AN APPROACH TO RECOGNITION IN 2D IMAGES OF 3D OBJECTS FROM LARGE MODEL BASES

J. Brian Burns Leslie J. Kitchen

COINS Technical Report 87-85

August 20, 1987

Abstract

An object recognition system is presented that is designed to handle the computational complexity posed by a large model base, an unconstrained viewpoint and the structural complexity and detail inherent in the projection of an object. The design is based on two ideas. The first is to compute descriptions of what the objects should look like in the image, called *predictions*, before the recognition task begins. This reduces actual recognition to a 2D matching process, speeding up recognition time for 3D objects. The second is to represent all the predictions by a single, combined IS-A and PART-OF hierarchy called a *prediction hierarchy*. The nodes in this hierarchy are partial descriptions that are common to views and hence constitute shared processing subgoals during matching. The recognition time and storage demands of large model bases and complex models are substantially reduced by subgoal sharing: projections with similarities explicitly share the recognition and representation of their common aspects. The original contribution of this paper is the automatic compilation, from a 3D model base, of a prediction hierarchy that can be used to recognize objects. A prototype system based on these ideas is demonstrated using a set of polyhedral objects and projections from an unconstrained range of viewpoints.

This research was supported in part by the by the Air Force Office of Scientific Research under contract number AFOSR-86-0021. Thanks to Al Hanson, Ed Riseman, Michael Boldt and John Dolan for comments, support and help with the figures.

Contents

1. Introduction

Object recognition is a central aspect of understanding visual information, helping us to relate what we are seeing to what we have experienced in the past.

The general framework being adopted here is object recognition as a matching problem, specifically between models of 3D objects and 2D images. This is a clear, powerful framework within which a wide range of image interpretation problems can be studied [Haralick78, Kitchen80, Brooks81, Ballard82, Besl85]. The basic idea is to explicitly represent long-term knowledge of the physical world in the form of models and expected model-to-data transformations. The models are collections of primitives and their relations that essentially describe expected scene structures; the transformations specify what these structures may look like in the incoming data. The result of the matching process, the actual model-to-data mapping, constitutes an interpretation of the data and hence the system's understanding of the current world. Within this framework, the general goals of the system designer are to develop a rich and reliable representation of the objects, an understanding of the transformation process, and an effective and efficient matching algorithm to make use of all this information.

In spite of much progress in model matching, there are crucial problems that have not received adequate attention. One is representing information about 3D objects in a way that makes matching them to 2D image data efficient and reliable. That is, the geometric analysis required to relate an arrangement of 2D image features to the structure and pose of 3D objects should be sufficient and yet not involve a lot of computation during the time-critical recognition task. Another problem is ensuring that storage and recognition time costs grow only slowly with respect to the size of the model base and the complexity of the models.

We emphasize these efficiency considerations because of the remarkable visual ability of humans to rapidly recognize such a large number and variety of objects from a range of viewpoints. For example, see Figure 1, which is a small sampling of the amazing assortment of objects, differing in subject, structure and detail, that people are capable of readily recognizing in the absence of any significant context. This ability is glaringly absent from all computer vision systems to date. While there are other sources of information that seem to be important, specifically scene context [Hanson78, Hanson85, Biederman85] and model-independent understanding of 3D structure [Marr82], these useful cues may quite often be unavailable, unreliable, or merely a first step towards a full interpretation. Hence it is of great importance to develop model-based systems capable of efficiently relating image data acquired from variable viewpoints to large 3D-model bases.

It is proposed here that the computational complexity posed by a large number of objects, an unconstrained viewpoint, and the structural detail inherent in even a single view can all be handled by a design based on the following principles:

- Before the recognition task begins, compute descriptions, called *predictions*, of what the objects should look like in the image, This reduces actual recognition to a 2D matching process, substantially speeding up recognition time for 3D objects.

- Represent all the predictions by a single, combined IS-A and PART-OF hierarchy called a *prediction hierarchy.* The nodes in this hierarchy are partial descriptions that are common to views and hence constitute shared processing subgoals during matching. The recognition time and storage demands of large model bases and complex models are reduced by subgoal sharing: projections with similarities explicitly share the recognition and representation of their common aspects.

- During recognition, if a prediction unique to an object is matched, determine the pose of this object. The pose can be calculated by using iterative techniques, where a typical view from the class of views associated with the matched prediction can serve as an initial pose estimate. The typical view should, in general, produce an initial estimate reasonable enough for simple refinement techniques to work.

The original contribution of this paper is the automatic compilation, from a 3D model base, of a prediction hierarchy that can be used to recognize objects. The rest of this paper is a review of the object recognition literature (Section 2) followed by a description of a prototype system which is demonstrated using a set of polyhedral objects and projections from an unconstrained range of viewpoints (Sections 3 to 6). This study is being performed in conjunction with an investigation of the visual properties of continuous surfaces [Callahan85] for the purpose of recognizing objects with non-convex, curved parts. An expanded recognition capability will come from the combination of these two lines of research.

2. Approaches to Object Recognition

Though recognition systems to date have only managed to address portions of the problem of 3D object recognition from a large model base, there is much to learn from them. This section is a review of some of the most relevant ideas, organized along two key aspects:

- Using 2D image data as evidence for the presence of 3D objects.

- Efficiently searching for matches between arrangements of image features and a large amount of model information.

Approaches to the first aspect are reviewed in Section 2.1, *Reasoning about 3D Objects and 2D images*, and the second is discussed in Section 2.2, *Matching Strategies.*

2.1 Reasoning about 3D Objects and 2D images

Existing techniques for relating 3D model information to 2D image data can be partitioned into two basic approaches: *prediction cycling* and *pre-recognition view analysis.*

In the former, represented by ACRONYM [Brooks81], the system cycles through a process of *prediction*, deciding which projected model feature to search for in the image; *observation*, searching for instances of the feature; and *back constraining*, using additional properties of the feature instance to further constrain the possible imaging geometry (e.g., object pose and focal length) and structural variations in the object. The feature selected during prediction is one that is most constant (i.e. constrained) in size, position and nature under the currently possible variations in pose and object structure. The use of more constrained features tends to reduce the number of actual image features that can satisfy the prediction and hence reduce spurious matches. The back-constraining process uses an algebraic constraint manipulation system (CMS) to narrow down the range in pose, other parameters of the imaging geometry such as focal length, and structural parameters of the object. If the back-constraining process produces an inconsistency (i.e. the CMS determines that the resulting system of constraints is unsatisfiable) then the matcher may abandon the current match and attempt to complete a different one.

Prediction cycling is advantageous in the sense that all the information required for the recognition of a given object is represented only by its 3D model and some general principles of projective transformation. This potentially makes the matcher fairly general and the information required compact. Though utilizing 3D descriptions of objects during matching has these advantages, it is computationally inefficient to have only 3D descriptions available during recognition. Instead of always generating predictions during recognition, as in ACRONYM, it may often be better to generate them before the recognition task begins. The process of understanding what is quasi-invariant about the appearance of a general class of objects from a range of viewpoints, and describing this in terms of features that can be recovered in the image, could often be expensive. It seems inappropriate to do this at every step in the search for matches. This is especially true at the earlier stages when one has not suffiently narrowed down the objects that may be present in the image, and hence the sources of the predictions.

Similarly, the process of interpreting the match as a 3D object in a given pose should often be done after much of the matching has occurred. In other words, it is often best to wait until the matcher finds a reasonably supported mapping between the image and a predicted, general view of an object. Then, the pose will be constrained to an extent that a simple refinement process can do the rest. Calculating the back-constraints every time a new piece of evidence is matched is unreasonable when the pose and model are not sufficiently narrowed down. The difficulties of manipulating partially constrained poses during matching has been emphasized by Lowe[85]: "the back-projection step is difficult from a mathematical viewpoint, since there are no simple, closed form solutions for determining the range of viewpoint parameters consistent with a given set of image matches". Even if such calculations could be done efficiently, they should only be done when the possible models have been sufficiently narrowed down.

In the alternative approach, *pre-recognition view analysis*, all expectations of what to look for in the image are generated before the actual recognition task. Hypothesizing the identity of an object then becomes a 2D matching problem and object pose analysis is initiated later during a verification phase when the possible objects have been sufficiently narrowed down. The characteristic-view based schemes of Chakravarty[82] and Wong[84], the property spheres of Fekete[84], the SCERPO system of Lowe[85], the principal views of Cooper[87] and the image-based descriptions of the VISIONS system developed in [Weymouth86] all roughly follow this method. Additionally, this approach has similarities with the photometric stereo interpretation system of Ikeuchi[87].

An important concept in pre-recognition view analysis is that of a *generic view* [Koenderink76], [Callahan85] (also called a *characteristic view* in [Chakravarty82] and [Wong84]). A *generic view* (GV) is a class of viewpoints that have the same set of features visible. The features are typically projections of straight, physical edge segments; though they can be more sophisticated, depending in part on the object surfaces considered [Callahan85]. Figure 2 shows an example of the GV's for an object and a selection of views within a single GV. The partitioning of the different views of an object into generic view classes is a useful way of reducing the 3D-2D matching problem to one that is more manageable: that of comparing 2D image data to 2D descriptions (one is valid for each GV). GVs not only simplify the prediction analysis, but also allow one to generate rich, informative predictions. This is because the GVs in some sense maximize the number of simultaneously visible features being considered together in the same description. Hence, a substantial amount of what tends to be fairly constant visual structure is available for description.

Unfortunately, both Chakravarty and Wong do not take advantage of this richness. Instead, they describe the views using edge-edge connectivity (i.e. which edge is connected to which) and junction type as the relations; little metric information is used to build descriptions of the view. Such schemes tend not to be very descriptive of the projection's shape or structure, nor are they robust outside of controlled blocks-world domains. In spite of this, the use of GVs as an initial step in the prediction process is a useful idea.

