basic design choices are made and principles of operation are laid out before detailed design decisions are made. There appears to be a similar stage of thinking in geospatial tasks, where sketch maps are used to reason through a problem. By *sketch maps,* we mean compact spatial representations that express the key spatial features of a situation for the task at hand, abstracting away the mass of details that would otherwise obscure the relevant aspects. Sketch maps today are typically drawn by hand on paper.

For computers to become useful partners in geospatial problem solving, they need to be able to work with sketch maps just as people do. Just as qualitative reasoning has proven valuable in software supporting conceptual design in engineering, we claim that qualitative spatial reasoning (Forbus, Nielsen, and Faltings 1991; Glasgow, Chandrasekaran, and Narayanan 1995) is essential for working with sketch maps. This article describes the progress we have made in a specific geospatial domain—battlespace reasoning—towards this goal. Warfare, while a regrettable aspect of human existence, remains one of the most complex and most important kinds of task that people do. Planning a battle requires coordinating a complex array of people and equipment to achieve sometimes subtle goals, in situations where there is great uncertainty and danger. Terrain plays a crucial role in military reasoning, because it affects movement, it can provide cover and concealment, and it affects the operation of sensors. Thus geospatial reasoning must play a major role in generating and reasoning about battle plans, called *courses of action.* Figure 1 is a screen shot of a COA drawn with nuSketch Battlespace.

The introduction of digital media into military operations has been slow for several reasons. One major problem is that commanders are adamant about not wanting to use mice and menus; they sketch, and they want to interact with software via sketching, just as they interact with their people. Dealing with sketch

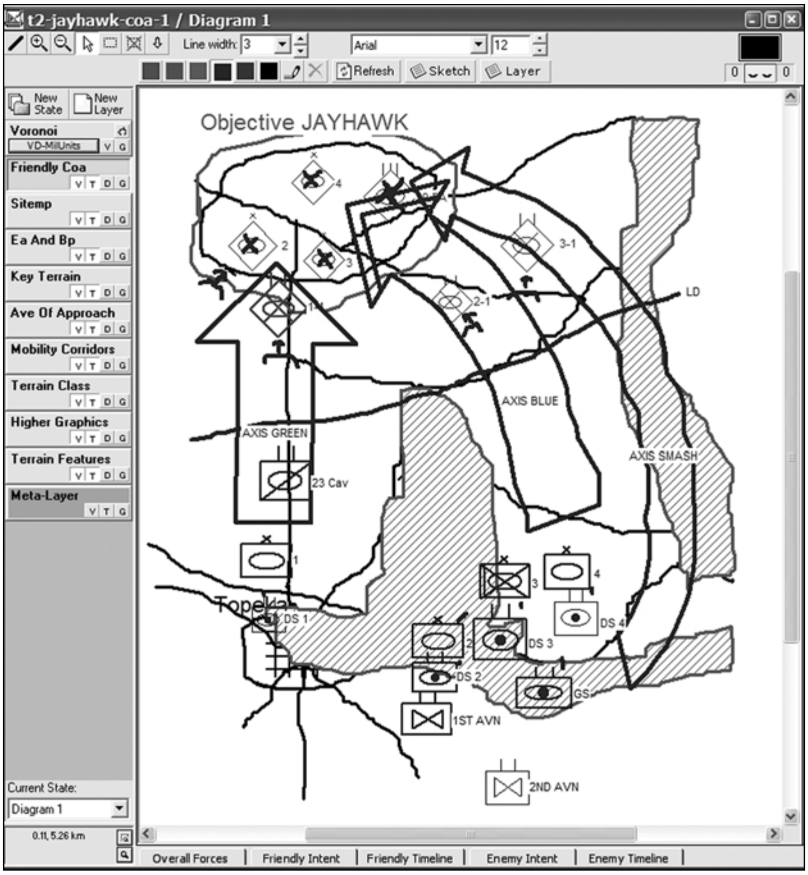

Figure 1. A COA Drawn with nuSketch Battlespace.

maps is a necessity for creating performance support tools for military operations. Although most of our experience has been in military tasks, the situation seems similar in other human geospatial reasoning tasks (Egenhofer 1997).

This article describes the techniques we have developed for qualitative spatial reasoning about sketch maps. We start by reviewing our approach to sketching and nuSketch Battlespace, our battlespace sketching software that has been used in several successful experiments. Next we provide an overview of the spatial representations of sketches and glyphs and the processing architecture that handles spatial computations. Then we describe the computation of spatial relationships, including qualitative topology and Voronoi diagrams. Path finding and position finding, two key tasks, are discussed next. We describe how these techniques are combined with analogical processing to provide a simple form of enemy intent hypothesis generation. Finally, we discuss plans for future work.

The nuSketch Approach to Sketch Understanding

Sketching is a form of multimodal interaction, where participants use a combination of interactive drawing and language to provide high bandwidth communication. Sketching is especially effective in tasks that involve space, such as geospatial reasoning. While today's software is far from being as fluent as sketching with a person, progress in multimodal interfaces has produced interfaces that are significantly more natural than standard mice or menu systems (cf. Cohen et al. 1997).

The typical approach in multimodal interfaces is to provide a more natural interface to legacy software systems by focusing on recognition (Alvarado and Davis 2001; Cohen et al. 1997). While this approach has led to useful systems, it has three serious limitations. First, today's statistical recognizers are not very good. Indeed, much of the multimodal literature focuses on using multiple modalities to overcome recognition problems in speech and ink. Our military users, based on their experience with previous multimodal interfaces, generally flatly refuse to use any system that requires speech recognition. Military operations are often conducted in noisy environments, under stress (which causes changes in vocal characteristics, which in turn requires retraining of recognizers), and there are potentially high costs associated with recognition errors. This is especially true in conceptual reasoning tasks, where achieving creative solutions requires focusing on the problem, not the interface.

