Center for Cognitive Architectures
University of Michigan
2260 Hayward Ave
Ann Arbor, Michigan 48109-2121

TECHNICAL REPORT
CCA-TR-2006-01

INCORPORATING VISUAL IMAGERY INTO A COGNITIVE ARCHITECTURE:

AN INITIAL THEORY, DESIGN, AND IMPLEMENTATION

Investigators

Scott Lathrop
John Laird

22 May 2006

Abstract: *Humans use visual imagery for a variety of purposes including reasoning about visual properties and spatial relationships, anticipating future events, recalling past experiences, and memorizing concepts and facts. Cognitive architectures have traditionally ignored visual imagery, but we have started to explore this phenomenon in the context of Soar. Soar provides the underlying control, memory, and learning mechanisms while visual imagery provides efficient representations and processes for finding an object's visual features and spatial relationships not explicitly encoded with symbols. This multi-modal approach enables the architecture to use the most efficient representation for the appropriate computation and requires less domain knowledge for visual-spatial type environments and tasks. This paper outlines our theory and the corresponding architecture, design, and implementation. We present initial results in two small problem domains.*

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Cognitive architecture research has traditionally focused on symbolic representations and computations. Other approaches in Artificial Intelligence and computer science have used quantitative, analog representations. The task domain often dictates what representation is required for the given problem, but for systems that attempt to generalize across a wide spectrum of tasks, multiple representations allow a system to take advantage of the most efficient and appropriate computational mechanism. This multi-modal approach is relevant for both scalability and robustness reasons.

This research effort investigates a multi-modal approach in a cognitive architecture using the human phenomena of visual imagery as our motivation and Soar as the underlying architecture. Soar provides the control of central cognition, traditional short term and long-term memories, problem space search, decision making, and learning mechanisms with general symbolic representations. Visual imagery, a separate cognitive component but controlled by central cognition, provides another source of knowledge represented in a modality specific, depictive, format. This multi-modal approach enables the architecture to use the most efficient representation as the situation dictates and requires less initial domain knowledge for visual-spatial type environments and tasks.

The goals of this research are the following:

(1) To determine how to incorporate the phenomena of visual imagery within the context of a cognitive architecture constrained by both psychological and biological evidence.
(2) To improve the capability of Soar, specifically in the area of spatial reasoning.
(3) To establish the class of problems where an imagery representation is computationally more efficient than a pure symbolic representation.

Providing agents with the functionality enabled by visual imagery would enhance cognitive architectures. Glasgow has already shown that computational imagery models apply to molecular scene analysis [1]. Robots and robotic devices that require higher level cognitive capabilities, decision support tools for military or transportation planning, and intelligent geographical information systems (GIS) are other applications that would also benefit. Other areas in computer science that this research may contribute to include simulations, games, spatial databases with three-dimensional models, visualization tools, information system monitoring and analysis software, and medical imaging software.

So why do we use imagery? From a computational viewpoint, we hypothesize that visual imagery provides more efficient structures and processes for specific tasks. The most predominate tasks are determining an object's visual properties (size, shape,

color, and texture) and reasoning about spatial, topological, and geometric relationships that the agent has not previously encoded symbolically.

Child development psychology hypothesizes that young children rely predominately on imagery when learning new information or memorizing concepts. Over time these concepts are more abstractly encoded in memory using a propositional or verbal representation [2]. In general, people use visual imagery to assist them with different types of cognitive tasks to include reasoning, planning, learning, and rehearsal. For example, consider the following questions. How many rooms are in your house? What is larger a baseball or a softball? What is closer to Ann Arbor, Michigan: South Bend, Indiana or Columbus, Ohio? Rather than mentally retrieving the empirical data, most people report creating a visual image of the scene and counting the number of rooms, comparing the sizes of the two balls, or estimating distance between the cities from their imagined map. Of course, being able to perform these tasks using visual imagery assumes:

(1) You have previously encoded a somewhat accurate visual representation of each object in the question and stored these encodings somewhere in memory.
(2) You have previously encoded the relationships between objects (such as the relative relationships between rooms in your house or between the cities) and stored these encodings somewhere in memory.
(3) You recognize that your current goal is to determine the number of rooms, compare the two balls, or determine the closest city to Ann Arbor.
(4) You are somehow able to access the previous encodings, combine them into an image, and then "examine" that image to form an answer to the query.
(5) You know where in memory to place the image's answer so you can formulate a verbal response.

These assumptions form the basis of our theory, which we describe in more detail in chapter 2.

To further elaborate the direction of our research, highlight imagery's depictive nature, and motivate future work, we provide an example from a military domain. Our current implementation is not advanced enough to encode all of the information in this domain, but this problem serves as an example of what we hope to achieve using mental imagery in the coming years (i.e., as part of a Ph.D. thesis). The domain consists of an Army scout platoon conducting a reconnaissance mission in order to determine the enemy's location, size, and orientation. The diamond shape objects in Figure 1 represent a single scout vehicle (six in all), the rectangle represents an enemy force of unknown size, the squares are buildings, the ellipse figures are hill masses, and the black squiggly line is a road. The north direction points toward the top of the figure. The scout platoon, consisting of three sections of two vehicles each, is moving north when a scout in section B reports contact with an enemy force to the north. Section B

sends the report to the platoon leader (our agent) who is currently in section C. The agent is not in position to observe section A, section B, the possible enemy, or the buildings. It can only see portions of the hill to its front. Its working memory contains the symbolic information shown in Figure 1. The agent now has to make a decision and issue orders to its subordinates.

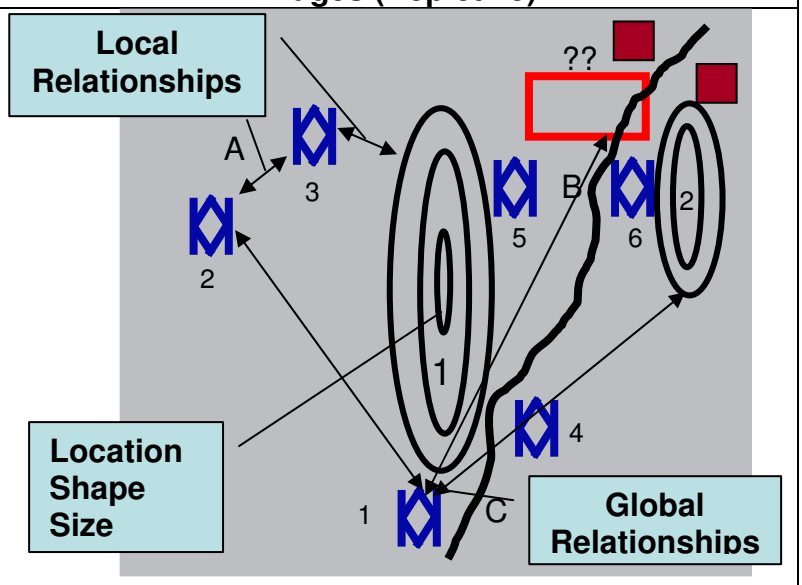
Symbols (Descriptive)	Images (Depictive)
Object (scout-1) …. Object (scout-6) Object (Hill-1) Object (Hill-2) Composite-Obj(section-A) Composite-Obj (section-B) Composite-Obj (section-C) Contains(section-A, scout-2, scout-3) Contains (section-B, scout-5, scout-6) Contains (section-C, scout-1, scout-4) north-of (section-B, section-C) west-of(section-A, Hill-1) south-of(section-C, Hill-1) west-of(section-B, Hill-2) north-of(enemy, section-B)	

Figure 1: Descriptive vs. Depictive Example

Symbols (Descriptive)	Visual Images (Depictive)
Explicit, abstract objects	Implicit, concrete, objects
Explicit relations	Implicit relations
Location, size, shape, orientation optional	Location, size, shape, orientation matter
Computationally efficient for maintaining semantic interpretations	Computationally efficient for maintaining global relationships

Table 1: Descriptive and Depictive (Adapted from Kosslyn [3])

Table 1 distinguishes the difference between a symbolic and visual representation. There are several symbols designating the scout vehicles, the sections, the two hills, and local relationships between the objects. The symbols denote an explicit, abstract object or relationship and represent the explicit knowledge available to the agent.

Assume for a moment that the agent can recreate the visual image in Figure 1. How would the agent do it? The agent knows its own location from a combination of its visual percepts and a map or GPS. Based on its viewpoint, it can construct an image of the situation using the explicit symbolic representations in working memory and underlying visual representations of each object stored somewhere else in memory.

The agent's image will not be a perfect replica of the current situation but detailed enough to infer new knowledge and assist in its decision making process.

There are several implicit, concrete objects and relationships in the image. For example, there are intersection points between the enemy red rectangle and the road. There is distance, direction, and angles between each of the objects. The objects have shape that encompasses space within the image. Each of these objects is located at a specific (x, y) location. There is a topological relationship with section C, section B, and the enemy force located along the same path (the road).

An important observation from this example is that a symbol denotes an abstraction of some object or thing. The agent (or knowledge engineer) abstracts away certain features that they believe are not relevant to the current problem. Visual imagery potentially provides a mechanism to reacquire concrete knowledge. In our example, the agent abstracts away features such as direction, size, shape, and location. It does not know the global relationships between objects. It could have maintained these relationships using symbolic processes, but at what cost? Keeping track of all the relationships between objects is an exponential cost in both time and space. In the visual image, you compute these relationships as required based on the visual location of the objects in question using special-purpose procedures that are optimized for computing spatial relations.

Another observation from this example is that the agent, using the image, can mentally simulate situations, and infer new relations. Suppose that the agent is trying to decide if it should move section A north to develop the situation further. Using the image, it could simulate the movement and infer that the new relationship between the suspected enemy force and section A does not intersect hill 1. Note that the shape and size of hill 1 makes a difference in making this calculation. Sending this knowledge back to central cognition and combining it with specific domain knowledge, enables the agent to infer that this action exposes section A to possible enemy fire.

Psychology has approached mental imagery from several angles including behavioral, cognitive, and neuroscience perspectives. In the early 20th century, behaviorists largely dismissed mental imagery, as it was only through introspection that one could describe their visual experiences. The behaviorists were very skeptical of introspection as a valid, scientific method. In the 1960s, however, cognitive psychology revived the role of mental imagery. Some form of a mental representation was the only plausible conclusion for cognitive theorists to explain the results of their imagery experiments. As a result, they evolved theories of the functional and computational mechanisms associated with mental imagery [4].

Psychologists now (almost) universally accept mental imagery as a cognitive function. The major question, and what psychologists have debated for the past 30 some years, is the format of these internal representations [3, 5, 6]. On one side of the argument, you have those who believe mental images are no different from other propositional or symbolic thought processes. The propositions, they argue, include not only verbal

semantics but also perceptual information [5, 7, 8]. At the other end of the imagery representational spectrum, there are those who hold that visual mental images are *quasi-pictorial* or *depictive* in nature, have an inherent underlying visual representation, and share mechanisms with vision. In the midst of this representational debate, Anderson raised a key observation that the representation is dependent on the processes that compute over those structures, and any theory must adequately articulate how the processes work in conjunction with the representation [6].

Kosslyn and those who share his view of imagery being a *quasi-pictorial* representation, profoundly influence our theory and implementation partly because we have pushed Soar to its limits using pure symbolic representations and partly because we believe visual imagery representations perform some calculations much more efficiently than pure symbolic approaches. Although the heart of our system is non-symbolic, we view imagery as multi-modal. The integration between the symbolic and the analog representations is vital to its capability.

Recent neurological evidence also affects our theory. Brain imaging studies demonstrate that visual imagery shares structures associated with vision. These studies show that the parietal cortex, temporal cortex, and occipital cortex are active during mental imagery experiments (Figure 2). The researchers believe the temporal cortex stores the visual information (the what) while the parietal cortex stores the spatial (where) information. During visual imagery, the two representations are combined in the occipital cortex (specifically the visual cortex) to form the mental image [9, 10].

