Visual Re-Representation in Creative Analogies

Jim Davies, Ashok K. Goel, Nancy J. Nersessian
Artificial Intelligence Laboratory, College of Computing
Georgia Institute of Technology, Atlanta, Georgia 30307, USA
jim@jimdavies.org, goel@cc.gatech.edu, nancyn@cc.gatech.edu

To appear in the 2003 the *IJCAI Proceedings of The Third Workshop on Creative Systems.*

Abstract

Visual representations seem to play a significant role in many creative analogies. In this paper, we describe a specific role of visual representations: two situations that appear dissimilar non-visually may appear similar when re-represented visually. We present a computational theory of multi-modal analogy in which visual re-representation enables analogical transfer in cases where there are ontological mismatches in the non-visual representation. We have developed a computer program, called *Galatea*, that implements a core part of our theory: it transfers problem-solving procedures between analogs containing only visual knowledge. In this paper, we describe both how Galatea accomplishes the transfer task using only visual knowledge and how it might be extended to support visual re-representation in multi-modal analogies.

1 Introduction

A central issue for research on creativity is that any solution to a problem has to start from what we already know. So, how is it possible to create novel solutions? There is ample evidence that analogy plays a central role in finding creative problem solutions, and that many of the most creative analogies involve cross-domain transfer [Nersessian, 1992; 2002; Bhatta and Goel, 1997a; 1997b; Thagard, 1992; Darden, 1983; Gentner and Stevens, 1983; Boden, 1990]. But how does one recognize similarity across domains and use it to arrive at a solution?

Clearly some kind of abstraction processes are involved in transferring problem solutions across domains. Most of the literature on analogy considers abstraction processes involving non-perceptual, amodal (e.g., linguistic) representations. We hypothesize that for some problems, the abstraction processes involve re-representation, changing the representation from an amodal format to a modal (e.g., visual) format. In these cases it is the similarity in the modal representation that enables analogical transfer.

To take an example, imagine that a reasoner is trying to figure out how to put batteries into a tape recorder, and there is an opportunity to use a source case in which film is put into a camera. Since *film* is a different entity from *battery*, and *tape-recorder* is a different entity from *camera*, analogical retrieval, mapping, and transfer will be hindered unless the reasoner can find some similarity between them. If the entities referred to are different things in the ontology of the representation language, then there is a mismatch between the semantics of representations of the two situations. This is sometimes called the *ontological mismatch* problem.

One way the two situations in the above example are similar is that they visually resemble each other: the batteries and the film canister are shaped like cylinders, and the tape recorder and the camera are shaped like rectangular prisms. In this example, the problem constraints pertain to the shape of the objects involved. Thus the visual similarity of the tape recorder and the camera (both may have cylindrical holes) is more relevant to the problem than, say, their functional similarity as recording devices, because their *shapes* have more to do with the placement of batteries and films than their *functions* do.

On our hypothesis, turning a non-visual representation into a visual one (visual instantiation) is one mechanism for resolving ontological mismatches. When a reasoner encounters an ontological mismatch in the non-visual representations of the target problem and the source case, it may dynamically create visual representations of the problem and the case, and transfer problem solving strategies (or solutions) between the generated visual analogs. The final solution of the non-visual problem involves specifying the visual representation back into non-visual form.

Note that a critical part of this process is transfer of problem solving strategy between two visual analogs. Thus, if our hypothesis is correct, it should be possible to transfer problem solving procedures from a source case to a target problem using representations that are purely visual. Therefore, the first core task in developing and evaluating our theory of multi-modal analogy is to develop and evaluate a computer program that can accomplish the transfer task using only visual knowledge.

Indeed, although above we outlined the case for visual re-representation in creative analogies, in many situations analogical problem solving may use visual representations from the outset. For example, problems in many design domains contain drawings, diagrams, animations, photographs, videos, etc. Instructions for assembling a complex artifact, for example, often are presented to people in a completely

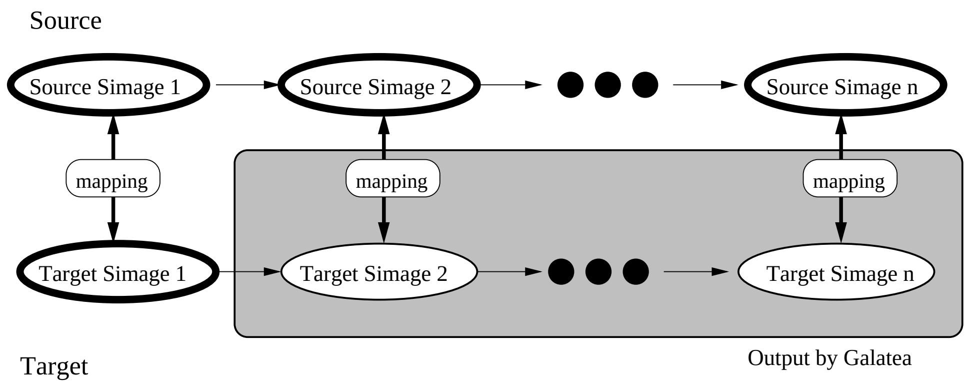

Figure 1: This Figure illustrates Galatea's input and output in the abstract. The knowledge states in the source case are depicted as ovals along the top of the figure. The knowledge states are represented as s-images. Transformations between the states in the figure are depicted as arrows. The target problem is depicted as the first oval along the bottom. All things in the gray box are output by Galatea.

diagrammatic from. Thus, establishing transfer of problem solutions using visual knowledge alone not only supports our theory of multi-modal analogies but also is an important task by itself.

This paper, then, has two goals: (i) describe our theory of visual re-representation in multi-modal analogies and (ii) describe an operational computer program called *Galatea* that implements a the transfer task in visual analogy. We begin with a description of Galatea that addresses visual analogy, and then show how it can be extended to visual re-representation in creative analogies.

We will use Duncker's classic fortress/tumor problem [Duncker, 1926] as a running example throughout this paper. [1]

2 Visual Analogy: Transfer

Problem solving in the fortress story in the Duncker example involves a series of knowledge states and transformations between them. A knowledge state specifies information about a specific configuration of elements and parameters that characterize the problem, and transformations are operations that change the configuration in a state and lead to a new knowledge state. The first knowledge state corresponds to the initial description of the problem. Starting from the first knowledge state in the fortress story, the first transformation is to break the army up into smaller armies. This leads to the second knowledge state containing smaller armies. The second transformation is to move the armies to different roads, and so on.