The second limitation is that today's recognition technologies work best with in small, tightly controlled domains. The kinds of glyphs to be drawn and the vocabulary and grammar for speech or natural language (NL) systems must be known in advance, along with data for training statistical recognizers. Experiments with multimodal interfaces in military tasks generally require unnatural restrictions on vocabulary (for example, the experimenters provide a list of legal names that can be used, or derive such a list by working in advance with the experimental subjects for each set of experiments). These restrictions sharply limit the potential utility of such systems. The situation with ink recognition is worse. Today's statistical recognizers are "which of *N*" systems. They often work reasonably well for tightly restricted sets of visual symbols (such as handwriting recognition) where there is a lot of training data available and the variability of symbols is not high. Some visual symbols in military tasks have this character (such as the symbols for military units). However, many do not. Rivers and roads are drawn similarly to each other, as are hills and mountain ranges. Their spatial location, extent, and shape matter. These properties are not dimensions of variation that can be ignored in order to achieve recognition. In general, sketch understanding requires visual understanding, not recognition.

The third limitation is the assumption that sketching is a more natural interface to legacy software. There are a wide variety of potential applications for sketching that require significant understanding of the domain that the sketch is about. These will be new types of AI systems, relying heavily on combinations of conceptual and visual reasoning. (We believe such systems will also heavily rely on qualitative representations and reasoning, because they bridge the visual and conceptual.) Even if recognition improves to human level or beyond, it is essential to address the problem of enabling software to construct a visual and conceptual understanding of what is being sketched

Our approach in the nuSketch architecture (Forbus, Ferguson, and Usher 2001; Forbus and Usher 2002; Forbus, Usher, and Chapman 2003) is quite different and complements traditional multimodal research. We avoid recognition issues by using clever interface design. We focus instead on providing richer visual and conceptual understanding of what is sketched. We have created two systems based on the nuSketch architecture: nuSketch Battlespace (nSB) (Forbus, Usher, and Chapman 2003) specialized for battlespace reasoning, and the sketching Knowledge Entry Associate (sKEA) (Forbus and Usher 2002), a general-purpose knowledge capture system. While sKEA also does geospatial reasoning when appropriate—the two systems share a common code base—we focus in this article on nuSketch Battlespace for brevity.

The key insight in the nuSketch approach is that, in human-to-human sketching, recognition facilitates communication, but recognition is not essential to it. Most people sketch, but few are artists. Typically, people verbally label what they are sketching, to ensure that those they are working with make correct identifications. To be sure, there are broadly used visual symbols within specific cultures and professions (such as stick figures, electronic circuit components, and map symbols). However, the conceptual range of entities that can be used in sketching is far larger than any visual symbology. Consequently, some form of conceptual labeling is essential for robust sketching. Speech and written language are two ways to provide such labeling, but importantly, they aren't the only ways, as we will soon describe.

We work around the limitations of today's recognition technologies in two ways. First, we use manual segmentation of ink into glyphs. Many multimodal systems use timeouts or lift the pen to automatically segment ink. These techniques only work if (1) users do not have to think hard about the problem they are solving, and hence exceed the timeout period, and (2) they are only drawing very simple shapes, so that they do not have to lift their pen to create a reasonable depiction. In our experience, both timeouts and pen-up methods cause users serious problems. Consequently, in nuSketch systems, users indicate when they have started to draw a glyph and when they are finished drawing a glyph by pressing buttons.

Conceptual labeling in nuSketch systems is accomplished by explicitly identifying the concept that is being depicted. For example, in nuSketch Battlespace, users click on a glyph bar to indicate the kind of concept that they are about to sketch, based on the visual language used by the U.S. military (U.S. Army 1997). There are hundreds of concepts that users routinely draw, requiring careful interface design. We use two interface techniques to keep glyph bars manageable in size. First, users construct many symbols compositionally, and our glyph bar interface supports this approach. Second, the interface uses layers that are functionally organized, providing contexts that restrict what concepts make sense. In sKEA, where users can express any concept in the subset of the Cyc KB that we are using, there are tens of thousands of concepts that users can draw upon, and there is no well-defined visual symbology for most of them. Consequently, conceptual labeling in sKEA is achieved by text completion on the names of concepts.

Manual segmentation and explicit conceptual labeling slightly reduce the naturalness of sketching. However, our users find that a small price to pay for complete elimination of recognition errors. To be sure, as recognition technologies improve, we would like to extend the nuSketch architecture to employ them. However, some tasks seem like they will always be out of reach for recognition-oriented systems. Consider knowledge capture in a new domain. The concepts that the users might introduce are not known in advance, so the data needed to train vocabulary, speech grammars, and ink recognizers simply isn't available. In such circumstances the nuSketch approach seems to be the only viable method.

The key insight in the nuSketch approach is that, in human-to-human sketching, recognition facilitates communication, but recognition is not essential to it.

Overview of nuSketch Battlespace

nuSketch Battlespace is designed to help users develop courses of action (COAs) for land forces. It uses a large knowledge base concerning specialized military concepts as well as general common sense. We use a subset of Cycorp's Cyc knowledge base contents, with extensions developed by our group for qualitative and analogical reasoning and by the Defense Advanced Research Projects Agency (DARPA) community for military concepts and reasoning. We use our own knowledge base and reasoning system (FIRE) instead of Cyc because it is optimized for our needs. FIRE is a federated architecture, with analogical matching and retrieval tightly integrated with other kinds of reasoning. The interface uses special-purpose interface techniques to enable users to specify conceptual information (including the types of entities being sketched, timing information, and intent of actions), organized into layers to control complexity. Users can sketch terrain, specialized areas and paths (such as engagement areas and axes of advance), position units, and assign tasks and the reasons for doing them. Since planning in uncertain situations often involves exploring multiple hypotheses, and plans can involve complex sequential behavior and conditionals, nSB enables users to describe and link multiple states into a *comic graph,* a visualization based on action-augmented envisionments (Forbus 1989). The interface techniques that enable us to avoid recognition are described by Forbus, Usher and Chapman (2003); here our focus is on the qualitative spatial reasoning the system performs.

nuSketch Battlespace has been successfully used in several experiments. First, an early version was combined with a natural language input system (by AlphaTech and Teknowledge) and BBN Technologies' CADET system that generates synchronization matrices in an experiment to see if active-duty military personnel could successfully create COAs. As described by Rasch, Kott, and Forbus (2002), commanders were able to generate COAs three to five times faster, without any degradation in plan quality. In DARPA's Rapid Knowledge Formation program, nSB was adopted by both teams to provide sketching and spatial reasoning services for their integrated knowledge capture systems. The KRAKEN system from the Cycorp team combined nSB with their natural language facilities, and the SHAKEN system from the SRI team combined nSB with their concept map facilities. In an evaluation run by an independent contractor in the fall of 2002, both teams were able to demonstrate that military subject-matter experts were able to author knowledge using these systems. Specifically, they were able to create knowledge useful for critiquing COAs without training in formal knowledge representation. In DARPA's Command Post of the Future program, we have received long-term, valuable formative feedback from a variety of retired military officers. Their feedback has helped us improve the system to the point where we can have generals doing analogies between battlespace states within an hour of sitting down with the software for the first time.