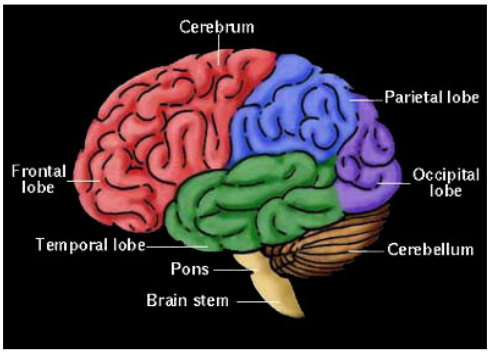


Figure 2: Brain Regions

AI researchers have made enormous contributions in areas related to visual imagery, but there have been few efforts to unify imagery within the constraints of a cognitive architecture. AI work in the areas of computer vision, robotics, spatial reasoning, and diagrammatic reasoning are important for the realization of a visual imagery model but do not singularly define it.

Kuipers' Spatial Semantic Hierarchy (SSH) is possibly the most complete spatial representation model. The SSH is a model of large-scale space (beyond an agent's sensory horizon) used primarily for robotic navigation and path planning. SSH is a multi-level representation with each level having both qualitative and quantitative modalities. The global knowledge of the environment increases as you move up the

hierarchy from very specific control laws, to causal schemas, to topological maps of places, paths, and regions. At the highest level, or what Kuipers calls the global metrical map, SSH combines the qualitative (symbolic) topological relationships with the local quantitative (analog) representations to form a metrical map with one global frame of reference. The merger of qualitative and quantitative representations is similar to our approach in combining symbolic local relationships with an object's geometric data. Kuipers claims that the inclusion of both qualitative and quantitative representations and their associated dependencies is important and a significant change from previous work [11]. These views are consistent with our multi-modal representation scheme for visual imagery although we are incorporating it within a general cognitive architecture rather than strictly focusing on specific robotic navigation schemes. In addition, our process is primarily top-down whereas Kuipers model is bottom-up with sensory input providing the local metric information.

Perhaps most influential in our research is Chandrasekaran's related work in diagrammatic reasoning and Forbus' efforts in qualitative spatial reasoning. Chandrasekaran's group combines diagrammatic reasoning with a problem solver. Similar to our approach, their goal is for an agent to solve problems using the representation most suited for the situation. Chandrasekaran concludes that diagrams provide emergent objects and relationships for "free" which is an attractive property for using diagrams in novel situations [12, 13]. Their perceptual and action routines map onto our notion of internal perceptions and internal actions, and the use of an alternate, non-symbolic visual representation to achieve gains in computationally efficiency is our primary motivation for incorporating imagery into a cognitive architecture.

Forbus and colleagues have developed sketching tools that assist decision makers with spatial reasoning. Also using a multi-modal approach, they have incorporated visual reasoning techniques to compute qualitative, spatial descriptions. Their combinations of viewpoint and type of sketch to describe the relationship between the visual and spatial frames of reference influence some of our symbolic representations in Soar and mapping to the imagery system [14, 15].

Cognitive architectures have largely ignored visual mental imagery and more specifically, depictive, metrical representations. Perhaps the most well known architectures, Soar and ACT-R [16-18], have traditionally focused their efforts on what Newell calls central cognition [17]. Other architectures, such as EPIC, emphasize the perceptual and motor components as much as central cognition but do not address visual imagery [19]. More functionally oriented architectures (rather than psychological plausible) such as ICARUS and Soar must use external processing or functions if heavy calculations are involved. Soar in particular has had numerous success in modeling human behavior to include pilots flying fixed-wing or rotary-wing aircraft, and soldiers conducting Military Operations on Urban Terrain (MOUT) [20-22]. In spite of these accomplishments, Soar's knowledge engineers admit to having to program the spatial and visual knowledge in an ad-hoc manner (for example, see [22]).

Kosslyn composed a detailed theoretical model and created a computational implementation to simulate and test his ideas but not within the constraints of a cognitive architecture [3, 23]. Glasgow and her colleagues built a computational model of imagery for a molecular scene analysis application. Their implementation, specified using array theory, included three types of representations for imagery to include a long-term memory and two distinct working memories for the visual and spatial components of imagery. Their long-term memory used a semantic network implemented with frames and "a-kind-of" and "part" hierarchies. Their visual working memory consisted of three dimensional spatial occupancy arrays. Each cell in the array corresponded to the electron density from X-ray diffraction experiments. The spatial working memory used symbolic arrays to encode and store the relationships between molecules. Their processes consisted of functions to manipulate each of these structures, to include creating, modifying, or deleting information in the frame representations, transforming visual representations, and functions to perform spatial reasoning [1].

There are many similarities between Glasgow's work and ours, to include initial separate representations for both visual and spatial information, the use of hierarchies and object composition and the treatment of images as three-dimensional. There are, however, two major differences. First, they built their model for a specific application while we are taking a more general approach. While they took significant strides to incorporate some of the key findings in mental imagery from psychology, they did not design it with the overarching constraint that it had to function within the framework of a cognitive architecture. Second, they represent their spatial information symbolically whereas we use a bi-modal approach storing local object relationships symbolically and realizing global spatial relationships using an analogous, metric representation from the combination of both visual and spatial sources of knowledge. Our scheme is more closely related to Finke's argument that imagery is manifested in a continuous space [24].

More recently, Barkowsky and his colleagues have specified the basic requirements for computational modeling with mental images for their conceptual architecture, *Casimir*. Their specifications include:

(1) Hybrid representation formats for both propositional and analogical structures
(2) Construction of mental images from pieces of knowledge retrieved from memory and dynamically added to increase specificity.
(3) Top-down and bottom-up distributed processing.
(4) Close coupling of the functional subsystems in mental imagery and visual perception [25].

Though it is unclear how some of these requirements are realized within a cognitive architecture, they are consistent with our theory. We address some of these issues in the next chapter.

CHAPTER 2

REQUIREMENTS, CONSTRAINTS, AND THEORY

The previous chapter outlined the background and motivation for visual imagery. We will now discuss our basic theory in the form of our requirements, constraints, and theory. The purpose of this chapter is to provide a basis for our architecture, design, and implementation decisions set forth in chapter 3.

When we speak of mental imagery, we are referring to it in the sense of visual imagery. That is, our research is centered on the visual modality of imagery rather than other forms of imagery that have been postulated such as auditory, olfactory, haptic (touch), or kinesthetic (or motor) imagery. Our definition of visual imagery is based on similar definitions found in [1, 2, 4, 9, 24, 26] but refined to emphasize our view of its role as a component in the cognitive architecture.

> Visual imagery is the structure and processes enabling the construction and interpretation of a *visual representation* in the absence of a visual stimulus. The cognitive architecture uses it as a *computationally efficient* process to obtain knowledge about objects' *visual features* and *spatial relationships* from both an egocentric and allocentric viewpoint.

We elaborate on two key points from our definition.

(1) Visual representation. The representation is metric and concrete. Central cognition constructs the image in a top-down manner with knowledge amalgamated from one or more memories.
(2) Computationally efficient. The structure and processes associated with visual imagery are computationally more efficient than symbolic operations when computing visual features and global relationships.

REQUIREMENTS AND CONSTRAINTS.

Our first and overarching constraint is that the visual imagery component must be integrated with the total cognitive architecture. The various architectural components have to constrain the design of each other and imagery is no exception. It cannot operate on its own accord, create whatever image it deems important, and periodically provide central cognition with its inferred knowledge. Rather central cognition must control visual imagery, as it is aware of the current problem space and associated goals. Central cognition triggers the construction of images and focuses visual imagery's attention during search to bring all knowledge to bear in support of the current problem space (Figure 3).

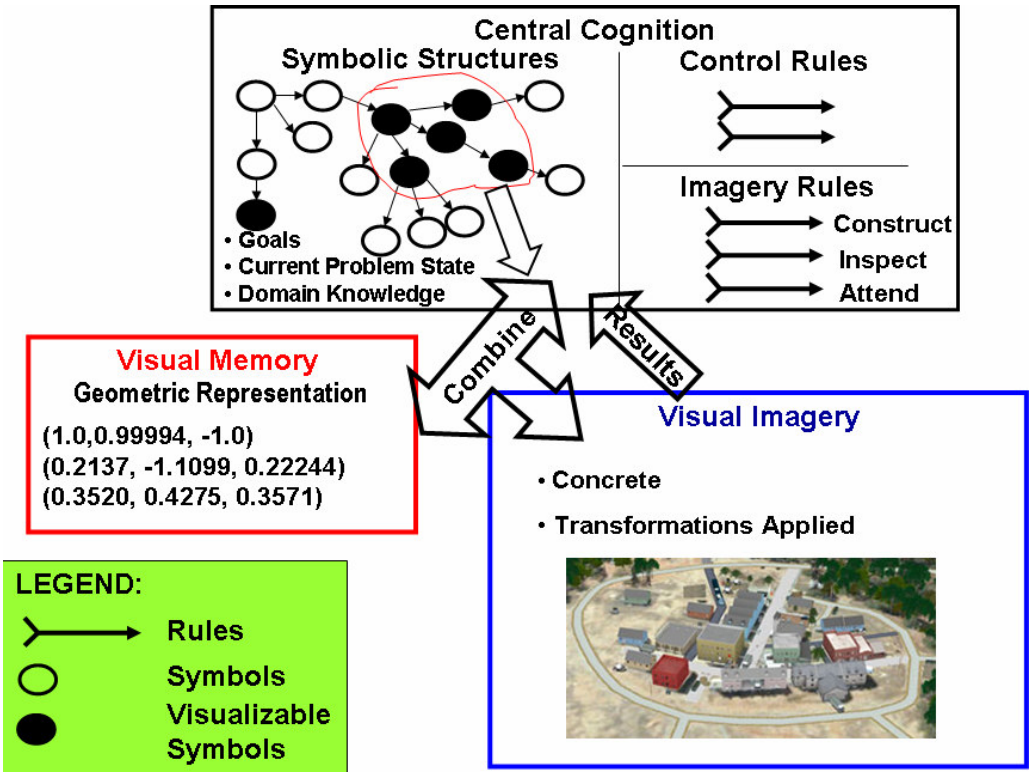


Figure 3: Central Cognition Controls Visual Imagery

Symbols provide the neutral, stable medium as theorized by Newell [17]. The result of vision processing divulges each object's underlying 3D geometric representation as ascribed by Marr [27]. Visual imagery constructs the image by assimilating the symbols and geometric representations, applies transformations based on local, symbolic relationships, and generates the image. The visual imagery system then acts as a straightforward computational device that searches the image for the desired knowledge and converts the depictive information back into an abstract symbolic representation for central cognition.

The imagery system is not just the visual imagery component. It also includes declarative symbols designating the "imaginable" or "visualizable" abstract objects (those objects so designated as having a visual representation) and the procedural symbolic expressions that control the construction and search processes. The visualizable object symbols are hierarchical. Each visualizable object may contain a collection of other visualizable objects designating the object's parts. If the object is a primitive part (i.e. it contains no subparts), then the object has an underlying geometric representation in visual memory.

A primitive object's geometric structure contains its shape (expressed through local coordinates), color, and texture. We impose a 3D representation, as it is consistent from a cognitive psychology and computer vision perspective. Pinker and Finke advocate a 3D representation for visual imagery [24, 28]. Marr and Ullman describe the vision process as transforming a raw, 2D primal sketch to a 3D representation [27, 29]. The end state representation for visual processing then is the start state for visual

imagery. We do not limit ourselves to 3D however. There may be instances where it is computationally advantageous to render the image to a 2D representation.
Those visualizable objects designated as composite objects (i.e. contain other visualizable objects) do not have an underlying geometric representation. Visual imagery builds and stores a composite object's metric representation by composing its subparts. Each composite object has symbolic structures designating the associations between pairs of sub-objects. These relationships are spatial (i.e. left of, right of, in front of), topological (i.e. connected, overlap, inside), or geometrical (i.e. parallel, perpendicular, intersect) and constrain the relative location in space between two objects. Symbolic relationships can be abstract in nature (left-of, right-of, in-front-of, behind) or concrete (a vector specifying the relationship). Specific continuous values such as a coordinate location, distance, or angle of rotation, may further constrain the relationship. Our hierarchy and composition constraints are motivated primarily by Kosslyn who advocates the construction of an analog image from the composition of an object's parts [2, 3, 23]. Other imagery researchers also support this view [1, 25].