[1] Experimental participants read a story about a general who must overthrow a dictator in a fortress. His army is poised to attack along one of many roads leading to the fortress when the general finds that the roads are mined such that large groups passing over will set them off. To solve the problem, the general breaks the army into smaller groups and they take different roads simultaneously and arrive together at the fortress. Participants are then given a tumor problem, in which a tumor must be destroyed with a ray of radiation, but the ray will destroy healthy tissue on the way in, killing the patient. The analogous solution is to have several weaker rays converging on the tumor [Duncker, 1926].

This analysis leads to two constraints on the visual representation of the source case containing the fortress story. First, a knowledge state will be represented as a diagram, and the transformations will be operations that can operate on the visual primitives in the diagram representing a state. Further, two successive states will be connected by only one primitive transformation. Second, visual information will be represented using symbolic, structured, *descriptive* representations. This is differentiated from *depictive* representations (e.g., bitmaps), where a depictive representation only "specifies the locations and values of points in space" [Kosslyn, 1994]. The symbolic representation provides the standard benefits of discreteness, abstraction, ordering, and composition.

2.1 Galatea: A Computer Implementation

Galatea is an operational program written in LISP. It implements the transfer of problem solving procedures in visual analogies. Figure 1 illustrates Galatea's input and output in the abstract. Each knowledge state is represented as a symbolic image or *s-image*. The reasoner takes as input a source case, an initial knowledge state in the target problem, and an analogical mapping between the s-image representing the first knowledge state in the source case and the initial knowledge state in the target problem. The source case is a complete problem solving episode. The system transfers the visual transformations from the source to the target, creating new target s-images and analogical mappings along the way. Figure 2 illustrates Galatea's input and output for the Duncker problem.

Galatea is intended to be able to operate on problems involving physical systems. It presently works on three problems: The Duncker problem, a case of scientific analogical reasoning by James Clerk Maxwell [Davies *et al.*, 2003], and a cake/pizza problem in which a single pizza must be distributed among several people (as briefly described below).

2.2 Knowledge and Representation

Covlan (for Cognitive Visual Language) provides an ontology of visual primitives and transformations. Table 1 shows

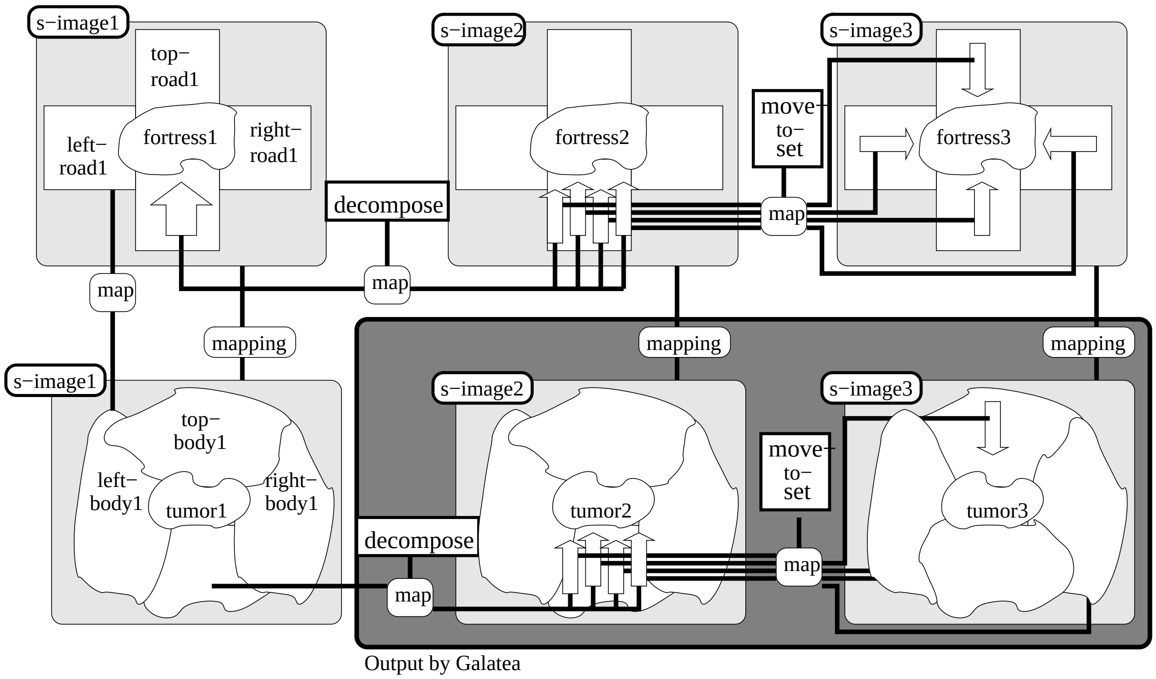

Figure 2: This Figure shows Galatea's input and output for the Duncker problem. The top series of s-images in the Figure shows the visual representation of the solved fortress problem. The bottom series of s-images shows the target tumor problem. The bottom left s-image is the initial state of the tumor problem. The darkly shaded box shows the output of the system.

Covlan's ontology of transformations.

Table 1	
Transformation name	**arguments**
move-to-location	object, new-location
move-to-touch	object, object2, new-location
move-above	object, object2
move-to-right-of	object, object2
move-below	object, object2
move-to-left-of	object, object2
move-in-front-of	object, object2
move-off-s-image	object, location
move-to-set	object, object2
rotate	object, direction
start-rotating	object, direction
stop-rotating	object
start-translation	object, direction
stop-translation	object
set-size	object, new-size
add-element	object, location (optional)
remove-element	object
decompose	object, number-of-resultants, type
scale	object, new-size

Each transformation is a function with arguments. All transformations operate on some *object*, and many have additional arguments as well. These transformations implement normal graphics manipulations such as translation (*move-to-location*, *move-to-touch*, *move-above*, *move-to-right-of*, *move-to-left-of*, *move-below*), rotation (*rotate*), and scaling (*set-size*). In addition there are transformations for adding and removing elements from the s-image (*add-element*, *remove-element*). Making topological changes of this kind to imagined physical systems has been shown in earlier work to be useful in problem solving [Griffith *et al.*, 2000; 1996].

Certain transformations (*start-rotating*, *stop-rotating*, *start-translation*, *stop-translation*) are changes to the dynamic behavior of the system under simulation. For example, *rotate* changes the orientation of an element once, as one might turn a chair to face a window. Such a transformation changes the position of an element between states. In contrast *start-rotating* sets an element in motion, as one might spin a top. A square that has been affected by this transformation would not simply be rotated in the next state, but actively rotating.

Covlan's ontology of *primitive visual elements* (Table 2) contains: *polygon, rectangle, triangle, ellipse, circle, arrow, line, point, spline,* and *text*. The elements are frame-like structures with slots that can hold values. For example, a *triangle* has a *location, size, height, width,* and *orientation*. All elements have a *location*, which is an absolute location on an s-image (e.g. *top, right*).