Representing Glyphs and Sketches

This section describes the underlying ontology of sketches that we use. The basic unit in a sketch is the *glyph.* Every glyph has ink and content. The *ink* consists of one or more polylines, representing what the user drew when specifying that glyph. (Each polyline includes width and color information in addition to its points.) The *content* is a conceptual entity—the kind of thing that the glyph is representing. For example, if a user drew a mountain range, there would be an entity created to represent the glyph itself and an entity to represent the mountain range. While each subsketch depicting the mountain range would have a distinct glyph, the contents of those glyphs would all be the same entity.

The type of a glyph's contents affects the interpretation of its spatial properties. For example, the spatial extent of glyphs representing mountains and lakes is taken to be the spatial extent of that terrain feature. On the other hand, the spatial extent of a military unit is ignored because the size of such glyphs by convention has nothing to do with its footprint on the ground, so only its centroid is used in spatial reasoning. Pathlike terrain features such as roads and rivers have a one-dimensional extent, but their width is not tied to the width of the line depicting them, since that would unduly burden our users' drawing abilities. In contrast, paths introduced in planning actions do have widths that are specified by special gestures during sketching, because they provide spatial constraints on the movements of units. (Regions just outside the path might be targets of artillery, for example, and avoiding friendly fire is an important task constraint.)

While some basic spatial properties of glyphs are computed (described later), we do not perform any detailed shape reasoning on the ink comprising a glyph, nor do we attempt to visually decompose it. We call this *blob semantics* because it focuses on spatial relationships between glyphs rather than detailed reasoning about the visual structure of glyphs themselves. While inappropriate for recognition based on detailed visual similarity of specific features, it is an excellent approximation for most geospatial reasoning, where the focus is on configural relationships between glyphs. Given the crude nature of sketch maps, people are unlikely to be extremely accurate at reproducing shapes.

A *sketch* consists of one or more subsketches. *Subsketches* represent a coherent aspect of what is being sketched, such as a state of a plan, or a more detailed depiction or distinct perspective on something. Logically, subsketches are Cyc-style microtheories, local descriptions that must be internally consistent. In nSB, every subsketch represents a battlespace state. States can be partial, and are hypothetical, observed, or planned. Visually, the user sees either a single subsketch at a time, or the *metalayer,* a special view where each subsketch is viewed as a glyph. The comic graph consists of these glyphs and relationships between them, expressed by drawing arrows between state glyphs.

Subsketches are composed of *layers*. In nSB, each layer represents a particular subset of information about a battlespace state. Examples include terrain features, friendly COA, and SITEMP (that is, enemy COA). Every glyph exists on some layer. The layers of a subsketch are spatially registered, that is, they share the same coordinate system. Distinct subsketches need not be spatially registered, although in nSB they tend to be. Logically, each layer in a subsketch is a microtheory. Visually, layers are depicted as overlays on a common workspace for that subsketch. The user can control whether or not a layer is visible, grayed out (which keeps layouts in focus without being distracting), or invisible, to control detail while sketching. nuSketch systems can also introduce new layers to display the results of their reasoning.

Spatial Processing of Glyphs

Spatial reasoning is carried out when a glyph is added or changed, and in response to queries from nSB reasoning facilities. nSB has two visual processors, which are threaded to enable computation while the user is thinking or sketching. We describe each in turn, as a prelude to the detailed discussion of the spatial operations.

The *ink processor* is responsible for computing basic spatial properties of glyphs and responding to queries concerning spatial relationships. Whenever a glyph is added or changed, basic spatial properties are computed for it, including a bounding box, area, overall orientation, and roundness. Qualitative topological relationships are automatically computed between the new glyph and other glyphs on its layer.

The *vector processor* is responsible for maintaining a set of Voronoi diagrams describing spatial relationships between types of entities, and for the polygon operations used in position-finding and path-finding. Any time a glyph is added or changed, once the ink processor has updated its properties, the Voronoi diagram(s) it is associated with are updated appropriately. When spatial constraints involving position-finding or path-finding need solving, the vector processor carries out the construction of obstacle and cost diagrams and the polygon operations needed to combine them.

Conclusions reached by these processors are added to the LTMS-based (Forbus and de Kleer 1993) working memory of the reasoner for that sketch. Time-stamped assertions are used as assumptions in visual conclusions drawn by the system, so that when glyphs are moved, resized, or deleted the appropriate conclusions are automatically retracted.

Spatial Relationships between Glyphs

Spatial relationships are the threads from which configural information is woven. Therefore computing them appropriately is a crucial problem for qualitative reasoning about sketches. We discuss four kinds of spatial relationships in turn: (1) Qualitative topological relationships, (2) Voronoi relationships, (3) positional relationships, and (4) relationships based on local frames of reference.

Qualitative Topological Relationships

We use the RCC8 algebra (Cohn 1996) to provide a basic set of qualitative relationships between glyphs. RCC8 is appropriate because it captures basic distinctions such as whether or not two glyphs are disjoint, touching, or inside one another. These distinctions are used in several ways. First, they are used in controlling when to compute other relationships: computing whether or not one entity is east of another is moot unless they are disjoint, for example. Second, they suggest conceptual interpretations of relationships between the contents of the glyphs that they relate. For instance, a touching relationship between two glyphs, which represent physical objects, suggests that their contents might be touching. Finally, domain-specific inference rules can use these relationships when needed, such as inferring containment of physical objects depicted based on one glyph being inside another.

Much of the work on RCC8 and other qualitative topological algebras has focused on using transitivity for efficient inference. For sketches the use of such tables is unnecessary, because we can simply calculate for each pair of glyphs what RCC8 relationship holds between them, based on the visual properties of their ink. By default, we compute RCC8 relationships between a glyph and everything else on its layer when it is created and whenever it changes. RCC8 relationships with glyphs across layers in the same subsketch can be computed on demand during domain-specific reasoning.