The imagery productions control the order in which the system adds objects to the image, focus the attention of the inspection process, and create new symbolic structures from knowledge returned from the imaging system. Transfer of knowledge between the various components has to be encoded and decoded in a representational form that the module understands.

The visual imagery component constructs the surface representations (or blobs) first and if the image inspector (search engine) requires more detailed representations to access the desired knowledge then the object's detailed geometric representations are applied. Image construction stops when the visual structure contains enough knowledge to answer central cognition's query or the image contains all of the objects associated with the composite object. The image inspector searches the visual representation for an object's visual properties or relationships. Central cognition uses images to answer specific queries (deliberate) or to look for emergent objects and relationships (automatic) in the context of the current problem space [2, 3, 13, 30].

Our final constraint is important as it relates to our assumption that visual imagery is a computationally efficient process. The visual representation has to be stored in a data structure(s) that supports computationally efficient processing and is flexible enough to render the structure to a 2D image if required.

As an example, Figure 3 shows the image generation of a set of buildings. Central cognition stores the symbols representing each visualizable object in the domain (buildings, roads). Some of the visualizable symbols contain other visualizable symbols (i.e. the image has roads and buildings, the buildings have floors, the floors have rooms, etc.). In addition these composite objects have knowledge concerning the relationships between pairs of their sub-objects (i.e. the set of buildings are inside the road, building A is in front of building B, etc.). Visual memory contains the underlying geometric representations for the primitive objects (buildings and the road). Central cognition's control rules trigger visual imagery rules. These rules control the construction and

inspection of the generated image. The visual imagery component constructs the image by combining the declarative, symbolic knowledge from central cognition with the concrete metric knowledge from visual memory. It then searches through this image to answer the specific query or find emergent objects and relationships and returns the results of this query back to the symbolic structures in central cognition.

THEORY.

Given the constraints and requirements, we can now briefly explain our five hypotheses concerning the nature and use of visual imagery. We will then follow with a summary of our theory.

(1) Symbolic representations are sufficient but are not the most efficient representation (performance, space) for visual and spatial relationship knowledge.

(2) Visual imagery is a subsystem of the total cognitive architecture but not part of central cognition. It is under the control of central cognition and communicates with it through internal perceptions and internal actions.

(3) Visual imagery shares some of the same structures as vision and has similar processes except that the source of information is internal (versus external) and the processing direction is top-down (versus bottom-up).

(4) During visual perception, an object's geometric representation (shape, color, texture) is stored in visual long-term memory.

(5) The composition of an object is stored in semantic memory. Semantic memory contains facts about the world. Static, local relationships (spatial, topological, and geometric*)* between objects are also stored in semantic memory. Dynamic, local relationships are stored in episodic memory. Episodic memory contains memories of what you remember.

Larkin and Simon, in their seminal paper titled "Why a Diagram is (Sometimes) Worth Ten Thousand Words" concluded that diagrammatic representations are computationally more efficient in knowledge search and object recognition when the knowledge can be indexed by location [31]. Chandrasekaran has demonstrated the advantages of diagrammatic representations as they reveal emergent objects and relationships not made explicit in pure symbolic representational schemes. He calls this notion the "free ride" [13, 32]. Kaufmann has looked at the functional role of imagery in problem solving and concluded that an analog concretization of visual representations and processing through simulation facilitates performance [26]. These observations form the basis for our first hypothesis.

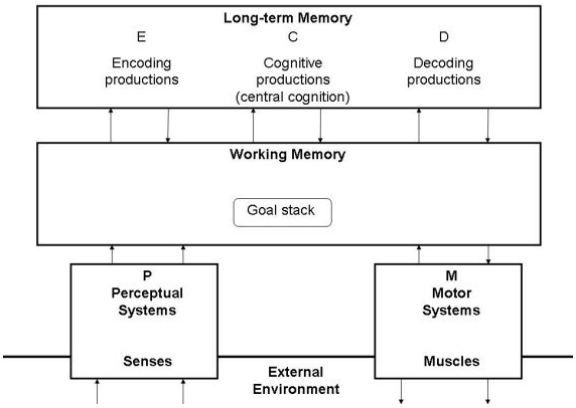


Figure 4: The total cognitive systems: perception, cognition, and the motor system. (Figure recreated from figure 4-15 in [17])

The second hypothesis attempts to answer the question, "Is visual imagery part of the total cognitive architecture?" Our answer is a definitive yes, but to answer that question it helps to understand what we mean by a total cognitive architecture. Newell described the architecture as the "system of mechanisms that accesses encoded knowledge about the external world held in memory and brings it to bear to select actions in the service of goals" (Figure 4). Perception (P) and motor (M) are considered outside of central cognition but part of the total cognitive system. According to Newell there may be other modules outside of central cognition. He further states that an imagery module may be such a component [17].

Related to our assumption and ultimate design decision to build the visual imagery component separate from central cognition, is our third hypothesis. Visual imagery shares some of the same structures and processes as vision. Aside from the most evident phenomena of people reporting that they see their visual images, there are also behavioral and neuropsychological studies showing that vision and visual imagery share similar mechanisms. Images have spatial and visual structure, exhibit similar dynamic characteristics, follow the same physics laws for motion, show the same motion aftereffects, appear to have the same resolution limits as objects imaged on the retina, interfere with concurrent visual tasks, and share a short term and long term visual memory [3, 9, 24, 33, 34].

Our fourth and fifth hypotheses are concerned with the origin of knowledge used in constructing the visual image. Hypothesis four states our assumption that the geometric data associated with an object is stored in a long-term memory structure associated with vision. This visual long-term memory, we assume from psychological and neurological evidence, is in the temporal cortex region of the brain. Since semantic memory is associated with facts about the world, we hypothesize that this memory component is where the static local relationships (spatial, topological, and geometric) between objects and their compositions are stored. Episodic memory, or what you remember, holds the dynamic, local relationships between objects. These two memories, we theorize, are in the parietal cortex regions.

In summary, our theory of visual imagery consists of a set of structures and processes working in conjunction with central cognition. Central cognition's memory includes procedural, episodic, and semantic memories and a short-term working memory that represents the current state of the problem space. Visual object features such as shape, color, and texture are stored in visual long-term memory after the visual system has finished processing and converting the 2D image to a 3D representation. Some symbolic objects are *visualizable* and these objects are composed of other *visualizable* objects. The objects and their local static relationships are stored in semantic memory and encoded at an abstract level. Dynamic local relationships are stored in episodic memory and updated as the situation changes. Each relationship may be further constrained with continuous values.

There are two cases for construction and use of a visual image: (1) automatic and (2) deliberate. In the automatic case, central cognition may be in a wait state or at an impasse in problem solving and automatically invokes imagery on a visualizable object that is relevant to the current problem state. It then initiates the relevant productions for constructing the composite object's image. These productions send commands to the imaging system to construct and inspect the image for emergent objects and relationships. If the imagery system finds the image is already constructed then it short-circuits the construction process and a search of the image begins immediately. If at any time during the process, new knowledge arrives from either another memory or perception, then the impasse is resolved, and the use of visual imagery is no longer required. The construction of the object's images to that point can be stored away for future use.

Central cognition may also deliberately initiate visual imagery with a specific request. In this case, central cognition augments a visualizable symbol's structure with a specific query. As described with the automatic initiation of imagery, the imagery system constructs the visualizable symbol's image prior to inspection. Again, visual imagery may short-circuit the construction process if the image is already constructed. The inspection process searches the image starting from the location relevant to the query and returns any results from the query to central cognition.

We have put forth our theory and supported it with relevant evidence. The next chapter explores the specific architectural design and implementation in this context.

CHAPTER 3

ARCHITECTURE AND DESIGN

The discussion of our theory in Chapter 2 provides the basis for our current architecture. Our software engineering methodology is iterative in nature rather than following a strict waterfall model. This chapter reflects the work of our first two iterations. We expect to modify our design as we continue to research the functional, structural, behavioral, and computational constraints.

In order to clarify the design, we will discuss it in the context of our "table setting domain". The purpose of this domain is to provide a very simple example that exercises the integration of imagery with a cognitive architecture. This domain is from one of our experiments discussed in the following chapter. The agent's goal in the "table setting domain", is to set a place setting for dinner (Figure 5). A strategy the agent may use to determine where to place the plate and utensils is to build and inspect a visual image and locate the center object (the plate). Once it knows the location of the plate, the agent can make its remaining decisions relative to the plate's location.

Figure 5: Table Place Setting

The agent begins the problem with the following knowledge.

(1) It has access to five objects (napkin, fork, plate, knife, spoon)
(2) A place setting is composed of a fork, napkin, plate, knife, and spoon
(3) The local relationships between the place setting's objects (i.e. the fork is on top of the napkin, the fork is left of the plate, the plate is left of the knife, the spoon is right of the knife).

Note that the agent does not have knowledge of global relations, such as the plate is in the center of the place setting or the spoon is the rightmost object.

The agent has two simulated commands it can give to its motor system: (1) move object to a *region,* where *region* is a place mat's left, center, or right region from an egocentric viewpoint; and (2) move object left-of, right-of, on-top-of, or underneath another object within a region.

What makes this problem difficult to solve efficiently is that the initial problem state provides no knowledge of the global relationships between objects. The agent has to

infer this knowledge by searching its problem space taking into consideration its local constraints. It can perform the search using predicate logic and inference rules based on a least constraining value heuristic, but the agent still has to make an initial random decision as to where to place the first object. For example, the agent can place the knife in the center region and then infer that it should place the plate in the left region because the plate is to the left of the knife. However, when it tries to set the fork to the left of the plate, it finds there is no room on the place mat. The agent has to backtrack and try another course of action. Of course, it could internally develop a plan to determine the solution prior to placing the objects, but this strategy still incurs a computational cost. On the other hand, if it could look at a visual image of the place setting (which is a concrete instantiation of the local constraints and each primitive object's shape) it could determine that the plate is the center object. It then possesses sufficient knowledge to solve the problem without random decisions.

One of the key design issues in our architecture is determining what data structure enables efficient search for knowledge in an image based on the spatial location of its objects. Another design issue is how the architecture supports efficient communication between the components and conversions between symbolic and metric representations. Finally, we have to address how the agent constructs the image in the first place so that it can use it for reasoning. Image construction is also computational overhead so we also have to consider its impact. With the example domain and its associated issues before us, let us discuss the architecture.

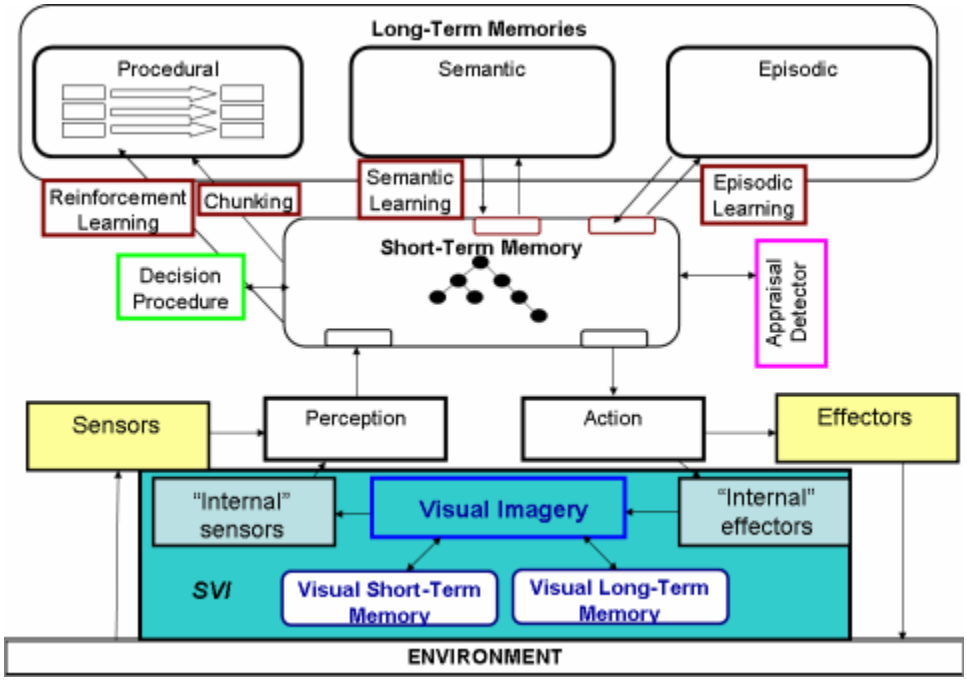


Figure 6: Soar Architecture

Figure 6 shows the Soar architecture. The blue highlighted components are the supplementary architectural modules we added to support visual imagery (hereafter called Soar Visual Imagery or SVI). We will discuss the major components involved in

visual imagery starting with central cognition (Soar) followed by the visual imagery component and visual memories (SVI).