In the fortress problem, the fortress is represented as a *spline*, the army as an *arrow* with *thickness* of *very-thick*. Likewise, in the tumor problem, the ray of radiation is represented as an *arrow* with *thickness* of *very-thick*, and the tu-

Table 2	
Primitive Element name	**attributes**
polygon	location, size
rectangle	location, size, height, width, orientation
triangle	location, size, height, width, orientation
ellipse	location, size, height, width, orientation
circle	location, size, height
arrow	location, length, start-point, end-point, thickness
line	location, length, end-point1, end-point2, thickness
point	location
spline	location, start-point, mid-point, end-point, thickness
text	location, length, letters

mor is represented as a *spline* (see Figure 2.) Two s-images are generated during processing; the final generated s-image of the tumor problem is represented in Figure 3.

In the fortress/tumor example, after the *decompose* transformation generates a number of smaller armies (by transforming a think arrow into thinner arrows), they must be dispersed to the various roads, in various locations in the image. In a previous version of Galatea [Davies and Goel, 2001; Davies *et al.*, 2003] each army arrow was *moved-to-location* individually to each road line. This solution was brittle because the number of roads to which the armies moved needed to match exactly the number of body areas the weaker rays moved to in the target.

The current version of Galatea uses *sets* to address this problem. By grouping the armies, roads, rays, and body parts into their own sets, the system adapts the solution in the source analog to accommodate differing numbers of any of these elements. Rather than using the *move-to-location* transformation on each army, it implements a new transformation *move-to-set* to the *set* of armies. The argument to this function is a *set* of roads. The *move-to-set* function takes one set and distributes its members around the locations of another set.

2.3 Inference and Processing

Galatea focuses on the transfer and adaptation stage of analogy. In particular, it adapts and transfers each transformation in the source problem to the target.

A transformation, such as *Decompose*, can be used to turn any primitive element into an arbitrary number of resultants, which is taken as an argument. An argument of a transformation can be an instance of one of three cases. First, the argument can be a literal, like the number *4* or the location *bottom*. Literals are translated directly.

Second, the argument could be a primitive element member of the source s-image. In this case, the transfer procedure operates on the analogous object in the target s-image. For example, in the fortress problem, the *soldier-paths* are moved to the *roads*. When *move-to-set* is transferred to the tumor problem, the argument *set-of-roads* is adapted to the analogous *set-of-body-areas.*

In the third case, the argument can be a function. Since this case does not occur in the Duncker problem, we will use another example to describe it. Let us suppose that a reasoner needs to feed six people with one Sicilian sheet pizza. An analog in memory of cutting a sheet cake for four people is used to generate a solution. Transfer is still difficult because somehow the *4* in the cake analog must be adapted to the number *6* in the source analog. Knowing how many pieces into which to cut the cake or pizza depends on the number of people in the problem. Since a set is different from a count of the set's members, some notion of count is needed. The use of *functions* as arguments to transformations helps address this problem. The cake analog is represented with a *function* that counts the number of people as its argument for the *decompose* transformation. This function has an argument of its own, namely the set of cake eaters, which during adaptation adapts into the set of pizza eaters. When the transformation is applied to the pizza, it counts the members of the set of people in the pizza problem (which results in six), and produces six pieces of pizza.

2.4 Algorithm

1. **Identify the first s-images of the target and source cases.**
2. **Identify the transformations and associated arguments in the current s-image of the source case.** This step finds out how the source case gets from the current s-image to the next s-image. In our example, the transformation is *decompose*, with *four* as the *number-of-resultants* argument (not shown).
3. **Identify the objects of the transformations.** The object of the transformation is what object the transformation acts upon. For the *decompose* transformation is the *soldier-path1* (the thick arrow in the top left s-image in Figure 2.)
4. **Identify the corresponding objects in the target problem.** The *ray1* (the thick arrow in the bottom left s-image) is the corresponding component of the source case's *soldier-path1*, as specified by the correspondences between the s-images (not shown). A single object can be mapped to any number of other objects. If the object in question is mapped to more than one other object in the target, then the transformation is applied to all of them in the next step.
5. **Apply the transformation with the arguments to the target problem component.** A new s-image is gener-

