Imagery as Compensation for an Imperfect Abstract Problem Representation

Samuel Wintermute (swinterm@umich.edu)
Electrical Engineering and Computer Science Department, 2260 Hayward
Ann Arbor, MI 48109-2121 USA

John E. Laird (laird@umich.edu)
Electrical Engineering and Computer Science Department, 2260 Hayward
Ann Arbor, MI 48109-2121 USA

Abstract

In this paper, we investigate the issues that arise when spatial abstractions do not capture all the details necessary for correct internal reasoning. We argue that in a general-purpose reasoning system, an imperfect abstract problem representation might be all that is available for any given problem. We propose that some forms of such imperfect representations are still useful in problem solving and can be the basis for heuristic transfer of learning between problem instances. However, there are cases when they are inadequate, such as for tasks where improper actions might have dire consequences. To compensate, an agent can use a concrete problem representation based on imagery in parallel with the abstract representation to predict the consequence of actions, thereby avoiding mistakes. A model is presented showing the usefulness of imagery to handle aspects of problem solving that the available high-level representation cannot.

Keywords: Mental imagery; spatial reasoning; transfer learning; cognitive architecture.

Introduction

In many AI systems and theories of cognition, perception is a process of taking raw sensory information, manipulating it, and creating an abstract representation that concisely captures useful properties of the world. Using this representation, internal reasoning can be performed without risking potentially costly or dangerous action in an external environment (e.g., Newell, 1990). In traditional AI systems, these problem representations resemble those used in problems like the blocks world (Figure 1). Here, the world is described by objects (the blocks and the table), along with properties relating objects, such as on(A,B). These objects and properties can be treated as logical variables and predicates. General rules can then be encoded, such as that moving any block X is possible if X is clear, rather than enumerating that fact for each block. These rules can be used to solve the problem internally. In addition, when the agent solves a problem, it can remember that solution, and if another problem is encountered with the same initial state and goal, the representation allows it to perfectly transfer the learned solution to the new problem. The precise size and position of the blocks may differ in the new problem, but that information is abstracted out of the problem representation. It is important that block A is on the table, not that it is located 2.12 cm to the left of block C.

We will call a representation like this that allows accurate internal search and transfer an *ideal* representation of the problem. However, as we shall argue later, creating an ideal representation of every problem may be difficult or even impossible for a generally intelligent agent faced with novel problems in complex, diverse environments. Thus, an agent must have the capabilities to succeed with problem representations that are less than ideal, which we call *imperfect* representations. The first objective of this paper is to further elaborate this point: that a generally intelligent agent is unable to create ideal problem representations in every case, but imperfect representations can be productively used instead.

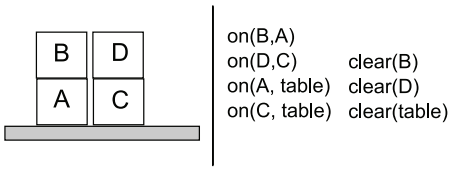

Figure 1. A simple blocks world problem.
Left: spatial representation
Right: abstract representation

Using an imperfect representation has costs, since there can be important details of the problem that are abstracted away. To compensate, an agent must have a way of retaining and using these details for precise inference. As an agent builds up its most abstract problem representation, intermediate representations may be built. In spatial perception, for example, sensory data might be used to build a representation of objects in 3D space, before being further processed into abstract symbols. Alone, this spatial representation is not very useful. Without some form of abstraction, knowledge must be learned or encoded about each specific spatial state, decreasing the generality of the agent. However, when used in concert with an abstract representation, a spatial representation can fulfill the need for precise inference. Operations within concrete representations such as this have been viewed as equivalent to human mental imagery (Lathrop, 2008). The second objective of this paper is to demonstrate how imagery can compensate for some of the consequences of using an imperfect abstract representation.