Lowe[85] suggests using *non-accidental* groupings of image features as object recognition cues. This amounts to analyzing the model for subsets of physical edges whose projections have view invariant aspects in their relative size, position or orientation. For instance, the ideal orthographic projection of a rectangle contains segments that are always parallel, as well as ones with always adjacent endpoints. This type of invariant analysis is useful in the sense that the expectations are valid over wide a range of viewpoints, giving one a parsimonious use of features. Also, the relational features chosen tend to reduce the number of false-alarms by accidental groupings. Though these are important goals, the general method needs to be extended to a more complete process of describing what the projections look like for the purpose of discriminating between a large number of objects and views. In other words, analysis of invariance is not enough; it is also important to describe the appearances in substantial detail. Specifically, there may be properties of the projection that have a narrow range of values over a wide range of viewpoints, but they are not strictly invariant. The process of compiling these descriptions into a data-base for efficient storage and access could perhaps use aspects of the invariants analysis described by Lowe.

The vision system described by Weymouth[86] constitutes a fairly rich collection of representation and control strategies. Nevertheless, there is a major emphasis throughout the work on what is called *image-based description*: that is, recognition by relating the image to a primarily 2D description of a scene, followed by a verification process involving 3D interpretation. The rich description of object appearances used by Weymouth could provide a useful starting point for scene analysis research. He stressed color, local texture measures, location in the image, size, gross shape descriptors and relative position of parts (and whole objects to each other). These aspects of an object's appearance could be fruitfully complemented and extended by incorporating the automatic methods for analysis of GVs and invariants discussed above.

Once a reasonably confident and complete match is isolated between expected data and actual image data, this mapping can be used to determine the position and orientation (pose) of the object in space. This is done by finding the pose in which the accumulative distance between the actual image and the projection of the object in this pose is minimized. Lowe[85] reviews both closed-form and iterative solution techniques. He argues for the use of iterative positional error minimization because it is currently standard practice in photogrammetry and the best closed-form method to date [Fischler81] is quite complex, with a quartic polynomial that "presumably must be solved by iterative methods" anyway. Additionally, since the set of simultaneous equations implied by a match is augmented by additional constraints (i.e., the system knows the rough view that the prediction is associated with) the solution neighborhood tends to be narrowed down considerably – enough to provide a good starting approximation for an iterative method. His method uses Newton-Raphson iteration which has quadratic convergence and corrects the rotational error by composing the initial orientation estimate with incremental rotations about the camera coordinate axes. These particular correcting rotations are useful because the derivatives of image position with respect to them can be expressed in a simple form. Lowe reports that the algorithm converges rapidly: the error is typically reduced to a negligible amount within 3 iterations (see Figure 3 for example results).

Before closing this section, it is important to note that there is an approach to pre-recognition analysis that is somewhat different to those just described. Goad[83] used prediction cycling as in ACRONYM, but pre-calculated and tabulated all calculations. He designed a special-purpose (i.e. single object) matching system by precomputing the predictions and back-constraints for an object from different poses and storing them in tables. He also pre-pruned the search-tree using mutual invisibility considerations. Goad finds a substantial speed-up in recognition time over ACRONYM, though it is not clear from his published results how effective his system was at finding matches.

There is a key difference between Goad's approach and the view analysis methods discussed above. In the view analysis approaches, instead of a simple prediction step that involves the estimate of the position of a single image segment, a much richer description is generated of what is expected in the image (given the presence of the object), a description that involves the arrangement of many segments. These substantially fuller descriptions of the views can be organized into a conceptual hierarchy, as will be discussed in the next section, to handle large numbers of objects and the complexity of the object itself, not just variations under camera movement. Goad's scheme is geared towards single-object recognition and it is not clear how it can be extended to work with a large model-base. In spite of this, his work stands as a good example of how pre-recognition analysis can speed up recognition considerably.

2.2 Matching Strategies

From the discussion above, it is clear that reducing the matching problem involved in recognition to one that is essentially 2D-2D is a useful starting point. Yet, it is also clear that this still leaves a task of some complexity. The predicted image must be effectively and efficiently compared with the actual image data and there must be some process of selecting the right prediction. In this section, existing matching methods will be reviewed, which can be roughly categorized as one of the following schemes: 1) generalized Hough transform, 2) local constraint-based relaxation, 3) one-shot selection, 4) iterative match growing and 5) hierarchical representations. These categories are not sharp, and it is also possible to combine the techniques; nevertheless this classification provides a useful starting point. The different methods tend to emphasize different aspects of the matching problem.

1) In *generalized Hough transform* approaches [Ballard81, Silberberg84], each image feature votes for each possible combination of pose parameters and structural parameters that could have generated it. These votes are accumulated in arrays indexed by the quantized parameters, and the best matches are selected by looking for peaks in the vote count. Such methods are useful in the sense that they tend to make the most out of the feature as evidence, in spite of heavy fragmentation, occlusion and local measurement error. The target object will tend to form a prominent peak in the parameter space in spite of such disturbances because many features will still be present and in the right position to vote for this peak. Yet, because the evidence for all possible parameter combinations is accumulated exhaustively and stored explicitly, this method is geared towards problems with small solution spaces. It is unlikely that such a small space could be used to differentiate between

objects from a large model-base seen from a variety of poses.

2) *Constraint-based relaxation* techniques essentially select the right model-image mappings by filtering out bindings that are inconsistent with others. The methods follow this basic pattern: assign, as potential labels, all promising model elements to all data elements; then, using measurable relations among the data elements and known relations (local constraints) between the model elements, iteratively rule out possible assignments. What should be left are sets of mutually consistent assignments called interpretations. This general idea has been incorporated into the designs of Davis[79], Kitchen[80], and Faugeras[84] (See also association-graph techniques in [Ballard82] and [Bolles82]. In these techniques, all the consistency analysis is done first and then cliques of consistent assignments are explicitly searched for. This is more expensive but more complete than relaxation). The relaxation process can be rather fast once all the initial associations are found: usually only a few iterations plus some simple post-processing are required. Additionally, the filtering generally involves testing local constraints, so that much of the processing can be done in parallel. The problem with it is that, for a large number of possible models and viewpoints, the initial assignment phase is prohibitively expensive. One must find a way to avoid considering most of the assignments in the first place.

The technique does have a possible role in situations where the number of possible labels is small and it is important to give exhaustive consideration to all possibilities. These situations may occur as subgoals of some larger problem. For an example of this type of design, see the hypothesis-generation step in Bolles[82].

3) There are several designs that have attempted to narrow down the possible objects and poses associated with the image data by a *one-shot selection* or ranking of the possible

candidates. The basic idea is to measure a small set of features whose existence or values essentially index into the model-pose space. Once a small number of models and poses have been selected, these methods look for more detailed evidence that confirms one model over the other and also refines the pose and other structural details. The focus-feature method of Bolles[82], the compact model scheme of Shneier[79], the model-selection process of Nevatia[74], the rule system of Hanson [85], the grouping processes of Lowe[84] and the property spheres of Fekete[84] all basically follow this scheme. Though breaking up the processing into these two phases is a useful idea, the single indexing step is not sufficient to properly narrow down the possible models and their 2D projections. In the context of matching a large number of 3D models, the necessarily small set of readily detectable features would essentially have to be ideal invariants. That is, they would have to be detectable almost all the time that the object is being viewed, but at the same time be very object-specific. This can hardly be expected, especially for a large model base and an unconstrained camera position.

4) The matching algorithm in Goad[83] follows a general approach that will be referred to here as *iterative match growing*. He represents the matching process as a best-first search through a space of single, partially complete interpretations. In other words, there are partial, one-one mappings between image and model segments that the system keeps track of and tries to complete in a best-first fashion. There is a single operator: adding a new image-model segment binding consistent with the partial match's already existing bindings. Branching points in the search occur when there are multiple possible instantiations (more than one image segment to bind the model segment to) or alternative model segments to search for in the image. This scheme is useful in the sense that most of the false bindings

between image segments and model segments are not tested. This is because the search for image and model segments to bind to at any given moment is constrained by the bindings currently being entertained. This is in contrast to the exhaustive search of the Hough techniques and the potentially involved initial assignment phase of the relaxation approaches. Also, since the search involves testing for a variety of image features (in this case, segments of different orientation, size and position, depending on the part of the search space it is in), it does not rely on a fixed, small set of features to narrow down the possibilities as in the one-shot cuing strategies above. In Section 2.1, it was concluded that Goad's overall 3D object recognition scheme may lack extendibility into multi-object recognition. In spite of this, Goad's idea of matching as a best-first search through a space of partial matches is a useful concept that could be incorporated into a more complete system.

5) One promising way of handling large numbers of models and views is by using *hierarchical representations* (Marr[77], Brooks[81], Mulder[85] and Weymouth[86]). These representations are generally built up by using IS-A and PART-OF links. An example of an IS-A hierarchy is Brooks' restriction graph of increasingly more restrictive constraints which supports the description of specific object structures as specializations of more general ones. This in turn potentially facilitates an efficient search for instances of a large family of different objects, since it allows the system to check some constraints once for all objects characterized by them in the hierarchy. With respect to recognition in *2D images*, we consider it to be more effective to organize the *predictions* in this fashion, not just the objects. The other type of hierarchy, the PART-OF hierarchy, describes models or their projections in terms of their part and subpart decompositions. These parts and subparts

can be used as intermediate interpretation subgoals in much the same way as the partially constrained objects in the restriction graph: any time two objects, two views or even two sections of the same view have parts in common, the processing for these parts can be shared. For example many projections of a house will contain parallelogram subparts; in fact, in some projections this part will occur several times as windows, doors, shutters, roof and wall.

