CONTEXT-BASED VISION: RECOGNIZING OBJECTS USING INFORMATION FROM BOTH 2D AND 3D IMAGERY

Thomas M. Strat and Martin A. Fischler
Artificial Intelligence Center
SRI International
333 Ravenswood Avenue
Menlo Park, California 94025

Abstract

This paper describes results from an ongoing project concerned with recognizing objects in complex scene domains, and especially in the domain that includes the natural outdoor world. Traditional machine recognition paradigms assume either (1) that all objects of interest are definable by a relatively small number of explicit shape models, or (2) that all objects of interest have characteristic, locally measurable features. The failure of both assumptions in a complex domain such as the natural outdoor world has a dramatic impact on the form of an acceptable architecture for an object recognition system.

In our work, we make the use of contextual information a central issue, and explicitly design a system to identify and use context as an integral part of recognition. In so doing, we provide a new paradigm for visual recognition that eliminates the traditional dependence on stored geometric models and universal image partitioning algorithms. This paradigm combines the results of many simple procedures that analyze monochrome, color, stereo, or 3D range images. By interpreting their results along with relevant contextual knowledge, a reliable recognition result is achieved, even in the face of imperfect visual procedures. Initial experimentation with the system on ground-level outdoor imagery has already demonstrated competence beyond what we believe is attainable with other existing vision systems[1].

1 Introduction

Much of the progress that has been made to date in machine vision has been based, almost exclusively, on shape comparison and classification employing locally measurable attributes of the imaged objects (e.g., color and texture) [2, 5, 6, 11, 13]. Natural objects viewed

[1]Supported by the Defense Advanced Research Projects Agency under contracts MDA903-86-C-0084, 89F737300, and DACA76-90-C-0021.

under realistic conditions do not have uniform shapes which can be matched against stored prototypes, and their local surface properties are too variable to be unique determiners of identity. The standard machine vision recognition paradigms fail to provide a means for reliably recognizing *any* of the object classes common to the natural outdoor world (e.g., trees, bushes, rocks, and rivers). In this paper and its predecessors [7, 8, 21, 22], we have outlined a new paradigm which explicitly invokes context and stored knowledge to control the complexity of the decision-making processes involved in correctly identifying natural objects and describing natural scenes.

Scene description is properly viewed as a problem in scientific discovery. Ultimately a collection of assertions must be provided, each assertion stating the identity and relevant attributes (e.g., spatial location) of some object depicted (or possibly invisible, but inferred to be present) in an imaged scene. There are two critical differences between the problem domain addressed in this paper and the class of problems capable of being solved by existing machine vision paradigms.

1.1 Hypothesis Generation

The first critical difference concerns hypothesis generation. The acceptance of either of two assumptions trivializes the hypothesis generation problem for conventional machine vision systems. Conventional systems have no effective machinery for hypothesis generation when both assumptions are invalid — in a sense, this failure of both assumptions is one of the main attributes of a complex domain.

The two assumptions are:

1. All objects of interest are defined by a relatively small number of explicit shape models. This makes it computationally feasible to exhaustively search for the presence of these models (via "geometric alignment") as a way of producing a suitable description of some given scene (as in [4] and [13], for example).

2. All objects of interest have characteristic features, homogeneous and locally measurable in an image (e.g., color or texture), which are reliable indicators of the object's identity. This either allows direct determination of the presence of objects using statistical decision theoretic methods (based on classification of the corresponding feature vectors); or permits the successful employment of a "universal" partitioning algorithm which finds regions (homogeneous in the given attributes) in the imagery corresponding to the objects of interest.

The validity of the second assumption allows a single universal procedure (the partitioning algorithm) to find regions in the imagery which are good delineations of the objects of interest. These regions provide an effective basis for generating the required hypotheses for object location and identity [6].

The failure of both assumptions has a dramatic impact on the form of an acceptable architecture/control-structure for an object recognition system facing a complex domain.

Not only are we required to introduce an explicit mechanism for computationally feasible hypothesis generation, but we must provide additional machinery for representing and accessing the supporting information necessary for such hypothesis generation. We are forced to cross the line from what has been called model-based vision to an "AI-complete" problem domain.

Much of our work has addressed devising the representations and control structures (i.e., the introduction of context sets in a production rule type framework) needed to merge the vision-specific and more general AI technologies.

1.2 Hypothesis Evaluation

The second critical difference between the approach we propose and existing machine vision paradigms is their distinct treatment of the scene objects to be recognized. Conventional machine vision paradigms define scene objects to be independent entities which can (and should be) isolated from the rest of the scene and then labeled on the basis of their differences from other objects for which we have names (and models). The system we describe in this paper, called Condor, treats objects as component parts of larger contexts (many different contexts for each object) from which they cannot be separated — like quarks in modern physics, they never appear in isolation and have no independent existence. Once a context is recognized, its individual components may be instantiated and given names.

The need to embed objects in more extensive contexts, rather than treating them as independent entities, is due to the following considerations (in complex domains):

1. The image appearance of an object can be quite variable, not only due to intrinsic shape variability, but also due to viewing conditions (e.g., resolution, occlusion, and lighting). The object's relationship to its surroundings is often a major factor in determining its identity — even for a human.

2. Some objects (such as a river or a bridge) cannot be defined, let alone recognized in an image, independent of their embedding in the surrounding terrain.

The architectural and computational implications of context-based definitions is of equal significance to those caused by the need to provide special machinery for hypothesis generation. Fortunately, much of the needed machinery can serve both functions. Thus, context sets not only define objects, but control the generation of (candidate) hypotheses from either 2D or 3D imagery and the subsequent evaluation of those hypotheses.

Having to deal with contexts, rather than independent objects introduces a major increase in computational complexity. Contexts are much more numerous than the objects they are composed of, and contexts are less precisely defined. The verification problem changes from identifying objects based on sufficient conditions (e.g., of similarity) to that of eliminating alternatives based on failure to satisfy necessary conditions. We are required to deduce much more about the nature of the overall scene — especially its physical structure.

To the extent that the Condor architecture, and its representations and partitioning of knowledge are successful in advancing the state-of-the-art in machine vision for complex natural scenes, we believe that this success will also contribute to increased competence in other non-vision related AI classification tasks in complex domains, but especially to those tasks which require decision making involving both iconic and symbolic information.