To date, research on mental imagery has focused on testing for its existence as a distinct phenomenon in humans and determining broad characteristics of mental processes involving imagery (e.g., Kosslyn et al., 2004). The issue of *why* imagery is useful, and thus why evolution has given us

the capability at all, instead of solely abstract reasoning, has received less attention. A common argument is that imagery allows problems to be represented in a form where certain types of inferences are computationally more efficient, as different information is directly available in an image than is available in a more abstract representation (Larkin and Simon, 1987, Huffman and Laird, 1992, Kosslyn et al., 2004, Kurup and Chandrasekaran, 2006, Lathrop, 2008). For example, systems for solving geometry problems have been built, and used to compare abstract and imagery-like problem solving (Larkin and Simon, 1987, Lathrop, 2008). In these experiments, the inferences possible with either representation were the same, but the imagery system was shown to be more efficient at making them.

This sort of comparison between systems is possibly a legacy of the imagery debate, where abstract reasoning and imagery are posed as alternative hypotheses to explain some capability. While the efficiency argument is persuasive (and seems to be true), using imagery to compensate for imperfections in abstract representations adds another, possibly more fundamental role for imagery: it can do things that cannot be done with the available abstract representation. Imagery and abstract reasoning are complements, not alternatives.

To explore these points, a domain and representation exhibiting imperfection will be presented. In this case, the imperfection is that objects with the same properties in the abstract representation are not completely interchangeable; it is not guaranteed that an action in the real world involving one object will have the same consequence as one involving another object, even though both objects have similar abstract descriptions. The objects and properties available are not completely equivalent to logical variables and predicates. Thus, basing action selection in the world on knowledge learned in terms of the abstract representation is not guaranteed to lead to success. However, it will be shown that despite its imperfection, the representation is still useful. In addition, to compensate for using heuristic imperfect knowledge, imagery is used to predict the consequence of actions, so that mistakes are avoided.

The model presented here is not intended to be a precise model that can be compared to human data, but can serve as the starting point for such a model.

The Modified Blocks World Domain

The argument we are presenting here is intended to be broadly applicable to any agent (human or AI) that reasons about suitably complicated spatial problems. To show the argument in the clearest possible terms, an exceedingly simple domain is necessary. Accordingly, we will use a modified version of the blocks world (Figure 2).

The basic problem is the same as in Figure 1, there is a set of blocks, and the agent's goal is to place them in a certain configuration. However, in this domain, the blocks are not freely stackable on the table. Instead, there are two fixed pegs attached to the table. Each block has a groove down the center of its back, which must be aligned with a peg

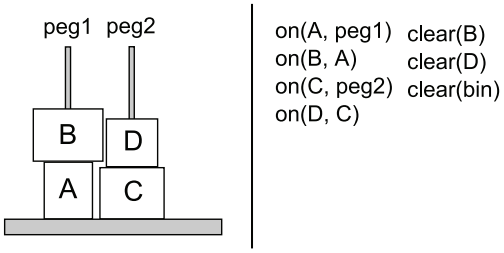

Figure 2. A modified blocks world problem.
Left: spatial representation
Right: abstract representation

when the block is placed. The effect is that all blocks must be centered relative to one of the pegs. Blocks cannot be placed on the table, but can be moved out of the way to a storage bin (not shown). In addition, the blocks are not all the same size, but vary slightly in their width and height. The pegs are close enough together that this variation can cause the blocks to interact in fairly complicated ways: whether or not a given block will fit on a given peg depends on the exact widths and heights of all the other blocks. We assume there is a fairly high cost to moving the blocks to and from the bin, so it is best to solve the problem by moving blocks between pegs, rather than simply moving them all to the bin and then building the goal configuration block-by-block. In addition, we assume that there is a very high cost to attempting to place a block on a peg where it will not fit, possibly the block jams in place and cannot be removed or might chip if it hits another block.

Consider the problem of encoding this domain abstractly, in terms of symbolic rules. The states of the problem can be described in similar terms to the states of a normal blocks world problem: which blocks are on top of which other blocks, which are clear, with the addition of a list of which pegs align which blocks. However, there is no compact symbolic description of the consequence of actions. It cannot be assumed that moving a block X to the top of a clear block Y will result in on(X,Y), it may instead result in jammed(X), depending on the exact sizes of the blocks below and in the other tower. The abstract representation of the problem does not have enough detail to accurately capture these relationships in anything more concise than a lookup table of the consequence of every action in every state.