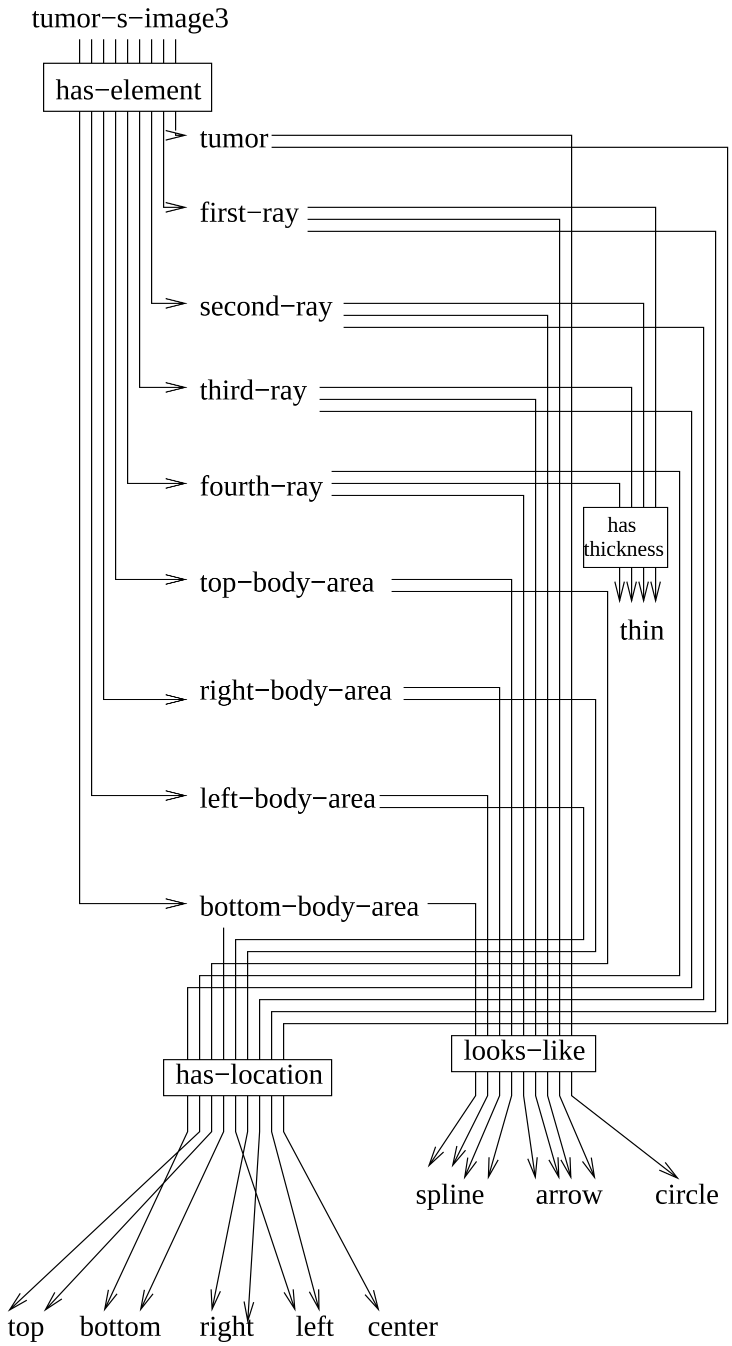

Figure 3: This Figure illustrates a portion of Galatea's representation of the third s-image in the tumor series illustrated in Figure 2. The third s-image in the tumor series represents the final solution generated by Galatea as a result of analogical transfer. The representation consists of a series of propositions, indicated in the Figure as labeled arrows connecting two elements. The *tumor-s-image3* is connected with a *has-element* relation to each element in the s-image. The elements in the s-image each have a location and are connected to a primitive visual element type with a *looks-like* relation. Each ray, represented as an arrow, also has a *thickness* —in this s-image, *thin*. Each arrow also has start and end points, also with locations (not shown in the figure). The s-image is connected to the s-image before it with a *transform-connection*. Not shown in figure are the maps that connect the elements of this s-image to the previous s-image, as well as the maps to the analogous source s-image.

ated for the target problem (bottom middle) to record the effects of the transformation. The *decompose* transformation is applied to the *ray1*, with the argument *four*. The result can be seen in the bottom middle s-image in Figure 2. The new rays are created for this s-image. Adaptation of the arguments can happen in three ways, as described above: If the argument is an element of the source s-image, then its analog is found. If the argument is a function, then the function is run (note that the function itself may have arguments which follow the same adaptation rules as transformation arguments). Else the arguments are transferred literally.

6. **Map the original objects to the new objects in the target case.** A transform-connection and mapping are created between the target problem s-image and the new s-image (not shown). Maps are created between the corresponding objects. In this example it would mean a map between *ray1* in the left bottom s-image and the four rays in the second bottom s-image. This system does not solve the mapping problem, but a mapping from the correspondences of the first s-image enable the mappings for the subsequent s-images to be automatically generated.
7. **Map the new objects of the target case to the corresponding objects in the source case.** Here the rays of the second target s-image are mapped to soldier paths in the second source s-image. This step is necessary for the later iterations (i.e. going on to another transformation and s-image). Otherwise the reasoner would have no way of knowing on which parts of the target s-image the later transformations would operate.
8. **Check to see if goal conditions are satisfied.** If they are, exit, and the problem is solved. If not, and there are further s-images in the source case, set the current s-image equal to the next s-image and go to step 1. If there are no further s-images, then exit and fail. Goal conditions are represented non-visually [Davies and Goel, 2001].

Galatea shows that visual knowledge alone, with no amodal knowledge, is sufficient for enabling analogical transfer, supporting a central hypothesis of our theory of creative analogies. It suggests a computational model of analogy based on dynamic visual knowledge that complements traditional models based on amodal knowledge. Although Galatea does not yet address the issues of retrieval and mapping, other implemented computer programs have (e.g. [Yaner and Goel, 2002; Ferguson, 1994]). Thus, we confidently conjecture that visual knowledge alone can enable the first three stages of analogy: retrieval, mapping, and transfer.

3 Multi-Modal Analogies: Re-Representation

Galatea addresses a core part of our theory of creative analogies: analogical problem-solving transfer using only visual knowledge. Our general theory of creative analogies suggests why and how visual reasoning is useful even with cases whose representations need not be visual. There is psychological evidence that humans make use of visual information when doing problem solving in general [Schrager, 1990; Farah, 1988; Casakin and Goldschmidt, 1999; Monaghan and Clement, 1999], but the details of what makes visual knowledge useful for analogy in natural and artificial reasoners is largely unknown.

Another way to frame this problem is that we do not know under what conditions it is useful for a reasoner to generate and process a visual representation. Our work on Galatea suggests that one reason to use visual representations is that ideas that are semantically distant with a non-visual representation (e.g. a marching army and a ray of radiation) may be semantically closer with a visual representation. Turning non-visual representations into visual ones (a process we call *visual instantiation*) is one possible solution to the *ontological mismatch* problem.

3.1 The Ontological Mismatch Problem

One kind of ontological mismatch occurs when the symbols representing two similar things are not the same. In a non-visual representation of the Duncker problem, the ray and the army are different symbols. Thus, without some notion of similarity between them, they cannot be aligned, which hinders analogical problem solving.

Ontological mismatches can be encountered during analogical retrieval, mapping, or transfer. In the retrieval stage, ontological mismatches can hinder retrieval of appropriate analogs. Psychological studies show that analogs are retrieved from memory based on surface similarity of the target analog to the retrieved source[Falkenhainer *et al.*, 1990]. Similar ideas represented with different symbols will fail to *appear* similar to the reasoner.

Upon retrieval of an analog, the reasoner might have trouble with *mapping* ideas that need to be aligned, such as the *tumor* with the *fortress*, because they are represented with different symbols.

Even if this mapping problem is overcome, the reasoner could still have a problem in transfer of the solution strategy. Let us suppose that the reasoner knows of a solved problem which involves breaking up an army into smaller groups. The army is represented as a group of constituent soldiers. The target problem involves a ray of radiation which must be turned into a number of rays with less intensity. The ray might be represented as energy, with a number associated with its intensity, a representation which that serves some other task (e.g. so that numeric intensities can be added). Thus, not having anticipated that the ray and army might need to be aligned, they could have been encoded with incompatible representations. The transformation applied to the army will not work on the ray because the representation of the ray, in this example, does not have constituent parts: breaking something into parts is different from dispersing energy.

The point of this analysis is to show that a reasonable non-visual representation can fail for transfer in analogical problem solving. It is possible to represent this problem with no ontological mismatches (e.g. [Holyoak and Thagard, 1989] do), but ontological mismatches are bound to occur in any large knowledge base [Lenat and Guha, 1990].