2 Background

The term 'recognition' in its most general sense involves associating linguistic labels with scene entities. The vast majority of research on recognition in machine vision relies on the use of a known geometric model of the object being "recognized." Such systems are often intended for use in an industrial setting where one or a small number of parts are to be located within a scene. The goal of these systems is the location and orientation of the objects of interest.

Some research has been directed toward relaxing the strict assumption of a fully specified geometric model. These techniques employ a parameterized model (as in Acronym [5]), or a generic model (as in Fua [9]). While much less restrictive in scope, these techniques all rely on shape as the primary attribute for recognition.

A third category of recognition research involves no dependency on stored geometric models. Recognition is attempted on the basis of cues besides shape, such as size, location, appearance, purpose, and context. Hawkeye [3], MSYS [2], and the approach described here, are examples of the few systems that have been designed without a primary reliance on geometric models.

For the natural world, precise geometric models of natural objects are not available, and existing techniques offer little insight on how to recognize natural scenes. There has been some work directed toward the goal of semantic understanding of natural outdoor scenes, but surprisingly, very little new work has been initiated in the last ten years [2, 11, 16, 17, 18, 23, 25]. All of these approaches begin by partitioning the image into regions, which presumably mirrors a "natural" decomposition of the scene into "objects." The regions are then analyzed in one way or another to determine their interrelationships, to merge them into larger regions, and ultimately, to assign each region a label that categorizes it semantically. This basic reliance on an initial "universal" partitioning is a critical weakness that we avoid in the approach offered in this paper.

3 Conceptual Architecture

The conceptual architecture of the system we describe, called Condor (for context-driven object recognition), is depicted in Figure 1. The input to the system is an image or set of images that may include intensity, range, color, or other data modalities. The primary output of the system is a labeled 3D model of the scene. The labels included in the output

Figure 1: Conceptual architecture of Condor

description denote object *classes* that the system has been tasked to recognize, plus others from the recognition vocabulary that happen to be found useful during the recognition process. An object *class* is a category of scene features such as

{sky, ground, geometric-horizon, skyline, foliage, bush, tree-trunk, tree-crown, tree, trail, ... }

A central component of the architecture is a special-purpose knowledge/database used for storing and providing access to knowledge about the visual world, as well as tentative conclusions derived during operation of the system. In Condor, these capabilities are provided by the Core Knowledge Structure (CKS) [20].

The conceptual architecture is much like that of a production system; there are many computational processes interacting through a shared data structure. Interpretation of an image involves the following four process types.

- Candidate generation (hypothesis generation)
- Candidate comparison (hypothesis evaluation)
- Clique formation (grouping mutually consistent hypotheses)
- Clique selection (selection of a "best" description)

Each process acts like a daemon, watching over the knowledge base and invoking itself when its contextual requirements are satisfied. All processing occurs asynchronously and each process is assumed to have access to sufficient computational resources. All processes have access to the entire knowledge base, but each type of process will only store the kind of information shown in the diagram (Figure 1).

3.1 Context sets

The invocation of all processing operations in Condor is governed by context through the use of *context sets:* an action is initiated only when one or more of its controlling context sets is satisfied. Thus, the actual sequence of computations, and the labeling decisions that are made, are dictated by contextual information (stored in the Core Knowledge Structure), by the computational state of the system, and by the image data available for interpretation.

A *context set* is a collection of context elements that are sufficient for inferring some relation or carrying out some operation on an image. Syntactically, a context set is embedded in a rule denoted by

$$L : \{CE_1, CE_2, \cdots, CE_n\} \Longrightarrow A$$

where L is the name of the class associated with the context set, A is an action to be performed, and the CE_i comprise a set of conditions that define a context. For convenience, we often refer to the entire rule as a context set.

Example: The context set {SKY-IS-CLEAR, CAMERA-IS-HORIZONTAL, RGB-IS-AVAILABLE} defines a set of conditions under which it is appropriate to use the operator BLUE-REGIONS to delineate candidate sky hypotheses.

There is a collection of context sets for every class in the recognition vocabulary. In theory, Condor performs the actions A that are associated with every *satisfied* context set.

A context set is satisfied only when the known context is sufficient to establish the truth of all its elements. Often it will not be possible to establish whether a context-set element is true or false, in which case the element is considered to be unsatisfied.

Visual interpretation knowledge is encoded in context sets, which serve as the uniform knowledge representation scheme used throughout the system. Context sets are employed in three varieties of rules.

- Type I: Candidate generation
- Type II: Candidate evaluation
- Type III: Consistency determination

Context sets of each type are constructed for each class in the recognition vocabulary. The most difficult part of building any AI system is encoding the knowledge that drives the system. Constructing context sets in Condor is tantamount to knowledge base construction and remains a critical task requiring a solid understanding of the limitations and applicability conditions of potential image understanding routines. Condor has been designed with this in mind, and offers several features that facilitate this process.

First, the construction task is eased somewhat by the separation of the knowledge base according to classes. Therefore, when constructing context sets for class L, the only other classes that must be considered are those that are immediately relevant for recognizing instances of class L.

Second, context sets need only define sufficient conditions for applying the associated operation — they need not attempt to define the full boundary of applicability. Thus, one can be quite conservative when constructing context sets, only encoding knowledge that is clearly relevant, and ignoring that which may be dubious.

Third, although it is desirable that the context sets and their associated operations be as infallible as possible, they need not be perfect. The entire architecture of Condor has been designed to achieve reliable recognition results, even in the presence of unreliable operators, imperfect evaluators, and faulty decision-makers. This is achieved primarily through the use of large numbers of redundant operations in every stage of processing, so that a single mistake is unlikely to affect the final interpretation.

Finally, some form of learning is essential if a large system with a broad range of competence is to be constructed. We have proposed a mechanism whereby context sets can be modified automatically, using the experiences of the system to refine the knowledge base incrementally [22]. The collection of context sets can be allowed to evolve, with or without human intervention.