Contrast this with the abstract problem representation used in the classic blocks world (Figure 1). The two are similar on the surface, but differ in an important way. In both problems, the state consists of a set of objects and a set of properties of those objects. In the first case, the problem can be completely solved in terms of this representation; rules can be written such that all future states of any problem instance can be predicted based on the initial state: the representation is ideal. However, this is not the case in the modified problem: the representation is imperfect.

The important difference between these two representations is that in the first case identities of objects can be treated as variables, where they cannot in the second. In the standard blocks world, the actual identities of the objects do not matter in determining the solution; what matters are the properties asserted about them. In this way,

the objects can be treated as variables. This is not the case in the modified blocks world, here, objects cannot be treated as variables. Accordingly, we will call this form of representation *object-dependent*.

While imperfect, this form of abstract representation is still more useful than none at all. In general, abstract problem representations eliminate irrelevant details, eliminate the need for detailed inference, allow faster learning, and allow learned knowledge to apply across different problem instances.[1] Consider if the only representation of the problem was at a perceptual level, such as a set of pixel-like points with no higher-level interpretation. An abstract representation, even if imperfect, provides a good measure of similarity between different states, where a concrete representation does not. Two states where the blocks are all on the table are probably very similar, even if they are not exactly the same, but could be represented by very different raw perceptions. Using an imperfect abstract representation can provide a valuable (although heuristic) measure of state similarity, as will be explored in detail in the next section.

Another, more basic, property of a good problem representation is that it can properly differentiate the states of the problem. That is, there should be no alternate paths of actions that can lead the agent to states that have the same abstract representation but are different in a way that will affect future actions. This is the Markov property, a property required for the proper use of reinforcement learning algorithms (Sutton and Barto, 1998). An imperfect, object-dependent representation can be Markovian, as long as object identities are part of the state: this is the case in the modified blocks world, the agent needs only to know the present state of the problem to make a decision; its action history is unimportant. If the problem representation used were too simple, for example, if it just included the 'clear' predicate, this would not be true.

Abstract Representation in a General System

It might be possible to create an ideal abstract representation of the modified blocks world, either by adding more properties, or by introducing new objects to the model that represent important areas of empty space. However, such a representation would be much more complicated than that needed in the unmodified problem in Figure 1, and would be specific to the problem at hand. If we are designing a general purpose AI system to solve spatial problems, or proposing a theory of human spatial reasoning, it seems inappropriate to require a completely different abstract representation for every problem: it takes complicated calculations to induce each object and property from a lower-level representation, and it is difficult to see how an AI system or person could perform exactly the calculations needed for any arbitrary problem it might encounter (see also Wintermute, 2009). This is related to the cognitive substrate hypothesis of Cassimatis (2006): it is more plausible to consider theories for general intelligence that use a small set of basic elements in different ways, rather than many different elements.

Theoretically, the best solution is then to develop a fixed, ideal abstract qualitative representation that can be used for any problem. If that is possible, the mechanism which creates that representation from perception does not need to change from problem to problem. However, the poverty conjecture of Forbus et al. (1991) states that this is impossible: if the conjecture is true (which it seems to be), "there is no purely qualitative, general-purpose, representation of spatial properties".

When a qualitative representation is augmented with a quantitative representation, though, a more complete reasoning system can be built. This idea has been previously linked to a need for mental imagery capability by Forbus (1993), who argued that a quantitative representation is necessary to compute problem-specific qualitative representations as needed. Our goal is to build a system that is as problem-independent as possible, though, so we take a slightly different approach, where an imperfect representation is built from problem-independent parts, and interaction with imagery supplements abstract reasoning.