CENTRAL COGNITION

The Soar kernel provides the fixed set of structures and mechanisms we call central cognition. It includes a set of long-term memories, a short-term working memory, a perceptual and motor interface, an emotional appraisal detector, and several learning mechanisms. Working memory consists of symbolic, declarative knowledge relevant to the current situation. It organizes knowledge as a hierarchy of states, operators (instantiations of which are steps through a problem space), existing goals, and domain knowledge. Long-term memories include procedural, episodic, and semantic memories. The memories are not directly available and must be searched to find knowledge relevant to the situation. Procedural memory is a production system that controls behavior by describing how and when to perform specific tasks. Episodic memory contains knowledge of previous events, created via cues in working memory. When an agent encounters a similar situation that was previously stored and procedural knowledge is incomplete or inadequate, the architecture brings an episode into working memory. The semantic memory contains general facts about the world encoded as declarative structures and again is retrieved and brought into working memory when procedural knowledge is lacking. [35-38].

The main control process in Soar is the decision cycle (Figure 7). During each cycle (hypothesized to be approximately 50 ms in humans), Soar searches for knowledge in an effort to progress through the current problem space. For example, setting the place setting may be the current problem space. During the input phase, perception augments a specific, fixed structure in working memory (called the input-link) with symbols that reflect current perceptions. In the proposal phase, parallel retrievals to the long-term memory structures elaborate the current state, suggest zero or more possible operators, and evaluate the potential operators. When Soar reaches quiescence, it enters the decision phase and, based on preferences, decides on an operator to apply. After the decision phase, Soar enters the application phase and adds persistent structures to working memory modifying its internal state. If the operator has commands for the motor system, it adds symbolic structures to a special fixed working memory structure called the output-link. Finally, during the output phase, if there are any symbols attached to the output-link, the motor system retrieves them for further processing. The cycle continuously repeats until someone disables the agent.

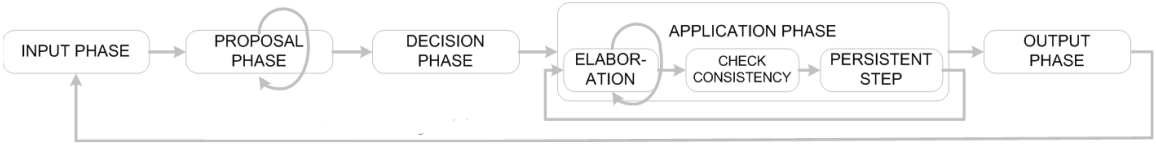


Figure 7: Soar Decision Cycle

To indicate which objects are imaginable and to support the hierarchy requirement, the agent designer designates symbols within Soar's working memory as visualizable objects. As discussed previously, a visualizable object is either a primitive object or a

composite object. Primitive objects have an underlying geometric representation while a composite object contains one or more visualizable (either primitive or composite) objects. Using a graph structure, Figure 8 shows an example of how an agent designer models a portion of the place setting in Soar. The napkin, fork, plate, knife, and spoon are primitive visualizable objects and the place setting is composed of these items. The agent designer can further decompose this object composition as required. That is, a knife may be a composite object made up of a handle and a blade, each with its own underlying geometric model (stored in visual memory).

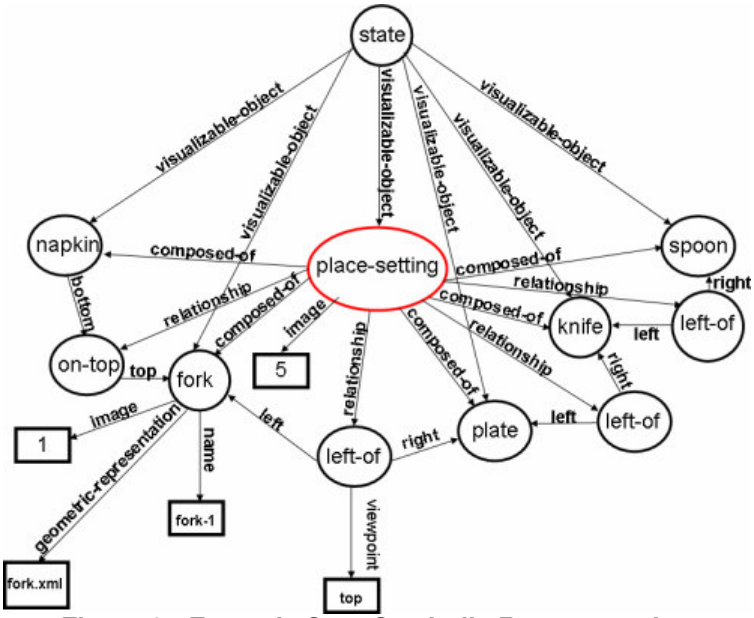


Figure 8: Example Soar Symbolic Representation

Each visualizable symbol includes an "image" attribute mapping the symbol to its image in visual memory (shown only for the fork and place setting in Figure 8). This mapping occurs after visual imagery constructs and stores the image of the object. Primitive objects contain a pointer to their geometric representation[1] or name a specific geometric shape (point, line, circle, triangle, rectangle, tetrahedron, etc.) as its value. Geometric shapes are used in domains where the agent wants to create a geometric figure, build an abstract diagram, or augment an existing image with a specific shape (such as adding a curve to represent a path).

Composite objects do not have geometric representations. Instead, they contain a set of local relationships. These relationships constrain the relative location between two object parts within the composite object and may be spatial (i.e. left-of, right-of, in-front-of), topological (i.e. connected, overlap, inside), or geometrical (i.e. parallel, perpendicular, intersect). The relationships are either abstract (i.e. left-of) or concrete (a specific vector). We decided to include concrete relationships for cases where the agent captured that information during the original perception of the object. We augment each relationship with a *top, front,* or *side* viewpoint (shown for the fork left-of

[1]In our initial implementation, this is the name of the XML file containing the model's vertices.

plate relationship in Figure 8) so that SVI knows the relationship's perspective prior to computing transformations. Additionally, the relationship attribute may include constraints further defining the relationship (not shown in Figure 8). There are distance, location, size, and angle constraints[2]. For example, object A may be halfway between object B and object C (distance constraint), object A may be at a specific location (x,y,z), object A may be two times as large as object B (size constraint), or object A is oriented 30 degrees with respect to object B (angle constraint).

Another attribute that a visualizable object may have is an *is-emergent* attribute (not shown in Figure 8). Visualizable objects with an *is-emergent* attribute are those objects that SVI finds during its visual inspection that were not part of the original specification. For example, if the agent creates a visual image with two lines that intersect, SVI returns to central cognition an emergent geometric object (a point). The object is emergent in that it was not explicitly represented in the original specification of the "two-lines-intersect" object.

The symbolic structures are currently stored in Soar's working memory. We hypothesize that some of this knowledge, such as static relationships between objects and object composition is stored in semantic memory and specific, dynamic instantiations of relationships are stored in episodic memory. This is the subject of future work as to how the memories in Soar interact. Below we discuss other considerations as to how these symbolic structures are combined with the geometric representations to form the final visual representation.

Within Soar's procedural memory, there are visual imagery operators that control the construction and inspection of visual images. These operators include *create, compose, add, query,* and *attend*. *Create, compose,* and *add* are commands to SVI that construct an image. *Query* is the operator specific to image inspection and *attend* monitors Soar's input-link for internal perceptions returned from SVI.

In order to build the place setting, cognition first has to find the geometric data associated with the primitive objects. The *create* operator, commands SVI to find the geometric data associated with the specified object in the operator and create an image of it. After creating the image structure, SVI provides central cognition with an internal perception containing a unique identification representing the object's image. The *attend* operator maps this identification symbol to the parent object's symbol. For example, in Figure 8, the *fork* symbol has an image attribute with a value of "1". This value is the unique identification of the fork image in visual memory. If SVI cannot find an underlying geometric representation for an object then it creates a random "blob" to represent the object and informs central cognition.

The second operator used in image construction is the *compose* operator. Given two objects with image identifications and a local relationship between them, Soar employs a *compose* operator instructing SVI to compose two objects into a single image. As an example, a compose operator may bind the fork and plate together into a single image.

[2] We have only implemented distance constraints at this time.

The operator is augmented with the image symbol of each object, and the relationship between the objects. After SVI composes the two objects, an *attend* operator maps the newly created image identification to the place setting's symbolic structure.

The final operator used in image assembly is the *add* operator. In this case, SVI has already created a skeleton image of the composite object with two or more visualizable objects. Another object exists that is part of the composite object, has a local relationship with one of the objects in the image, but is not yet in the image. Therefore, we do not want to compose a new image but rather add the object to the existing image relative to the object already in the image. For example, if the fork and plate are in the composite's image, then to add the napkin to the image, an *add* operator is applied.

In addition to visual imagery's construction operators, there is an operator to inspect or *query* the image. The query command specifies the type of search visual imagery performs. In the deliberate case, cognition has a specific query it wants answered. It may be as simple as what is the center object to more complex as to what is the orientation and distance from object A to object B[3]. In the place setting domain, it wants to know what the center object is.

In the automatic case, cognition invokes visual imagery not to answer a specific question but rather to determine if there are any emergent objects or relationships relevant to the current problem state. The *query* operator commands SVI to search for these features. Emergent objects may be simple geometric shapes such as points, line-segments, or triangles or regions of objects that appear to form a single entity because of their relatively close proximity. Emergent relationships include angles, congruent geometric figures, or global relationships between two or more objects[4]. As in image construction, an *attend* operator monitors the input-link for responses to these queries and creates the associated symbolic structures within working memory.

VISUAL IMAGERY

As shown in Figure 6, the visual imagery component receives commands to construct and interpret an image from central cognition. Because visual imagery is not part of the Soar kernel, there is a communication protocol between the two components. To support the communication, we created a framework of "internal" perceptions and "internal" actions. The internal action interface receives symbolic commands from central cognition, converts those symbols to concrete descriptions recognizable by the cognitive component, and then sends the transformed command to that component. The perceptual interface performs the opposite transformation. That is, it receives the "internal sensing" from the cognitive component transforms it into a symbolic structure that Soar recognizes and places it on Soar's input-link. For visual imagery, a *SoarImager* class parses the symbolic commands, marshals the appropriate arguments, and calls a function in an *Imager* class (the heart of SVI) that handles the request.

[3] The current implementation contains support for basic abstract spatial relationship queries (left-of, right-of, center-of, in-front-of, behind,)
[4] The current implementation finds points, line-segments, triangles, angles, and geometric congruency relationships.

When the *Imager* is finished processing the request it returns its results (image identification or query results) to the *SoarImager* for conversion back into a symbolic representation. Except for the parsing of the command, these operations are all O(1) to support efficient communication.