3.2 Hypothesis generation

The customary approach to recognition in machine vision is to design an analysis technique that is reliable in as many contexts as possible. In contrast to this tendency toward large, monolithic procedures, the strategy embodied in Condor is to make use of a large number of relatively simple procedures. Each procedure is competent only in some restricted context, but collectively, these procedures offer the potential to recognize a feature in a wide range of contexts. The key to making this strategy work is to use contextual information to predict which procedures are likely to yield desirable results, and which are not.

While it may be extremely difficult to write a recognition procedure that is competent across many different contexts, it is often quite easy to devise a procedure that works well in some specific context. For example, finding foliage that is silhouetted against the sky is far simpler than finding foliage in general. Similarly, finding foliage in an environment where only a single species of tree occurs, is easier than finding foliage associated with any kind of tree. By assembling a collection of such context-specific procedures, it has been possible to recognize foliage in many different situations under a wide variety of conditions.

A collection of recognition procedures is associated with each class in the recognition vocabulary. Of course, no procedure, not even one applied in very restricted contexts, will be sufficiently reliable that its results can be accepted with confidence. Accordingly, the output of each procedure is treated as a candidate hypothesis.

A *candidate hypothesis* is any image feature that is potentially an instance of some specified class. In most of our examples, an image region is associated with each candidate, but in general, a candidate is any hypothesis that asserts the presence of some object in the 3D scene depicted in the image being analyzed. Candidates are generated by specialized operators using both intensity and range data and every candidate is associated with the class of which it is potentially an instance.

Figure 2: Schematic diagram of data dependency in Condor. A separate organization of information is required for each clique being formed.

A large portion of the Condor architecture is devoted to sorting out the better candidate hypotheses from the poorer ones. Figure 2 shows the generation and subsequent processing of candidates throughout the system.

The invocation of recognition procedures is governed by candidate generation context sets which define the conditions under which it is sensible to employ each recognition procedure.

Example: Horizontal surface patches are likely to be part of the ground, but they can only be computed when range data is available. The context set in the following rule controls the invocation of this operator.

GROUND : {CLIQUE-IS-EMPTY, DENSE-RANGE-IS-AVAILABLE }
$\Longrightarrow$ HORIZONTAL-SURFACE-PATCHES

The elements in a candidate generator context set encode the assumptions that were made when the associated operator was written. This formalism ensures that each operator will only be employed in circumstances in which it can reasonably be expected to succeed. The context set not only identifies an applicable procedure, but also supplies the information to establish intelligently the inevitable parameters (such as a threshold or a window-size) associated with that operator.

Obviously, context sets can be very specific, very generic, or anywhere in between. It is intended that candidate generator context sets be provided that span this range. One encodes very specific context sets for operators that work well only in very special circumstances, presumably a context that has some special significance to the larger goals of the embedded system. Generic operators that provide reasonable performance over a broad range of contexts, are employed when the more competent specialized procedures are not applicable. Generally, the more candidate generator context sets that are provided, the more operators that will be applicable in any given context. Ideally, there will always be multiple operators invoked so that the system need never rely on a single routine.

It should be clear that it is possible to make use of large, carefully constructed procedures when they exist. Thus, if one has already expended a great deal of effort tuning a large, monolithic recognition procedure, it can be incorporated into Condor alongside any other operators that might also exist.

The interaction of context sets across classes is of interest. The context elements in one context set may refer to the existence of other labeled entities. For example, a tree trunk candidate generation routine may require knowledge of the ground surface as part of its context. Whenever a need for recognition of other classes is detected, Condor adds that class to its list of labels that are actively being recognized. In this way, when Condor is tasked to recognize a specific class from its target vocabulary, it will automatically seek to instantiate other classes from its recognition vocabulary that are relevant.

3.3 Clique formation

The result of the candidate generation process is a collection of candidates for each label in the active recognition vocabulary. Because the operators cannot be expected to be sufficiently robust, extra steps must be taken to find those candidates that truly are instances of their associated classes.

To obtain this increase in reliability, we make use of a principle of maximal coherency which holds that the best interpretation of an image is the one which coherently explains the greatest portion of a scene. Candidates that are not consistent with a partial image interpretation cannot be part of the final interpretation. The goal, similar in spirit to that employed by McKeown in SPAM [15], is to find a mutually consistent set of candidates that explains as much of the image as possible.

A set of mutually consistent candidate hypotheses, called a *clique,* represents a possible interpretation of the image. Condor builds a number of cliques and chooses the "best" one as its final interpretation. Naturally, it would be computationally infeasible to generate all possible cliques — instead, cliques are generated in a special order (described in the next section) to increase the likelihood that a good interpretation is found early. Thus, the longer that Condor analyses an image, the better its interpretation is likely to be.

Inconsistency is determined by context-specific procedures whose application is mediated by context sets (see also Figure 2).

Example: A candidate for ground cannot extend above the skyline.
GROUND : { CLIQUE-CONTAINS(skyline) } $\Longrightarrow$ PARTIALLY-ABOVE-SKYLINE

As was the case with candidate generation, the inconsistency determination routines are assembled into context sets that encode the assumptions necessary for their successful application. Each operator tests a candidate for consistency with all the incumbents already present in a clique. If any satisfied context set finds a candidate to be inconsistent, then it is not admitted into that clique, although it may participate in other cliques. Thus, consistency determination context sets provide necessary (but not sufficient) conditions for clique inclusion.

A clique contains a collection of candidates annotated with inferred 3D properties and relations. The inconsistency operators encode geometric and physical relationships that must be consistent with known facts about the environment and the various semantic classes. The operators may involve either 2D image-plane computations or such 3D constraints as size, support, orientation, occupancy of solid objects, etc. The 2D constraints are useful for rapidly eliminating some candidates when they are easily seen to be inconsistent, or when sufficient 3D information cannot be established to allow more sophisticated spatial reasoning procedures to be applied. The consistency determination context sets include context elements that specify what 3D information must be known. Their use causes an attempt to infer that information if it is not already known.

3.4 Candidate comparison

The search for the largest coherent set of candidates can be combinatorially infeasible without further constraint — the number of potential cliques is exponential in the number of candidates. For this reason, cliques are generated in a special order.