3.2 Resolution of Ontological Mismatches

In our theory, ontological mismatches encountered in non-visual representations are resolved by providing a level of visual abstraction at which two different symbols are similar. This process of visual instantiation offers a means for resolving ontological mismatches different from, say, using a type hierarchy. For example, the ray in the *fortress* and the *tumor* may not be under any same superordinate category. Representing them both as *splines*, however, shows a similarity between these distant concepts. This kind of visual abstraction works especially well in conditions under which the visual properties of the objects represented are related to the properties relevant to the current task (See Figure 4.)

For the retrieval task, the reasoner may visually instantiate the elements in question, and find similarity through comparison of the visual symbols. For example, the *ray* and the *soldier-path* might both instantiate to *arrows*, at which point the reasoner can identify the similarity.

The mapping stage outputs alignments between elements of the source and target initial problem states. As in retrieval, visual instantiation of the target and the source can help align the symbols. For example, visual instantiation of the *tumor* and *fortress* as *splines* abstracts them to the same symbol, enabling alignment.

In the transfer task, the transformations that connect the different knowledge states in the source are transferred to the target. The elements in the target problem that a transferred transformation affect are analogous to the elements that get affected in the source. Sometimes, however, there can be problems in transfer due to ontological mismatches. For example, as discussed above, trying to transfer the *break-up* transformation to the ray in the Duncker problem will not work because the ray does not have constituent parts.

Visual representations can be used as an intermediate level of abstraction in the transfer task as well. Let us suppose that in solving the Duncker problem, both the army and the ray get visually instantiated as a *line*. The *break-up* transformation, too, gets visually instantiated as the *decompose* visual transformation. In the generated visual representation, the transfer of the transformation occurs without hindrance because *decompose* can apply equally well to both lines.

In Figure 5, the top two ovals represent the first two knowledge states of a non-visual representation of the source case in the Duncker problem, connected with a *break-up* transformation. Also input is the initial state of the target problem and the analogical mapping between them. The grayed area is generated by the reasoner.

The reasoner first tries to transfer the *break-up* transformation directly, but cannot because *break-up*, as mentioned above, only works on things with constituent parts. At this stage, the reasoner generates a visual representation of the solved source case and the target problem. For example, the large marching army and the ray are represented as thick arrows. All the elements of these analogs are turned into visual primitives, including the *break-up* transformation, which instantiates into *decompose*, which is a visual transformation that takes a visual object and turns it into smaller objects of the same type. In this case, it turns the thick lines in the visual target into thin lines. Unlike *break-up*, the *decompose* function transfers from the source to the target, because both *break-up* and *distribute* share the same visual abstraction, *decompose*.

Now that the problem is solved in the visual representation, it is re-specified back into the non-visual representation. This can be done because *decompose* translates not only into *break-up* but also into *distribute*, which takes some intensity value and breaks it up into some number of elements with a weaker intensity.

4 Discussion

The ontological mismatch problem has been identified and extensively studied in the context of large knowledge bases, specially inter-operable knowledge systems. Various models of analogical problem solving resolve the ontology mismatch problem in different ways.

Case-based reasoning appears to assume that memory is so massively populated and well organized and the retrieved case so similar to the target problem that ontological mismatches simply will not occur — if the source is so similar to the target that it need only be "tweaked" to get the desired solution, then there is simply no ontological mismatch. In contrast, Yarlett and Ramscar [Yarlett and Ramscar, 2000] specifically address the ontological mismatch problem in analogical reasoning. Their system takes two different symbols and evaluates their similarity using Latent Semantic Analysis [Landaur, 1998], a database of correlations between all words representing their co-occurrence in a text. The analogical mapper treats as identical any pair of symbols which correlate above a specified threshold. In our theory, objects and operations that appear different in a non-visual representation may look more similar in a different, visual representation.

The literature on visual analogy is small but rapidly growing. ANALOGY is an early computer program that performed visual analogies [Evans, 1968]. It solved multiple choice visual analogy problems of the kind found on intelligence tests (e.g. A:B::C:?). It does this by describing how to turn A into B, and then testing into which choice might C turn into in a similar manner.

Although the structure-mapping theory does not specifically address the ontological mismatch problem, it is applicable to both non-visual and visual representations. According to this theory [Falkenhainer *et al.*, 1990], two ideas are considered similar if the idea's properties and the relations between it and surrounding elements are the same as the relations between another idea and its surrounding elements. For example, an electron is similar to a planet because it revolves around some body (nucleus or star). Galatea's use of multiple kinds of arguments for transformations (e.g., literals and functions) is similar to that of the structure-mapping theory.

GeoRep [Ferguson and Forbus, 2000] connects visual and non-visual knowledge in a different way than Galatea. First, it takes in line drawings as inputs and outputs the visual relations in it. Then, it takes the visual relations as input and outputs domain-specific causal descriptions. Covlan's visual primitives are similar to that of GeoRep.

Like Galatea, LetterSpirit is a model of analogical transfer using visual representations [McGraw and Hofstadter, 1993].