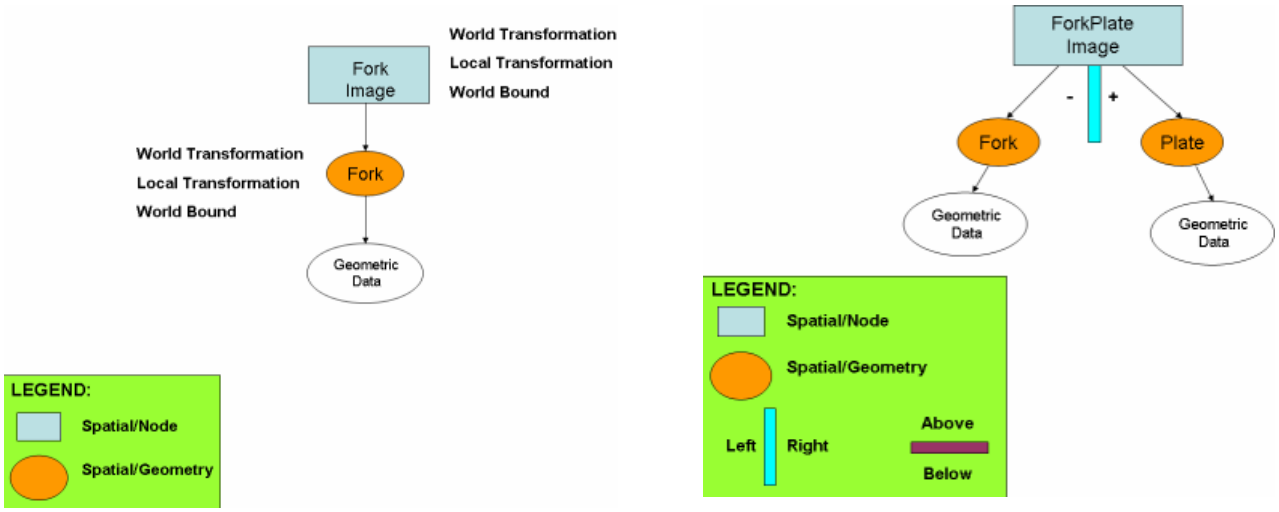
Within SVI, a key design decision was determining the data structure to represent the visual image. As discussed in chapter 2, the data structure has to be computationally efficient in terms of time and ideally in terms of space also. As much as possible, it has to partition the space into discrete regions to support efficient search and have a hierarchical structure to support both logical relationships (i.e. a fork is part of a place setting) and spatial relationships (the fork is to the left of the plate). We desired it to be flexible enough to support dynamic addition and deletion of objects. It has to combine the geometric representations with the spatial, topological, and geometric relationships, support three-dimensional representations, and render to a two dimensional image.

We looked at several data structures including spatial occupancy arrays, quad trees (or their 3D counterpart, octtrees), and scene graph structures. We decided on the scene graph structure in conjunction with a binary space partitioning (BSP) tree. Graphics engines commonly use scene graphs to manage vertices, transformation, and bounding volume information. BSP trees are used for sorting a scene primarily for culling purposes [39]. The scene graph provides a structure that combines both an object's geometric and spatial (i.e. transformation) data. It is hierarchical in nature supporting logical and spatial grouping of objects. The composite object's spatial information includes all of its sub-objects' spatial information and treats them as one entity for simulated movement (i.e. for simulation). The scene graph is primarily used in graphics applications to support efficient rendering to 2D images so it supports our requirement to create 2D pixel-based images if we decide this is important for feature extraction. We can also model quad trees/occtrees with scene graphs. Based on our requirements and constraints, the scene graph is much more flexible, efficient, and powerful for sorting the objects than the other data structures we investigated. The following few paragraphs explain the scene graph and its advantages over the other structures for visual imagery.

The scene graph is a directed acyclic graph (DAG) but often implemented as a tree. Each leaf node of the tree is a primitive object containing the set of vertices associated with it. The root and intermediate nodes represent composite objects. In this regard, the scene graph structure maps directly onto our hierarchical, symbolic representation of visualizable objects in Soar. Figure 9 and Figure 10 show the scene graph after a *create* fork, *compose* fork and plate, and appropriate *add* commands have respectively finished processing.

As shown in Figure 9, the fork and plate leaf nodes point to their geometric data. Any other scene graph(s) created with one of these objects will share this data. The data sharing reduces storage costs and is why the scene graph is considered a DAG. It fulfills our requirement to keep the geometric data separate from the spatial information, combining them as necessary to form the image.

Each node in the scene graph stores local and global transformation information. Coupled with an object's geometric data, this information specifies each object's location, orientation, and size within its local and global frame of reference. The hierarchical nature allows the scene graph to propagate the transformation data down the hierarchy. For example, if the place setting is moved, that transformation will propagate to all the objects in the place setting thereby updating their global representations.



(a) Fork **(b) Fork and Plate**

Figure 9: The Scene Graph Data Structure

Another property of the scene graph supporting our requirements is that each node serves as a container for its children objects, effectively representing a region of space. The search for object locations is more efficient as each node can determine which one of its child nodes subsumes the object. To make the search more efficient, SVI uses BSP nodes in instances where SVI can separate two objects based on their relationships (i.e. left-of allows the two objects to be separated, intersect does not). The BSP tree taken as a whole represents the entire space, and each node in the tree represents a convex subspace. Since scene graph implementations use the composite design pattern [40], we can interchange the nodes of the graph between basic scene graph nodes and BSP nodes. The final structure may be a combination of both basic scene graph nodes and BSP nodes.

As an example, a *compose* operator may bind the fork and plate together into a single image as shown in Figure 9b. Since these two objects have a relationship which spatially separates them (i.e. left-of), SVI creates a BSP node to bind the two objects together. The rectangle in the center is the third child of the BSP node and represents the separating plane. The object on the positive side (the plate) of the separating plane's normal is in the plane's positive region and the object on the negative side (the fork) is in the plane's negative region. Figure 10 shows the final place setting structure where every intermediate node is a BSP node. The separation of space into two discrete regions at each level makes the search process more efficient. For example, if we naïvely attach all the place setting objects (fork, spoon, plate, etc) to the scene

graph's root node, then to determine the right most object we have to perform a linear search through all the objects. With the BSP structure, the search is binary (assuming a balanced tree).

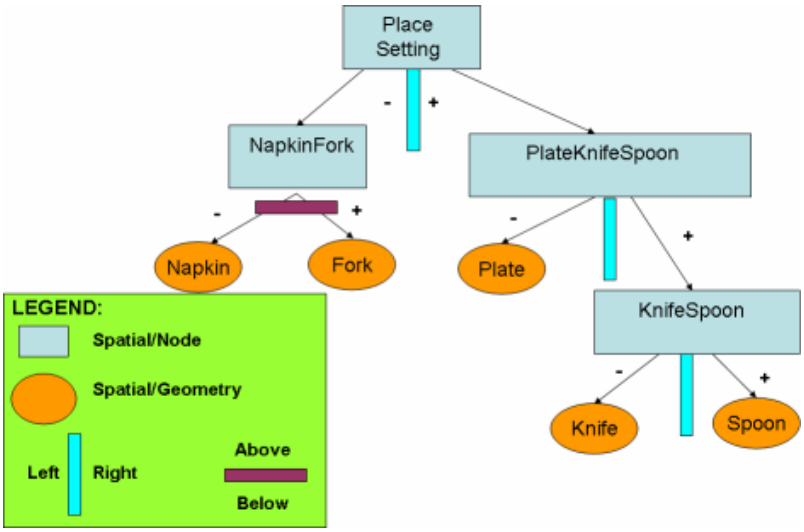


Figure 10: The Complete Scene Graph Data Structure for a Place Setting Image

A key function in the construction process for *composing* and *adding* objects is *SetRelationship*. For each relationship, the operator designates one object as the base object (the object already in the scene graph if *adding*) and one object as the relative object. The *SetRelationship* function translates the relative object to the base object's location, and based on its relative relationship, performs the local transformation. Unless there is a distance constraint, the object's are separated using a heuristic based on each object's *area of influence* [41].

The scene graph/BSP tree is more advantageous in comparison to spatial occupancy arrays. Occupancy array algorithms are quadratic in the number of grid cells and effectively require you to reacquire the 3D structure and objects for any spatial reasoning at viewpoints different from the original viewpoint . With an occupancy array or pixel based image, if we want to simulate movement in the image for further reasoning, we would have to have create another occupancy array to replicate the movement requiring much more space and time. With the scene graph, we simply apply the transformation at the node representing the object we are simulating and then prorogate the transformation down the tree. In addition, if we decide that we have to render the scene graph to a 2D pixel-based image to assist with feature extraction, then the scene graph is primed for this to occur. We do in fact render the image during testing and debugging to allow a human engineer to see what Soar is "imagining" (Figure 11).

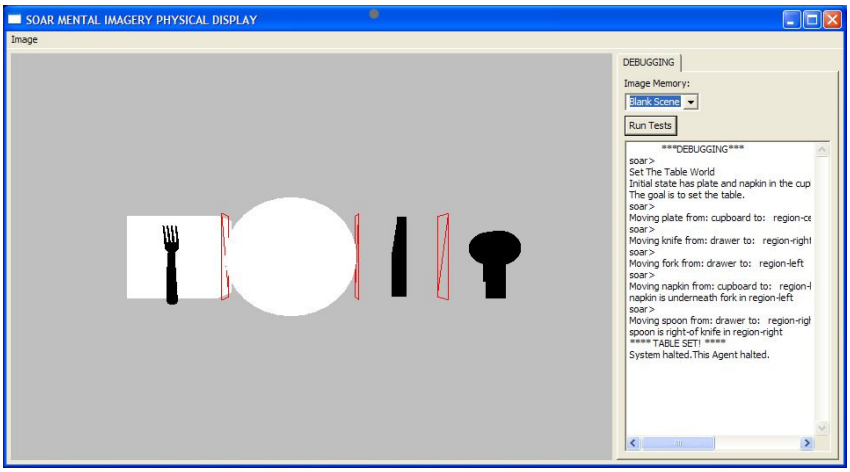


Figure 11: Soar's Visual Imagery Physical Display

The advantage of using a BSP tree over a quadtree/octree representation is that those structures require more space than a BSP tree as each level of the tree has four/eight children nodes to partition the region even if some regions are empty. The BSP tree only partitions the space if the partitions contain objects. Additionally, with a quadtree the subdivision is static whereas with the scene graph/BSP tree we have much more flexibility. In fact, we can represent a quadtree/occtree with a scene graph/BSP tree.

A disadvantage of the scene graph/BSP tree structure is that some relationships do not separate the space discretely. This is also a problem with the quadtree. Maintaining both the logical and spatial structure of the tree can become cumbersome to manage. When adding or moving objects within an image, SVI has to take into account the constraints imposed by objects already in the image. Algorithms determining where to insert objects in the scene graph so that relationships remain consistent is a constraint satisfaction problem (CSP) and the subject of future research.

Figure 6 shows two visual memory structures. Visual short-term memory is simply the scene graph that SVI is currently processing. Visual long-term memory stores an object's geometric representations (i.e. its model vertices) along with any constructed visual images. In our implementation, visual long-term memory is a set of XML files storing a primitive object's geometric representations and a hash table in main memory to store the scene graph structures. During image construction, SVI reads an object's geometric representation from the file into main memory and stores it along with its associated scene graph structure in a hash table. The hash table's entries contain a pointer to the associated scene graph along with another hash table that stores a pointer to each object in the scene. This time for space tradeoff is necessary to facilitate efficient search starting from a specific object's location. For example, when *adding* objects to the scene we desire immediate access to the object already in the image. If the fork and plate are already in the image and an *add* operator directs SVI to add the knife to the left of the plate, we want SVI to immediately locate the plate. Secondly, when performing queries, such as "what objects are right of the plate", we want the search to start from the plate. Finding these objects should ideally be an O(1) access rather than having to search the scene graph structure.

We generically designed the core visual imagery component so that one can consider it for use with any cognitive architecture. The *Imager* class and its associated colleague classes provide the majority of the SVI functionality. We implemented the *Imager* and associated classes in C++. We used Java to implement the *SoarImager,* perception, action, and the visual display components.

CHAPTER 4

RESULTS

We experimented with our initial implementation in two domains. As we were initially interested in how we could realize our overall theory coupled with building the visual imagery component from the ground up, we decided to keep our initial experiments within the confines of simple internal problem solving tasks.

Our evaluation criteria for the overall architecture consist of the following. We did not use every criteria element in our experiments, and we do not mean to imply that this is a complete list of our evaluation criteria.

(1) **Computational Efficiency**
   a. Time in Soar kernel (central cognition)
   b. Time in Soar Visual Imagery Component
   c. Time in Soar-SVI communication
   d. Total Time.