At any point during the processing of an image, there will be a collection of candidates for each label to be instantiated. Some of these candidates are obviously better examples of the class denoted by the label than are others. By first building cliques from the best candidates of each class, we are much more likely to encounter good cliques early in the search (typically several within the first half-dozen cliques). Condor has used this *best-first* strategy to successfully avoid the combinatorics that would otherwise prevent recognition.

The task here is to order the candidates within each class so the better ones may be added to cliques before the others. The difficulty is choosing a suitable metric to accomplish this ordering. For most classes of interest in the outdoor world, there is no single evaluation metric that gives a reliable ordering. One could conceive of multiple metrics that evaluate the candidates along various dimensions, but that would still leave the problem of comparing multi-dimensional evaluation vectors. In order to justify a weighted sum of the vector components, one would have to make the unlikely assumption of some form of independence. A similar independence assumption would be required if the evaluation measures were to be given a probabilistic interpretation and combined using probability theory.

The solution we have adopted is to make use of multiple evaluators, but not to assume that they are independent in any way. Instead, they are used to compare two candidates for a given label, with each evaluator casting a vote for the candidate it ranks higher. If all evaluators favor one candidate over another, a preference ordering between the candidates is established. Otherwise, no ordering is imposed. The net effect of comparing (pairwise) all candidates for a given label is to impose a partial order on those candidates. The candidates at the tops of the partial orders are tested for consistency with the cliques before those below them.

An *evaluator* is a function that scores the relative likelihood that a candidate for a class is actually an instance of that class. The evaluators that apply in any context are described by *candidate evaluation context sets.*

Example: When viewed obliquely, the ground usually exhibits a horizontally striated texture. HORIZONTALLY-STRIATED is a function that measures this property within a candidate region.

GROUND : {CAMERA-IS-HORIZONTAL} $\Longrightarrow$ HORIZONTALLY-STRIATED

As before, the context sets allow the relevant knowledge to be subdivided into manageable pieces. The elements of each context set encode the conditions under which a relatively simple-minded evaluation function gives meaningful information. It is intended that many evaluation functions be provided with context sets, so that robust comparisons result whenever a unanimous vote occurs. One candidate is preferred over another only when all evaluators occurring in satisfied context sets score it as least as high as the other candidate.

As always, context-set elements that refer to other object classes cause other computations to be triggered. Satisfied context-set elements also provide information for setting parameters that may be required by the associated evaluation functions.

The structure of the comparisons is noteworthy because it contrasts with the way comparisons are performed in nearly every other recognition system. The usual approach is to partition an image and to consider which of several potential class labels is the best description of a region. In Condor, we start with several partitions (candidates) and consider which of several candidates is the most likely instance of a class. For example, a conventional recognition system would consider whether a particular region was more likely to be a tree trunk or a road or a bush. Condor would have several potential delineations of a tree trunk and would consider which is the best description of the trunk.

This departs from conventional approaches in two significant ways. First, comparing candidate regions for a given label requires knowledge of the semantics of that label only, whereas the customary approach of comparing two labels for a given region requires knowledge of the relationships between many semantic categories. When considering which candidate is the best tree trunk, Condor needs to know only about tree trunks and related categories (such as branches, roots, and the ground). In contrast, to decide what label to assign to a given region using a conventional approach, one must be able to compare any pair of labels. This requires knowledge of the relationships between every pair of semantic categories, and grows rapidly as new classes are added to the recognition vocabulary. The Condor orientation provides a basis for believing that sufficient knowledge might eventually be encoded in the system to allow robust comparison even in a large-scale system.

Second, we enforce the condition that the comparisons lead to a preference only if one candidate is clearly a better choice than another. With this conservative approach, we can reap additional computational savings by pruning large portions of the search for maximally consistent cliques. For example, if candidate C_1 is clearly a better instance of class L than candidate C_2 in the context of a particular clique, and C_1 is found to be inconsistent with that clique, then C_2 can be eliminated as a potential member of that clique as well. Ruling out C_2 may eliminate other candidates recursively. Thus we avoid the need to test the consistency of C_2 and any of its inferiors. Furthermore, it may at times be impossible otherwise to establish C_2 as inconsistent, in which case this pruning step prevents the clique from being contaminated with a bad candidate. Although it does not follow logically that C_2 cannot be a class L instance (i.e., a less likely candidate may indeed be consistent), its elimination is a powerful heuristic that is nearly always warranted by the computational savings that results. We can afford to chance the elimination of a valid candidate because we simultaneously generate additional cliques that may happen to avoid repeating an unjustified elimination. Thus even when some generators yield unreliable candidates, and the comparisons make occasional mistakes, it may still be possible to build a clique that yields a completely accurate semantic labeling of an image.

3.5 The recognition process

Let us summarize the processing steps that have been described so far (Figure 2). For each label in the active recognition vocabulary, all candidate generation context sets are evaluated. The operators associated with those that are satisfied are executed, producing candidates for each class. Candidate comparison context sets that are satisfied are then used to evaluate each candidate for a given class, and if all such evaluators prefer one candidate over another, a preference ordering is established between them. These preference relations are assembled to form partial orders over the candidates, one partial order for each class. Next, a search for mutually coherent sets of candidates is conducted by incrementally building cliques of consistent candidates, beginning with empty cliques. A candidate is nominated for inclusion into a clique by choosing one of the candidates at the top of one of the partial orders. Consistency determination context rules are used to test the consistency of a nominee with candidates already in the clique. A consistent nominee is added to the clique; an inconsistent one is removed from further consideration with that clique. Further candidates are added to the cliques until none remain. Additional cliques are generated in a similar fashion as computational resources permit. Ultimately, one clique is selected as the best semantic labeling of the image on the basis of the portion of the image that is explained and the reliability of the operators that contributed to the clique.

Each of the processing steps occurs simultaneously in our conceptual view, but there are some implicit sequencing constraints. Candidate evaluators begin to construct partial orders as soon as candidates become available. Incremental addition of candidates to cliques begins as soon as partial orders are available. Theoretically, there is no need to wait for one stage to complete before latter stages are begun, but it may be desirable when computational resources are limited.