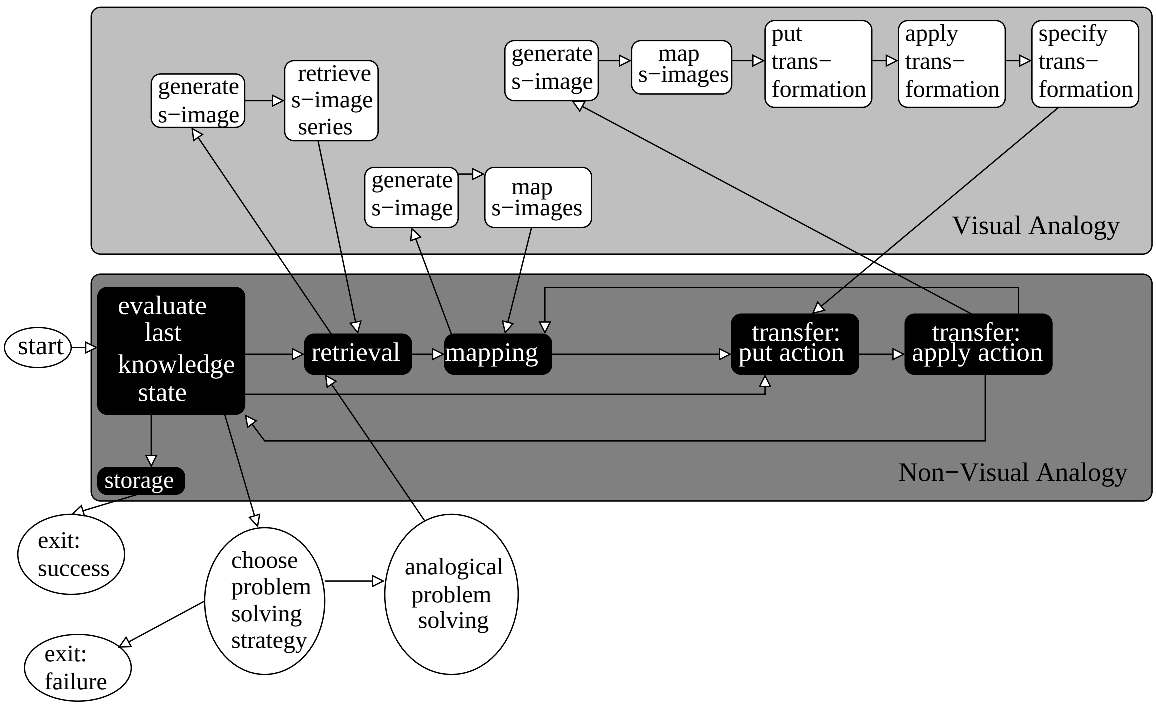

Figure 4: This Figure illustrates the computational process for multi-modal analogies. First the current last knowledge state in the target problem is evaluated. If it satisfies the goal conditions, then the reasoner stores the solution and exits. Else it may elect to use analogy to address the problem. Retrieval can occur non-visually, but failing that, the reasoner may generate s-images for the target problem, and try to retrieve based on those (processes in the top shaded box are visual processes). When a source case is retrieved, the reasoner attempts to map the elements. Again, the generated s-images can be used to facilitate mapping. With a mapping in place, the reasoner may attempt to transfer the solution from the source to the target by transferring operators and applying them. If the transfer processes fails due to an ontological mismatch, the reasoner may use generated s-images to resolve the conflict. The procedure transformations is re-specified back into non-visual operators, and are evaluated in the non-visual representation.

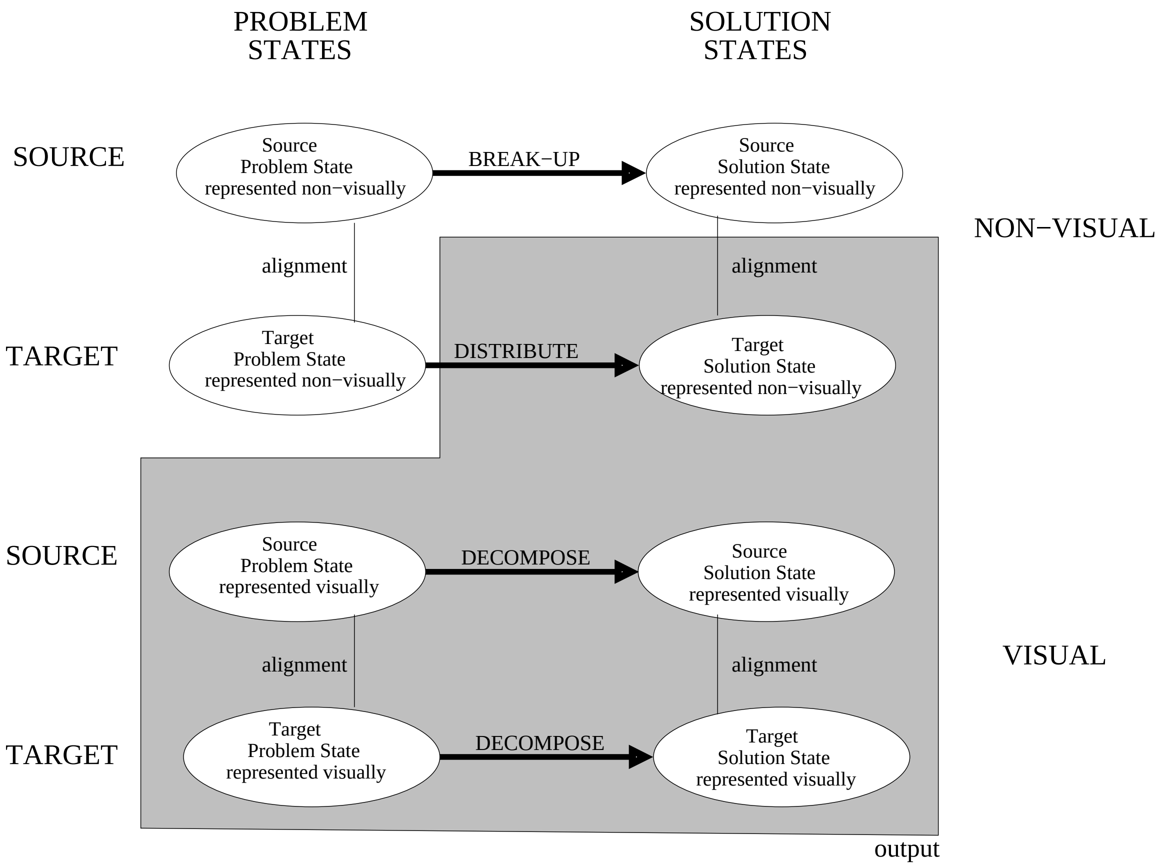

Figure 5: This Figure illustrates visual re-representation in the transfer stage of analogical problem solving. When the reasoner cannot directly transfer the *break-up* transformation from the source case to the target problem, it creates a visual abstraction of the knowledge states and transformations. The non-visual *break-up* transformation instantiates to the visual transformation *decompose.* Transfer of *decompose* from the visual source to the visual target is now possible. After the transfer, the reasoner specifies the transferred visual transformation back into the appropriate non-visual transformation, *distribute.* Galatea implements the processing in the bottom half of this figure.

It takes a stylized seed letter as input and outputs an entire font that has the same style. It does this by determining what letter is presented, determining how the components are drawn, and then drawing the same components of other letters the same way. The analogies between letters are already in the system: the vertical bar part of the letter *d* maps to the vertical bar in the letter *b*, for example. A mapping is created for the input character. For example, the seed letter may be interpreted as an *f* with the cross-bar suppressed. When the system makes a lower-case *t*, by analogy, it suppresses the crossbar.

Like ANALOGY, LetterSpirit transfers single transformations/attributes (e.g. crossbar-suppressed) and therefore cannot make analogical transfer of procedures (e.g. moving something, then resizing it) like our theory can. In contrast, one can see how Galatea might be applied to the font domain: The stylistic guidelines in LetterSpirit, such as “crossbar suppressed” are like the visual transformations in our theory: it would be a transformation of removing an element from the image, where that element was the crossbar and the image was a prototype letter *f*. Then the transformation could be applied to the other letters one by one. In this way our theory has more generality than LetterSpirit, which by design only works on alphabets.

