Using Perceptual Syntax to Enhance Semantic Content in Diagrams

Pourang Irani	Colin Ware	Maureen Tingley
Visual Interfaces Laboratory	*Data Visualization Research Lab*	*Department of Mathematics*
University of New Brunswick	*Center for Coastal and Ocean Mapping*	*University of New Brunswick*
Box 4400 Fredericton, NB	*University of New Hampshire*	*Box 4400 Fredericton, NB*
Canada E3B 2A3	*Durham, NH 03824*	*Canada E3B 2A3*
(506) 453-4566	*(603) 862-1138*	*(506) 458-7343*
pirani@nbnet.nb.ca	*colinw@cisunix.unh.edu*	*maureen@math.unb.ca*

Diagrams are essential in documenting large information systems. They capture, communicate and leverage knowledge indispensable for solving problems and are conceived to act as "cognitive externalizations".[1] A diagram provides a mapping from the problem domain to the visual representation by supporting cognitive processes that involve perceptual pattern finding and cognitive symbolic operations.[2] However not all mappings are equivalent, and to be effective a diagram's representation needs to be embedded with characteristics such that meaningful patterns can be easily perceived. Consequently a diagram's effectiveness depends, to some extent, on how well it is constructed as an input to our visual system.[3,4]

In our research, we focus on a class of diagrams commonly referred to as graphs or node-link diagrams. Nodes representing entities, objects or processes, and links or edges representing relationships between the nodes characterize them. Their most common form is that of outline circles or boxes denoting nodes, and lines of different types representing links between the nodes. Entity-relationship diagrams, software structure diagrams and data flow models are examples of node-link diagrams used to model the structure of processes, software, or data.

Currently the most widely used graphical language for modeling complex systems is the - Unified Modeling Language (UML). UML contains a suite of diagramming techniques that allow one to model various aspects of a software system,[5] a real-time application,[6] or an enterprise structure.[7] Its versatility in several application areas results from the rich semantics it seeks to model. For example class diagrams in UML model software structures and include methods for depicting inheritance and composition. When these semantics are used in the realm of enterprise modeling for example, UML can capture relationships between organizations or relationships between the corporation and its employees. However, although considerable attention has been given to making these UML notations general and complete, the actual choice of graphical notations appears to be somewhat arbitrary; only an expert in the field can easily read them.

In this paper we first discuss aspects of structured object recognition theory and show how this can be used to make 3D diagrams that are more easily analyzed and remembered. We present the results of a new set of studies suggesting how careful mapping of problem semantics to 3D diagram structures can make the meaning of the diagrams easier to “read” with minimal training.

Structured Object Perception

Theories of object perception can be roughly divided into two general approaches: image-based and structure-based perception. Image-based theories propose that human perception stores multiple views of an object in memory; thus specifying processes for matching the stored views to what is being perceived.[8] Structure-based theories place emphasis on the extraction of the 3D structural information of objects for recognition.[9] This extraction results in decomposing the image into perceptual primitives consisting of 3D solids such as cones, cylinders, and ellipsoids, along with information about how they are inter-connected. Each theory has strong evidence supporting its validity and we believe that it is plausible that both mechanisms are used in a hybrid manner at several layers of the recognition process. However, for our purposes the structure-based theories are more interesting because they suggest that if information structures can be mapped into structured objects then the structure will be automatically extracted as part of normal perception.

We have taken as our starting point the theories of structural object recognition. Marr and Nishihara proposed a model in which our visual system extracts information from the 2D contour or silhouette structure of the viewed object.[10] The silhouette is decomposed into regions of concavity that facilitate the extraction of sub-parts of the image. Transformations within our visual system enable us to translate the sub-parts of the silhouette into a set of 3D generalized cones. Figure 1 illustrates a crudely drawn animal that we nevertheless readily perceive as having distinct head, legs, torso, and tail parts. Marr and Nishihara also proposed a mechanism whereby the axes of the parts become cognitively connected to draw a structural skeleton.

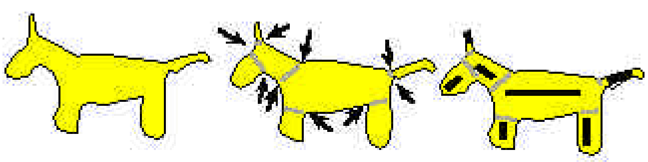

Figure 1. According to Marr and Nishihara[9], concave sections of the silhouette define sub-parts of the object. These points are critical in defining a structural skeleton. Adapted from Marr and Nishihara.[9]

Biederman elaborated Marr's initial theories in two significant ways.[11] He extended the set of generalized cones defined by Marr by describing them based on geometrical properties of the silhouette in the 2D plane including co-linearity, symmetry, parallelism, curvature and co-termination (the contours meet at a point, e.g. a cone). He devised a set of 36 primitives he termed "geons" (geometrical ions). A sample set of geons and objects constructed with them is depicted in Figure 2. A second significant contribution by Biederman is his description of the structural composition of the decomposed geons from the image. The decomposition of an object results in a geon structural description (GSD), consisting of geons, their attributes, and their relations with adjacent geons. The structural description contributes to object constancy, i.e. if two views of an object result in a similar GSD, then they are recognized as equivalent objects by our perceptual system.

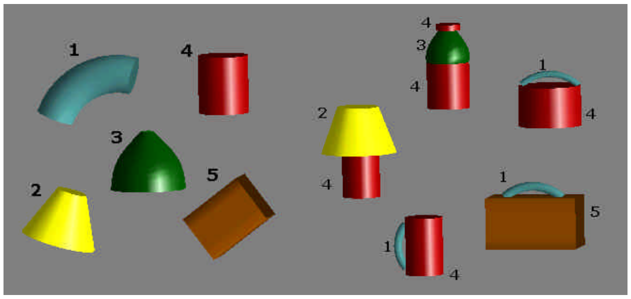

Figure 2. (a) Geons are object primitives in Biederman's theory. (b) When connected in a particular structural relationship they can define an object. (c) Different connections of the same geons can result in different objects as the figure shows geons 1 & 4 can give different objects.

The structural description is not purely topological. As Figure 3 illustrates, two objects, a human figure and a table, can have identical geons and an identical skeleton in terms of its topology, but still be identified as very different objects. In describing the representational capacity of his set of geons, Biederman suggests looking at the number of readily discriminable relations between any pair of geons. These relations are largely viewpoint independent, preserve their two-dimensional silhouette structure, and are categorical.[12] The following set of relational rules is based on the set proposed by Biederman. We have added an additional containment rule.