The interaction among context sets is significant. The addition of a candidate to a clique may provide context that could trigger a previously unsatisfied context set to generate new candidates or establish new preference orderings. For example, once one bush has been recognized, it is a good idea to look specifically for similar bushes in the image. A candidate generation context set that includes an element that is satisfied only when a bush is in a clique implements this tactic.

Similarly, as cliques evolve, the partial orders for each class may change. Ideally, one should wait for all candidate generation and comparison activity to subside before nominating a candidate into a clique. We regard this synchronization as an implementation issue that is best resolved by tailoring the strategy to the particular computing architecture employed. Given sufficient computational resources, all synchronization strategies will attain the same interpretation.

It is important to remember that multiple cliques will be in various stages of construction simultaneously. Each clique has its own partial orders from which to choose, although many candidates will be identical in several or all of the cliques. Context set satisfaction is determined individually for each clique.

4 Implementation of Condor

4.1 Processing Sequence

All of the computations carried out by Condor are controlled by context sets. At any given time, there might be many satisfied context sets whose operators could be invoked. Condor as implemented evaluates context sets in an order that is designed to provide additional information rapidly. For example, it is sensible to build all partial orders as completely as possible before starting to build cliques, although this is not required by the conceptual architecture. Although the context sets are evaluated in a fixed order, their satisfaction depends on the context so far derived. Thus, the order in which operators are invoked depends primarily on the contextual information. The order of context set evaluation we have chosen serves mainly to accelerate the interpretation of images.

The sequence of operation in Condor is summarized in Figure 3. The serialization of an inherently parallel architecture is complicated by the interdependencies among the processing

Figure 3: Sequence of computation

steps. When first presented with an image and tasked to recognize a target vocabulary, Condor generates candidates and compares them to impose a partial order on the candidates in the target vocabulary. Any additional classes that are found to be of use are added to the active recognition vocabulary and are processed similarly. Next, a candidate from the top of one of the partial orders is added to a clique. This changes the context relevant to that clique, so the candidate generation process is repeated and the partial orders are reevaluated in that new context. A comprehensive caching mechanism[2] is employed to prevent reevaluating any operations that have not changed. A new nominee is chosen from the tops of the partial orders and checked for consistency with the clique. If it is found to be consistent, it is added to the clique and removed from its partial order. If inconsistent, it is removed from further consideration for membership in that clique, although it may join another clique later. The inconsistent nominee is removed from its partial order along with any candidate over which it is preferred. This cycle is repeated until no candidates remain for nomination, thus completing the development of the first clique.

Additional cliques are generated by iterating the entire process. Any operations that occurred before construction of the first clique began need not be repeated since their context

[2]The caching mechanism associates a result with an operation and all its parameters. When the same result is needed for constructing another clique, the value is retrieved rather than recomputed.

is still valid. To accomplish this, the system is rewound to that point and construction of the second clique begins. Condor generates different cliques by nominating candidates in different orders. Many strategies exist for selecting different orders and the heuristic nominating function can be modified to implement them. The strategy that Condor standardly uses is to seed each clique with a candidate that had been ruled out by an earlier clique, thereby guaranteeing that a new and different clique will result.

After each clique is completed, it is compared with the best previous clique to determine which interpretation of the image is better. There is no theoretically sound way of comparing two cliques, and the method we employ is somewhat ad hoc. Each clique is scored on the basis of the portion of the image that is explained and the reliability of the operators that generated the candidates in the clique. The higher scoring clique is retained and additional cliques are generated until a scoring threshold is exceeded or available computation time is exhausted. At that point, the highest scoring clique is accepted as the best interpretation of the image, and the candidates it contains are considered to have been recognized.

The contents of this best clique are then used to update the 3D model of the environment. Newly found objects are inserted in the Core Knowledge Structure. Candidates depicting previously known objects are used to update the location, size, shape, and appearance of that object in the CKS. The name of the operator that successfully delineated each object in the image is stored with the object so that it might be invoked again when that object next comes in the field of view. The result is an updated model of the visual world, that will provide more context for the recognition of objects in subsequent images.

4.2 Representation of context

Because Condor has been designed to make use of a persistent store of information about the visual world, it is necessary to provide a mechanism for its representation. Condor requires access to scene objects based on their location and various semantic properties. This role is filled by the Core Knowledge Structure.

The CKS is an object-oriented knowledge/database that was originally designed to serve as the central information manager for a perceptual system [19, 20]. The following four facilities of the CKS are of particular importance for Condor.

4.2.1 Multiple Resolution

The CKS employs a multiresolution octree to locate objects only as precisely as warranted by the data. Similarly, a collection of geometric modeling primitives are available to represent objects at an appropriate level of detail. In parallel with the octree for spatial resolution is a semantic network that represents things at multiple levels of semantic resolution. Condor's recognition vocabulary is represented as nodes in the semantic network, which allows the system to refer to objects at an appropriate level in the abstraction hierarchy.

4.2.2 Inheritance and inference

The CKS uses the semantic network to perform some limited types of inference that ease the burden of querying the data store. Thus, query responses are assembled not only from those objects that syntactically match the query, but also from objects that can be inferred to match given the relations encoded in the semantic network. For example, the CKS can be queried for all trees within 10 meters of any dirt road, and will find all such trees regardless of whether they were originally categorized as oaks or pines or whether any roadway was present when they were instantiated in the database. Spatial inference is provided based on geometric constraints computed by the octree manipulation routines. Inheritance of attributes that are unspecified is performed in a similar fashion. For example, a query for all objects taller than 5 meters will be satisfied by all trees not specifically known to be shorter than 5 meters, but not satisfied by any rocks (unless they are known to be higher than 5 meters).

4.2.3 Conflicting data

One of the realities of analyzing imagery of the real world is that conflicts will result from mistakes in interpretation and from unnoticed changes in the world. The database used by Condor must not collapse when conflicting information is stored. The CKS treats all incoming data as the opinions of the data sources, so logical inconsistencies will not corrupt the database. Similarly, values derived through multiple inheritance paths are treated as multiple opinions. This strategy has several advantages and disadvantages. Rather than fusing information as it arises, the CKS has the option of postponing combination until its results are needed. This means that the fusion can be performed on the basis of additional information that may become available, and in a manner that depends on the immediate task at hand. Some information may never be needed, in which case the CKS may forego its combination entirely. The disadvantages are the need to store a larger quantity of data and a slowed response at retrieval time. For an object recognition system like Condor, the CKS seems to provide the right tradeoff.