The VAMP systems are analogical mappers [Thagard *et al.*, 1992]. VAMP.1 uses a hierarchically organized symbol/pixel representation. It superimposes two images, and reports which components have overlapping pixels. VAMP.2 represented images as agents with local knowledge. Mapping is done using ACME/ARCS [Holyoak and Thagard, 1997]. The radiation problem mapping was one of the examples to which VAMP.2 was applied. [Croft and Thagard, 2002] created a computational model DIVA which does analogical mapping using ACME. What it maps are three-dimensional representations in hierarchically organized scene graphs. Things in the graph can be associated with behaviors, so it can represent dynamic systems. This system was a mapping system, and though it deals with the Duncker problem, it does not transfer the solution procedure.

MAGI [Ferguson, 1994] uses the structure-mapping theory to find examples of symmetry and repetition in a single image. JUXTA [Ferguson and Forbus, 1998] uses MAGI in its processing of a diagram of two parts, and a representation of the caption. It outputs a description of what aligns with what, along with important and distracting differences. It models how humans understand repetition diagrams.

Like Galatea, MAGI, JUXTA, and the VAMPs use visual knowledge. But unlike Galatea their focus is on the creation of the mapping rather than on transfer of a solution procedure. MAGI’s and our theory are compatible: a MAGI-like system might be used to create the mappings that our theory uses to transfer knowledge. The theory behind the VAMPs is incompatible because they use a different level of representation for the images.

Though not an implemented computer program, the image-schema theory of Lakoff and Johnson [Johnson, 1990; Lakoff and Johnson, 1983] says that humans use metaphors pertaining to their bodies to reason about external situations. Our theory is similar in that it uses perceptual abstraction to find similarity between ideas and to reason about external ideas. Our ideas differ in that their image-schemas are multi-sensory and based on bodily action, where our theory, though it does not exclude such representations, focuses on visual abstractions.

The perceptual symbol system theory [Barsalou, 1999] holds that all mental representations are perceptual in nature, so that all reasoning operates on perceptual symbols. On this view, what we have called the non-visual or amodal representation is actually perceptual as well, and what we are calling the visual level of representation is a more abstract perceptual representation.

Our theory of visual re-representation in creative analogies evolves from our earlier work on analogical reasoning, and shares two central themes with it. The first central theme is the development of content accounts that support analogical reasoning. The Ideal system [Bhatta and Goel, 1997a; 1997b], for example, used structure-behavior-function (SBF) models for supporting analogical remindings, mappings and transfer in the context of conceptual design. The Torque system [Griffith *et al.*, 2000; 1996] used SBF models for analogical remindings, transfer and evaluation in the context of scientific problem solving. Qian and Gero [Qian and Gero, 1996] use similar Function-Behavior-Structure models to support analogy-based design.

The second central and consistent theme is the use of abstractions to facilitate analogical reasoning. In the Ideal system, for example, analogical transfer was enabled by behavior-function abstractions of SBF models. Ideal provided a content account of the behavior-function abstractions in the form of generic teleological mechanisms and generic physical principles. The Torque system similarly provided a content account of generic structural transformations and used them to facilitate analogical reasoning.

One hypothesis of the present work is that analogical transfer is a difficult part of the analogical problem solving process — that even with the appropriate source retrieved and the correct mapping, transfer and adaptation can prove difficult. We are currently running a psychological experiment to test this hypothesis. Experimental participants are given the fortress story, the tumor problem, and a mapping between them, and are asked to solve the problem. We have a diagram group which is given abstract diagrams of the fortress and tumor problems, and a control condition in which no diagram is given. If our theory is correct, both conditions will have difficulty, because transfer is difficult, but the diagram condition will have a higher rate of successful transfer of the analogous solution.

In our future work we will expand Galatea to demonstrate how ontological problems in non-visual representations can be aided by visual instantiation and visual reasoning.

4.1 Conclusion

The above work leads us to our main conclusions, the first pertaining to analogy and the second to creativity. First, Galatea shows that visual knowledge alone is sufficient for the transfer task. This is in contrast to earlier theories of analogical transfer that rely on non-visual knowledge. Second, our theory of creative analogies says that visual representations can be used as an intermediate representation that a reasoner can use to find similarities between otherwise dissimilar objects.

According to [Mednick, 1962] reasoners are creative because they can make semantic connections between dissimilar things. Solving the Duncker problem by transferring the solution from the fortress problem to the tumor problem requires creativity of this kind. Our theory shows how this might occur: by changing representations to visual and back again, cognitive and artificial reasoners can make connections among distant objects, strategies and ideas.

References

[Barsalou, 1999] Lawrence W. Barsalou. Perceptual symbol systems. *Behavioral and Brain Sciences*, 22, 577-609.

[Bhatta and Goel, 1997a] Sambasiva R. Bhatta & Ashok K. Goel. A functional theory of design patterns. In the *Proceedings of IJCAI-97*. pages 294–300, 1997.

[Bhatta and Goel, 1997b] Sambasiva R. Bhatta & Ashok K. Goel. Learning Generic Mechanisms for Innovative Design Adaptation. *Journal of Learning Sciences,* 6(4):367-394, 1997b.

[Boden, 1990] Margaret A. Boden. *The Creative Mind.* Basic Books, 1990.

[Casakin and Goldschmidt, 1999] H. Casakin & G. Goldschmidt. Expertise and the use of visual analogy: Implications for design education. *Design Studies*, 20:153–175, 1999.

[Croft and Thagard, 2002] D. Croft, & Paul Thagard. Dynamic imagery: A computational model of motion and visual analogy. In L. Magnani (Ed.), *Model-based reasoning: Scientific discovery, technological innovation, values.* New York: Kluwer/Plenum, 2002.

[Darden, 1983] Lindley Darden. Reasoning by analogy in scientific theory construction. In Michalski, *Proceedings of the International Machine Learning Workshop*, pp. 32–40, 1983

[Davies and Goel, 2001] Jim Davies, & Ashok K. Goel. Visual analogy in problem solving. *Proceedings of the International Joint Conference on Artificial Intelligence 2001.* pp 377–382. Morgan Kaufmann publishers, 2001

[Davies *et al.*, 2003] Jim Davies, Nancy J. Nersessian, & Ashok K. Goel. Visual models in analogical problem solving. To appear in *Foundations of Science 2003, special issue on Model-Based Reasoning: Visual, Analogical, Simulative.* Magnani, L. & Nersessian, N. J., Eds., 2003.

[Duncker, 1926] K. Duncker (1926). A qualitative (experimental and theoretical) study of productive thinking (solving of comprehensible problems). *Journal of Genetic Psychology*, 33:642–708, 1926.