(2) **Functional Capability**. What new capability does it provide?

(3) **Knowledge**. How much knowledge is required?
   a. Long term procedural knowledge (number of productions)
   b. Short term (number of working memory elements)
   c. Visual long term memory
      i. Number of images
      ii. Size (bytes)

(4) **Problem Solving**.
   a. Number of Soar decision cycles
   b. Number of visual imagery accesses

(5) **Constraints**. Must work within the given behavioral, structural, functional, and computational constraints.

Experiment #1 (Table Setting). The first experiment, as described in the previous chapter, was the table setting experiment. The purpose of the experiment was to explore the conditions under which central cognition may not have the required knowledge available in the problem space to make an effective decision and to compare the computational efficiency between searching for that knowledge symbolically versus searching for it using a metric representation.

In order to solve the problem the agent has to search through the problem space determining where to place the objects in the place setting. We created three agent strategies for solving the problem. The first agent, using only central cognition and symbolic representations, searches through the problem space by randomly placing objects on the place mat until it reaches the goal state (**RANDOM**). The agent continually compares the current place setting state with the goal state. If it is in the goal state, it declares success and stops. The second agent, similar to the first agent has only central cognition's symbolic representations, but uses a least constraining

value heuristic to direct its search through the problem space (**LCV**). For example, if the agent has already placed an object in the center region and knows of another object that is left of the object in the center region then it will place that object in the left region rather than in the right region. The third Soar agent realizes it does not have enough knowledge to determine precisely where to place the first object so it sets up a request to visual imagery to determine the center object (**SVI**). Once visual imagery provides it with the knowledge of the center object, it is able to place the rest of the objects based on their local relationships.

This experiment has the following characteristics for the agent using visual imagery: deliberate retrieval, inferring global relationships from local relationships, and exploiting 2D visual object properties (size and shape) to infer new relationships. Since there are an even numbers of objects in the place setting, the agent can infer the center object only by exploiting the object's shape and size. Once the agent determines the center object, the problem space search becomes linear in the number of remaining objects.

We evaluated the domain with three criteria: (1) number of productions required (knowledge), (2) number of decision cycles required (a count of reasoning steps in Soar), and (3) time to solve the problem. The results, shown in Figure 12, Figure 13, and Figure 14, are the median for each category over 30 trials.

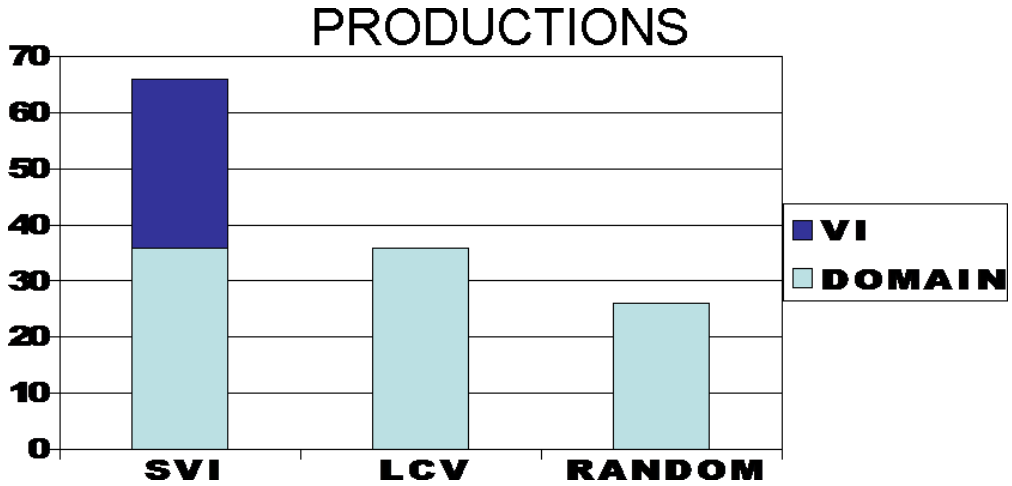


Figure 12: Number of Productions Per Table Setting Agent.

VI = Visual Imagery Productions
Domain = Domain Productions

The number of productions compares each agent's initial knowledge. Visual imagery incurs the overhead of an additional 30 productions to provide control. These production rules include *create, compose, add, query,* and *attend* operators discussed in chapter 3. Both the SVI and LCV agent require 36 domain productions while the RANDOM agent only requires 26 productions, as it did not use any heuristics to control its search.

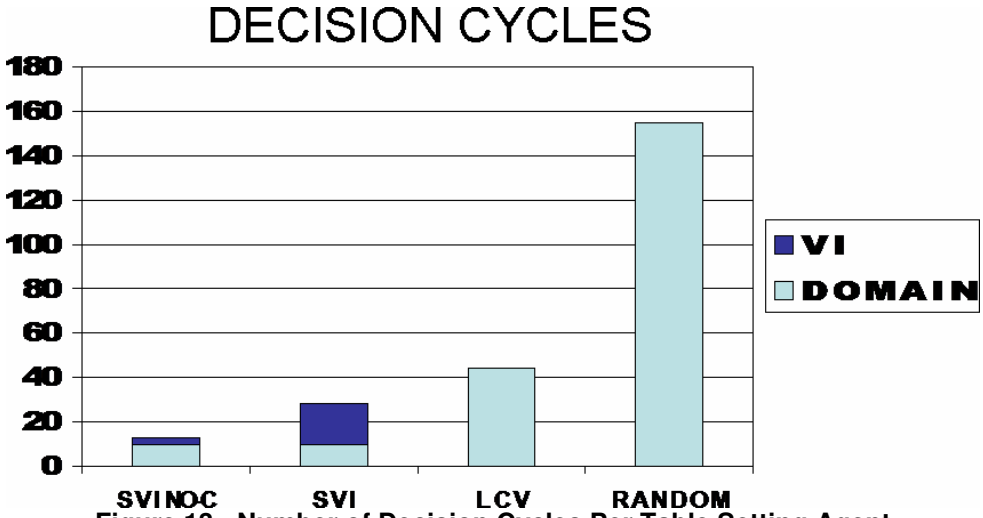


Figure 13: Number of Decision Cycles Per Table Setting Agent

Figure 13 shows the number of decision cycles required per agent. As expected, the SVI agent requires less decision cycles (28) than the LCV and RANDOM agents do, as once it knows the center object, its search becomes linear. The LCV agent requires a few more decision cycles (44) but its entire search is within central cognition. The RANDOM agent searches aimlessly for the correct set up and requires a median of 155 decision cycles to achieve the goal.

The SVI agent incurs an overhead of 15 decision cycles to construct the image, but this overhead is an initial cost only. Once it builds the image, the agent does not have to reconstruct it again. To demonstrate this initial cost for both the number of decision cycles and total time, we included the results for an agent who does not have to construct the image because it already exists in visual long-term memory (SVI NO-C). As expected, SVI NO-C requires half the decision cycles (13) as SVI.

Figure 14 shows each agent's cumulative time along with the time spent in the Soar kernel (central cognition). The figure also shows the time the SVI agent spent in visual imagery. The SVI agent performed slightly better in terms of performance than the LCV agent, and as expected both significantly outperformed the RANDOM agent. The SVI NO-C (no image construction) agent was twice as fast as the LCV agent with no recorded time spent in the visual imagery component (only one access was made). The data provides promising evidence in support of our hypothesis that, for certain tasks, analog representations may be more computational efficient. There may have been other strategies we could have programmed for a Soar only agent to improve its performance. This would require adding more initial knowledge and require the knowledge engineer to determine another strategy. This requirement is not an overwhelming issue in this small domain, but in larger domains can become more burdensome.

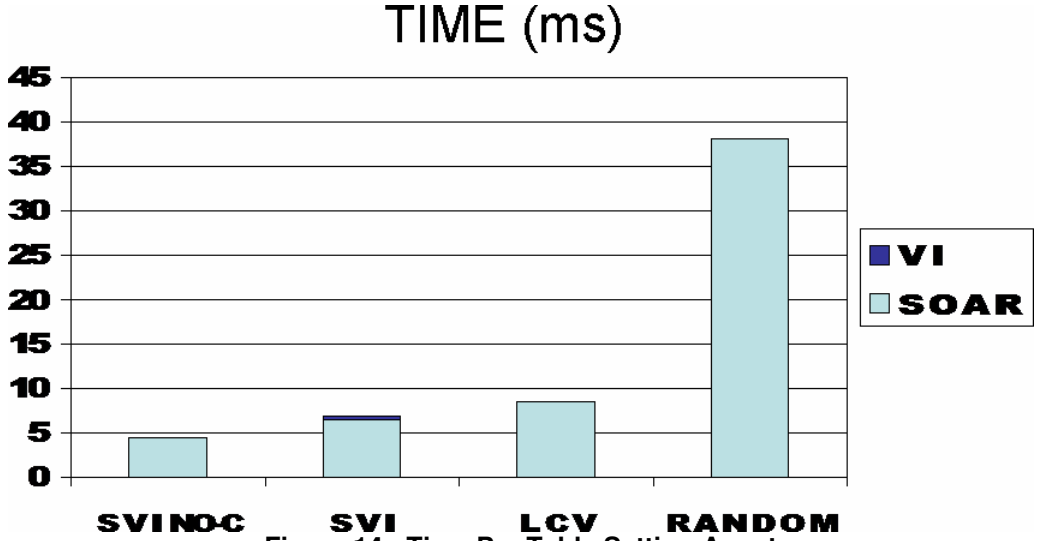


Figure 14: Time Per Table Setting Agent

VI = Time spent in visual imagery component
Soar = Time spent in Soar Kernel

The final criteria we evaluated were what, if any, new functional capability visual imagery may provide. In this domain, being able to infer the center object was a global relationship we could not infer with just symbolic reasoning using Soar unless we had more knowledge about each object's size and shape and production rules to reason with that knowledge. Of course, visual imagery has access to this information because of our assumption that it shares structures with the vision system. Another way to interpret the data is that visual imagery is calculating a valuable heuristic. Given enough knowledge, a symbol system could also calculate the heuristic. So the question is which representation is more efficient in calculating the heuristic? As with all heuristics, we do not want the time computing it to be more than just performing the search without it. The results provide us with an indication that visual imagery can be more efficient in these spatial domains.

Experiment #2 (Geometry Problem). Our second experiment derives from a paper written by Larkin and Simon in 1987, titled "Why a Diagram is (Sometimes) Worth Ten Thousand Words". Their paper provides examples where they argue that diagrammatic reasoning provides benefits in terms of computational efficiency. In this problem domain, an agent has to prove the congruency of two triangles given the initial relationships between four lines. The initial specification includes neither the objects nor the relationships required to solve the problem. Finding, recognizing, or creating the appropriate elements is necessary [31]. The purpose of this experiment was to explore our theory in a different domain, where finding emergent objects and relationships is important. We also wanted to see if our design choices applicable for the table-setting domain carried over to this problem.

The problem statement is, given four lines (A, B, C, D) where line A is parallel to line B, Line C intersects Line A, and Line D bisects the line segment formed by the intersection of line C with line A and line B , prove that the two triangles formed are congruent (Figure 15).

To prove the two triangles are congruent, an agent can employ one of a couple geometry rules. We designed the agent using the same rule Larkin chose, which is the angle-side-angle (ASA) rule. The rule states that if two angles and the included side of one triangle are congruent to two angles and the included side of another triangle, the triangles are congruent. In Figure 15, if the agent shows E1=E2, c=b, e1=e2 then it proved the triangles are congruent.