RR1: Color and texture are surface properties of geons that play a secondary role in perceptual object classification. These properties may aid in the recognition process, but do not play a major role in entry level classification.
RR2: Verticality - Geon A can be ABOVE, BELOW or BESIDE geon B.
RR3: Centering - Objects can be connected on or off-center. For example, human legs are connected to the right and left of the bottom of the torso. Human arms are at the top of the torso.
RR4: Connection relative to elongation - Most geons are elongated, and connecting to the long face versus the short face has important perceptual semantics. Humans and four-legged animals are differentiated in this way.
RR5: Relative size - One geon is larger or smaller than another.
RR6: Containment - An important perceptual task is identifying objects enclosed within larger components. This relationship is inherently hierarchical. However, strict containment can only be displayed using transparency.

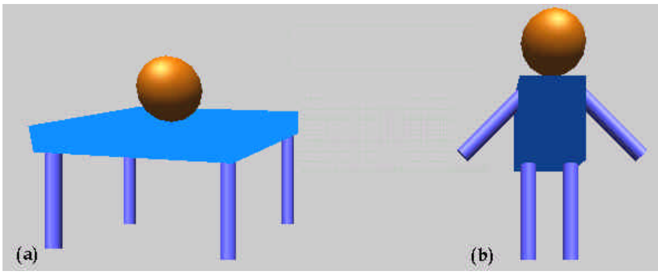

Figure 3. Same Geons and topological arrangement result in different objects. The Geon Structural Description (GSD) is important for identification to take place.

Geon Diagrams

If, we indeed identify structured objects through the mechanisms described above then we should be able to apply this theory to making diagrams that are more effective. We call such diagrams "geon diagrams" because they are loosely based on Biederman's geon theory. To evaluate this concept we have carried a series of studies that compare diagrams constructed with 3D geon primitives to various types of 2D diagrams.[13] The results have been remarkable. Subjects in our experiments identified substructures 40% faster and about twice as accurately using the geon diagrams compared to corresponding UML diagram. They also could recognize geon diagrams they had seen briefly with far greater accuracy (18% vs. 39% error rate). As an additional test we repeated the experiments without surface attributes (color and texture) for the geon diagrams and without the corresponding labels on the UML diagrams. The result was the same, we found that subjects made half as many errors recognizing the geon diagrams compared to their equivalent 2D UML diagrams. All of these findings were highly significant statistically.

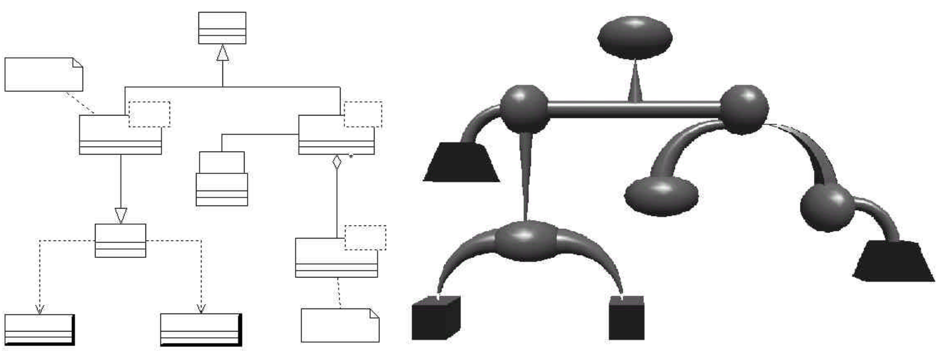

Figure 4. UML and Geon equivalent of a structured diagram. The relationships between nodes did not represent any actual system.

However because there are many differences between UML and geon diagrams other than the use of 3D shape primitives we conducted a further set of studies comparing shaded diagrams with the same diagrams as flat outline shapes as shown in Figure 5. These results were also highly significant. The use of 3D shaded primitives resulted in much more accurate sub-structure identification (11.4 vs. 21% errors) and shorter times

(4.1 seconds vs. 5.2 seconds). The subjects also accurately recognized more 3D diagrams than 2D-silhouette diagrams (20% vs. 34% errors).[14]

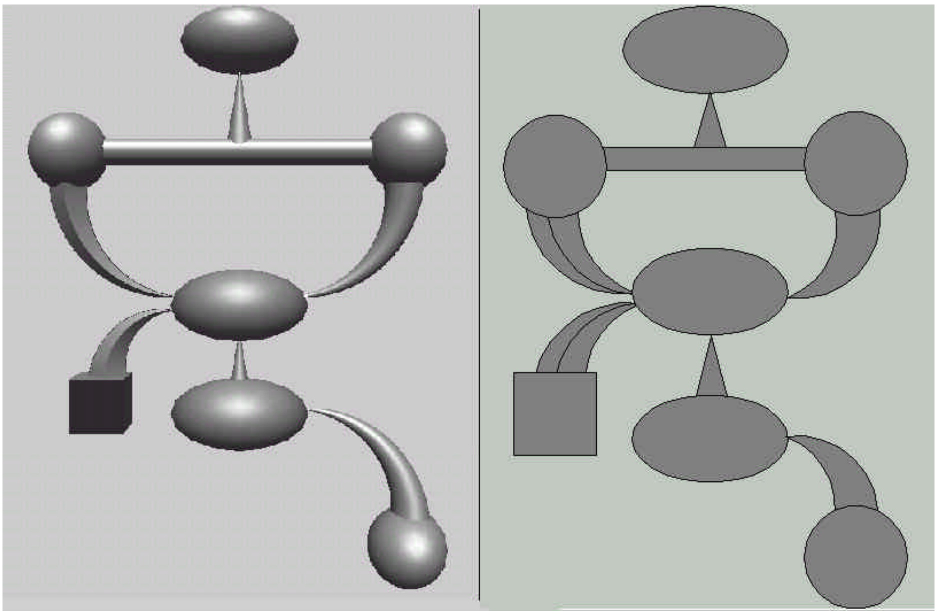

Figure 5. Geon and equivalent 2D-silhouette diagram.

The results of our experiments strongly suggest that visual parsing and recognition of diagram structures are facilitated by the use of 3D shaded primitives compared to box and line diagrams (such as UML) and 2D silhouette equivalents. Thus using 3D shaded components for diagram elements can facilitate interpretation and recognition of diagrams.

Perceptual Semantics

Having shown that geon diagrams are easy to remember and to visually parse, we now turn our attention to the issue of perceptual semantics. The work of Biederman and others suggests that certain spatial relationships, such as "on-top-of" or "unbalanced" may have a kind of immediately understandable perceptual meaning. We reasoned that if we could map the semantics of systems modeling into this perceptual semantics we could make diagrams that are easier to read. The remainder of this paper is devoted to reporting this new line of research.