[Evans, 1968] T. G. Evans A heuristic program to solve geometric analogy problems. In *Semantic Information Processing* edited by Marvin Minsky MIT Press, Cambridge, MA. (1968)

[Falkenhainer *et al.*, 1990] Brian Falkenhainer, Kenneth D. Forbus, and Dedre Gentner. The Structure mapping engine: algorithm and examples. *Artificial Intelligence,* 41:1-63, 1990.

[Farah, 1988] M. J. Farah. The neuropsychology of mental imagery: Converging evidence from brain-damaged and normal subjects. In J. Stiles-Davis, M. Kritchevsky, and U. Bellugi (Eds.) *Spatial Cognition– Brain bases and development*. pp.33–59. Hillsdale, NJ. Erlbaum, 1988.

[Ferguson, 1994] Ronald W. Ferguson. MAGI: Analogy-based encoding using regularity and symmetry. In *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society* pages 283-288. Atlanta, Georgia: Lawrence Erlbaum Associates, 1994.

[Ferguson, 1994] Ronald W. Ferguson. MAGI: Analogy-based encoding using symmetry and regularity. In Ashwin Ram & Kurt Eiselt (Eds.), *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society* (pp. 283-288). Atlanta, GA: Lawrence Erlbaum Associates (1994).

[Ferguson and Forbus, 1998] Ronald W. Ferguson, & Kenneth D. Forbus. Telling juxtapositions: Using repetition and alignable difference in diagram understanding. in Keith J. Holyoak and Dedre Gentner, & B. Kokinov (Eds.), *Advances in Analogy Research* (pp. 109-117). Sofia, Bulgaria: New Bulgarian University. (1998).

[Ferguson and Forbus, 2000] Ronald W. Ferguson, & Kenneth D. Forbus. GeoRep: A flexible tool for spatial representation of line drawings, *Proceedings of the 18th National Conference on Artificial Intelligence.* Austin, Texas: AAAI Press, 2000.

[Gentner and Stevens, 1983] Deidre Gentner and A. L. Stevens. *Mental Models* Lawrence Erlbaum, Hillsdale, NJ, pp 99–130, 1983.

[Griffith *et al.*, 1996] Todd W. Griffith, Nancy J. Nersessian and Ashok K. Goel. The role of generic models in conceptual change. In *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society*, pages 312–317 Lawrence Erlbaum, Mahwah, NJ, 1996.

[Griffith *et al.*, 2000] Todd W. Griffith, Nancy J. Nersessian and Ashok K. Goel. Function-follows-form transformations in scientific problem solving. In *Proceedings of the Twenty-Second Annual Conference of the Cognitive Science Society*. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2000.

[Holyoak and Thagard, 1989] Keith J. Holyoak, & Paul Thagard, P. A computational model of analogical problem solving. In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning*. Cambridge:Cambridge University Press. pages 242–266, 1989.

[Holyoak and Thagard, 1997] Keith J. Holyoak and Paul Thagard The analogical mind. *American Psychologist.* 52(1) 35–44 (1997).

[Johnson, 1990] Mark Johnson. *The Body and the Mind: The Bodily Basis of Meaning, Imagination, and Reason.* University of Chicago Press, Chicago, 1990.

[Kosslyn, 1994] Stephen M. Kosslyn. *Image and Brain: The Resolution of the Imagery Debate.* MIT Press, Cambridge, MA, 1994.

[Lakoff and Johnson, 1983] George Lakoff and Mark Johnson. *Metaphors We Live by.* University of Chicago Press, Chicago, 1983.

[Landaur, 1998] T. K. Landaur. Learning and Representing verbal meaning: The latent semantic analysis theory. *Current Directions is Psychological Science*, 7 (5) pp 161-164, 1998.

[Lenat and Guha, 1990] Douglas Lenat, & R. Guha, R. *Building Large Knowledge Based Systems: Representation and Inference in the CYC Project.* Addison-Wesley Publishing. Reading, MA, 1990.

[Mednick, 1962] M. T. Mednick. The associative basis of the creative process. *Psychological Review*. 3:220–232, 1962.

[McGraw and Hofstadter, 1993] G. McGraw and Douglas R. Hofstadter. Perception and creation of alphabetic style. In *Artificial Intelligence and Creativity: Papers from the 1993 Spring Symposium*, AAAI Technical Report SS-93-01, AAAI Press, 1993.

[Monaghan and Clement, 1999] Monaghan, J. M. & Clement, J. (1999). Use of computer simulation to develop mental simulations for understanding relative motion concepts. *International Journal of Science Education* 21(9), 921–944.

[Nersessian, 1992] Nancy J. Nersessian (1992). How do scientists think? Capturing the dynamics of conceptual change in science. In Giere, R. N. (ed.)*Cognitive Models of Science.* University of Minnesota Press. Minneapolis, MN, 1992.

[Nersessian, 2002] Nancy J. Nersessian. Maxwell and "the Method of Physical Analogy": Model-based reasoning, generic abstraction, and conceptual change. In: *Essays in the History and Philosophy of Science and Mathematics*, D. Malament, ed. 129–166. Lasalle, Il: Open Court, 2002.

[Qian and Gero, 1996] L. Quian and John Gero. Function-Behaviour-Structure Paths and Their Role in Analogy-Based Design. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing,* 10: 289-312, (1996).

[Schrager, 1990] Jeffery Schrager. Commonsense perception and the psychology of theory formation. In Schrager, J. & Langley, P. (Eds.) *Computational Models of Scientific Discovery and Theory Formation.* Morgan Kaufman, San Mateo, CA. 437–470, 1990.

[Thagard, 1992] Paul Thagard. *Conceptual Revolutions*. Princeton University Press, 1999.

[Thagard *et al.*, 1992] Paul Thagard, D. Gochfeld, and S. Hardy. Visual analogical mapping. In *Proceedings of the 14th Annual Conference of the Cognitive Science Society,* pages 522–527, Hillsdale, Erlbaum, 1992.

[Yarlett and Ramscar, 2000] D. Yarlett & M. Ramscar. Structure-mapping theory and lexico-semantic information. In Gleitman, L. R., and Joshi, A. K. (Eds.) *Proceedings of the Twenty-Second Annual Conference of the Cognitive Science Society.* Lawrence Erlbaum. Mahwah, NJ. pp. 571–576, 2000.

[Yaner and Goel, 2002] Patrick W. Yaner and Ashok K. Goel, Retrieving 2-D Line Drawings by Example. In *Diagrammatic Representation and Inference: Proc. Diagrams 2002*, volume 2317 of Lecture Notes in Artificial Intelligence, pp. 97-99, Springer-Verlag, 2002.