Voronoi Relationships

Following Edwards and Moulin (1998), we use Voronoi diagrams to compute a variety of spatial relationships. Recall that given a set of spatial entities (called *sites,* typically points), a Voronoi diagram consists of edges that are equidistant from a pair of points. The Delauney triangulation is the dual of the Voronoi, consisting of a set of arcs between sites that have an edge between them in the Voronoi diagram. As Edwards and Moulin (1998) describe, the Delauney triangulation provides a reasonable approximation to visual proximity, in that two sites are proximal exactly when there is an edge connecting them in the Delauney triangulation. Moreover, a number of approximations to spatial prepositions can be computed, including between and near. Again, these are approximations: It is known that, psychologically, spatial prepositions depend on functional and conceptual information as well as spatial information (Coventry 1998, Feist and Gentner 1998). However, so far we have found them adequate for sketch maps.

Voronoi computations are defined in terms of sites being points, but glyphs have significant spatial extent. Consequently, adding a glyph to a Voronoi diagram involves adding sample points along the outer contour of the glyph's ink, each of which is treated as a site. These sites are marked with the glyph they derived from, so that while the Voronoi computations are done on the sampled sites, the results are expressed in terms of relationships between the glyphs. For example, two glyphs are *siteAdjacent* exactly when there exists a sample site on each glyph that is connected by an edge in the sample-level Delauney triangulation.

A key design feature in any system using Voronoi computations is what diagrams should be computed. We use several diagrams to capture different notions of proximity: A terrain-only diagram is useful for characterizing free space, and a units-only diagram is useful for grouping units, for example.

Positional Relationships

Positional relationships provide qualitative position and orientation information with respect to a global coordinate frame. Positional relationships between contents are expressed in terms of compass directions. For example, a tank brigade can be south of a mountain and east of a bridge. Not all glyphs can participate in such relationships. The task of securing a bridge, while represented by a glyph in the sketch, is not itself something that participates in positional relations, although the location at which it occurs can.

A key design choice is what positional relationships should be computed. It might seem at first that, like RCC8 relationships, it could be worth computing positional relationships between every pair of RCC8-DC glyphs. This turns out to be a terrible strategy, both in terms of computational effort and in terms of the usefulness of the results. Computationally, positional relationships are used to provide concise summaries (if communicating a situation) and to provide a framework for describing the layout of a situation (for instance when computing spatial analogies). Consequently, we limit the automatic computation of them to pairs of geographic features and compute positional relations for other appropriate entities on demand.

Other Frames of Reference

Another type of positional relationship links two entities based on a local coordinate system. For example, if two entities are related to an oriented path, it is useful to talk about one entity being ahead, behind, or at the same location along that path. nuSketch computes such relationships on demand, using projection of the centroids of the entities to the closest point on the path to determine their relative position.

Some entities have a distinct orientation, even without having a pathlike extent. Military units, for example, have fronts, flanks, and rears. Again, we compute such relationships on demand, based on orientation information associated with the entities.

Position Finding

Some of the most interesting implications of sketch maps involve constructing places: The good sites for a park, in an urban planning task, or a good site for an ambush, in a military setting. We use conceptual knowledge of the contents of glyphs, combined with spatial reasoning on their ink, to automatically construct regions that satisfy spatial and functional criteria.

Two important constraints in military spatial reasoning are (1) *fields of fire,* that is, what can someone's weapons hit? and (2) *visibility,* that is, what can someone see?

Some kinds of terrain features, such as mountains, block weapons and thus provide cover. Other kinds of terrain features, such as forests, block visibility and thus provide concealment. Cover and concealment are important concepts in military reasoning, because they provide protection from the enemy and deny them information. Finding regions of the sketch that satisfy these properties is a critical spatial operation. For example, finding positions that provide concealment is an important subtask in planning (or detecting) a potential ambush.

Our position-finding technique relies on polygon operations over relevant subsets of glyphs. Depending on the constraint(s) to be satisfied, some glyphs are treated as obstacles. New regions are constructed by projections from seed locations, subject to obstacle constraints. Regions that must satisfy multiple constraints are computed by combining the regions constructed for each constraint. The polygon operations of union, intersection and subtraction thus enable the conjunction, disjunction, and complement of constraints, respectively.

For each task, domain knowledge indicates what kinds of terrain entities should be treated as obstacles. Table 1 indicates for example whether or not particular types of terrain provide cover and/or concealment, derived from conversations with military experts. These distinctions are very coarse. They do not consider the footprint of a unit on the ground relative to the size of the terrain feature, nor do they consider specific kinds of weapons or sensors, for instance. Nevertheless, this information is the kind of default categorization that is useful in early-stage conceptual reasoning. Of course, for later-stage planning more detailed information, as might be obtained via a GIS, would be appropriate. As with engineering domains, conceptual reasoning with sketch maps frames the problems, and indicates where more detailed information is needed.

Terrain Type	Concealed?	Cover?
Mountains	Yes	Yes
Hills	Yes	Yes
Open/rolling hills	No	Yes
Forest	Yes	Partial
Scrub	Yes	Partial
Jungle	Yes	Partial
Swamp	No	No
Desert	No	No
Lake	No	No
River	No	No
Bridge	No	No
City	Yes	Yes
Road	No	No

Table 1. Concealment and Cover Provided by Different Terrain Types.

Let us consider concealment to see how position-finding works. Suppose we are trying to find all regions where someone could hide from us. For each unit on our side, a new polygon is constructed by ray casting to represent the region that is visible from that unit. (If there is numerical information as to limits of visibility, the polygon is also clipped using that information.) Let V be the union of these polygons, representing all of the areas that we can see. Let W be the polygons that results from subtracting out places where units cannot be (for example, in lakes) from the entire sketch. (Notice that we allow polygons to have holes.) Then the set of polygons $W - V$ constitutes the places where an enemy could hide. Fields of fire and cover are computed similarly, using cover constraints and weapon ranges.

Path Finding

Planning and following routes is one of the major purposes of maps, and so path finding is an important capability for sketch maps. As with position finding, domain constraints are used to define what are obstacles, and hence by implication what is free space. What is an obstacle can depend on the type of unit moving: Forests are considered untrafficable for vehicles, for example, but trafficable by infantry. The costs of movement depend on the type of terrain. For example, it takes longer for infantry to move through a swamp than through a desert. In military planning, estimates of trafficability are often computed based on complex formulae involving specific details of vehicles and properties of soil and vegetation (for example, rod cone index and stem spacing)

Terrain Type	Armor	Infantry
Mountains	SR	R
Hills	R	UR
Open/rolling hills	UR	UR
Forest	SR	R
Scrub	UR	UR
Jungle	R	R
Swamp	R	R
Desert	UR	UR
Lake	SR	SR
River	SR	SR
Bridge	UR	UR
City	R	UR
Road	UR	UR

Table 2. Trafficability Constraints.