3. Overview of Project

The problem currently being studied is the compilation and use of a prediction hierarchy to recognize polyhedral objects using straight-line segments detected in the image. This is considered a good initial problem because polyhedra can be used to represent approximately many objects in the world and, more important, polyhedra have sufficient richness and complexity to fully test our ideas, without requiring the setting up of complex machinary needed to represent say curved surfaces. Additionally, there exist reasonably reliable techniques for extracting straight edges from images [Burns84], [Boldt86]. The actual objects used to demonstrate the design are shown in Figure 4. The objects have differences and similarities in various dimensions and part of the problem is to take advantage of both. The similarities in their visual structure, such as occurrences of parallelograms or of certain types of line junctions, must be utilized by the recognizer to make the search for a match efficient. The differences in visual structure, such as height-to-width proportions, must be utilized to discriminate between the objects. Additionally, the variations in visual appearance caused by variations in the camera must be taken into account while doing the structural analysis.

The camera geometry used is as follows. The *viewpoint* of the camera is taken to be the position of the image origin in the object coordinate system. Currently, the projection is taken to be orthographic (i.e. focal length at infinity). Also, since the objects are expected in any pose, all predictions generated are invariant to re-scaling, translation and rotation in the image plane. This means that there are only two degrees of camera variation that have a significant effect on the projections being described by the predictions: the two angular components of the viewpoint that sweep out a *viewing sphere* about the object.

As described in Section 1, our basic approach is to treat recognition as a 2D matching problem. This is done by characterizing what the objects look like from different viewpoints before the recognition task begins. In addition to view analysis, the pre-recognition phase will also include extensive organization of the expectations in the form of a PART-OF/IS-A hierarchy for efficient storage and matching.

A top-level view of the algorithm is blocked out in Figure 5 and proceeds as follows:

1. *Compile the prediction hierarchy from the set of 3D models given.* The hierarchy is compiled by starting with a small set of simple and very general structural predictions and then iteratively searching for commonly occurring combinations or specializations of these predictions across all objects and views, eventually isolating predictions that characterize the projections of fairly specific objects from a range of views.

2. *Match predictions to input image.* During the recognition phase, look for matches between segment descriptions and actual sets of image segments by a combined search of the prediction hierarchy and image data base. The hierarchy is used as an organized network of recognition subgoals.

3. *Refine the pose estimate.* For each promising match, calculate the pose more precisely given the image-model mapping and an initial estimate of the pose implied by the prediction matched (i.e., some typical viewpoint from the set of those that could satisfy the prediction matched). For pose estimation refinement, the iterative method described in [Lowe85] could be used.

The integration of these three processes could be more sophisticated: for instance, the pose-refinement process could start up any time a reasonable match is generated, while the matcher continues to search for new ones, possibly guided by the result of the pose analysis. However, the pressing issue at this point concerns the basic design of these steps; investigating further how they might fit together is outside the scope of this paper.

Following a description of the nature of the predictions and their representation (Section 4), the processes that compile them and use them for matching are discussed and demonstrated (Sections 5 and 6).

4. Predictions and Their Representation

A *prediction* is a statement concerning some structural aspect of the image of an object. For example, this may be as simple and general as an assertion that there exists a pair of parallel segments in the projection; or as complex as a description of an image unique to some object. A prediction is represented here as a relational graph; the elements in the graph are projected straight-line segments. The relations associated with arcs in the graph mutually constrain the orientations, positions and sizes of pairs of segments. For example, the relation *parallel* constrains two segments to having the same orientation. The predictions can be viewed as conjunctions of these relations over formal segment arguments;

for example, see the description of a parallelogram in Figure 6.

A relation between a pair of projected segments used in the predictions is represented by ranges of four relational measures, u, v, α and s (see Figure 7). Call one segment s_1 and the other s_2. The vector (u, v) is the position of segment s_2 relative to s_1: it is the displacement of an endpoint of s_2 from an endpoint of s_1 measured along s_1 and normal to s_1, divided by the length of s_1. The angle between them, α, is measured counterclockwise from s_1 to s_2; and s is the relative scale or length ratio of s_2 over s_1. A relation is defined as an *extent box*, i.e. intervals in u, v, α and s, in order to capture the variation over ranges in viewpoint. For instance, projecting a pair of parallel object segments over all possible viewpoints will generate a set of measurements that have a single value (zero) in the α dimension but some non-trivial extent in the others. Similarly, the family of projections of a pair of object segments that share an endpoint can be represented by a relation that has the value zero in both position components (u, v). A relation between projected segments is considered useful if it is valid over a wide range in viewpoints and its extent box is small in volume (for example, consider the two view-invariant relations just mentioned). The latter property is important if the relation is to help characterize an object's projection with a specificity sufficient to discriminate the object from a large number of other objects and from chance arrangements of image segments. Although invariant relations are clearly useful [Lowe85], they alone are in general not enough to fully characterize projections. For instance, proportions are often strong characterizations of object structure, but the length measurement ratios that represent them are often not strictly view invariant. For example, the tall box in Figure 4 has a height to width ratio that is significantly different from the cube over a large range in views and no other property can be used to differentiate them.

It should be clear from the above discussion that a prediction, since it may be composed of non-invariant relations, may only be valid over a restricted set of views for a given object. A prediction *instance* is a set of model segments, a mapping from the model segments to the segments of the prediction's relational graph and a range of viewpoints from which the prediction is valid for these segment bindings. See, for example, the *Y junction* prediction of Figure 8. For a given model base, there is a set of such instances associated with each prediction and a *cumulative visibility* that is the total area of all their visibility regions on the viewing sphere, across all objects.

Any given prediction in the *prediction hierarchy* is implicitly some relational graph, but explicitly it is almost always described as some combination or specialization of other predictions (see Figure 9). In such cases, it is a *derivative* of the other predictions. A prediction is a *specialization* of another if it can be described by adding new relations or narrowing the extent boxes of existing ones. For example, the rhombus can be described by adding a relation between the bottom and side segments of the more general parallelogram prediction that constrains their ratio of length to be one. A prediction is a *combination* of other predictions if it can be described as a conjunction of these other predictions. Predictions may be combined in various ways, depending on the segment mappings between the whole prediction and its parts. See, for example, the triangular prism prediction of Figure 9. The mappings are collectively called the *arrangement* of the combination.

5. Prediction Hierarchy Compiler

In Section 4 it was shown that a prediction hierarchy is composed of predictions that are combinations or specializations of other predictions. The initial predictions of the hierarchy (those that all the other predictions are derivatives of) ought to be those that are valid for very large numbers of objects and are essentially invariant to viewpoint (such predictions are like the invariant binary relations discussed above and in [Lowe85]) – and the final predictions ought to be valid for only a few objects and over a narrow enough range of viewpoints that pose estimate refinement and verification can easily be performed. A useful way to build such a structure is to start with a small set of simple, ubiquitous structural predictions and then iteratively search for useful combinations or specializations, eventually isolating predictions that characterize the projections of specific objects. By using this iterative construction approach we limit the combinatorial complexity, and hence processing time, to a manageable level. This is because the system is never performing subgraph isomorphism analysis over large graphs: it is always comparing combinations of small numbers of parts.

5.1 Criteria for Selecting Prediction Derivatives

In building the prediction hierarchy, it is important to understand which of the possible combinations and specializations of simpler predictions are useful to explicitly represent as nodes in the hierarchy. Considering the set of object projections that satisfy a given prediction, useful derivative predictions are those that are also valid for some significant fraction of this set. For the recognizer to efficiently search for possible objects and views, a derivative prediction should occur frequently enough that its satisfaction tends to dichotomize the instances of the prediction it is derivative of.

The following is a more precise description of how the compiler selects derivatives every iteration. Recall that an *instance* of a prediction is a mapping between the prediction segments and some model segments – and the range of viewpoints from which this mapping satisfies the prediction constraints (Fig. 8). Another useful concept for this discussion, *cover*, is defined as follows. Consider a given prediction p and one of its derivatives, p'. An instance of p' ($i_{p'}$) covers an instance of p (i_p) if the set of model segments that $i_{p'}$ maps to is a superset of the segments that i_p maps to (Fig. 10). A derivative, p', covers some subset of the derivatives of p if for all instances in this subset there exists an instance of p' that covers it.

The initial, simple predictions have a large number of instances across the objects, views and even within a single view and the final predictions have only a few instances. For the matcher to use the hierarchy to efficiently search for matches to these final predictions given matches to the initial predictions, the derivatives p' selected each iteration for each intermediate prediction p by the compiler should have the following properties:

1. The number of instances of each p' should tend to be much less than that of p.

2. Each instance of p should be covered by an instance of some p'.

3. The number of derivatives, p', per prediction, p, should be low.

(1) The number of instances of p' should tend to be much less than that of p. This is because a prediction's set of instances represents the possible matches between image segments that satisfy the prediction's constraints and the models – and the recognition process should be able to narrow down the set of possible models, views and model parts at a reasonable rate each step in the matching process. Since p' always has constraints that do not exist in p, it is not satisfiable everywhere that p is. In fact, the instances of p covered by p' are usually a small set of the total. Unfortunately, this does not mean that most

derivatives of p necessarily have fewer instances than p – there is a redundancy problem that can actually *increase* the number of instances of p' over p. It often occurs that more than one instance of p' covers a given instance of p. This happens sometimes when the derivative p' is a combination of p with itself or other predictions and instances of these combinations overlap heavily. For example, the view of a tetrahedron in Figure 11 contains 12 instances of the coincidence prediction. Two derivatives of it, *snake* and *trihedral junction*, both cover the coincidence instances, but the trihedral junction prediction does so with much fewer instances (4 instead of 12). Selecting snake as a derivative of coincidence hardly reduces the number of possible matches, and is thus a poor choice.

(2) Each instance of p should be covered by an instance of some derivative of p to reduce dead-ends during the search for matches. Say that there is an instance of p that is *not* covered, and the model segments that it maps the prediction segments to are $\{m_1, m_2, m_3\}$. Also say that the image includes the projection of $\{m_1, m_2, m_3\}$ from a viewpoint such that the constraints in p are satisfied. If the recognition system matches p to the projection of $\{m_1, m_2, m_3\}$ it cannot use this match as a step in the process to matching predictions unique to the model projected. This is because there is no path in the hierarchy from p to such a goal prediction – no set of additional constraints that can be tested to isolate that model and view from others possible given p.