The Model

To demonstrate these points, a simple example model has been implemented. Using this model, it will first be shown that an imperfect abstract representation can still be useful to an agent, as it can provide a Markovian state representation and a heuristic basis for transferring knowledge between problem instances. Building on this, it will then be shown that imagery capability can overcome some of the problems inherent in using an imperfect representation. Imperfections in the representation can lead the agent to mistakenly believe dangerous actions are good, but if action outcomes can be inferred through imagery, actual execution of those actions can be avoided.

Consider a model where the abstract state is represented as a set of objects, and properties describing qualitative relationships between those objects, as on the right-hand side of Figures 1 and 2. In addition, there is a concrete representation describing the same situation, similar to the left side of each Figure. In this case, this is a set polyhedrons described by 3D coordinates.

A model of this form for the modified blocks world task in Figure 2 has been implemented using the Soar cognitive architecture (Laird, 2008). Symbolic processing in Soar provides a basis for reasoning with an abstract representation. A recent extension to Soar, SVS, provides specialized processing for spatial and visual information (Wintermute and Lathrop, 2008, Wintermute, 2009). SVS contains the concrete problem representation, from which the abstract representation is built.

[1] In addition, employing abstract representations is implicitly required in any psychological model connecting language and spatial reasoning (e.g., Ragni and Steffenhagen, 2007).

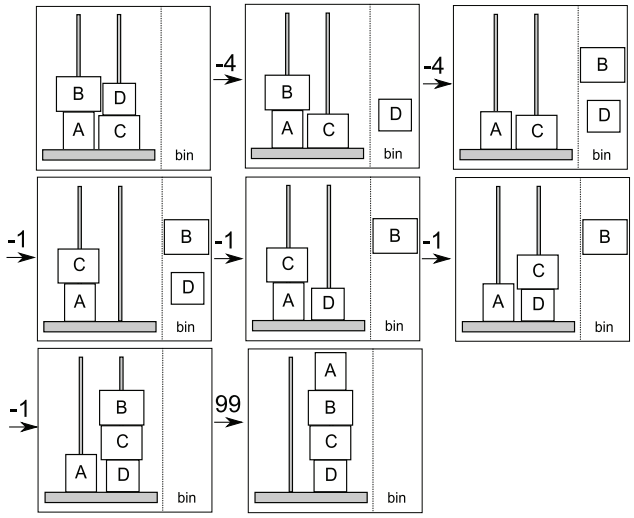

Figure 3. Optimal solution to a modified blocks world problem. The reward for the next state is shown at each state transition.

All of the primitive operations needed to construct the abstract representation are provided by built-in, problem-independent processes in Soar/SVS. These include the ability to extract simple properties such as object connectivity, distance, and direction from the spatial representation. The needed abstract properties can be built from these simpler properties, for example, on(A,B) is true if A is adjacent to B in the vertical direction.

The actions available to the agent at each state are to move each clear block (those at the top of a tower or in the bin) to the top of either tower or to the bin. Moving the same block twice in a row is prohibited, unless that is the only action possible, or a collision has occurred, in which case the colliding block is the only block that can be moved.

Symbolic rules in Soar encode knowledge about how the abstract problem representation is built from the primitives provided by SVS, and the knowledge needed to execute in the problem: which actions are available based on the current state, and whether or not the goal has been achieved.

To learn to solve the problem faster, reinforcement learning (RL) is used over the abstract representation in Soar's working memory (Nason and Laird, 2005). Through experience, the agent learns the value of executing a given action in a given abstract state. This value is in terms of rewards received for environmental interactions. In this case, the agent gets a reward of 100 for solving the problem, -1 for a normal action moving a block to one of the two possible towers, -4 for moving a block to the bin, and -200 for causing a collision by placing a block where it cannot fit. This reward structure encourages the agent to solve the problem in the fewest number of steps, avoiding the bin if possible, and avoiding collisions.