(U.S. Army 1997). This level of analysis can be automated using AI techniques, but it requires GIS data and a wealth of detail to do so (Donlon and Forbus 1999). For sketch maps, we have developed a simpler technique, for two reasons. First, sketch maps are often used in the early stages of planning, when many details have not yet been decided. Second, sketch maps are lower resolution than GIS systems, and hence are better suited for rough estimates than detailed calculations. Consequently, we use a simplified qualitative theory of trafficability, closer to the heuristic guidelines that we have seen used by commanders.

There is a standard qualitative representation for trafficability in military terrain analysis that divides space into regions that are unrestricted terrain (abbreviated *UR* or "go"), restricted terrain (abbreviated *R* or "slow go"), and severely restricted terrain (abbreviated *SR* or "no go"). Instead of demanding detailed descriptions of terrain, we assign trafficability categories based on the overall type of terrain. Since moving on foot is fundamentally more flexible than vehicles, our qualitative trafficability theory simplifies the vast array of units into two distinctions: armor versus infantry. Table 2 shows the trafficability implications of the terrain types in nSB.

Terrain regions can intersect, which slightly complicates these assignments. For example, a road over a mountain range or through a swamp is still UR, while a lake in mountains remains SR. Given a sketch, we compute a single obstacle and cost diagram by finding the maximal partition under intersection of these regions, and assigning costs to regions with two terrain types based on rules like those above.

In path finding, SR regions are treated as obstacles, and R regions are treated as higher-cost for travel than UR regions. We use A* search over the terrain Voronoi diagram to find paths. That is, junctions and edge midpoints of Voronoi cells in free space are the nodes of the graph, with the cost of each path segment being determined by the product of the path segment length and its intersection with the cost diagram. Once a path is found, it is smoothed in a post-processing step, to reduce its dependence on the details of the tessellation and thus improve its appearance.

Our Voronoi-based path planner is the third approach we tried. Originally we had used a bitmap-based approach (Forbus, Mahoney, and Dill 2001), but we were unable to make those techniques fast enough for interactive-time operation. Following Davis (2000) we also tried A* search over a quad tree representation for path planning. In the game Star Trek: Armada™ used as an example in Davis (2000), obstacles were few in number and the details of their shapes were not terribly important, so quad trees were satisfactory. In sketch maps of terrain, we found that quad trees were not as useful, since they are based on a tessellation that is not particularly aligned with the shapes of the constraining obstacles. On the other hand, the Voronoi diagram tracks the shape of the obstacles by definition, and provides reasonable resolution where it is needed.

Example: Hypothesizing Enemy Intent by Analogy

To illustrate the utility of these ideas, we demonstrate how they are used in the nSB subsystem that hypothesizes possible enemy actions. The inputs are a sketch of a precedent and a sketch representing the current situation. The output consists of a new layer, which illustrates how, in the current situation, the enemy might attempt something similar to what they did in the precedent. We have simplified the general problem in several ways. First, we only generate hypotheses about a single enemy task, for constraint solving tractability. Second, we only consider precedents and situations consisting of single battlespace states. Third, we are providing the precedent as part of the input, rather than retrieving it automatically from a memory of experiences. However, even with these simplifications, this task represents a significant advance in the state of the art in combining analogical and spatial reasoning.

Figure 2 shows an example precedent. In it, a small enemy unit (Bait) is trying to escape Alpha Battalion, which is planning to destroy it at EA Killzone. Unbeknownst to Alpha, this is a trap. Berserker Division, hiding behind the mountain range, attacks Alpha from the rear as Alpha goes after Bait, causing considerable damage. This precedent was created with nSB in the usual way, using a template-based interface to describe why the task was successful. In this case, the ambush is successful because the attacker was concealed and could travel to an engagement area on Alpha's path.

Figure 3 shows an example current situation, from another sketch. Your unit (Bravo) sees an enemy unit (Bait) trying to escape, and you are tempted to go after it. But, having heard about what happened to Alpha, you are worried. Using nSB, you can ask for hypothesized enemy tasks about the current situation based on the precedent sketched state. Its answer is shown in figure 4. There are two places that an enemy unit might be hiding to carry out an ambush similar to what happened before. The rest of this section describes how results like this one are computed.

A key aspect of our approach is the use of humanlike analogical processing for comparisons. Our goal is to ensure that, within the limitations of our representations, things that look alike to human users will look alike to the software. This *shared similarity constraint* enables the software's conclusions to be more trusted by the user. We achieve a shared sense of similarity by using cognitive simulations of human analogical processing, over representations that approximate human visual representations. The cognitive simulation of analogical matching we use is the Structure-Mapping Engine (SME) (Falkenhainer, Forbus, and Gentner 1989). SME is backed by considerable psychological evidence (Gentner and Markman 1997). Of most direct relevance here is evidence that the structural alignment processes it models are operating in human visual processing (Ferguson 2000), which makes using SME a reasonable choice for mixed visual and conceptual analogies. The shared similarity constraint has proven to be a valuable constraint on representation and reasoning choices, and has guided many of the representation and processing choices described in this article.