(3) The number of derivatives per prediction should be low. This number represents the branching factor at each node in the hierarchy. Large branching factors can slow the match search process down. Picking devivatives that cover large subsets of a prediction's instances helps to reduce the number of derivatives necessary to ensure that all instances of the prediction are covered.

A derivative-selection procedure that approximates the general conditions above has been incorporated into the compiler implemented and is described in the next section.

5.2 Iterative Hierarchy Construction

For the experiment reported here, *parallelism* and *endpoint coincidence* are used as the initial set of simple and general predictions; and the iterative process is stopped when all of the predictions without derivatives (the ones at the top of the hierarchy) are either associated with single objects or, if they are satisfied by projections of more than one object, there are no describable differences between their projections. The prediction hierarchy compiler implemented iteratively combines predictions and then follows this up with specialization of predictions associated with multiple objects that are not distinguished by combinations of parallelism and end-point coincidence relations alone.

For each iteration of the combination process, the system isolates useful combinations by (1) finding predictions that often appear together in the same projections and then (2) selecting a subset of these commonly occurring combinations that approximately satisfy the criteria discussed in Section 5.1.

The co-occurrences are found in the following fashion. All instances of all predictions are stored in data structures called *visibility maps.* There is a visibility map for each object; the maps are arrays of cells indexed by the two viewpoint parameters, making a discrete sampling of the view sphere about the object. Each cell lists prediction instances visible from the associated viewpoint and object; and with each prediction is a list of cells that contain it. To find frequent co-occurrences between some prediction p and other predictions, the system looks for predictions that appear in the same cells as p

and accumulates the total number of cells for each that do. To keep the number of combinations analyzed down to a manageable size, we consider only co-occurring *pairs* in whose arrangements the part-to-whole mappings overlap for at least one whole segment.

After finding commonly occurring combinations, the compiler tries to select a small subset of them that collectively and efficiently cover the instances of the predictions that they are derivatives of. Consider a prediction p and its derivatives p'. The derivatives are iteratively selected in the following way:

1. The compiler keeps track of the instances of p not covered yet.

2. The next derivative selected is one the covers the largest number of previously uncovered instances but does so with the fewest number of instances of its own. Specifically, pick p' whose ratio of these two numbers is highest.

3. The selection process stops if only a few instances of p are left uncovered and the derivatives thus far unselected have a large number of instances themselves.

Note that this iterative selection process is an inner loop to the iterative process that is building up the prediction hierarchy level by level. This inner loop works within a level to select a useful set of derivatives at that level.

5.3 Experiment

The prediction hierarchy compiler design was tested on the models in Figure 4. The resulting hierarchy is shown in Figure 12. There are six levels of the hierarchy; the average path length between the initial nodes and goal nodes (object matches) is 3.9. The total number of nodes (predictions represented) is fifteen. Considering that the hierarchy is capable of being used to distinguish five objects from almost all viewpoints (with an average of 8 segments per view) – and the predictions are represented efficiently as combinations of simpler ones, this appears to be a reasonable result.

The iterative combination left the tall box and cube indistinguishable. This was corrected by the specialization process by adding segment length ratio relations between two pairs of segments. This conjunction of two proportional relations was satisfied by the tall box projections over most of the view sphere, and satisfied by none of the projections of the cube.

6. Recognition

The object recognizer finds correspondences between segments detected in a given image and the segments of some model in the model base by an organized search of the prediction hierarchy and image data base. Like parsing and other interpretation problems, there are many ways to perform such a search (e.g., top-down, bottom-up or some combination of both). A fairly straight-forward method of search was implemented for the purpose of demonstrating the usefulness of the prediction hierarchy. The search proceeds in a bottom-up fashion by iteratively selecting a previously satisfied prediction (an image-prediction match), attempting to find additional evidence in the image to satisfy its derivative predictions (i.e., testing new constraints on already matched image segments for specializations and searching for parts for combinations), and storing any new image-prediction matches for further expansion. Figure 13 shows the results of this matching process using the hierarchy in Figure 12 and a synthetically generated set of image segments.

7. Conclusions

This report presents an approach to object recognition developed to handle the computational complexity posed by a large model base, an unconstrained viewpoint and the structural detail inherent in the projection of an object. The approach stresses extensive pre-recognition view analysis and organization of the resulting expectations into a PART-OF/IS-A structure called a *prediction hierarchy*. A major contribution of this report is the design of a prototype system that automatically compiles a prediction hierarchy for the recognition of polyhedral objects. The design was implimented and demonstrated on a set of simple objects. Further experiments will involve larger model bases, more complex models, and image noise. This research has been done in conjunction with studies of the visual properties of piecewise smooth surfaces [Callahan85] for the purpose of recognizing objects that may have curved and non-convex parts. An expanded recognition capability will come from the combination of these two lines of research. Current design refinements are centered on the cost/benefit analysis of adding a prediction to the hierarchy and the matching control strategies ([Weymouth86] and [Draper87]), including the use of image context for match selection.

REFERENCES

[1] Ballard, D.H. and D. Sabbah, "On Shapes", proceedings of IJCAI-81, Vancouver, pp. 607–612, 1981

[2] Ballard, D.H. and C.M. Brown, *Computer Vision*, Prentice Hall, Englewood Cliffs, NJ, 1982.

[3] Besl, P. and R. Jain, "An Overview of Three-Dimensional Object Recognition", RSD-TR-19-84, Univ. of Mich., Ann Arbor, Nov. 1984.

[4] Biederman, I., "Human Image Understanding: Recent Research and a Theory", CVGIP, vol. 32, pp. 29–73, 1985.

[5] Boldt, M. and R. Weiss, "Geometric Grouping of Lines", CVPR, pp. 489–495, 1986.

[6] Bolles, R.C. and R.A.Cain, "Recognizing and Locating Partially Visible Objects: The Local-Feature-Focus Method", IJRR-1, no. 3, p.57–82, 1982.

[7] Brooks, R.A., "Symbolic Reasoning Among 3D Models and 2D Images", Artificial Intelligence, vol. 17, pp. 285-348, Aug. 1981.

[8] Burns, J.B., Hanson, A.R. and E.M. Riseman, "Extracting Straight Lines", IEEE Trans. Pattern. Anal. Mach. Intell., v.8, no.4, pp. 425–455, Jul. 1986.

[9] Callahan, J. and R. Weiss, "A Model for Describing Surface Shape", CVPR, pp. 240–245, 1985.

[10] Chakravarty, I. and H. Freeman, "Characteristic Views as a Basis for Three-Dimensional Object Recognition", Proceedings of the SPIE, Vol. 336: Conf. on Robot Vision, Arlington, pp. 37–45, 1982.

[11] Cooper, P. and S. Hollbach, "Parallel Recognition of Objects Comprised of Pure Structure", DARPA Image Understanding Workshop, pp. 381–391, 1987.

[12] Davis, L., "Shape Matching using Relaxation Techniques", IEEE-PAMI, vol.1, no.1, pp. 60–72, 1979.

[13] Draper, B.A., R.T. Collins, J. Brolio, J. Griffith, A.R. Hanson and E.M. Riseman, "Tools and Experiments in the Knowledge-directed Interpretation of Road Scenes", DARPA Image Understanding Workshop, pp. 178–193, 1987.

[14] Faugeras, O., "New Steps Toward a Flexible 3-D Vision System for Robotics", Proc. ICPR-7, pp.796–805, 1984.

[15] Fekete, G. and L.S. Davis, "Property Spheres: a New Representation for 3-D Object Recognition", Proceedings of the IEEE Workshop on Computer Vision: Representation and Control, Annapolis, pp. 192–204, 1984.

[16] Fischler, M. and R. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography", Comm. ACM, Vol. 24, no. 6, pp. 281–395, 1981.

[17] Goad, C., "Special Purpose Automatic Programming for 3D Model-Based Vision", Proc. DARPA Image understanding Workshop, Arlington, pp. 94–104, 1983.

[18] Hanson, A. and E. Riseman, "VISIONS: A Computer System for Processing Scenes", in *Computer Vision Systems*, Hanson and Riseman (Eds.), Academic Press, pp. 303–333, 1978.

[19] Hanson, A. and E. Riseman, "A Methodology for the Development of General Knowledge-Based Vision Systems", in *Vision, Brain, and Cooperative Computation*, M. Arbib and A. Hanson, Eds., MIT Press, Cambridge, 1986.

[20] Haralick, R., "Scene Analysis, Homomorphisms and Arrangements", in *Computer Vision Systems*, Hanson and Riseman (Eds.), 1978.

[21] Ikeuchi, K. "Precompiling a Geometrical Model into an Interpretation Tree for Object Recognition in Bin-picking Tasks", DARPA Image Understanding Workshop, pp. 321–339, 1987.

[22] Kitchen, L. "Relaxation Applied to Matching Quantitative Relational Structures", IEEE SMC-10, no.2, pp. 96–101, 1980.

[23] Koenderink, J.J. and A.J. van Doorn, "The Singularities of the Visual Mapping", Biological Cybernetics, vol. 24, pp. 51–59, 1976.

[24] Lowe, D., "Visual Recognition from Spatial Correspondence and Perceptual Organization", Proc. IJCAI-9, pp. 953–959, 1985.

[25] Marr, D. and N.K. Nishihara, "Representation and Recognition of the Spatial Organization of Three-Dimensional Objects", Proc. Royal Soc. B. 200, pp. 269–294, 1977.

[26] Marr, D., *Vision*, W.H. Freeman, 1982.

[27] Mulder, J.A., "Using Discrimination Graphs to Represent Visual Knowledge", Unversity of British Columbia Laboratory for Computational Vision, T.R. 85-14, 1985.