UML class diagrams present a view of software structure; in particular, they depict the objects that exist, their internal structure, and their relationships with one another. They do not show temporal or causal information.[5] The notation has been derived from three main sources Booch, Rambaugh and OMT, through the efforts of the Object Management Group (OMG) to standardize software modeling semantics and notations. UML offers a rich semantic base which we use for deriving our visual representations. We are not augmenting UML notation by suggesting the use of different shaped boxes or different types of edges. Instead, we are exploring the use of certain visual constructs to enhance the understanding and intuitiveness of semantics such as those available through UML. We hope that our findings can be generalized to other diagramming applications.

We carried out our investigation in a three-stage process. In the first stage we constructed a number of different visual representations for each of the following modeling concepts: 1) Generalization (A is-a B), 2) Dependency (A depends-on B), 3) Strength of relationship (some relationships are stronger or weaker than others), 4) Multiplicity of relationship (e.g. one-to-many), and 5) Aggregation (A has-a B). In each case a perceptual principle was used to construct at least one of the instances. The other members of the set were made up of what we though to be reasonable alternatives. In the second stage we conducted a multi-part evaluation study whose goal was to find out if subjects agreed on which mappings were the best. This experiment was made of five parts each evaluating the representation that best fit the five semantics enumerated above. The experiment was conducted on 40 volunteer students, 20 of whom were familiar with software diagramming notation, in particular UML (experts). The remaining twenty students had not been exposed to any form of diagram modeling and had not been trained in understanding software semantics (novices). The interface to the experiment was a web browser; HTML and Javascript were used to resize and randomize the appearances of the images. In the third stage we created diagrams using the best mappings and evaluated how well they were perceptually understood.

For clarity we have separated each of the modeling concept mapping sub-experiments, together with the evaluation results, in the following sections.

Generalization

In abstract terms, generalization is the task of grouping concepts that fit a given pattern under a common header by moving from the more particular to the more general. This capability of the human mind has led to the concretization and refinement of ideas over the centuries. In software modeling the semantic of generalization is used to classify objects based on their common functionality. The UML notation guide defines generalization as being the relationship between a more general element and a more specific element that adds additional information to it.[15] It is commonly also referred to as inheritance (the specific object inherits properties of the general object) and is casually referred to as an "is-a" relationship. Thus objects can belong to a common class; for example, rotweilers, boxers and retrievers are all dogs. The perceptual principle here is based on Biederman’s claim that the primitive shapes play a primary role in object classification, whereas surface properties such as color and texture, play a secondary role (see RR1 defined in our introduction).[11]

Representing Generalization

The purpose of this part of the experiment was to determine whether shape had a stronger influence than color in classifying objects of the same kind. If shape were to be a better cue, then we could suggest using same shaped primitives to denote objects of the same kind. This is radically different than what is available in UML in which objects "of-the-same-kind" are connected by a solid line arrow with a closed arrowhead pointing to the more general class.

We considered two cases, one which contained objects made of a single geon primitive (one-component case) and the other which consisted of images containing two primitive shapes (two-component case). The procedure used for each case was similar and so they only differed in their representations.

Representations - One-Component case. We constructed three sets of images. Each set contained eight shaped primitives labeled A to H (Figure 6). The types of primitives were different from one set to another. Each set contained two primitives with the same color but different shapes, and two others that had the same shape but different surface color. For example in the set depicted in Figure 6, A and G were the same color (red) while E and C were the same shape (barrels). All the other shapes in the set had either a different color or different shape than the two pairs described above.

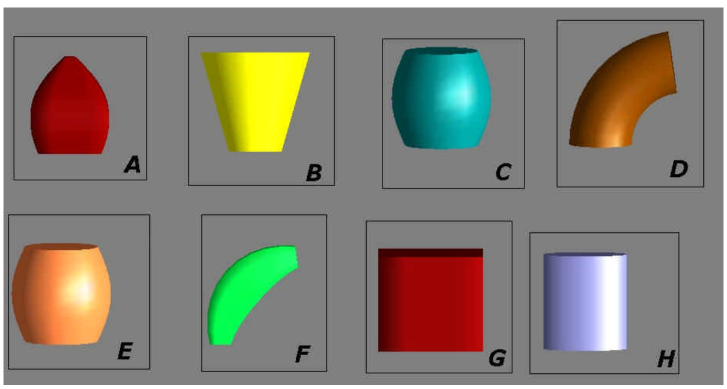

Figure 6. Which two objects are of the same kind? Sample set of representations containing only one-component. Contains a matching color pair (A&G) and a matching pair of shapes (C&E).

Representations - Two-Component case. In this case three sets of pictures were created each containing six images. Each image was comprised of two geon primitives (Figure 7) labeled A to F. For the images in all the sets we used one major (bigger) and one minor (smaller) geon primitive to construct the image. We created each set with two images containing major and minor components of the same color and different shapes (color-pair B and D) and two others whose minor and major components were made of the same shaped geons with different colors (shape-pair C and E). All the other images in the set contained primitives whose major and minor components were not colored as our selected color-pair and did not have the same arrangement of primitives as our shape-pair.

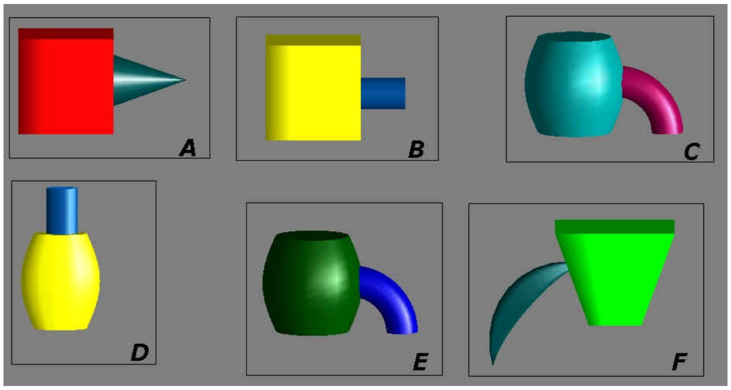

Figure 7. Which two objects are of the same kind? Sample set of two components. Contains a pair with matching major and minor shaped components (C&E) and matching color coding on major and minor components (B&D).

Evaluating Generalization

Subjects were shown the three sets of images for the one-component case and the three sets of images for the two component-case. They were asked to *select using your best judgment the pair of images that are of the same kind* which they recorded on the experiment handout sheet. For each of the two cases, the three sets were shown in different order for all the subjects.