Condor uses the multiple opinion facility to store the attributes of recognized objects. Each attribute value is annotated with the image in which it was identified, its time of acquisition, and time of recognition. In so doing, it is possible to reason about the validity of the stored data, and to react accordingly. The opinion mechanism is also used to store multiple cliques in Condor. Each candidate is stored in the CKS as the opinion of the clique to which it pertains.

4.2.4 User interface

Although Condor is designed to be a fully automated recognition system, a comprehensive user interface is invaluable for development and debugging. The CKS provides a menu-driven query mechanism that is useful for inspecting the intermediate states of computation. In addition, the CKS has been integrated with SRI's Cartographic Modeling Environment [12] to provide a capability of generating synthetic views of terrain. This allows one to visualize the contents of the database from an arbitrary viewpoint by rendering a synthetic image. Doing so provides a window into the information that Condor is assuming as it interprets an image.

4.3 Context set construction

Context sets are the key to any recognition abilities that Condor demonstrates. While we have not yet evolved a precise procedure for designing context sets, we can provide some insight based on our experience in building context sets for natural object recognition.

Type I context sets (candidate generation) are constructed based on an assessment of what operators may work for each label in the recognition vocabulary. Based on a representative sample of imagery from the target domain, we composed image processing operations that work reasonably well in various circumstances from either intensity or range data. Factors that influenced the choice of which operators to include were the likelihood of success, the ease of implementation, the lack of any alternative operators, and the availability of existing code. Table 1 lists the types of operators that are actually employed by Condor to generate candidates (for the experimentation site in the foothills near Stanford University). For each operator, the assumptions that it requires are encoded as context elements in a context set that controls the invocation of the operator. These context elements limit the situations in which the operator will be applied, ensure the existence of any required data, and establish the parameter settings associated with the operator.

Algorithm	Explanation
ASSOCIATION	Finds connected sets of pixels in a binary image
STRIATIONS	Finds the orientation and strength of local texture
DELINEATION	Finds line-like structure
OUTLINING	Finds the boundary of a region
THRESHOLDING	Uses scale-space techniques to choose thresholds
EDGE FINDING	Any of several well-known edge-finding routines
CONTRAST enhancement	Stretches the histogram of an image
SMOOTHING	Low-pass filter
HISTOGRAMMING	Computes a histogram and associated statistics
TEXTURE	Any of several well-known algorithms for measuring texture
SEGMENTATION	Completely partitions an image using KNIFE [14]
DENSE STEREO	Computes a dense depth image using CYCLOPS [1]
SPARSE STEREO	Computes depths at some easily correlated points [10]
HOMOGENEITY	A noise tolerant algorithm for measuring local homogeneity

Table 1: Candidate generation operators

Type II context sets (candidate evaluation) are assembled from evaluation metrics that can be used to compare two candidates. Context elements that define the conditions under which the metrics are meaningful are collected into context sets for each label in the recognition vocabulary. The metrics themselves need not order candidates perfectly, but should perform substantially better than a random ordering. Condor requires a unanimous vote of all applicable metrics before ordering two candidates, so a faulty metric is likely to leave some candidates unordered but not reverse ordered. It is important that preferences be correct when they are made. Non-preferences will require more cliques to be searched but will not lead to incomplete recognition results. Table 2 shows some of the evaluation metrics that are used by Condor.

Evaluation metric	Explanation
ABOVE-GEOMETRIC-HORIZON	Raised objects are more likely found above the horizon
ABOVE-SKYLINE	Raised objects above the skyline are preferred
BELOW-GEOMETRIC-HORIZON	Prefer ground candidates below the horizon
BELOW-SKYLINE	Prefer ground candidates below the skyline
BLUE	Prefer blue sky candidates on a sunny day
BRIGHT	Prefer bright sky candidates
ELLIPSOIDAL	When range data is available, prefer ellipsoidal bushes and tree-crowns
ELLIPTIC	Prefer bushes and tree-crowns that are shaped like ellipses (in 2D)
GREEN	Prefer green grass in the winter and spring in California
HIGHLY-TEXTURED	Prefer foliage candidates that are highly textured
HORIZONTAL	Prefer ground candidates that are horizontal (in 3D)
HORIZONTALLY-STRIATED	Prefer ground candidates that exhibit horizontal striations
NEAR-TOP	Prefer sky candidates that are near the top of the image
NO-SKY-BELOW	Prefer bush and rock candidates that are not above the sky
REASONABLE-SIZE	Prefer trees and bushes that are sized appropriately
SIMILAR-COLOR	Prefer candidates that are similar in color to known objects
SIMILAR-TEXTURE	Prefer candidates that have similar texture as a known object
UNDEFINED-RANGE	Prefer sky candidates that is uncorrelated in stereo
2D-VERTICALITY	Prefer tree trunks that are approximately vertical in the image
3D-VERTICALITY	When range is available, prefer tree trunks that are vertical

Table 2: Evaluation metrics

Type III context sets (consistency determination) define the conditions under which inconsistency of a candidate with a clique can be established. Any constraints that make it impossible for a candidate hypothesis to be valid given the assumption that the candidates already in the clique are correct, are encoded and assembled into Type III context sets. It is important that inconsistent candidates be correctly identified so that physically impossible cliques are not constructed. However, it is not necessary that a complete definition of consistent candidates be encoded. This asymmetry was designed specifically because it is far simpler to specify what could not be a tree, for example, than it is to specify what is a tree. Some of the consistency determination constraints that are used by Condor are listed in Table 3.