When intent hypotheses are requested, nSB runs SME on the two descriptions, which are states from sketches. The descriptions include both visual and conceptual information. SME derives a set of candidate inferences about the current situation based on the comparison. So far, this is simply SME doing what it normally does. Next, the set of candidate inferences is searched to see if there is a hypothesized task that acts on a blue unit. Such a task represents

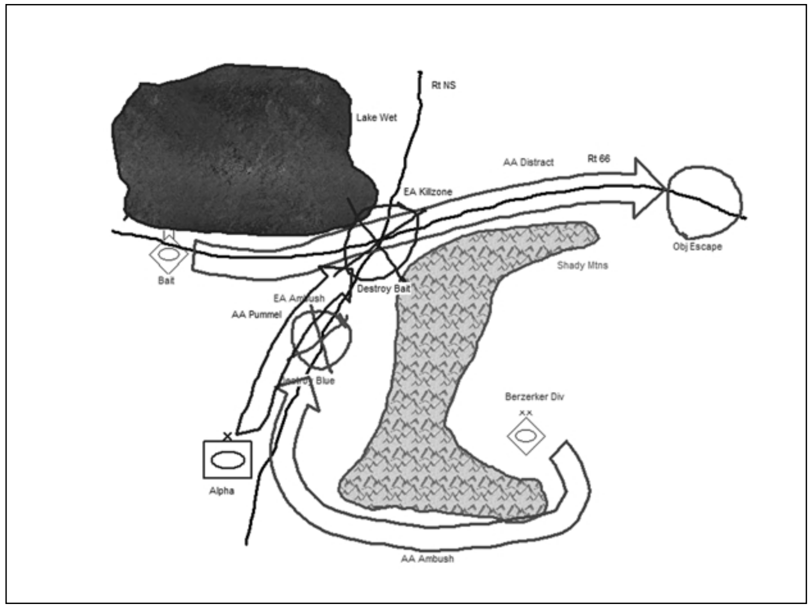

Figure 2. An Ambush.

something the enemy might be doing, if it can actually be made to work in the current situation. Specifically, a candidate inference like

```
(objectActedOn
    (:skolem Object-50) Object-64)
```

which means "there is a task that is like Object-50 (the destroy task in the precedent) that is acting on Object-64 (Bravo) in your situation" The expression (:skolem Object-50) is an *analogy skolem* (Falkenhainer, Forbus, and Gentner 1989), an entity whose existence is postulated based on the correspondences found via the comparison. A destroy task aimed at one of your units is a kind of thing that one wants to know about, so nSB attempts to construct an entity (or entities) that satisfy the constraints suggested by the analogy. This is an example of the skolem resolution problem in analogical reasoning.

If such a task is found, a new entity is created to represent that task. Such new entities are called *analogy plunks,* in honor of the notion of plunks in constraint solving (Stallman and Sussman 1977). Once a plunk has been made, SME is reinvoked to mine the analogy further by extending the mapping with the new information (Forbus, Ferguson, and Gentner 1994), which leads to new candidate inferences concerning the task. This additional information often involves new skolems (for example, the attacker, the location, and the path in this case) and their properties. These new skolems are plunked in turn, and the mapping extended further, until a complete set of constraints is obtained. A critical component of this new information is the explanation about why the

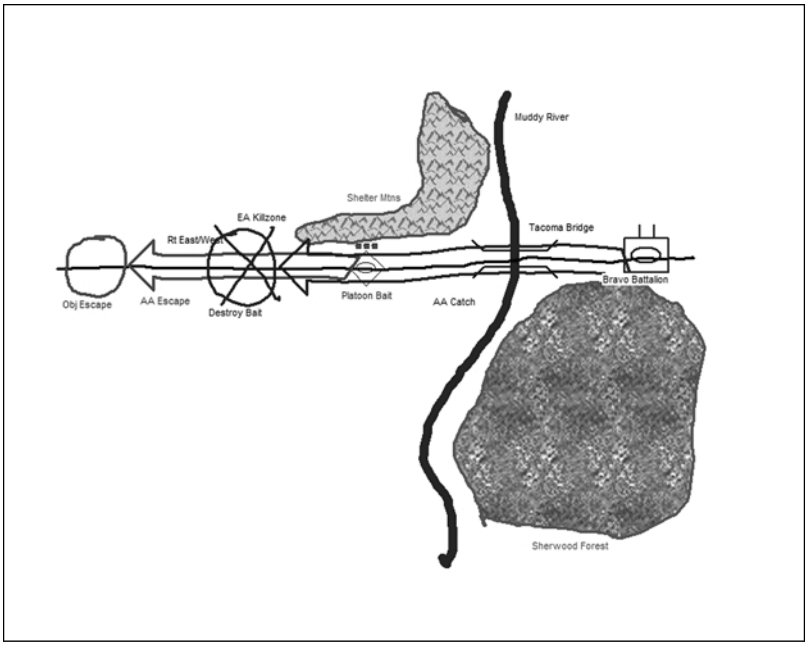

Figure 3. Current Situation.

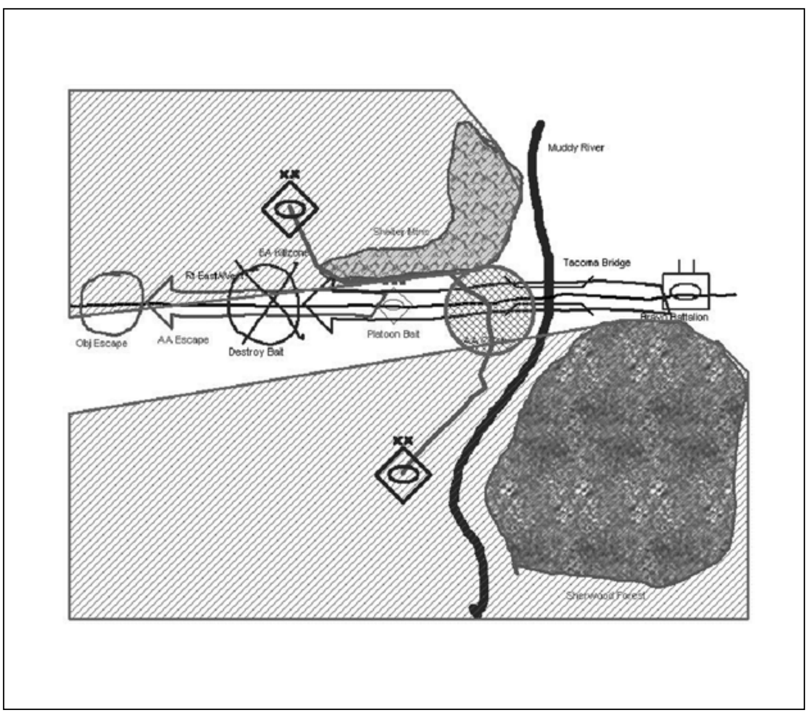

Figure 4. Two Possible Ambush Hypotheses.

task succeeded. This explanation is used to determine whether the hypothesized task is applicable in the current situation.