Results and Discussion. We did not observe any statistically significant differences between responses from novice and expert subjects (test for null hypothesis of agreement between novices and experts yielded P-values of 0.403 for the one-component-case and 0.3091 for the two-component case). Therefore the results were combined. Overall, 92% of the responses favored same shaped objects, whereas 8% of the responses favored the same color. This suggests that shape is a better stimulus for identifying objects of the same kind than is color.

With the defined set of 36 geon primitives[11] a large system is limited in the number of different distinct objects that can be represented. By using shape to identify objects of the same kind, a diagram could not be formed by more than 36 distinct objects. This limitation might be resolved by creating objects with pairs of primitives as was done for the two-component case of this part of the experiment. However, designing compound objects that express single-object identity might be challenging.

The use of same shaped objects for generalization will also be problematic when a system is modeled using multiple inheritance. Two objects of different shapes generalize into a third object whose representation may or may not be a combination of the earlier two. On the other hand, multiple inheritance is regarded by some as an improper structure in software modeling.[16] This has led to the exclusion of this feature in Java, for example.

Dependency

In common parlance dependency means a relationship in which an entity is supported by another. Children financially depend on their parents, humans depend on breathing clean air, and engines depend on gasoline. Similarly, in software modeling terms, dependency describes a relationship in which changes to one component can cause changes to the state of the dependent component. Therefore the dependent module is unstable when changes are made to the entity it depends upon. A goal in proper software modeling is to reduce the number of dependencies.[5]

Certain spatial representations are easily visually recognized.[17] These visual properties include such relations as "longer or shorter", "thinner or thicker", and "above or below". Biederman suggests that the spatial property of verticality (see RR2) makes up for more than 80% of arrangements between visual components during object perception.[11] Glasgow and Papadias use such spatial properties to construct models that lower the computational costs for AI systems to retrieve and understand the representations of visual images.[18] In her study Petre concluded that "secondary notation" such as relative positioning of nodes (e.g. placing two linked or unlinked nodes near each other) plays an important role in conveying meaning.[19]

Representing Dependency

The purpose of this part of the experiment was to derive a representation that was best suited for understanding the semantics of dependency. If our perceptual system has deeply engrained within it certain spatial properties to describe and classify objects then perhaps we could use such properties to depict dependency.

Representations. We constructed five different ways of representing the dependency relationship as illustrated in Figure 8. To show that the red cylinder depends-on the blue barrel, these representations consisted of a broken tube (A), a connected tube (B), the cylinder on-top-of the barrel (C), disconnected objects (D), and the cylinder on-the-bottom-of the barrel (E). Representation A most closely resembles the representation of dependency used in UML which consists of a dashed line with an open arrowhead going from the dependent to the depended.[5]

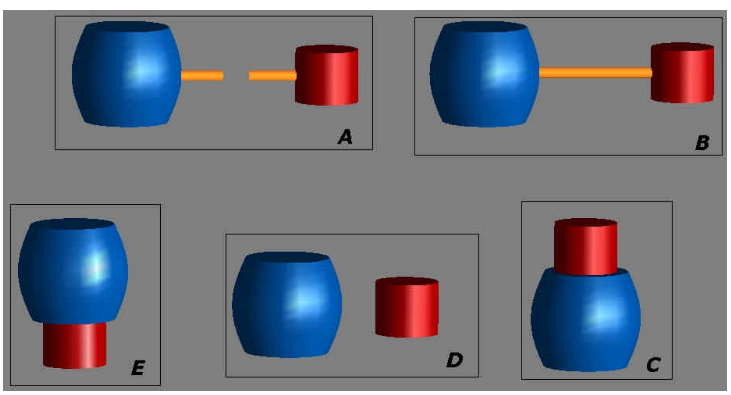

Figure 8. Which representation best denotes the red cylinder depends-on the blue barrel? A sample set containing representations that were ranked for showing dependency.

Evaluating Dependency

Subjects were asked to *rank from 1-5 (best-to-worst) the representation denoting that one object depends on another*. In the case of the representations illustrated in Figure 8, the relationship was that of the red cylinder depending on the blue barrel. They recorded their rankings for all three sets of representations. Each subject was shown the representations in a different random order.

Results and Discussions. A χ^2 test on the results shows that there were no statistically significant differences in selecting the best representations between novices and experts (P-value of 0.49 for null hypothesis of agreement between novices and experts). Therefore the results were combined and in Chart 1 the average rankings of all 40 subjects is shown. A top-down test of correlation on the average rankings shows a strong agreement between all 40 subjects for the best-ranked representations (P-value < 0.0001 for null hypothesis of no correlation between rankings chosen by 40 subjects). As seen in the chart below, dependency is best depicted using the on-top-of representation (C). This representation is significantly better than the second best representation of a connected tube (B) (with 95% confidence, the probability that any subject, novice or expert, would choose C over A is between 0.58 and 0.86).

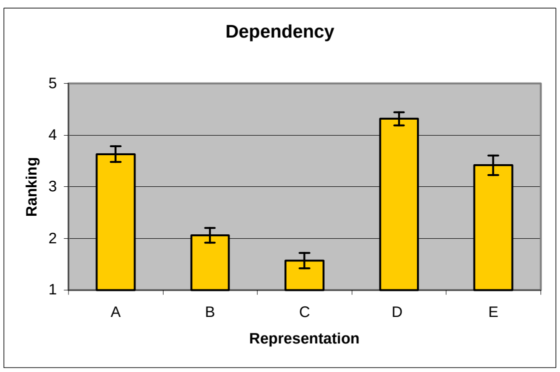

Chart 1. Average rankings for dependency. 1 - best, 5 - worst.

Representing dependency using the spatial property on-top-of is not commonly seen in software structure diagrams but can be found in other types of visual representations. Organizational charts, which stack different parts of the corporation on a pyramid, make use of such a spatial organization between the represented entities. In such diagrams there is an implicit assumption of the dependency between objects on-top-of and on-the-bottom-of the pyramid. A drawback of such a representation is the amount of space required to show several entities on-top-of any given object.

Relationship Strength

A common semantic found in structured diagrams is strong and weak relationships. This semantic is not directly modeled using UML but is common in other types of structured diagrams such as Entity Relationship diagrams.[20] In UML class diagrams, a strong aggregation is referred to as a composition and as such can denote a strong relationship between two entities. Biederman suggests that during object recognition our perceptual system differentiates parts of an object based on their relative sizes (see RR5).

Representing Relationship Strength

The purpose of this sub-experiment was to examine whether the perceptual mechanism of discriminating based on relative ratios can be applied toward the semantic of relationship strength.

Representations. We created six different ways of representing the strength of a relationship as illustrated in Figure 9. These representations consisted of a thick connected tube (A), adjacent entities (B), a long connected tube (C), a conic connection (D), disconnected entities (E), and a short connected tube (F).