[28] Nevatia, R., "Structural Descriptions of Complex Curved Objects for Recognition and Visual Memory", AIM-250, Stanford AI Lab, October 1974.

[29] Shneier, M., "A Compact Relational Structure Representation", Proc. of IJCAI-6, pp 818–826, 1979.

[30] Silberberg, T.M., D. Harwood and L.S. Davis, "Object Recognition Using Oriented Model Points", U. of Maryland CS-TR-1387, 1984.

[31] Weymouth, T.E., "Using Object Descriptions in a Schema Network for Machine Vision", Ph.D. Thesis, COINS, U. Mass., Amherst, 1986.

[32] Wong, E. and K. Fu, "A Graph-theoretic Approach to Model-Matching in Computer Vision", Proc. Workshop on Computer Vision: Representation and Control, pp. 106–111, 1984.

8. Figures

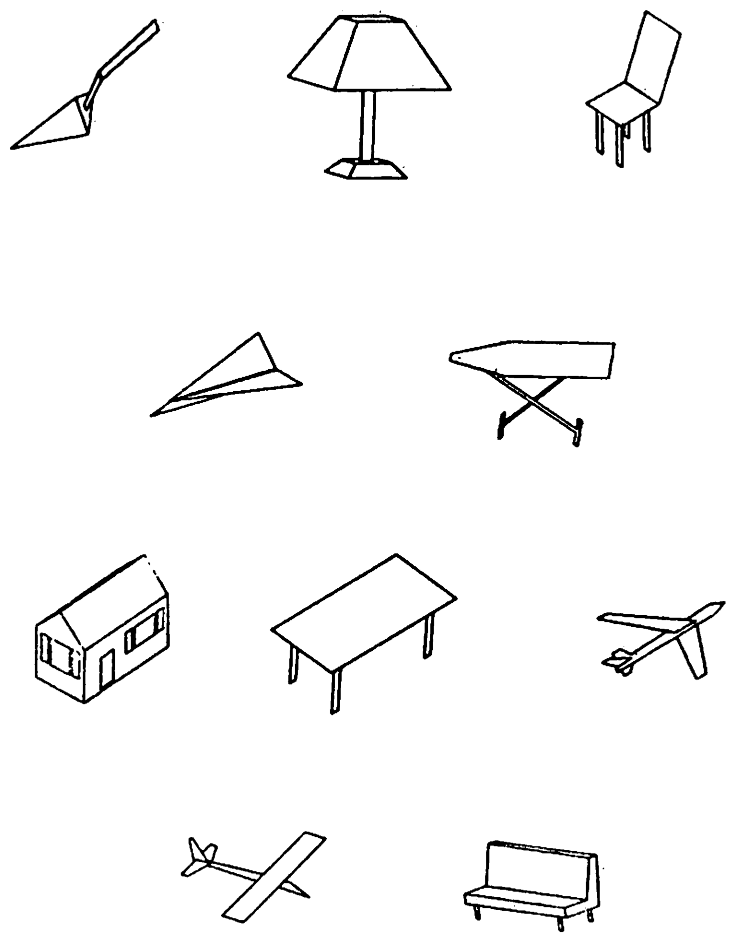
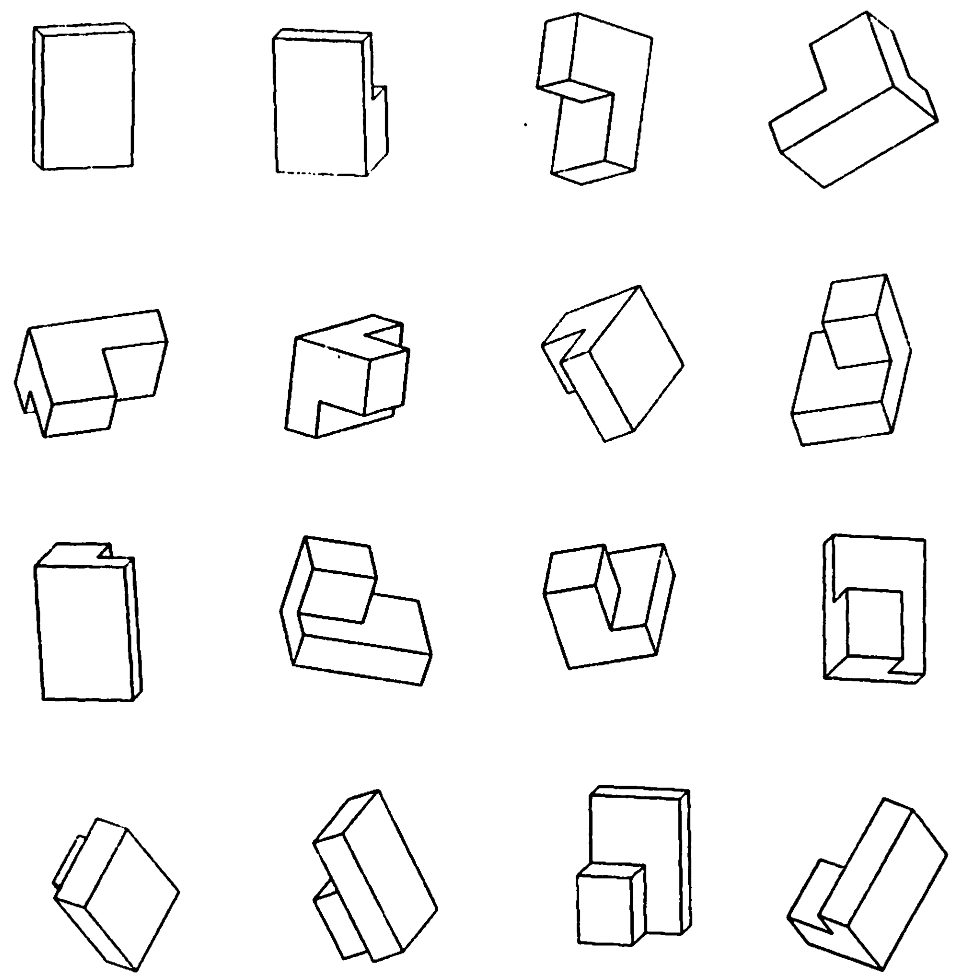
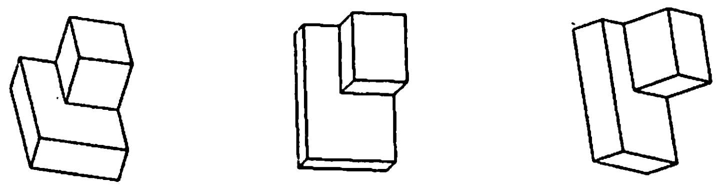


Figure 1: A sampling of the amazing variety of objects readily recognized by humans.



(a)



(b)

Figure 2: (a) GVs for a polyhedron, where 2 views are in the same class if the same segments are visible and unfragmented. (b) A sampling of views from a single GV.

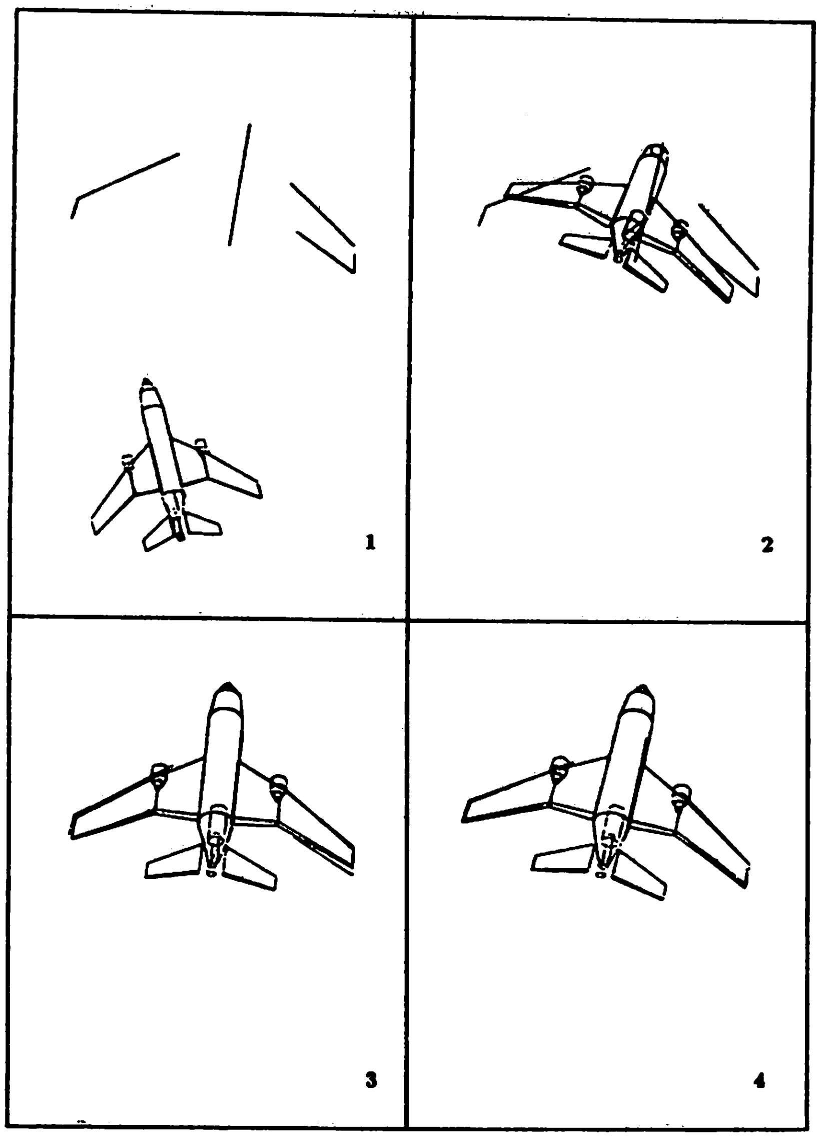


Figure 3: An example of the pose refinement process developed by Lowe working on a modeled airplane. The initial estimate of position and orientation is shown in the box at the upper left. The program is also given the mapping between edges in the model and the displayed 2D lines. The first three iterations of convergence towards a least-square solution of viewpoint are shown in the other boxes.