Learning and Transfer with an Imperfect Abstract Representation

If the abstract representation available to the agent is object-dependent, and therefore imperfect, how useful is

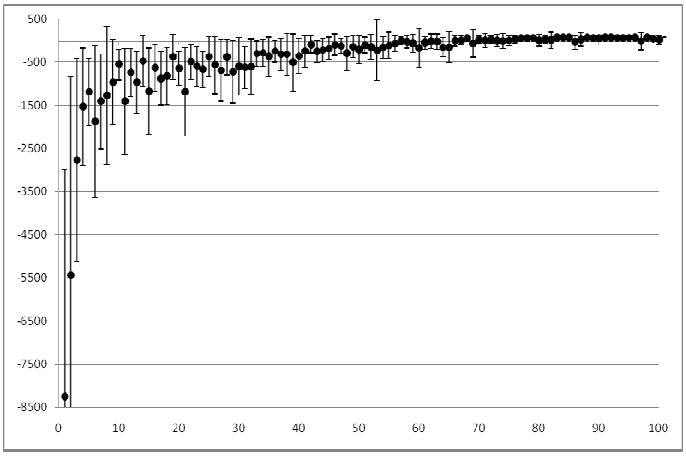

Figure 4. Performance on the problem in Figure 3, total reward (y) vs. episode (x). Results are averaged over 10 trials, error bars show standard deviation.

that representation? We will consider two dimensions in evaluating the usefulness of a representation: if it can be used by the agent to learn to solve the problem well, and if it can be used to transfer learned knowledge to other, similar problems. To better separate the effects of using an imperfect representation from the effects of using imagery, imagery is not initially used for these experiments.

The model described above has been used to solve the modified blocks world problem shown in Figure 3. The goal state of the problem is a tower, A on top of B on top of C on top of D, all on peg2. The optimal solution is shown in the figure; it achieves a total reward of 87. Ten trials of this problem were run, each trial consisting of 100 episodes over which an RL policy was learned.

Results for this experiment are shown in Figure 4. The agent solved the problem optimally as early as the 50th episode, although average performance always remains slightly sub-optimal due to the exploration policy. From this data, it is clear that the representation available to the agent was sufficient to allow the problem to be solved. [2]

A long-lived agent will encounter many different problems in its lifetime, and it is undesirable that each encountered problem must be solved completely from scratch, as in the previous experiment. Rather, a better strategy is to identify a new problem as similar to a previously-solved problem, and transfer the solution of the old problem to the new problem. One of the benefits of using an abstract problem representation is that irrelevant details are discarded, so this mapping between problem instances is simple. Mapping would be extremely difficult without an abstract representation. In an ideal representation, if the abstract state of the current problem is the same as that in an old problem, the problems can be

[2] Soar-RL's q-learning algorithm was used, with a learning rate of .3 and discount factor of .9, and epsilon-greedy exploration with an epsilon of .1. The actual results obtained are particular to the Soar agent involved, though, since Soar-RL takes into account other minor factors outside of the description provided here.

solved in exactly the same way, even though low-level details might differ.

In this case, the representation is object-dependent, but in order to map problems, it can be assumed that the objects are variables – that the blocks in the new problem are functionally the same as those in the old. The mapping is not completely reliable, but can still provide a substantial benefit. To show this, an agent was trained on a simpler instance of the problem, where no blocks were wide enough to cause collisions. The initial state and goal were otherwise the same. After 250 episodes, the policy learned was transferred to the problem in Figure 4, by assuming the objects involved were the same. The agent was again run for 100 instances, as shown in Figure 5. As can be seen, although the policy initially caused a large negative reward, the agent quickly learns to solve the problem well, much faster than when it is not provided with transferred knowledge (in Figure 4). This shows that, even though the agent does not create an ideal representation of the problem, the abstract nature of its representation can provide a substantial transfer benefit. Including objects in the representation, even though they can't be treated exactly as variables, is still very valuable.

Imagery Compensates for an Imperfect Abstract Representation

While the transfer performance above is much better than what is possible with no prior knowledge, there is certainly room for improvement. For a long-lived agent encountering many problems, the common case for performance might well be the far-left data point on Figure 5, the very first time a new problem instance is encountered. There does not seem to be a straightforward ideal abstract representation of the modified blocks world problem that could be built by a task-independent perception system. Because of this, there is no reason to expect the agent could somehow improve its abstract representation to be able to optimally solve any new instance of the problem on the first try.