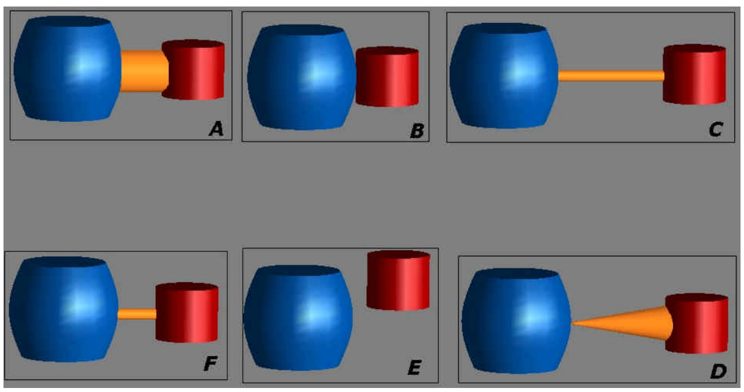

Figure 9. Which image shows best that the red cylinder and blue barrel are strongly related? One of the three sets being ranked for showing relationship strength.

Evaluating Relationship Strength

Subjects were asked to *rank from 1-6 (best-to-worst) the representation denoting that one object is strongly related to another*. For the example in Figure 9, subjects were asked to rank the representations according to how effectively each denoted a strong relationship between the blue barrel and red cylinder. For each subject, the representations appeared in a different random order.

Results and Discussion. A χ^2 test on the results showed that novices and experts were in perfect agreement on their choice of rankings (test of null hypothesis of agreement yields P-value of 1.0). The results were thus combined and the average rankings for 40 subjects are summarized in Chart 2. A top-down test of correlation shows that all 40 subjects were in strong agreement in selecting the top best rankings (P-value < 0.0001 for null hypothesis of no correlation between rankings chosen by 40 subjects) and as depicted in Chart 2 the best representation for showing that two entities are strongly related is a thick connection between the objects (A). This representation is significantly better than its alternative representation (B) (with 95% confidence, the probability that any subject, novice or expert, chooses A over B is greater than 0.99). These results strongly suggest that using a relatively larger size for a connection can depict strength of relationship.

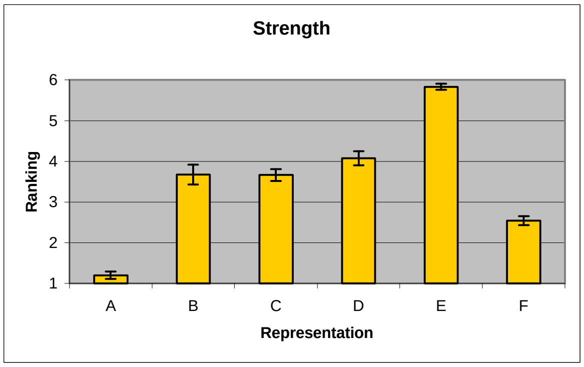

Chart 2. Average rankings for the six representations for relationship strength. 1 - best, 6 - worst.

Multiplicity (or cardinality)

Numbers are a relatively recent invention in human evolution. In many instances skills involving counting or knowing the exact quantity is not essential in providing a stable image. For example, a shepherd need not count to know whether his group is complete.[21] When children are asked to copy a figure made with counters, they do not use the exact number of counters (even if they know to count) but do justice to the shape of the figure.[22]

A common method for representing this semantic attribute is the use of an asterix (*) or an exact numeral (such as 1, 2, etc.) over the link and on the side of the entity which is associated in multiples. Other common notations are 1..*, 0..* or *. However, numbers are learnt symbols and not perceptually immediate.

Representing Multiplicity

The purpose of this part of the experiment was to derive a representation that would represent that an entity is associated with multiple instances of another.

Representations. We constructed five different ways of representing the multiplicity attribute of a relationship as illustrated in Figure 10. These representations consisted of a conic connection with multiple glyphs (A), disjoint objects (B), multiple connecting tubes (C), single connecting tube (D), simple conic connection (E). In this illustration, the representations were created to depict the blue barrel being associated to multiple instances of the red horn.

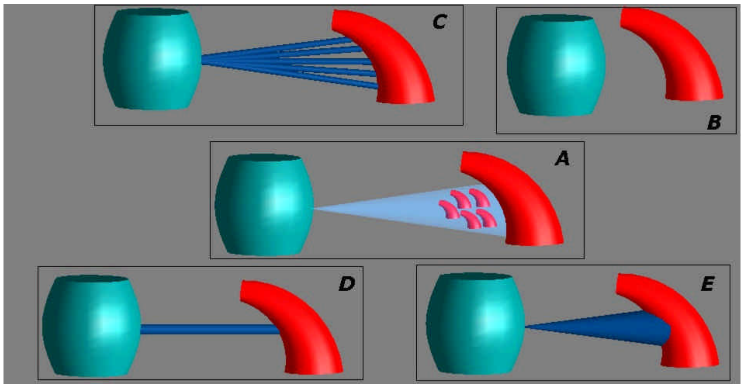

Figure 10. Which representation best depicts that the blue barrel is associated to multiple instances of the red horn? Sample set used in ranking representations of multiplicity.

Evaluating Multiplicity

Subjects were asked to *rank from 1-5 (best-to-worst) the representation denoting that one object is associated to multiple copies of another object*. They were presented with three sets of images. For the set shown in Figure 10 subjects were asked to rank the representation that best denotes the blue barrel is associated to multiple copies of the red horn. For each subject the order of the representations was random.

Results and Discussion. A χ^2 test on the results shows that both groups of subjects strongly agree on rankings (test of null hypothesis of agreement yields P-value of 0.18). We therefore average the results, which are summarized in Chart 3. A top-down test of correlation on the average rankings shows strong agreement between all 40 on subjects on their rankings (P-value < 0.0001 for null hypothesis of no correlation between rankings chosen by 40 subjects). As seen in Chart 3, multiplicity is best represented using multiple connecting tubes (C). This representation is significantly better than its alternative (A) (with 95% confidence, the probability that any subject, novice or expert, chooses A over B is between 0.52 and 0.82).

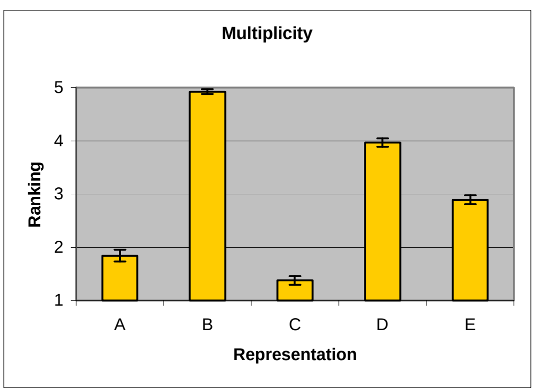

Chart 3. Average rankings for the six representations for relationship strength. 1 - best, 5 - worst.