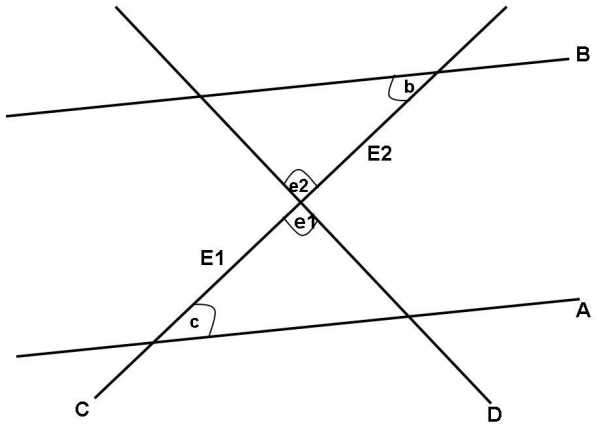


Figure 15: Geometry Problem

The characteristics of this problem are the use of automatic retrieval, relationships with constraints, search for emergent objects and relationships, and the requirement for a sense of direction. The agent using visual imagery does not have a specific query. It only knows that it does not have enough knowledge to solve the problem so it engages its visual imagery system. Unlike the table-setting domain, this domain incorporates relationships with constraints. In this problem, there is a constraint on the intersection relationship between line C and line D. The lines could not intersect anywhere. Their intersection was at the midpoint of the line segment formed by the intersection of line C with line A and line B. When building the image structure, SVI has to consider this constraint.

We created two agents to solve the problem. The first agent only used central cognition and symbolic representations (**Soar Only**). This agent's productions consist of what Larkin and Simon call perceptually enhanced productions. These productions create geometric objects that an average person looking at the diagram of the problem could see (vertices, line segments, angles, and triangles). In addition to the perceptually enhanced productions, the agent also has rules to prove congruency relationships (i.e. alternate interior angles are congruent). The agent continues to create these objects and relationship until it has enough knowledge to prove the triangles are congruent. The problem requires the productions to perform a lot of pattern matching as Soar creates several objects and relationships that are not required to solve the problem.

These objects increase the number of relevant combinations that could possibly lead to the problem's solution.

As we were developing this "Soar Only" agent, we realized that the agent could only solve the problem if it was given a "sense of direction." The agent needs a "sense of direction" to determine which angles are associated with each triangle. Without direction, it can only narrow the choice down to four possibilities. Therefore, we provide this knowledge to the agent, as without it, the agent cannot solve the problem. We will explain this problem in more detail shortly.

The second agent (**SVI**) used visual imagery to help it solve the problem. The SVI agent builds a visual image of the situation from the geometric relationships. After building the image structure, it searches it for emergent objects and relationships such as the vertices, line segments, angles, triangles and congruent relationships. After finding these objects and relationships, it returns that information to central cognition. In this case, central cognition has enough knowledge to select and execute the operator proving that the triangles are congruent. This agent does not run into the "sense of direction" problems as described for the "Soar Only" agent, as metric information is available to determine direction and ultimately the angles belonging to the triangles. Our evaluation criteria were the same as in the first experiment.

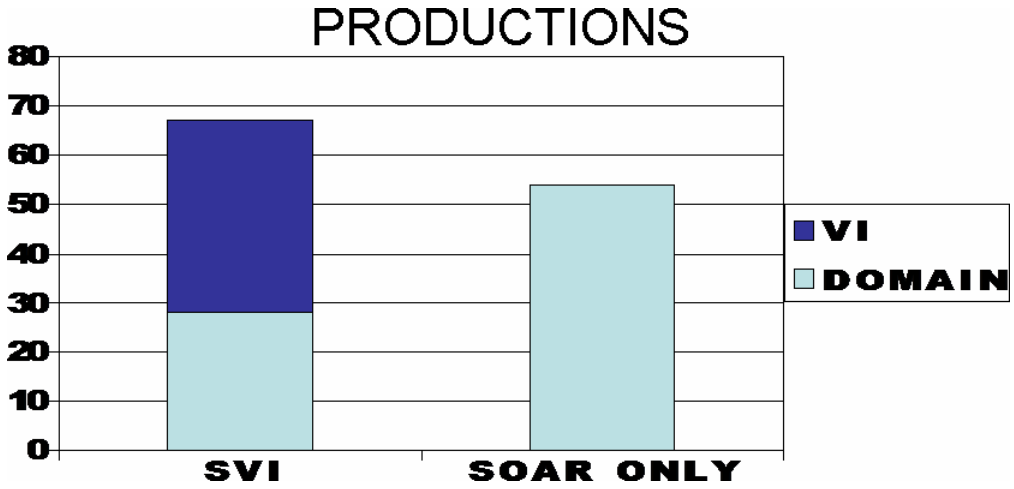


Figure 16: Number of Productions Per Agent.

VI = Visual Imagery Productions
Domain = Domain Productions

As in the table-setting experiment, the "Soar Only" agent required less total productions (Figure 16). The SVI agent reused the productions from the table-setting agent for *creating, composing, adding, querying, and attending*. We had to add nine more domain-independent visual imagery productions to provide additional functionality. These productions included two productions for composing Geometric Objects as compared to domain objects (objects with an underlying geometric model), productions to compose and add objects with constraints, and productions to attend to the emergent objects and emergent relationships. We expect the productions controlling visual

imagery to continue to increase as we add more functionality but to eventually stabilize and generalize over time.
Figure 17 shows the number of decision cycles each agent required to solve the problem. As the problem was deterministic, the number of decision cycles was the same for all trials. The SVI agent required 21 decision cycles, 18 of which were spent constructing and inspecting the image. The "Soar Only" agent required 35 cycles.

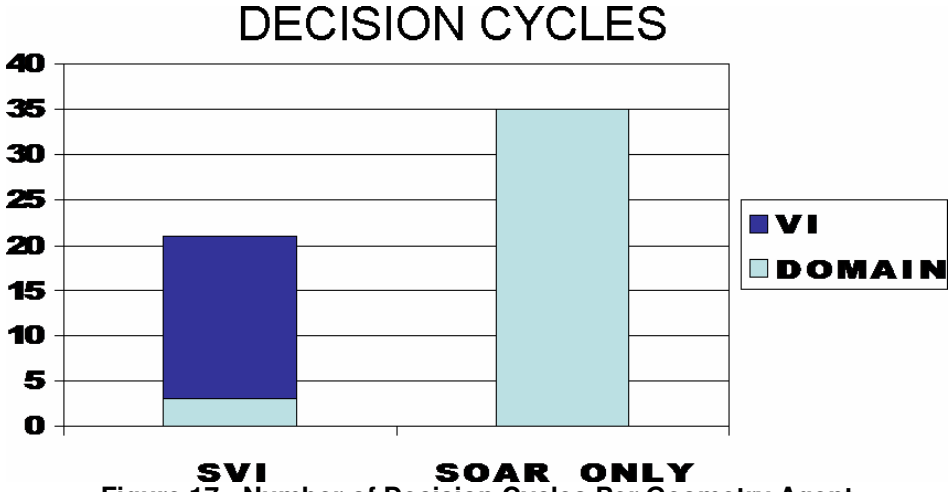


Figure 17: Number of Decision Cycles Per Geometry Agent

Figure 18 indicates that the SVI agent was an order of magnitude faster than the "Soar Only" agent. The "Soar Only" agent spent a vast majority of its time in the last five decision cycles as it had a lot of objects and relationships to consider and these created a large computational load on the Soar production matcher. We expect we could improve the performance of this approach by rearranging conditions and reducing the number of variables the productions have to match, but are not convinced this will make the result significantly different as we could also improve the SVI agent's performance by streamlining its productions and modifying some algorithms in its visual imagery component.

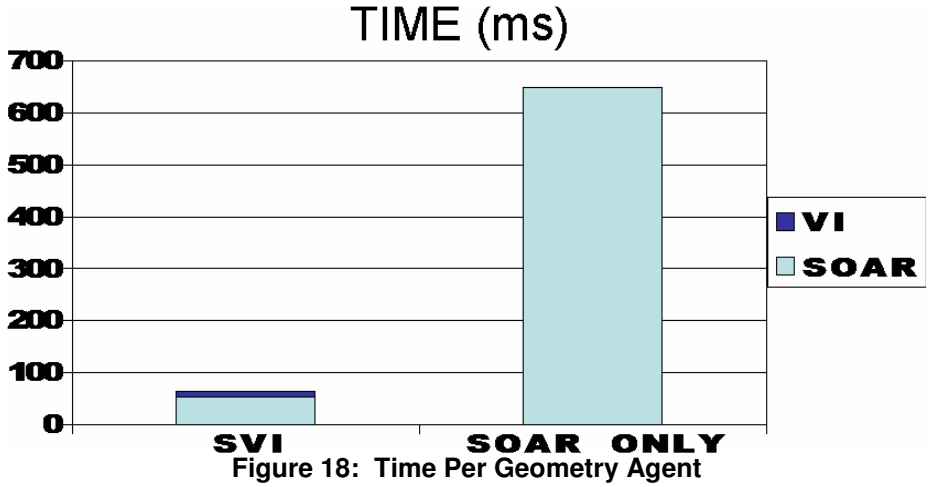


Figure 18: Time Per Geometry Agent

VI = Time spent in visual imagery component
Soar = Time spent in Soar Kernel

Again, our final measure for the domain was whether visual imagery provides any new functional capability. The "sense of direction" was important in order to determine the triangle's angles. The metric information in visual imagery provided the agent this "sense of direction" that it could not infer with symbols and the initial problem statement alone. Figure 19 highlights this issue. The Soar only agent creates the vertex labeled 'c' in Figure 19. It then creates the angles associated with this vertex by naming each of the regions (1, 2, 3, and 4) in space created by the intersection of these two lines. The problem arises when it tries to create the triangle. How does the agent know which angle (1, 2, 3,or 4) belongs to this triangle's vertex? It cannot infer the actual angle without additional, specific knowledge about the domain (i.e. if the vertex is the intersection of line A and line C then assume the angle is in region 4).

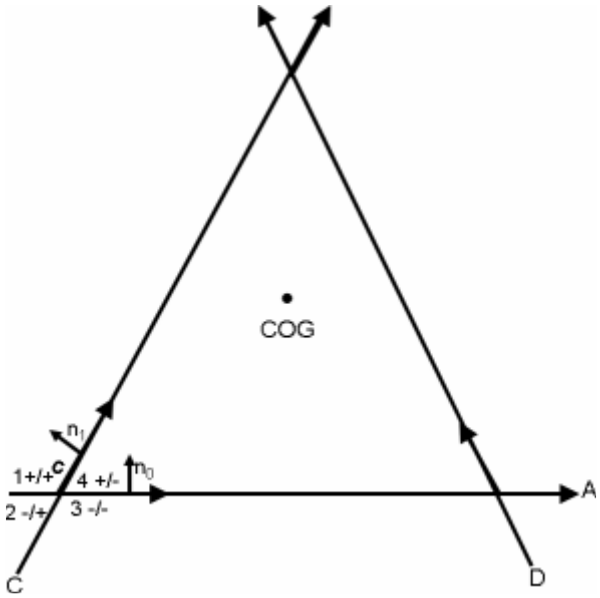


Figure 19: "Sense of Direction"

The SVI agent, however, can take advantage of the image's underlying metric information and associated direction to determine the angles belong to the triangle's vertices. One possible algorithm is, for each vertex, determine which angle is on the inside of the triangle. The general algorithm is:

1. Finds the triangle's center of gravity (COG).
2. For each line segment of the triangle, determine what side of the line (+/-) the COG is (taking advantage of the line's direction and normal). For example, the COG is on the positive side of line A and on the negative side of line C.
3. For each triangle vertex
    a. Based on the results from step 2, determine which region (+/+, +/-, -/+, -/-) the vertex's angle associated with the triangle falls. For example, the triangle's angle associated with vertex c is in the +/- region (4). That is, the angle is on the positive side of line A and the negative side of line C. The actual value of that angle is the arc cosine of the dot product between the line's vectors. The dot product always returns the angle in region four.

Other angle's values may be calculated from this result (i.e. region 2 angle is the same as region 4, region 1 and 3 angles are 180 – region 4 angle).

Some of our concluding observations from this domain are that the scene graph/BSP data structures are an adequate representation although we need to continue to work through the constraint satisfaction problems of where to insert objects in the structure. For this domain, SVI could separate the two parallel lines using the BSP node, but it had to add the remaining lines as children of a standard scene graph node.

Another potential issue with visual imagery is that there may be numerous emergent objects and relationships arriving as internal perceptions indicating a future requirement for an attention mechanism. Another question that continues to arise, is what is emergent? In this problem domain, it is what is "obvious to a human viewer" as defined by Larkin and Simon. What is obvious to one is not always necessarily obvious to another.