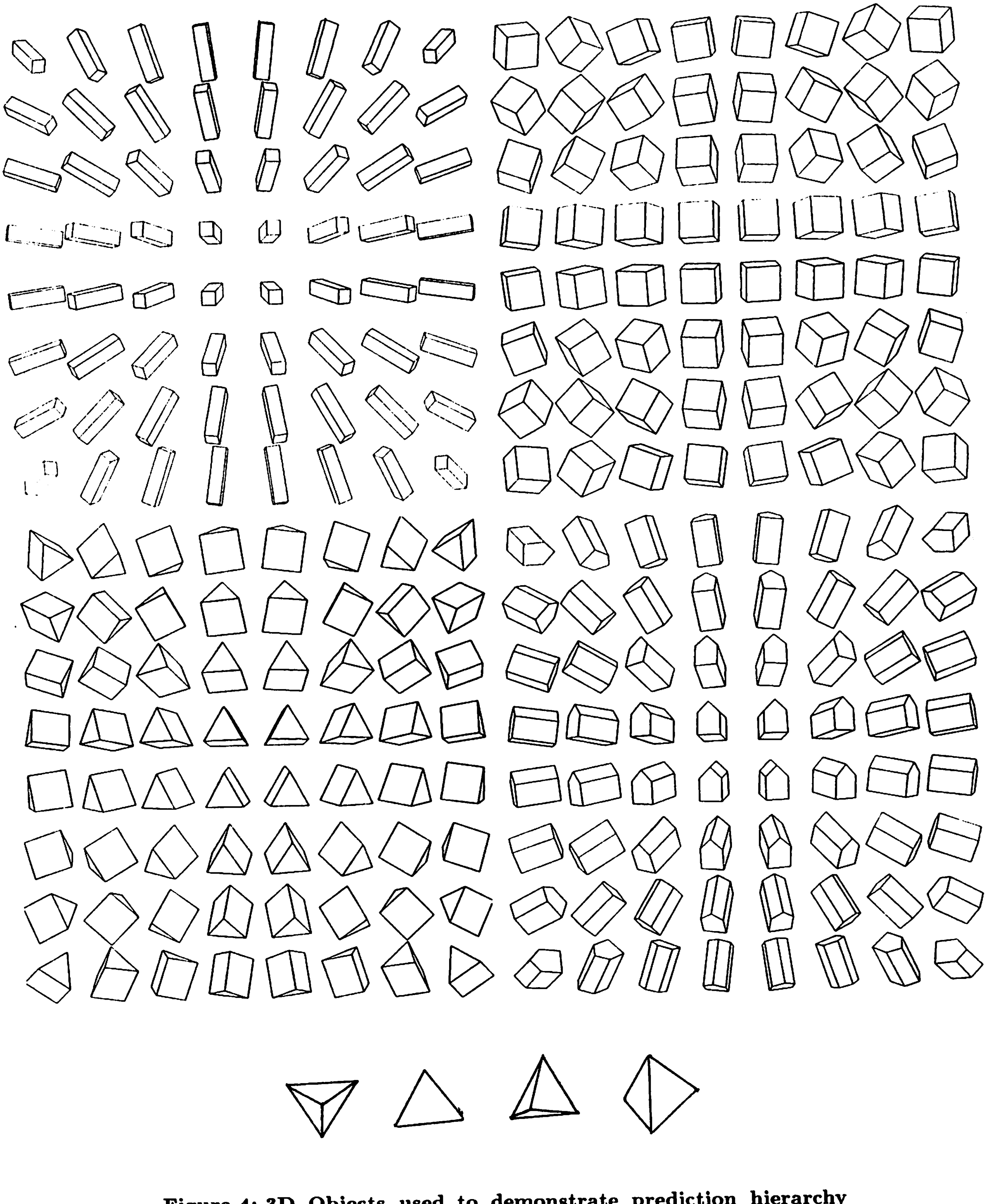


Figure 4: 3D Objects used to demonstrate prediction hierarchy compiler and matcher (*tall box*, *cube*, *triangular prism*, *house*, *tetrahedron*). A section of the views sampled is shown for each.

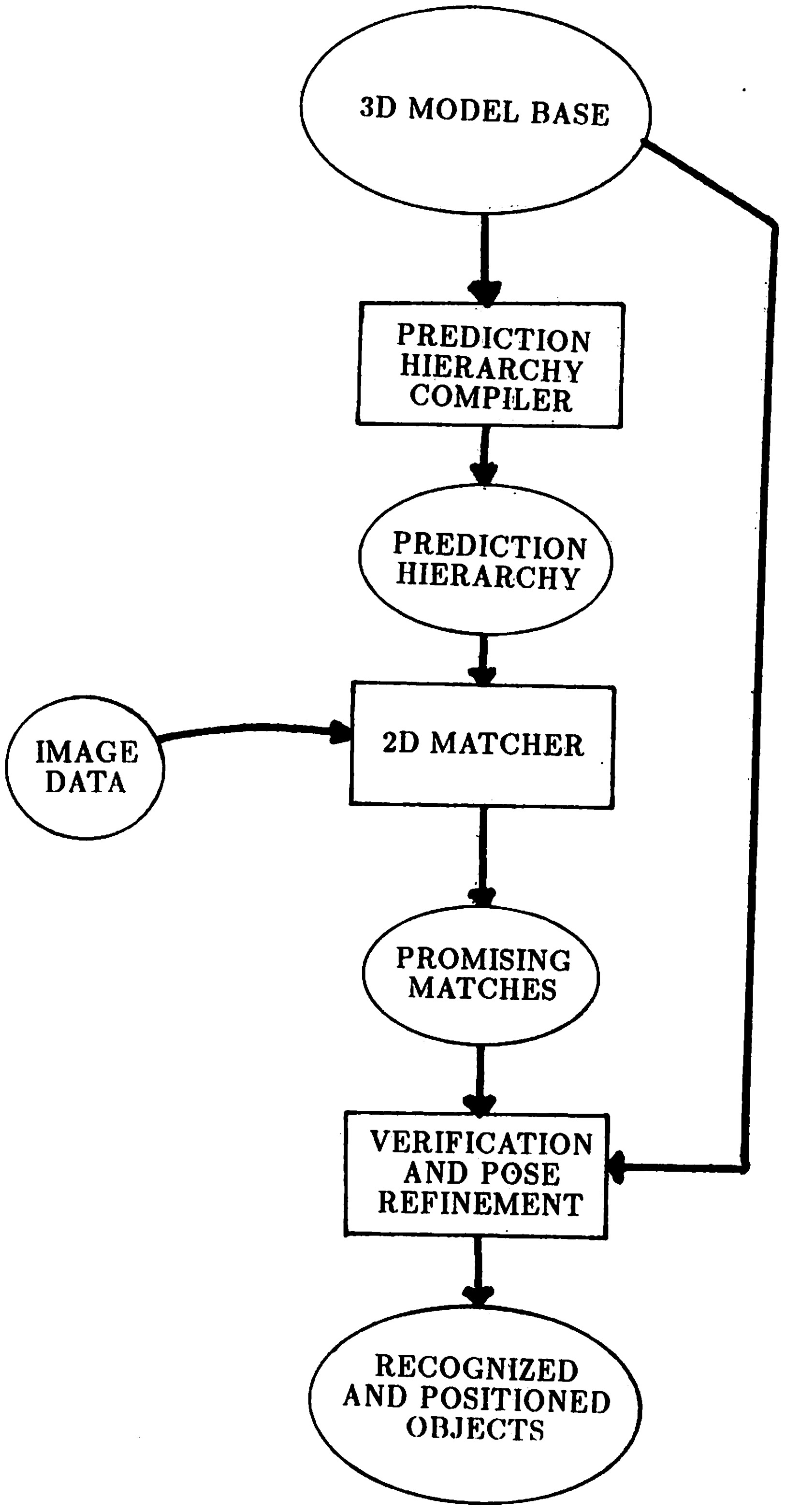
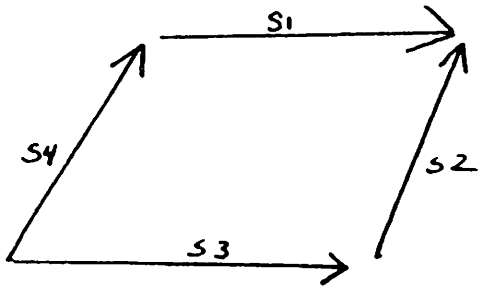
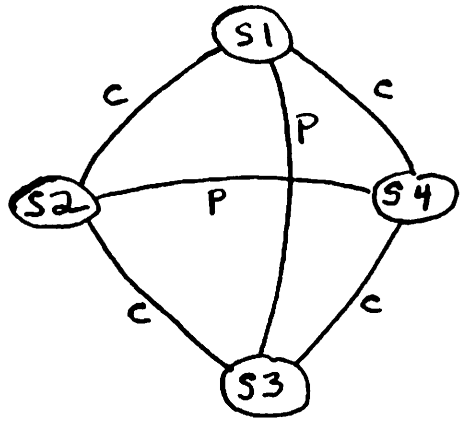

3D MODEL BASE
PREDICTION HIERARCHY COMPILER
PREDICTION HIERARCHY
IMAGE DATA
2D MATCHER
PROMISING MATCHES
VERIFICATION AND POSE REFINEMENT
RECOGNIZED AND POSITIONED OBJECTS

Figure 5: The basic recognition system design.