There are obvious problems with this representation of multiplicity. People are likely to interpret multiple representations in an overly literal sense. For example, if three branches are shown, exactly three instances will be inferred. On the other hand there is a theory that humans and animals have a perceptual sense of numbers, but it is strictly limited. We may be only able to naturally separate 1, 2, and possibly 3 objects. If there are more objects they are simply perceived as a lot.[23]

Aggregation

In UML methodology, aggregation describes a special form of association in which an object contains another. In software engineering terms, aggregation is also referred to as a "has-a" relationship. For example, an organization has-a president. A strong aggregation is referred to as a composition. There is evidence suggesting that our perceptual system is capable of separating an object when it is seen as being contained within other objects.

Representing Aggregation

The purpose of this part of the experiment was to establish a representation suitable to denote an aggregation, part-of, or containment relation.

Representations. We constructed five different ways of representing aggregation as illustrated in Figure 11. These representations consisted of a connected tube (A), an object-on-top-of (B), disjoint objects (C), connection with containment (D), and containment (E).

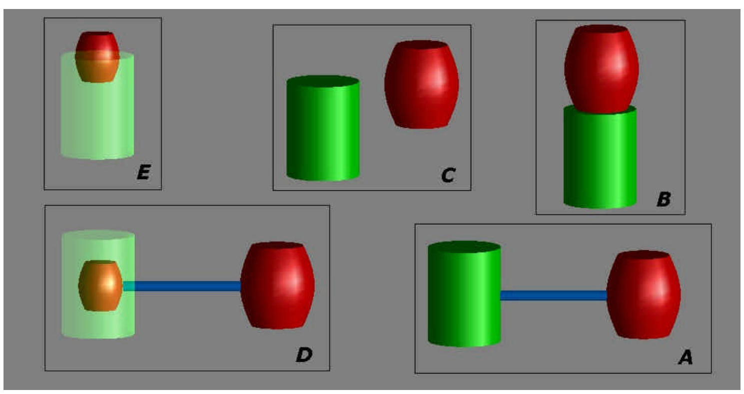

Figure 11. Which image best depicts that the red barrel is contained within the green cylinder? Sample set used in ranking aggregation.

Evaluating Aggregation

The subjects were asked to *rank from 1-5 (best-to-worst) the representation denoting that one object is contained within another*. In the case of the representations given in the set in Figure 11, subjects were asked to rank the representation that best denoted that the red barrel is contained within the green cylinder.

Results and Discussion. A χ^2 test on the results shows that both groups of subjects strongly agree on rankings (test of null hypothesis of agreement yields P-value of 0.74). This result allows us to average the rankings of all 40 subjects as summarized in Chart 4. A top-down test of correlation on the average rankings shows that subjects are consistent in their ranking (P-value < 0.0001 for null hypothesis of no correlation between rankings chosen by 40 subjects). The results show that the best depiction for aggregation is a connection with containment (D). This representation is not significantly better than its alternative (E) (with 95% confidence, the probability that any subject, novice or expert, chooses A over B is between 0.47 and 0.76).

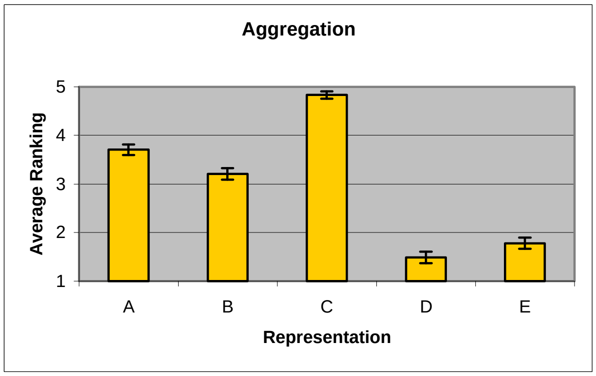

Chart 4. Average rankings for the five representations for aggregation. 1 - best, 5 - worst.

The Perceptual Syntax

Figure 12 summarizes the set of best representations derived from the experiment. This can be regarded as the basis for a kind of natural visual semantics for drawing diagrams, which will be easily interpreted. In the following section we present the full set of rules defining geon diagrams.

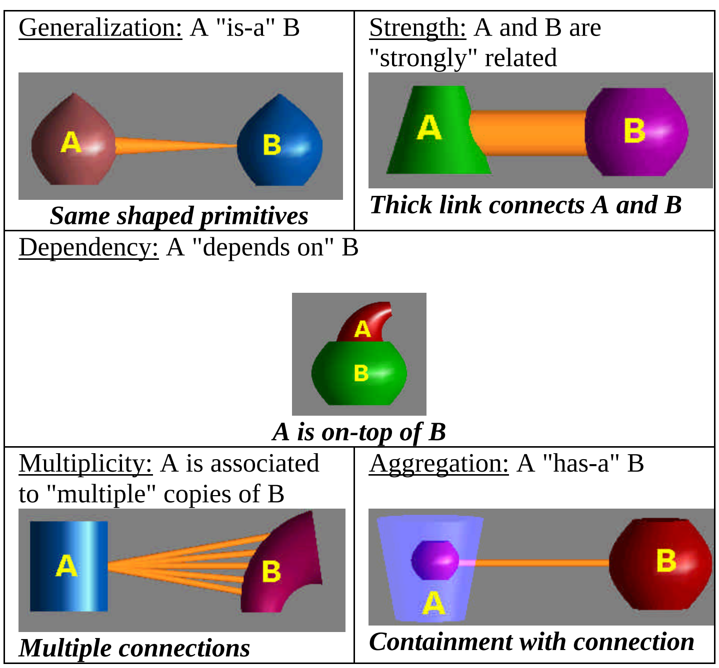

Figure 12. Perceptual notation for showing each of the five semantics investigated in our experiment: Generalization, Relationship Strength, Dependency, Multiplicity, and Aggregation.

Applying Theories of Perception to Drawing Diagrams

Together with results from our previous studies, the results described above help us define rules for what we call the "geon diagram". We have used Biederman's term "geon" as it nicely describes the idea of a 3D shape although we do not necessarily endorse the particular set of 3D shape primitives in Biederman's theory. We define five rules relating to the use of geons as 3D primitives, three rules that relate to the layout of the geon structure and nine additional rules for portraying certain semantics.