There is some high-level knowledge the agent could use outside of the RL policy, though. It is clear that causing a collision by attempting to move a block where it cannot fit is never a good idea; the action always results in a huge negative reward and must be undone. The RL policy captures this implicitly through the learned values of certain actions, but the knowledge is not well-transferred between problems, since whether or not a collision will occur depends on the exact sizes of the blocks in the problem, information thrown out when making the abstract representation. However, if the agent has some means for making collision predictions at a concrete spatial level, it is possible that collisions can be foreseen and avoided.

In this case, the agent described above can use the imagery functions in SVS. SVS has built-in means to interpret simple commands such as "imagine block B on top of block A" – it simply copies the polyhedron describing the block to the new location. In this way, the consequences of an action can be determined by creating an image of its result and inspecting the imagined scene (using the same means as regular perception) and detecting any collisions.

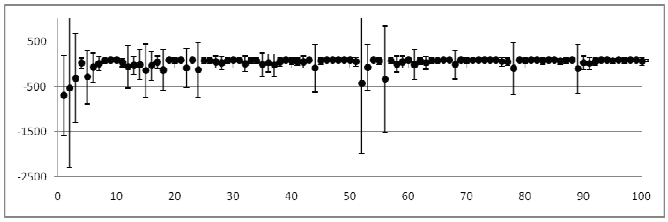

Figure 5. Performance on the problem in Figure 4, after transferring a policy from another instance. Results are averaged over 10 trials. (same axes and scales as Figure 4)

Accordingly, the agent has been modified to imagine the consequence of each action before executing it. If a collision results, it chooses the next-best action (according to the RL policy) instead. This agent was tested on its first encounter with the Figure 3 problem, after learning a policy over 250 episodes of a simpler (collision-free) problem. 35 trials were done, each with its own learned policy. The same policies were also tested without imagery. In 14 trials, the imagery agent performed optimally in its first encounter with the new problem. For 33 of 35 trials, more reward was received by using imagery than not. In one trial, both performed equally, and in one trial the non-imagery agent performed better (exploration was possible; in this case, the imagery agent explored a very unproductive path).

Discussion

To summarize the argument presented above:

- Abstract, qualitative representations of spatial problems are useful.
- A generally-intelligent agent must solve many different types of spatial problems.
- The poverty conjecture implies that there is no single qualitative representation that could perfectly capture all of those problems.
- It is unlikely that wildly different problem-specific abstract representations could be built every time a new problem is encountered.
- This leads us to look at the possibility of using imperfect abstract representations which might be built out of problem-independent parts.
- One type of such representation is an object-dependent representation, where it cannot be reliably assumed that objects with the same properties are interchangeable.
- An object-dependent representation can still be useful to differentiate states, and to provide a basis for heuristically transferring knowledge between problem instances.
- However, since this transfer is heuristic and inexact, high-cost, useless actions could still be performed in new problem instances.
- If an imagery system is used, and the consequences of actions can be predicted, those actions can be avoided.

The example agent provides a simple illustration of these points, but much more work must be done to determine how

generally the principles apply, and what bearing this has on psychological models.

AI Concerns

Many of the principles behind the design of this system result from the goal of building a general, problem-independent AI system. Two of the most important claims toward this goal are that a fixed system can build a useful (though possibly imperfect) qualitative representation for any arbitrary problem, and that relevant imagery operations exist for arbitrary problems.

Substantial further work is needed to adequately prove these claims. For the first claim, we know that building an imperfect representation is much simpler than building a perfect representation. In addition, the same basic system used here has been successfully used for several different tasks (e.g., car motion planning in Wintermute, 2009) where states have been built out of the same basic elements (object intersections, directions, and distances) without the need for architectural modification. How far this same system can continue to be used remains to be seen.