Consistency constraint	Explanation
ABOVE-SKY-REGION	Most objects must not be completely off the ground
LEANING	Objects that lean too much are unsupported
MISMATCHED-BRIGHTNESS	The intensity of sky, for example, cannot vary too much
NOT-SUPPORTED-BY-GROUND	Most plants must be rooted in the ground
OVERLAPS-IN-IMAGE	Some hypotheses that are inconsistent in 2D are ruled out
PARTIALLY-ABOVE-SKYLINE	The ground cannot extend above the skyline
PARTIALLY-BELOW-GEOMETRIC-HORIZON	The sky cannot extend below the horizon

Table 3: Consistency constraints

5 Example of natural object recognition

Figure 4: A typical image from the Stanford foothills

To illustrate the basic processing sequence, Condor was tasked to recognize the sky, the ground, and the foliage, appearing in the image shown in Figure 4. This relatively easy image was acquired in the foothills behind the Stanford University campus in the afternoon of a sunny day using an ordinary 35mm camera. To make the description as clear as possible, some of the machinery incorporated in Condor has been deactivated while creating this example. In particular, no prior knowledge of the terrain or features on that terrain is used.

5.1 Candidate generation

Condor begins by generating candidates for each of the classes in the target vocabulary. The relevant candidate generation context sets are shown in Table 4. Tables 5 and 6 show the relevant Type II and Type III context sets used in this example.

#	Class	Context elements	Operator
1	SKY	CLIQUE-IS-EMPTY	SEGMENTATION-REGIONS
2	SKY	CLIQUE-IS-EMPTY	WEAKLY-TEXTURED-REGIONS
3	SKY	CLIQUE-IS-EMPTY	WEAKLY-STRIATED-REGIONS
4	SKY	CLIQUE-IS-EMPTY	BRIGHT-REGIONS
5	SKY	CLIQUE-IS-EMPTY $\wedge$ SKY-IS-CLEAR $\wedge$ RGB-IS-AVAILABLE	BLUE-REGIONS
6	SKY	LAST-SELECTED-CANDIDATE-IS(sky)	SIMILAR-REGIONS
7	GROUND	CLIQUE-IS-EMPTY	SEGMENTATION-REGIONS
8	GROUND	CLIQUE-IS-EMPTY $\wedge$ CAMERA-IS-HORIZONTAL	HORIZONTAL-STRIATION-REGIONS
9	GROUND	CLIQUE-IS-EMPTY $\wedge$ DENSE-RANGE-IS-AVAILABLE	HORIZONTAL-SURFACE-PATCHES
10	GROUND	LAST-SELECTED-CANDIDATE-IS(ground)	SIMILAR-REGIONS-REGIONS
11	FOLIAGE	CLIQUE-IS-EMPTY	TEXTURE-ABOVE-THRESHOLD
12	FOLIAGE	CLIQUE-IS-EMPTY	VEGETATIVE-TRANSPARENCY
13	FOLIAGE	CLIQUE-IS-EMPTY $\wedge$ RGB-IS-AVAILABLE	GREEN-REGIONS
14	FOLIAGE	LAST-SELECTED-CANDIDATE-IS(foliage)	SIMILAR-REGIONS
15	FOLIAGE	CLIQUE-IS-EMPTY $\wedge$ DENSE-RANGE-IS-AVAILABLE	HIGHLY-FRACTAL-REGIONS
16	RAISED-OBJECT	CLIQUE-IS-EMPTY	SEGMENTATION-REGIONS
17	RAISED-OBJECT	CLIQUE-IS-EMPTY	VERTICAL-STRIATION-REGIONS
18	RAISED-OBJECT	CLIQUE-IS-EMPTY $\wedge$ DENSE-RANGE-IS-AVAILABLE	DENSE-REGIONS-ABOVE-GROUND
19	RAISED-OBJECT	CLIQUE-IS-EMPTY $\wedge$ SPARSE-RANGE-IS-AVAILABLE	SPARSE-REGIONS-ABOVE-GROUND
20	RAISED-OBJECT	LAST-SELECTED-CANDIDATE-IS(complete-sky)	NON-SKY-REGIONS-ABOVE-SKYLINE
21	COMPLETE-GROUND	LAST-SELECTED-CANDIDATE-IS(geometric-horizon)	REGION-BELOW-GEOMETRIC-HORIZON
22	COMPLETE-GROUND	LAST-SELECTED-CANDIDATE-IS(ground)	UNION-OF-GROUND-REGIONS
23	COMPLETE-GROUND	LAST-SELECTED-CANDIDATE-IS(skyline)	REGION-BELOW-SKYLINE
25	COMPLETE-SKY	LAST-SELECTED-CANDIDATE-IS(sky) $\wedge$ SITE-IS(Stanford-hills)	UNION-OF-SKY-REGIONS

Table 4: Type I Context Sets: Candidate Generation

While generating candidates for the sky label, context set 5 was not satisfied because no color image is available and context set 6 was not satisfied because no candidates have been selected yet for inclusion in a clique. Context sets 1–4 are satisfied and the sky candidates they generate are shown in Figure 5a. Notice that three of the candidates (910, 912, and 914) are fairly similar — Condor must eventually sort out which one(s) to include in each clique based on how well they fit in the context of other members in the clique.

Ground candidates are generated by context sets 7–10 and are shown in Figure 5b. Foliage candidates are generated by context sets 11–15. The candidate generation context sets for raised-object are used to generate additional foliage candidates because foliage is a subcategory of raised-object in the abstraction hierarchy. The foliage candidates are depicted in Figure 5c.

5.2 Candidate comparison

Next, Condor compares the candidates for each class to construct the partial orders. Candidate evaluation context sets 41–53 are used for evaluating sky candidates. Only context sets 41, 42, 43, and 49 are satisfied. Their associated operators are used to evaluate each of the sky candidates and the results are assembled in Table 7. Each evaluator returns a score between 0.0 and 1.0. Only the relative magnitude of this score for each evaluator is meaningful. The scores are not normalized across evaluators because there is no basis to do so.