G1: Major entities of a system should be presented using simple 3D shape primitives (geons).
G2: The links between entities can be represented by the connections between geons. Thus the geon structural skeleton represents the data structure.
G3: Minor sub-components are represented as geon appendices - small geon components attached to larger geons.
G4: Geons should be shaded to make their 3D shape clearly visible.
G5: Secondary attributes of entities and relationships are represented by geon color and texture and by symbols mapped onto the surfaces of geons.

Although geons are 3D shape primitives, theories of shape extraction rely heavily on a clear silhouette. For this reason a good 2D layout will also be important in determining how easily a geon structural description can be identified. Thus we add the following layout rules:

L1: All geons should be visible from the chosen viewpoint.
L2: The geon diagram should be laid out predominantly in the plane orthogonal to the view direction.
L3: Junctions between geons should be made clearly visible.

To provide rules for constructing effective diagrams we need to extend the syntax of structural description to include mapping of semantics to data elements. The results of our experiments suggest that certain types of "naturally" occurring semantic rules can be used:

SM1: Similarity, Generality - Geons with same structural geometrical composition (or shape) can be used to denote objects of the same kind.
SM2: Gravity - If geon A is on top of geon B this suggests that geon A is supported by geon B. In addition gravity determines that structures are perceived as either being stable or unstable.
SM3: Enclosure - shows that Geon A is contained within Geon B. Syntactically this can be shown as an internal component attached to the same primitive geon on the outside.
SM4: Ordinality - to show multiple associations between two entities a series of attachments can best denote such a relationship.
SM5: Strength of connection - Using a thicker connection as opposed to a thinner one can denote a stronger relationship between two entities.
SM6: Sequence - Geons arranged in a line become a metaphor for a chain of operations or some other linear structure.

SM7: Symmetry - Some information structures have symmetry and a symmetrical arrangement of geons should be used to show this.
SM8: Central, Peripheral - If a component is of central importance to a structure this can be illustrated through its position and by the location of the interconnections.
SM9: Size - Larger components can be mapped to an understanding of superiority in areas such as finances, economics, and politics.

Validating the Perceptual Syntax

In order to evaluate the geon diagram syntax and semantics we created geon diagrams to model real world examples. For example, Figure 13 illustrates one of the diagrams modeling an academic conference whose components are attendees, speakers, A/V equipment, etc. Using this diagram and others like it, we conducted an experiment to evaluate whether the notations of the geon diagrams were intuitive and easily describable. A group of 35 students (different from the 40 subject used earlier) who were unfamiliar with software modeling semantics found in UML participated in the experiment.

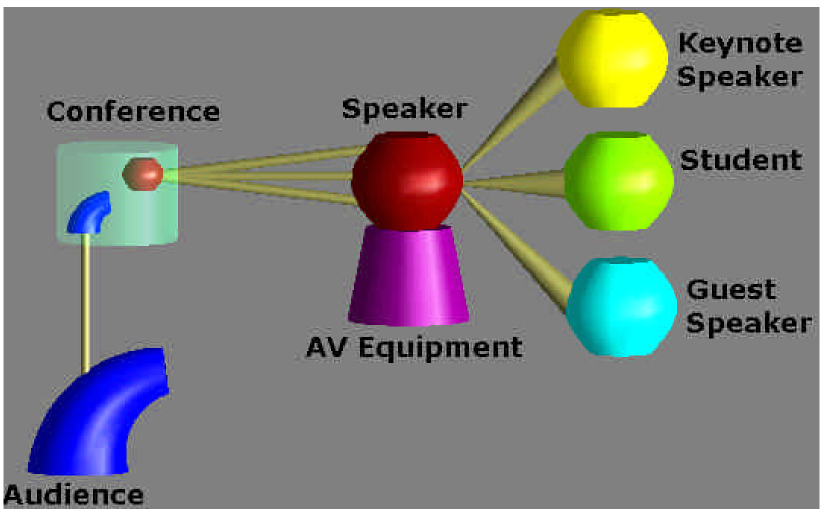

Figure 13. Geon representation for communicating structural information between representable entities in a conference.

Validation Method. A set of 3 UML diagrams and equivalent geon diagrams were created for the experiment. The entities in the diagrams were arbitrarily labeled as we were primarily validating the syntax and did not want to influence the choice of semantic based on the labeling (Figure 14 and Figure 15). In a classroom setting, the subjects were presented with the UML diagrams and equivalent set of geon diagrams. They had no previous experience with either type of diagramming convention. They were asked to describe the relationships between several pairs of entities in the diagram using multiple choice answers. The choices included:

- is of the same kind as
- depends on
- has many
- has one

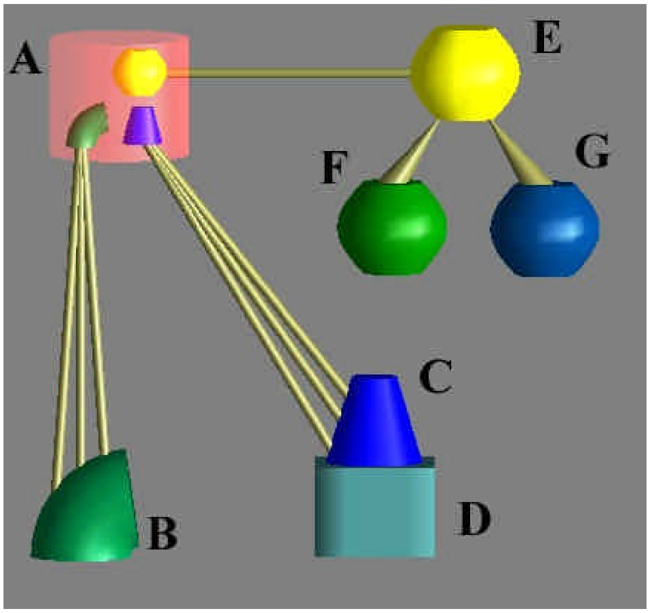

Figure 14. Geon diagram representation used in validating the syntax.

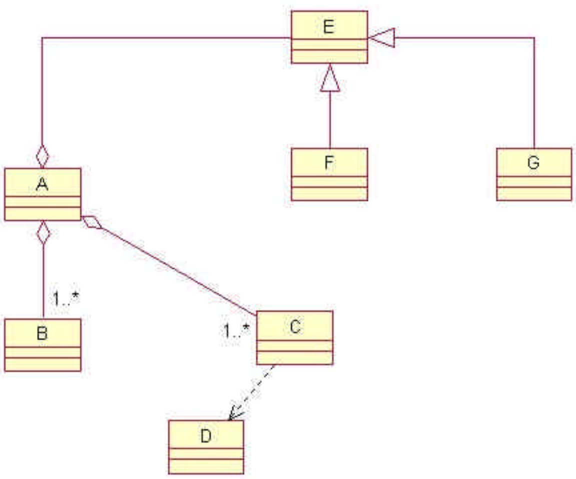

Figure 15. UML representation equivalent to the geon representation of figure 14.