Once all of the information about the hypothetical task is mined from the analogy, the system must determine if the task is plausible. In the current system, we only take into account spatial constraints, ignoring factors such as relative combat power. Specifically, we solve for the locations and paths involved in the task, to see if we can find positions and a path that satisfy the task's constraints. Each combination of locations and path defines a way for that task to be executed in the current situation. For example, the engagement area for the hypothesized destroy task can be anywhere along the axis of advance for blue, the starting point for red is a region that cannot be seen by blue, and the path must start at red's location and end at the engagement area.

We use the spatial reasoning techniques described previously to solve these constraints and construct the appropriate positions and path. All consistent solutions found are presented to the user via a new layer depicting the solution, as shown in figure 4.

Note that path finding is defined with respect to start and end points, whereas the start and end locations were only constrained by regions. Since sketch maps are by nature coarse, we simply use the centroid of a region when necessary, and display both the concrete location and the constraint region. In a performance support application this is a reasonable solution because accurate optimization can depend on more information than the sketch map has, and once alerted to a general possibility, in our experience users are quick to see improvements. For creating game AIs it will be useful to optimize automatically, for example, place the division at the northern edge of the mountain and attack from behind.

Other Related Work

Qualitative spatial reasoning has often focused on mechanical systems (cf. (Forbus, Nielsen, and Faltings 1991; Stahovich, Davis, and Shrobe 1996), but some have focused on navigation and locations (cf. Kuipers 2000). None have focused on supporting the kind of complex reasoning that occurs in the military domain. Efforts in the synthetic forces literature start with GIS data rather than sketch maps. While terrain analysis is starting to be used in the computer game industry, the analyses are often carried out by hand, typically by annotating maps during level design. Winston (1982) was the first to model the use of precedents in supporting reasoning; our system uses a more sophisticated model of analogical reasoning and more complex reasoning to generate results, making it closer to case-based reasoning systems (Leake 1996).

Discussion and Future Work

We have argued that sketch maps provide an important arena for qualitative spatial reasoning, using battlespace reasoning as a source of examples. We have described the qualitative spatial representation and reasoning facilities in nuSketch Battlespace, a multimodal interface system that focuses on reasoning rather than recognition. We have shown that these facilities can be combined with analogical reasoning to do a sophisticated task, a subset of enemy intent hypothesis generation. While this article has focused exclusively on the military domain, we believe these ideas are applicable to a number of domains, such as architecture and urban planning. Planning for drainage, scenic views, and assessing environmental impacts, for instance, seem to involve similar computations and reasoning.

While these capabilities are a significant advance in the state of the art, much research remains before human-quality spatial reasoning facilities will be achieved. We see three key problems to address: (1) Optimization within constraint solutions, such as picking optimal combinations of starting and ending positions and paths. This will be very important for supporting war-gaming, where one wants to see how a plan survives the best that an opponent might throw at it. (2) Sketch retrieval, that is, automatically finding precedents (cf. Gross and Do 1995) to be used in generating enemy intent hypotheses and COAs. We plan to use our MAC/FAC model of similarity-based reminding (Forbus, Gentner, and Law 1995) for this. (3) Moving beyond blob semantics, that is, using more information about glyph shapes in matching and retrieval. Our shared similarity constraint suggests that shape descriptions need to be guided by results in visual psychology to the extent possible (Ferguson 2000; Saund and Moran 1995).

As these techniques advance, we intend to apply them in three ways. First, we plan on adding more performance support tools to nSB, such as trafficability calculators and COA critiquers, to help users generate better plans. Second, we plan on using it in intelligent tutoring systems for military training. Finally, we plan on providing interfaces to war game engines, both as a way of providing war-gaming for performance support, and as an interface to commercial computer games. Discussions are already underway with several computer game design studios concerning the use of our spatial reasoning techniques in their upcoming games. We are also currently constructing a two-player war-game, nuWar, based on nSB, to provide an experimental platform for building systems that can learn strategies and tactics by watching capable human players and for future intelligent tutoring systems.

Acknowledgements

This research was supported by the DARPA Command Post of the Future and Rapid Knowledge Formation programs, and by a National Science Foundation ROLE grant. We thank our military experts, both active duty and retired, for their sustained and valuable feedback. Special thanks to Major Robert Rasch, Major James Donlon, and Major William Turmel for helping design and implement the nuSketch Battlespace interface. John Everett provided useful suggestions about the article.

References

Alvarado, Christine; and Davis, Randall 2001. Resolving Ambiguities to Create a Natural Sketch Based Interface. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence.* San Francisco: Morgan Kaufmann Publishers.

Cohn, A. 1996. Calculi for Qualitative Spatial Reasoning. In A*rtificial Intelligence and Symbolic Mathematical Computation, Lecture Notes in Computer Science, Volume 1138,* eds. J. Calmet, J. A. Campbell, and J. Pfalzgraf, 124-143. Berlin: Springer Verlag.

Cohen, Paul R., Johnston, M.; McGee, D.; Oviatt, S.; Pittman, J.; Smith, I.; Chen, L.; and Clow, J. 1997. QuickSet: Multimodal Interaction for Distributed Applications. In *Proceedings of the Fifth Annual International Multimodal Conference (Multimedia '97),* 31-40. New York: ACM Press.

Coventry, K. 1998. Spatial Prepositions, Functional Relations, and Lexical Specification. In *Representation and Processing of Spatial Expressions,* eds. P. Olivier and K. P. Gapp. Mahwah, N.J.: Lawrence Erlbaum Associates, Inc.

Davis, I. 2000. Warp Speed: Path Planning for Star Trek™: Armada. In Artificial Intelligence and Interactive Entertainment: Papers from the 2000 AAAI Spring Symposum, AAAI Technical Report SS-00-02, Menlo Park, Calif.

Stallman, R. and Sussman, G. 1977. Forward Reasoning and Dependency-Directed Backtracking in a System for Computer-aided Circuit Analysis. *Artificial Intelligence* 9(2): 135-196.

Donlon, J. J.; and Forbus, K. D. 1999. Using a Geographic Information System for Qualitative Spatial Reasoning about Trafficability. Paper presented at the Thirteenth International Workshop on Qualitative Reasoning (QR99), Loch Awe, Scotland. 6-9 June.

Edwards, G.; and Moulin, B. 1998. Toward the Simulation of Spatial Mental Images Using the Voronoi Model. In *Representation and Processing of Spatial Expressions,* ed. P. Olivier and K. P. Gapp. Mahwah, N.J.: Lawrence Erlbaum Associates, Inc.