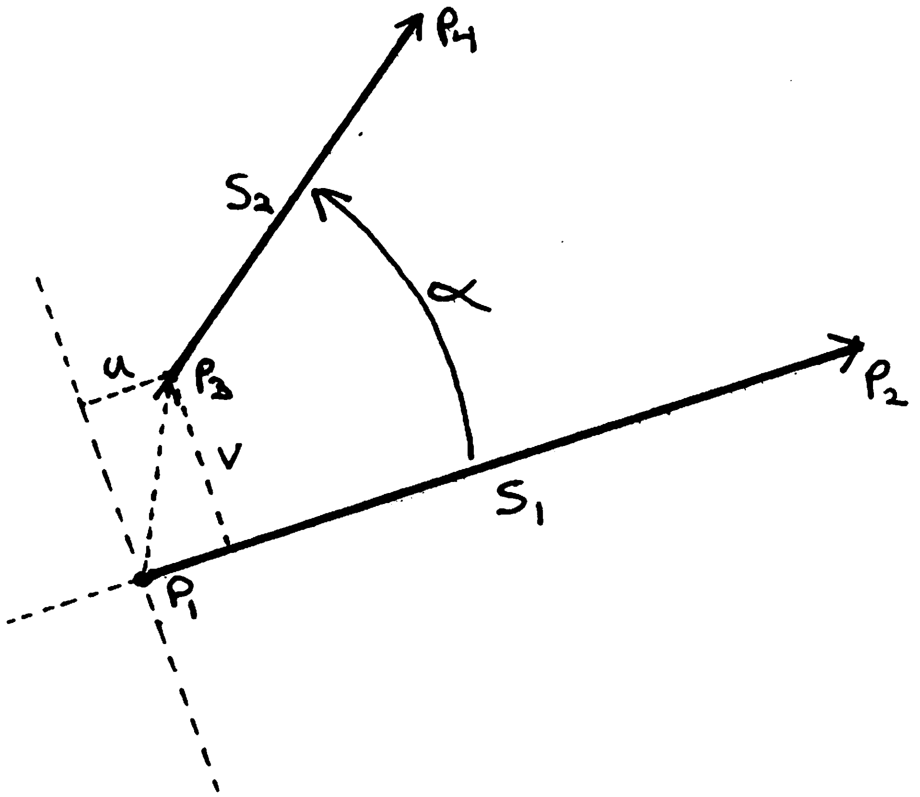


c = coincident endpoints
p = parallel



```
(define parallelogram (s1 s2 s3 s4)
  (and (coincident s1-head s2-head)
       (coincident s2-tail s3-head)
       (coincident s1-tail s4-head)
       (coincident s3-tail s4-tail)
       (parallel s1-tail s3-tail)
       (parallel s2-tail s4-tail)
       )
  )
```

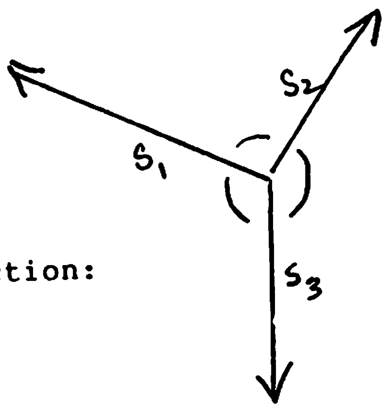
Fig. 6. A parallelogram and its relational graph representation. In the internal representation, the segments have arbitrary but fixed endpoint orderings (head/tail). This is to differentiate the parallelogram from other, distinct structures such as a four-way junction (which are otherwise identical).



```
Relative measurements:
---------------------

   u = displacement of p3 from p1 in direction along s1,
        divided by the length of s1
   v = displacement of p3 from p1 in direction normal to s1,
        divided by the length of s1
   alpha = angle between s1 and s2 measured counter-clockwise from s1
   s = ratio of size, length(s2)/length(s1)


Relation as extent box in measurement space:
-------------------------------------------

  R(s1,s2) = (and (min-u ≤ u ≤ max-u)
                  (min-v ≤ v ≤ max-v)
                  (min-alpha ≤ alpha ≤ max-alpha)
                  (min-s ≤ s ≤ max-s)
                  )
```

Figure 7: Representation of image segment relations as an extent box in space of relative position, orientation and size measurements.

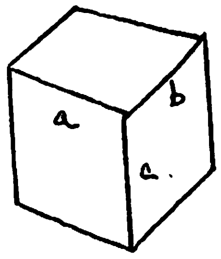
Y-junction:



Pseudo-code description:

```
(Define Y-junction (s1 s2 s3)
    (and (coincident s1-tail s2-tail)
         (coincident s2-tail s3-tail)
         (coincident s3-tail s1-tail)
         (0 ≤ alpha(s1,s2) ≤ 180)
         (0 ≤ alpha(s2,s3) ≤ 180)
         (0 ≤ alpha(s3,s1) ≤ 180)
         )
    )
)
```

Instance of prediction:



```
mapping s1 <- a
        s2 <- b
        s3 <- c
```

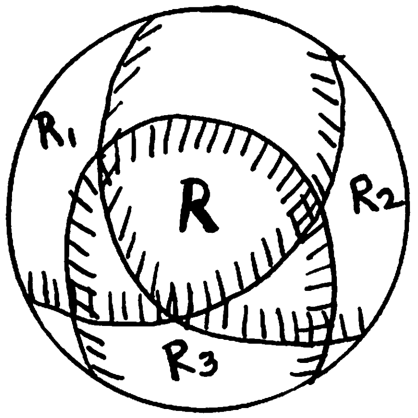


Figure 8: Example of a prediction and its instance. The *Y junction* and an instance of it in the projection of a cube over a set of view angles. The instance's visiblity region (R) is the intersection of the regions of the relations that make it up: (R1) $0 \leq \alpha(s1, s2) \leq \pi$ (R2) $0 \leq \alpha(s2, s3) \leq \pi$ (R3) $0 \leq \alpha(s3, s1) \leq \pi$

```
(define parallelogram (s1 s2 s3 s4)
  (and (coincident s1-head s2-head)
       (coincident s2-tail s3-head)
       (coincident s1-tail s4-head)
       (coincident s3-tail s4-tail)
       (parallel s1-tail s3-tail)
       (parallel s2-tail s4-tail)
       ))
(define triangle (s1 s2 s3)
  (and (coincident s1-head s2-tail)
       (coincident s2-head s3-tail)
       (coincident s3-head s1-tail)
       ))
(define triangular-prism (s1 s2 .. s8)
  (and (parallelogram s1 s2 s3 s4)
       (parallelogram s5 s2 s6 s7)
       (triangle s3 s7 s8)
       ))
(define rhombus (s1 s2 s3 s4)
  (and (parallelogram s1 s2 s3 s4)
       (same-size 1 2)
       ))
```

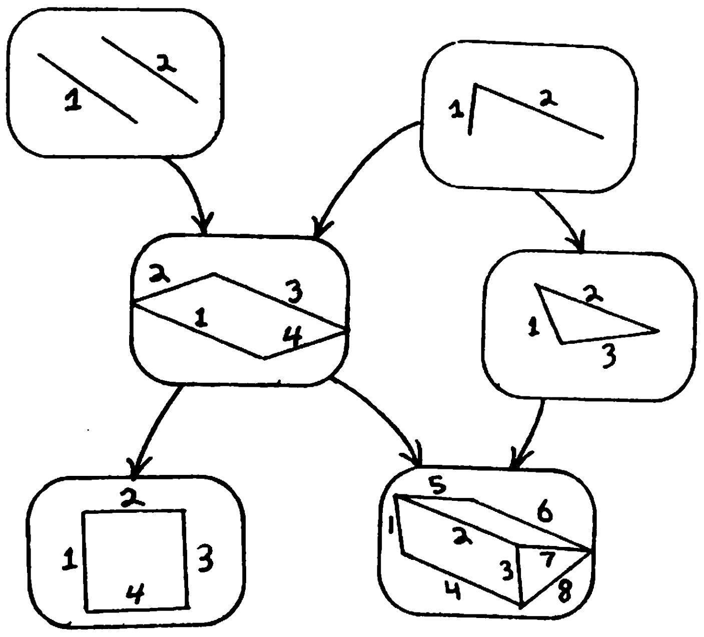
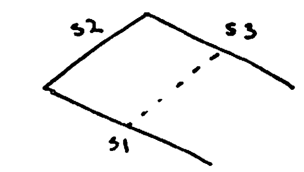
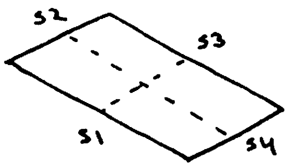


Figure 9: Predictions as specializations or combinations of other predictions.

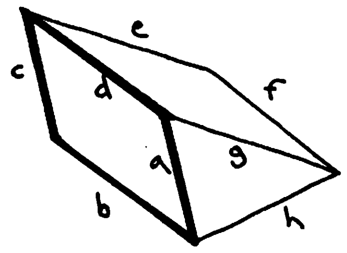


```
(define p (s1 s2 s3)
  (and (coincident s1 s2)
       (coincident s2 s3)
       (parallel s1 s3)
       )
  )
```



```
(define p' (s1 s2 s3 s4)
  (and (p s1 s2 s3)
       (coincident s3 s4)
       (coincident s4 s1)
       (parallel s2 s4)
       )
  )
```

(a) Predictions p and p'.

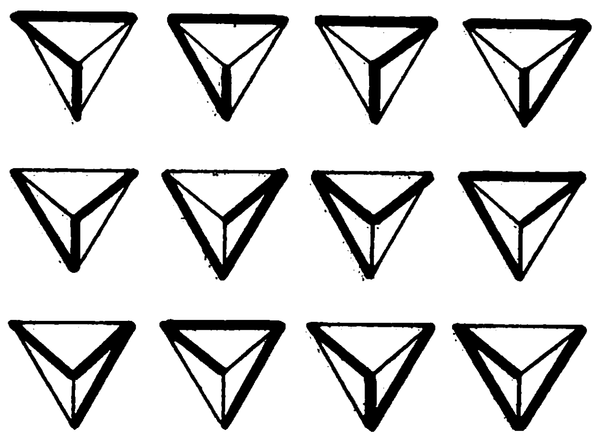


(b) Model m.