For the second claim, that imagery can be used in a problem-independent manner, again, substantial further work is needed. A general argument can be made that complicated actions can be represented more easily through precise forward simulation in imagery than through more abstract means (Wintermute and Laird, 2008, Wintermute, 2009). However, in the case covered here, motion simulation was not used. Rather, the agent used a simple fixed language to describe the consequence of an action ("imagine the A centered on top of B"). The implications of this approach have yet to be fully characterized.

Imagery in Psychological Models

The model presented here is not intended to be a precise psychological model, but even without precision, the model does make some basic psychological predictions: people will tend to imagine the consequences of actions when the problem cannot be easily captured in a simple abstract representation, and when the wrong move could be costly. Imagery in this way has a functional role in planning.

A similar hypothesis, motivated by behavioral data from a motor planning experiment, was presented by Johnson (2000). Johnson's hypothesis is that "movement selection involves mentally simulating candidate response options in order to evaluate their consequences". While Johnson's work involves judgment over intrinsic properties of motor imagery (the comfort of a certain grip), and we have instead looked at spatial aspects of imagery, the principles here still apply. Specifically, a plausible argument for *why* people would use motor imagery in planning is that the abstraction of the problem available to the human reasoning system does not contain enough information by itself to determine whether a certain action in the experiment will be comfortable or not. This imperfection in the abstract representation is present because the human brain's abstraction-performing machinery has not evolved specifically for the problem tested in the experiment, but instead to cover a wide variety of tasks.

Acknowledgments

This research was funded by a grant from US Army TARDEC.

References

Cassimatis, N. L. (2006). A Cognitive Substrate for Achieving Human-Level Intelligence. *AI Magazine*, *27*(2).

Forbus, K. D. (1993). Image and substance. *Computational Intelligence*, *9*(4), 377-378.

Forbus, K. D., Nielsen, P., & Faltings, B. (1991). Qualitative spatial reasoning: the CLOCK project. *Artificial Intelligence*, *51*(1-3), 417-471.

Huffman, S., & Laird, J. E. (1992). Using Concrete, Perceptually-Based Representations to Avoid the Frame Problem. In *AAAI Spring Symposium on Reasoning with Diagrammatic Representations*.

Johnson, S. H. (2000). Thinking ahead: the case for motor imagery in prospective judgements of prehension. *Cognition*, *74*(1), 33-70.

Kosslyn, S., Thompson, W., & Ganis, G. (2006). *The Case for Mental Imagery*. New York: Oxford U. Press.

Kurup, U., & Chandrasekaran, B. (2006). Multi-modal Cognitive Architectures: A Partial Solution to the Frame Problem. In *Proceedings of The 28th Annual Conference of the Cognitive Science Society* .

Laird, J. E. (2008). Extending the Soar Cognitive Architecture. In *Proceedings of the First Conference on Artificial General Intelligence*.

Larkin, J. H., & Simon, H. A. (1987). Why a Diagram is (Sometimes) Worth Ten Thousand Words. *Cognitive Science*, *11*(1), 65-100.

Lathrop, S. D. (2008). *Extending Cognitive Architectures with Spatial and Visual Imagery Mechanisms*. PhD Thesis, University of Michigan.

Nason, S., & Laird, J. E. (2005). Soar-RL: integrating reinforcement learning with Soar. *Cognitive Systems Research*, *6*(1), 51-59.

Newell, A. (1990). *Unified theories of cognition*. Harvard University Press Cambridge, MA, USA.

Ragni, M., & Steffenhagen, F. (2007). Qualitative Spatial Reasoning: A Cognitive and Computational Approach. In *Proceedings of the 29th Annual Conference of the Cognitive Science Society*.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press.

Wintermute, S., & Laird, J. E. (2008). Bimodal Spatial Reasoning with Continuous Motion. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI-08)*. Chicago, IL.

Wintermute, S., & Lathrop, S. D. (2008). AI and Mental Imagery. In *AAAI Fall Symposium on Naturally Inspired AI*.

Wintermute, S. (2009). Integrating Reasoning and Action through Simulation. In *Proceedings of the Second Conference on Artificial General Intelligence*.