Egenhofer, M. (1997) Query Processing in Spatial-Query-by-Sketch. *Journal of Visual Languages and Computing* 8(4): 403-424.

10. Falkenhainer, B., Forbus, K., Gentner, D. (1989) The Structure-Mapping Engine: Algorithm and Examples. *Artificial Intelligence* 41(1): 1–63.

Feist, M.; and Gentner, D. 1998. On Plates, Bowls and Dishes: Factors in the Use of English IN and ON. In *Proceedings of the Twentieth Annual Meeting of the Cognitive Science Society,* 345–349. Mahwah, N.J.: Lawrence Erlbaum Associates, Inc.

Ferguson, R. W. 2000. Modeling Orientation Effects in Symmetry Detection: The Role of Visual Structure, 143. In *Proceedings of the Twenty-Second Annual Conference of the Cognitive Science Society.* Mahwah, N.J.: Lawrence Erlbaum Associates, Inc.

Forbus, K. 1989 Introducing Actions into Qualitative Simulation. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence,* 1273–1278. San Francisco: Morgan Kaufmann.

Forbus, K. and de Kleer, J.,1993. *Building Problem Solvers*. Cambridge, Mass.: The MIT Press.

Forbus, K., Ferguson, R. and Gentner, D. 1994. Incremental Structure-Mapping. In Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society. Mahwah, N.J.: Lawrence Erlbaum Associates, Inc.

Forbus, K.; Ferguson, R.; and Usher, J. 2001. Towards a Computational Model of Sketching. In Proceedings of the 2001 International Conference on Intelligent User Interfaces (IUI'01). Association for Computing Machinery, New York, N.Y.

Forbus, K., Gentner, D. and Law, K. 1995. MAC/FAC: A Model of Similarity-based Retrieval. *Cognitive Science* 19(2) (April-June): 141–205.

Forbus, K.; Mahoney, J. V.; and Dill, K. 2001. How Qualitative Spatial Reasoning Can Improve Strategy Game AIs: A Preliminary Report. Paper Presented at the Fifteenth International Workshop on Qualitative Reasoning (QR01), San Antonio, Texas, 17–19 May.

Forbus, K.; Nielsen, P.; and Faltings, B. Qualitative Spatial Reasoning: The CLOCK Project. *Artificial Intelligence* 51(1-3): 417–471.

Forbus, K. and Usher, J. 2002. Sketching for Knowledge Capture: A Progress Report. In Proceedings of the 2002 International Conference on Intelligent User Interfaces (IUI'02). Association for Computing Machinery, New York, N.Y.

Forbus, K.; Usher, J.; and Chapman, V. 2003. Sketching for Military Courses of Action Diagrams. In Proceedings of the 2003 International Conference on Intelligent User Interfaces (IUI'03). Association for Computing Machinery, New York, N.Y.

Gentner, D., and Markman, A. B. 1997. Structure Mapping in Analogy and Similarity. *American Psychologist* 52(1): 45-56.

Glasgow, J.; Chandrasekaran, B.; and Narayanan, N. (eds.) 1995. *Diagrammatic Reasoning*. Menlo Park, Calif.: AAAI Press / The MIT Press.

Gross, M.; and Do, E. 1995 Drawing Analogies—Supporting Creative Architectural Design with Visual References. In Third International Round-Table Conference on Computational Models of Creative Design (HI'95), M-L Maher and J. Gero (eds), 37–58. Technical Report, University of Sydney, Sydney, Australia.

Kuipers, B. 2000. The Spatial Semantic Hierarchy. *Artificial Intelligence* 119(1–2): 191–233.

Leake, D. (ed.) 1996. *Case-Based Reasoning: Experiences, Lessons, and Future Directions*. Menlo Park, Calif.: AAAI Press / The MIT Press.

Rasch, R.; Kott, Al; and Forbus, K. 2002. AI on the Battlefield: An Experimental Exploration. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence and the Fourteenth Innovative Applications of Artificial Intelligence Conference,* 906–912. Menlo Park, Calif.: AAAI Press/The MIT Press.

Saund, E. and Moran, T. 1995. Perceptual Organization in an Interactive Sketch Editing Application. In *Proceedings of the Fifth International Conference on Computer Vision,* 597–604. Los Alamitos, Calif.: IEEE Computer Society.

Stahovich, T. F.; Davis, R.; and Shrobe, H. 1996. Generating Multiple New Designs from a Sketch. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence,* 1022–1029. Menlo Park, Calif.: AAAI Press/The MIT Press.

U.S. Army 1985. *Operational Terms and Graphics*. U.S. Army Field Manual No. 101–5-1. (Marine Corps Reference Publication No. MCRP 5-2A), 21 October. Washington, D.C.: Government Printing Office.

U.S. Army. 1990. *Terrain Analysis*. U.S. Army Field Manual FM 5-33, 11 July. Washington, D.C.: Government Printing Office.

Winston, P. 1982. Learning New Principles from Precedents and Exercises. *Artificial Intelligence* 19(3): 321–350.

Kenneth D. Forbus is the Walter P. Murphy Professor of Computer Science and Education at Northwestern University. His research interests include qualitative reasoning, analogy and similarity, sketch understanding, spatial reasoning, cognitive simulation, reasoning system design, articulate educational software, and the use of AI in computer gaming. He received his degrees from the Massachusetts Institute of Technology (Ph.D. in 1984). He is a Fellow of the American Association for Artificial Intelligence and a Fellow of the Cognitive Science Society. He is an associate editor of *Cognitive Science* and serves on the editorial boards of *Artificial Intelligence,* AAAI Press, and the *Journal of Game Development.* His e-mail address is forbus at northwestern.edu.

Jeffrey Usher is a research programmer in Northwestern University's Qualitative Reasoning Group. He received his M.S. degree in computer science from Northwestern University and his B.S. degree in electrical engineering from Purdue University. His research interests include using hand-drawn sketching as computer input, spatial reasoning, analogical reasoning, attentive user-interfaces, and digital assistants. His e-mail address is usher at cs.northwestern.edu.

Vernell Chapman is a student at Northwestern University currently working toward a B.S. in computer science. His interests include procedural rendering, programming language theory, distributed systems, knowledge representation, military strategy, cinematography, and nineteenth century English literature.