Results. The results are summarized in the Table 1. They show that there were almost five times as many errors deciphering relationships between entities using UML notation than using the perceptual syntax.

	Geon	UML
Error Rate	11.5 %	53.6 %

Table 1. Comparison of error rates in identiyfing relationships in geon diagrams vs. UML diagrams.

Conclusion

Perception researchers have theorized that complex structured objects are automatically parsed by the visual system into component parts, together with a structural skeleton. We have shown that applying this theory leads to a perceptual syntax can be applied to constructing diagrams that are easier to interpret and remember. We have developed a set of rules defining what we call geon diagrams, based on perceptual theory and evaluated this with a series of experiments. The results suggest that mapping information structures into connected structures built with 3D geon primitives will make the information easy to read.

The major contribution of the present paper has been to add semantics to the geon diagram design rules. We have developed ways of representing the following abstract relationships between entities: generalization, dependency, relationship strength, multiplicity and aggregation. Our experimental results suggest that the best representations can be understood intuitively without training. This leads to a kind of natural semantics for diagrams. In a comparison with UML diagrams, we found that using our semantic rules greatly facilitated the task of identifying relationships between elements. However, we recognize that our semantic mappings are far from complete. For example a study of the perceived meaning of different kinds of linking objects: broken tubes, dashed tubes, cone-shaped tubes, transparent tubes might further enrich our graphical vocabulary. Much remains to be done.

There are tradeoffs inherent in creating geon diagrams. Due to the amount of real-estate they can consume, the complexity of what can be represented using these kinds of primitives may be less than what is possible using more cryptic line and box diagramming techniques. There is also a possible tradeoff between the very concrete nature of geon diagrams and the representation of certain abstractions. If, for example, size is used to represent magnitude, then it becomes difficult to represent objects or concepts that have arbitrary magnitude. Labeling is an important element of diagrams and is another issue that must be addressed when using geon primitives. It may be difficult to show text as clearly wrapped on a 3D-shaded object. If the object is textured this is especially likely to interfere with the readability unless the texture is subtle.

One of the results from the evaluation stage was that there were no differences in interpretation of the geon-based notation between UML novices and experts presumably because the task did not directly involve modeling. This suggests that our new notation can possibly help experts and non-experts interact. In many projects, interaction between the client, manager, and programmer is essential for the proper development of the system, so a diagramming system that is more easily accessible to non-experts could be useful. We are not, of course advocating the abandonment of UML notations. They are the most highly evolved graphical modeling tools we have. However, we hope that for new applications we may start to see the development of diagrams that take advantage of diagrams made with 3D shape primitives.

Acknowledgment

We gratefully acknowledge the support of an NSERC PGS-B award to the first author and an NSERC research grant to the second author. Discussions with Jane Fritz have helped focus some of the ideas. We also thank Richard Spacek who helped with the manuscript and the volunteers who participated in our experiments.

References

1. J. Zhang, "The nature of external representation in problem solving," *Cognitive Science*, Vol. 21, No. 2, 1997, pp. 179-217.
2. J.H. Larkin, and H.A. Simon, “Why a diagram is (sometimes) worth ten thousand words.” *Cognitive Science*, Vol. 11, 1987, pp. 65-99.
3. S.M. Kosslyn, *Elements of Graph Design*, W.H. Freeman and Company, NY, 1994.
4. C. Ware, *Information Visualization: Perception for Design*, Morgan Kaufman, 2000.
5. M. Fowler, *UML Distilled: Applying the Standard Object Modeling Language*, Addison-Wesley, 1996.
6. B.P. Douglass, *Real-Time UML*, Addison-Wesley, 1998.
7. C. Marshall, *Enterprise Modeling with UML: Designing Successful Software through Business Analysis*, Addision-Wesley, 1999.
8. S. Ullman, "Aligning pictorial descriptions: An approach to object recognition," *Cognition*, Vol. 32, 1989, pp. 193-254.
9. D. Marr, *Vision: A computational investigation into the human representation and processing of visual information*, Freeman, San Fransisco, CA, 1982.
10. D. Marr and H.K. Nishihara, "Representation and recognition of the spatial organization of three-dimensional shapes". Proceedings of the Royal Society of London, 1978, B, 200, pp. 269-294.
11. I. Biederman, "Recognition-by-Components: A Theory of Human Image Understanding," *Psychological Review*, Vol. 94, No. 2, 1987, pp. 115-147.
12. I. Biederman, and P.C. Gerhardstein, “Recognizing depth-rotated objects: Evidence and conditions for three-dimensional viewpoint invariance”. *Journal of Experimental Psychology: Human Perception and Performance*, Vol. 19, 1993, pp. 1162-1182.
13. P. Irani and C. Ware, "Diagrams Based on Structured Object Perception," *Proc. Advanced Visual Interfaces AVI’2000*, Palermo, Italy, May 2000, pp. 61-67.
14. P. Irani and C. Ware, "Should the Elements of Diagrams Be Rendered in 3D?," *Late Breaking Hot Topics, IEEE Information Visualization 2000 CD-ROM Proc*., Salt Lake City, Utah, Oct. 2000.
15. UML Notation Guide, Version 1.1, Rational Software Corporation, 1997.
16. C.S. Horstmann, *Computing Concepts with Java 2 Essentials*, John Wiley & Sons, 1999.
17. J.I. Glasgow, N. Hari Narayanan, and B. Chandrasekaran, "Diagrammatic Reasoning: Cognitive and Computational Perspectives", MIT Press, 1995.

18. J.I. Glasgow and D. Papadias, "Computational Imagery", *Cognitive Science*, Vol. 17, No. 3, 1992, pp. 355-394.
19. M. Petre, "Why looking isn't always seeing: readership skills and graphical programming," *Communications of the ACM*, Vol. 38, No. 6, June 1995, pp. 33-44.
20. P.P. Chen, "The Entity-Relationship Model -- Toward a Unified View of Data," *ACM Transactions on Database Systems*, Vol. 1, No. 1, 1976, pp. 9-36.
21. R. Arnheim, *Visual Thinking*, Berkeley, LA, University of California Press, 1969.
22. J. Piaget, *The Child's Conception of Number*, W. W. Norton & Company, Inc., NY, 1952.
23. D. Stanislas, *The Number Sense: How the Mind Creates Mathematics*, Oxford University Press, NY, 1998.