CHAPTER 5

OPEN RESESEARCH QUESTIONS

Several open research questions remain, and we highlight a few here.

(1) Neurological evidence shows that the visual cortex is active during imagery. Does this imply a 2D rendering of the image to infer further objects and relationships? Is there more than one primary data structure and associated processes used for visual imagery?
(2) What are the inherent relationships between visual imagery and vision?
(3) Truth Maintenance. When a relationship structure changes in central cognition's working memory should this automatically change the relationship in the image or should the image update prior to the next inspection? If visual perception reveals a change in a relationship between objects, does the visual image update itself? If it does update itself, does it notify central cognition/Soar of the change?
(4) Emergent features. What defines an emergent feature? How long (or deep) does SVI search for emergent features?
(5) Constraint Satisfaction. Where does SVI insert objects into the scene graph structure? An example from Anderson highlights this problem [43]. Assume there are two counties, A and B, each with two cities, x and y (Figure 20). The problem with representing this in a straight hierarchical fashion is that the relationship at higher levels in the hierarchy (i.e. county A is west-of county B) do not apply at the lower levels (city x is east-of city y). This issue is a common problem in human spatial reasoning. It does not necessarily invalidate the hierarchical representation but highlights the fact that cognition must account for these exceptional cases and either answer wrong or realize it must "look closer".

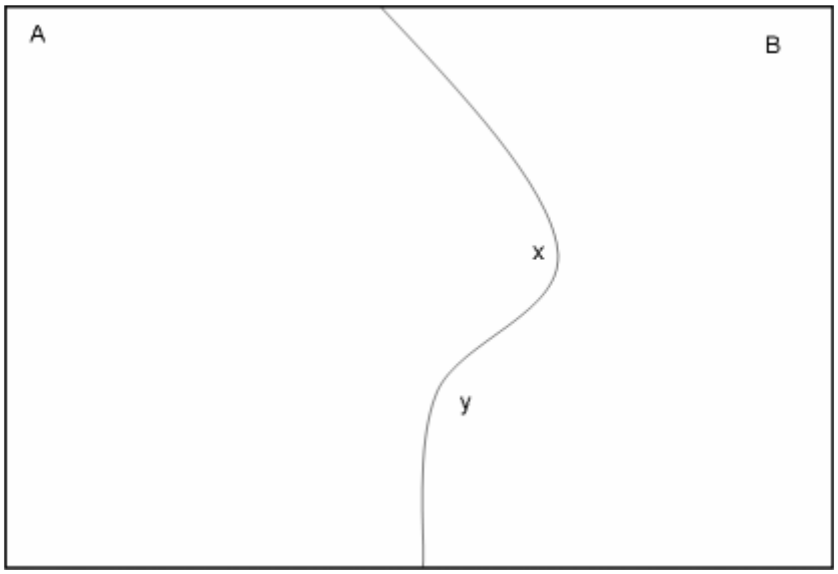


Figure 20: County Hierarchical Example

(6) What is the basic “instruction set” for communication between central cognition and visual imagery?
(7) Image decay and visual memory capacity. For now, we assume created images are stored in memory forever and that an agent can add an infinite amount of objects to an image. We realize these are not psychological valid assumptions.

These issues remain in front of us. We have outlined our theory and initial implementation of visual imagery within the constraints of a cognitive architecture, provided background as to why we believe the architecture must commit to a multi-modal approach, and presented results in two different domains that, although small, at least provide initial evidence we are on the right track. As we submit the theory and corresponding implementation to more demanding environments, our conjectures will solidify into more substantial and convincing empirical evidence.

REFERENCES

[1] J. Glasgow and D. Papadias, "Compuational Imagery," *Cognitive Science*, vol. 16, pp. 355-394, 1992.

[2] S. M. Kosslyn, "Mental Images and the Brain," *Cognitive Neuropsychology*, vol. 22, pp. 333-347, 2005.

[3] S. M. Kosslyn, *Image and Mind*. Cambridge: Harvard University Press, 1980.

[4] The Stanford Encyclopedia of Philosophy [Online]. Available: http://plato.stanford.edu/

[5] Z. Pylyshyn, "Mental Imagery: In search of a theory," *Behavioral and Brain Sciences*, vol. 25, pp. 157-238, 2002.

[6] J. R. Anderson, "Arguments Concerning Representations for Mental Imagery," *Psychological Review*, vol. 85, 1978.

[7] Z. Pylyshyn, "What the mind's eye tells the mind's brain: A critique of mental imagery," *Psychological Bulletin*, vol. 80, pp. 1-24, 1973.

[8] D. E. Kieras, "Beyond pictures and words: Alternative information-processing models for imagery effects in verbal memory," *Psychological Bulletin*, vol. 85, pp. 532-554, 1978.

[9] S. E. Palmer, *Vision Science Photons to Phenomenology*. Cambridge, Massachusetts: The MIT Press, 1999.

[10] W. L. Thompson and S. M. Kosslyn, "Neural systems activated during visual mental imagery: A review and meta-analyses," in *Brain mapping II: The systems*, J. C. Mazziotta and A. W. Toga, Eds. New York: Academic Press, 2000, pp. 535-560.

[11] B. Kuipers, "The Spatial Semantic Hierarchy," *Artificial Intelligence*, vol. 119, pp. 191-233, 2000.

[12] B. Chandrasekaran, U. Kurup, B. Banerjee, J. R. Josephson, and R. Winkler, "An Architecture for Problem Solving with Diagrams," in *Diagrammatic Representation and Inference*, A. Blackwell, K. Marriot, and A. Shimojima, Eds. Berlin: Springer-Velag, 2004, pp. 151-165.

[13] B. Chandrasekaran, U. Kurup, and B. Banerjee, "A Diagrammatic Reasoning Architecture: Design, Implementation and Experiments," presented at Proceedings AAAI Spring Symposium 2005, 2005.

[14] K. Forbus, D., J. Usher, and V. Chapman, "Sketching for military courses of action diagrams," in *Proceedings of the 8th international conference on Intelligent user interfaces*. Miami, Florida, USA: ACM Press, 2003.

[15] K. D. Forbus, J. Usher, and E. Tomai, "Analogical Learning of Visual/Conceptual Relationships in Sketches," presented at AAAI, Pittsburgh, Pennsylvania, 2005.

[16] J. E. Laird, A. Newell, and P. S. Rosenbloom, "Soar: An architecture for general intelligence.," *Artificial Intelligence*, vol. 33, pp. 1-64, 1987.

[17] A. Newell, *Unified Theories of Cognition*. Cambridge, Massachusetts: Harvard University Press, 1990.

[18] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin, "An Integrated Theory of the Mind," *Psychological Review*, vol. 111, pp. 1036-1060, 2004.

[19] D. E. Kieras and D. E. Meyer, "An Overview of the EPIC Architecture for Cognition and Performance with Application to Human-Computer Interaction," *Human-Computer Interaction*, vol. 12, pp. 391-483, 1997.

[20] R. M. Jones, J. E. Laird, P. E. Nielsen, K. J. Coulter, P. G. Kenny, and F. V. Koss, "Automated Intelligent Pilots for Combat Flight Simulation," *AI Magazine*, vol. 20, pp. 27-42, 1999.

[21] M. Tambe, W. L. Johnson, R. M. Jones, F. M. Koss, J. E. Laird, P. S. Rosenbloom, and K. B. Schwamb, "Intelligent Agents for Interactive Simulation Environments," *AI Magazine*, vol. 16, pp. 15-39, 1995.

[22] R. E. Wray, J. E. Laird, A. Nuxoll, D. Stokes, and A. Kerfoot, "Synthetic adversaries for urban combat training," *AI Magazine*, vol. 26, pp. 82-92, 2005.

[23] S. M. Kosslyn, *Image and brain - The resolution of the imagery debate.* Cambridge: MIT Press, 1994.

[24] R. A. Finke, *Principles of mental imagery*. Cambridge, MA: MIT-Press, 1989.

[25] S. Bertel, T. Barkowsky, D. Engel, and C. Freksa, "Computational Modeling of Reasoning with Mental Images: Basic Requirements," presented at 7th International Conference on Cognitive Modeling, Trieste, 2006.

[26] G. Kaufmann, "Mental Imagery and Problem Solving," in *Cognitive and Neuropsychological Approaches to Mental Imagery*, *NATO ASI Series*, M. Denis, J. Engelkamp, and J. T. E. Richardson, Eds. Dordecht / Boston / Lanchester: Martinus Nijhorff, 1988, pp. 231-240.
[27] D. Marr, *Vision*. San Francisco: Freeman, 1982.
[28] S. Pinker, "A Computational Theory of the Mental Imagery Medium," in *Cognitive and Neuropsychological Approaches to Mental Imagery*, *NATO ASI Series*, M. Denis, J. Engelkamp, and J. T. E. Richardson, Eds. Dordecht / Boston / Lanchester: Martinus Nijhorff, 1988, pp. 17-32.
[29] S. Ullman, "Visual Routines," Massachusetts Institute of Technology 1983.
[30] T. Helstrup, "Imagery as a Cogntive Strategy," in *Cognitive and Neuropsychological Approaches to Mental Imagery*, *NATO ASI Series*, M. Denis, J. Engelkamp, and J. T. E. Richardson, Eds. Dordecht / Boston / Lanchester: Martinus Nijhorff, 1988, pp. 241-250.
[31] J. H. Larkin and H. A. Simon, "Why a Diagram is (Sometimes) Worth Ten Thousand Words," *Cognitive Science*, vol. 11, pp. 65-99, 1987.
[32] B. Chandrasekaran, J. R. Josephson, B. Banerjee, and U. Kurup, "Diagrammatic Reasoning in Support of Situation Understanding and Planning," presented at Proceedings of Army Science Conference, Orlando, FL, 2002.
[33] F. Peronnet, M. Farah, and M. Gonon, "Evidence for shared structures between imagery and perception," in *Cognitive and Neuropsychological Approaches to Mental Imagery*, *NATO ASI Series*, M. Denis, J. Engelkamp, and J. T. E. Richardson, Eds. Dordecht / Boston / Lanchester: Martinus Nijhorff, 1988, pp. 357-362.
[34] D. L. Gilden, R. Blake, and G. Hurst, "Neural adaptation of imaginary visual motion," *Cognitive Psychology*, vol. 28, pp. 1-16, 1995.
[35] J. Lehman, J. Laird, and P. Rosenbloom, A Gentle Introduction to Soar, An Architecture For Human Cognition: 2006 Update* [Online]. Available: http://ai.eecs.umich.edu/soar/sitemaker/docs/misc/GentleIntroduction-2006.pdf
[36] A. Nuxoll and J. E. Laird "A Cognitive Model of Episodic Memory Integrated With a General Cognitive Architecture," in *International Conference on Cognitive Modeling* Pittsburgh, Pennsylvania, 2004.
[37] S. Nason and J. E. Laird, "Soar-RL: Integrating Reinforcement Learning with Soar," in *International Conference on Cognitive Modeling*. Pittsburgh, Pennsylvania, 2004.
[38] R. P. Marinier and J. E. Laird, "A Cognitive Architecture Theory of Comprehension and Appraisal," in *Agent Construction and Emotion*. Vienna, Austria, 2006.
[39] D. H. Eberly, *3D Game Engine Architecture, First Edition: Engineering Real-Time Applications with Wild Magic* Morgan Kaufman Publishers Elsevier Inc, 2005.
[40] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*: Addison-Wesley Professional Computing Series, 1995.
[41] D. Kettani and B. Moulin, "A Spatial Model Based on the Notions of Spatial Conceptual Map and of Object's Influence Areas," in *Proceedings of the International Conference on Spatial Information Theory: Cognitive and Computational Foundations of Geographic Information Science*: Springer-Verlag, 1999.
[42] M. Buro and T. Furtak, "RTS Games and Real-Time AI Research," presented at Behavior Representation in Modeling and Simulation Conference (BRIMS), Arlington VA, 2004.
[43] J. R. Anderson, *Cognitive Psychology and Its Implications*, Sixth ed. New York, NY: Worth Publishers, 2005.