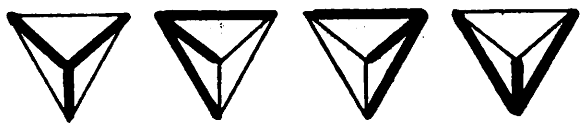
i_p :	$i_{p'}$:
$1 \leftrightarrow b$	$1 \leftrightarrow b$
$2 \leftrightarrow c$	$2 \leftrightarrow c$
$3 \leftrightarrow d$	$3 \leftrightarrow d$
	$4 \leftrightarrow a$

(c) Instances of p and p' on model m.

Figure10: An example of *covering*: (a) a prediction p and one of its possible derivatives, p', (b) a model, m, whose projection satisfies both p and p' and (c) instances of both predictions, i_p and $i_{p'}$, that map the prediction segments to actual segments of m. Note that $i_{p'}$ covers i_p because the model segments it maps to are a superset of those mapped to by i_p.



(a)



(b)

Figure 11: A view of the tetrahedron that contains 12 instances of the *endpoint coincidence* prediction. The instances of two of its derivatives, (a) *snake* and (b) *trihedral junction*, are shown. Both derivatives cover all coincidence instances, but the trihedral junction prediction does so with much fewer instances (4 instead of 12). The snake prediction has three segments connected in a chain that is not closed in a loop; and the trihedral junction has three segments with each pair coincident, using the same endpoints.

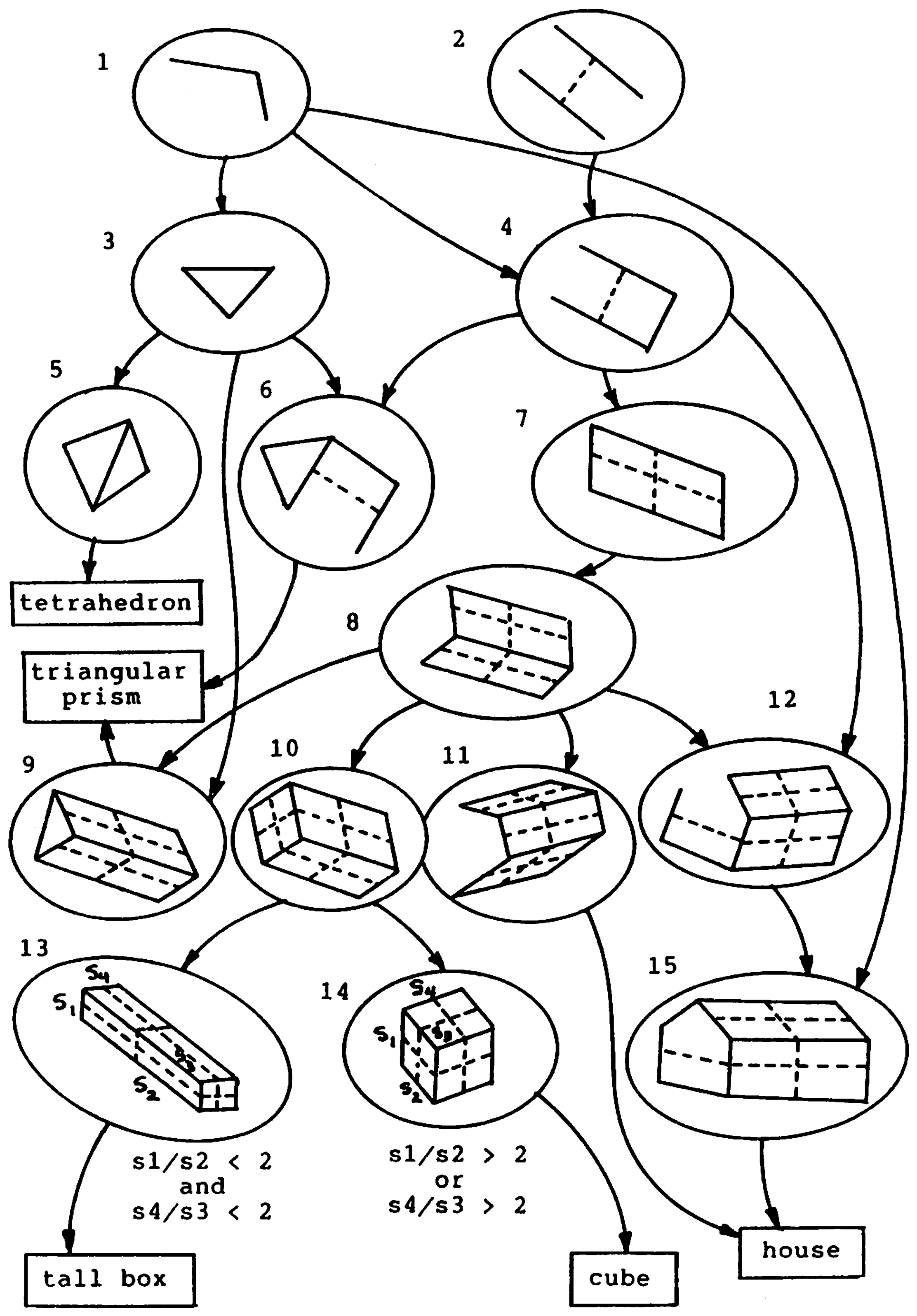


Figure 12: The resulting prediction hierarchy compiled from views of the objects in figure 4. The nodes represent predictions and arrows indicate combination and specialization links. The predictions are represented graphically by segments and dashed lines for parallel relations.

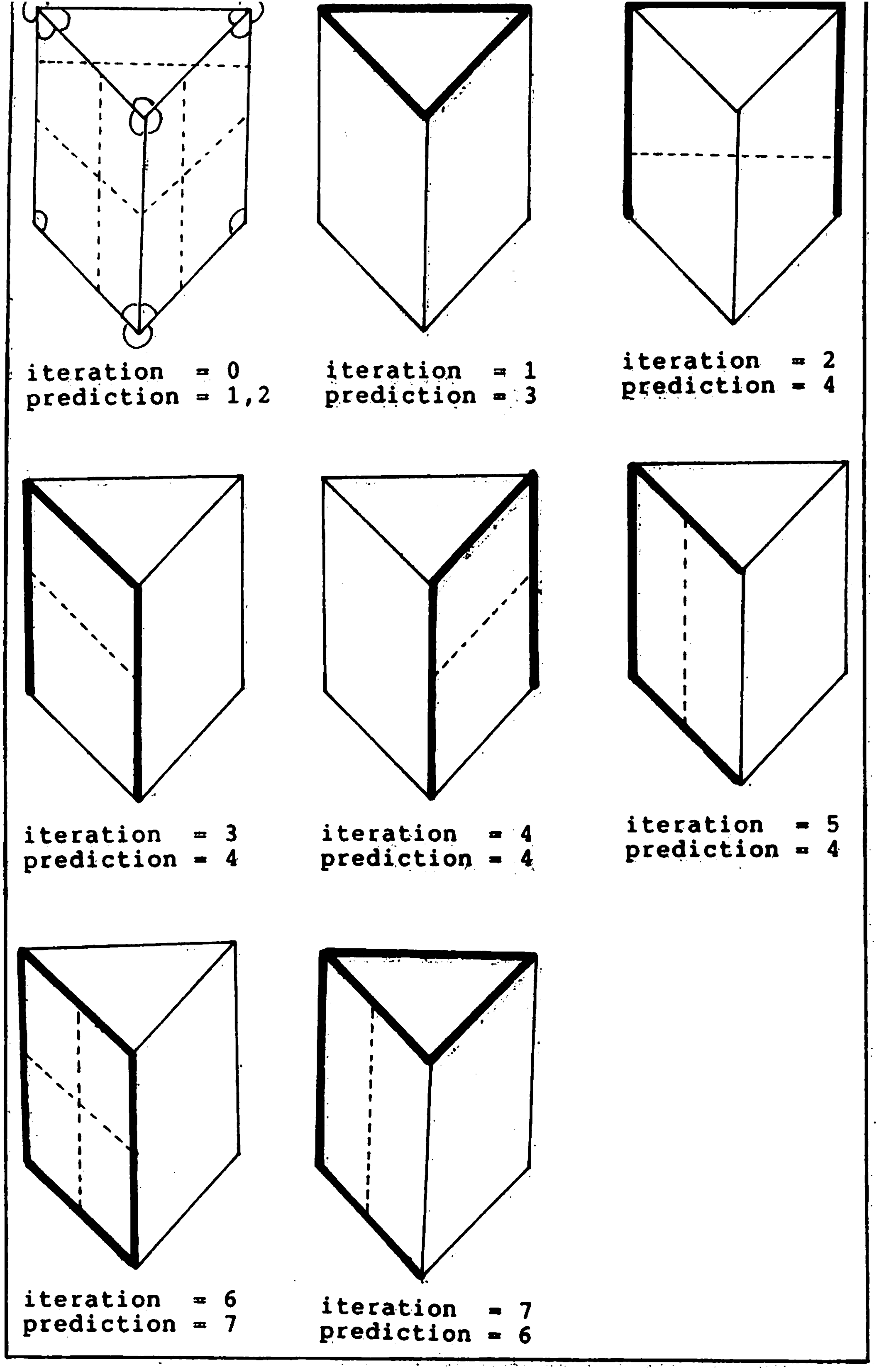


Figure 13: **Example run of the matcher using the prediction hierarchy in Figure 12. The matcher is initialized (iteration=0) by finding all instances of the initial predictions (coincidence and parallel). The matcher then iteratively searches for matches between combinations and specializations of already matched predictions and the image. A prediction unique to the triangular prism object was matched to the image at iteration 7.**