#	Class	Context elements	Operator
41	SKY	ALWAYS	ABOVE-HORIZON
42	SKY	SKY-IS-CLEAR ∧ TIME-IS-DAY	BRIGHT
43	SKY	SKY-IS-CLEAR ∧ TIME-IS-DAY	UNTEXTURED
44	SKY	SKY-IS-CLEAR ∧ TIME-IS-DAY ∧ RGB-IS-AVAILABLE	BLUE
45	SKY	SKY-IS-OVERCAST ∧ TIME-IS-DAY	BRIGHT
46	SKY	SKY-IS-OVERCAST ∧ TIME-IS-DAY	UNTEXTURED
47	SKY	SKY-IS-OVERCAST ∧ TIME-IS-DAY ∧ RGB-IS-AVAILABLE	WHITE
48	SKY	SPARSE-RANGE-IS-AVAILABLE	SPARSE-RANGE-IS-UNDEFINED
49	SKY	CAMERA-IS-HORIZONTAL	NEAR-TOP
50	SKY	CAMERA-IS-HORIZONTAL ∧ CLIQUE-CONTAINS(complete-sky)	ABOVE-SKYLINE
51	SKY	CLIQUE-CONTAINS(sky)	SIMILAR-INTENSITY
52	SKY	CLIQUE-CONTAINS(sky)	SIMILAR-TEXTURE
53	SKY	RGB-IS-AVAILABLE ∧ CLIQUE-CONTAINS(sky)	SIMILAR-COLOR
61	GROUND	CAMERA-IS-HORIZONTAL	HORIZONTALLY-STRIATED
62	GROUND	CAMERA-IS-HORIZONTAL	NEAR-BOTTOM
63	GROUND	SPARSE-RANGE-IS-AVAILABLE	SPARSE-RANGES-FORM-HORIZONTAL-SURFACE
64	GROUND	DENSE-RANGE-IS-AVAILABLE	DENSE-RANGES-FORM-HORIZONTAL-SURFACE
65	GROUND	CAMERA-IS-HORIZONTAL ∧ CLIQUE-CONTAINS(complete-ground)	BELOW-SKYLINE
66	GROUND	CAMERA-IS-HORIZONTAL ∧ CLIQUE-CONTAINS(geometric-horizon) ∧ ¬ CLIQUE-CONTAINS(skyline)	BELOW-GEOMETRIC-HORIZON
67	GROUND	TIME-IS-DAY	DARK
71	FOLIAGE	ALWAYS	HIGHLY-TEXTURED
72	FOLIAGE	ALWAYS	HIGH-VEGETATIVE-TRANSPARENCY
73	FOLIAGE	CAMERA-IS-HORIZONTAL	NEAR-TOP
74	FOLIAGE	RGB-IS-AVAILABLE	GREEN
76	RAISED-OBJECT	SPARSE-RANGE-IS-AVAILABLE	SPARSE-HEIGHT-ABOVE-GROUND
77	RAISED-OBJECT	DENSE-RANGE-IS-AVAILABLE	DENSE-HEIGHT-ABOVE-GROUND
78	RAISED-OBJECT	CAMERA-IS-HORIZONTAL ∧ CLIQUE-CONTAINS(complete-sky)	ABOVE-SKYLINE

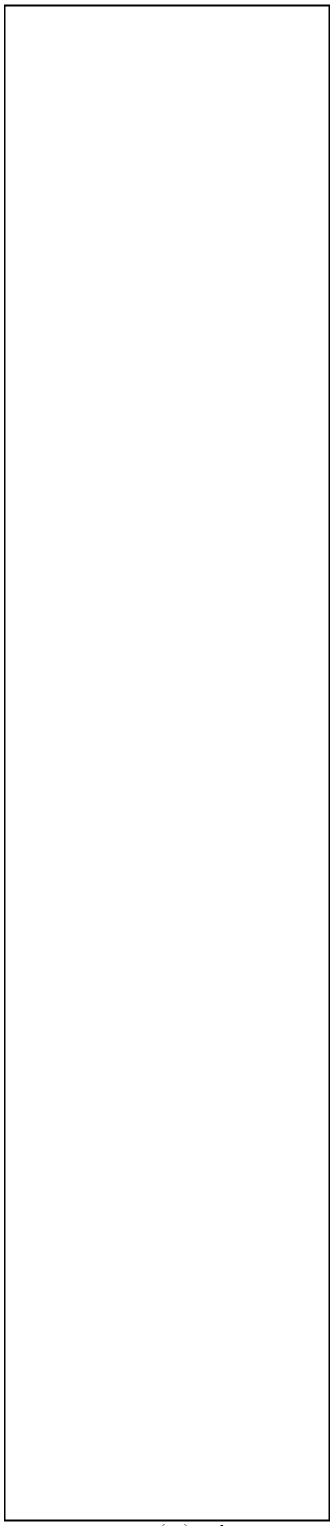
Table 5: Type II Context Sets: Candidate Evaluation

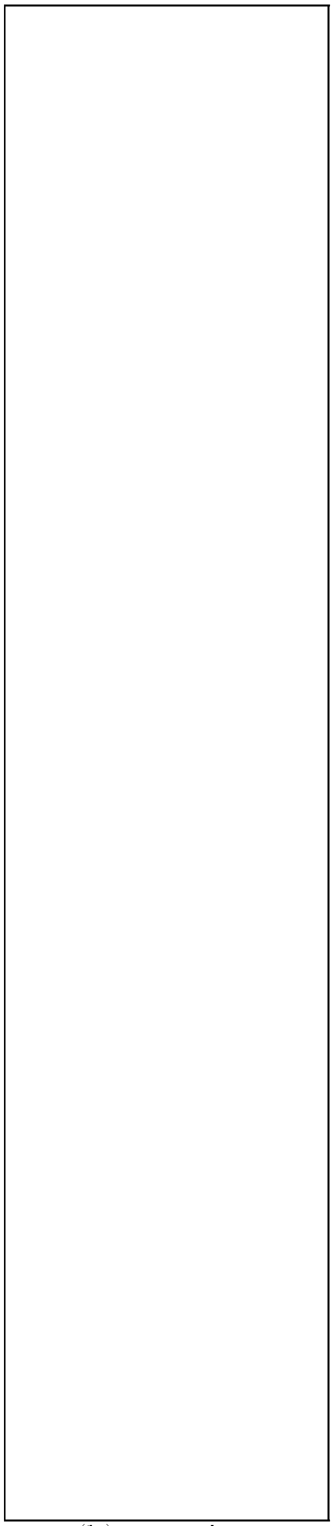
#	Class	Context elements	Operator
81	SKY	GEOMETRIC-HORIZON-KNOWN	PARTIALLY-BELOW-GEOMETRIC-HORIZON
82	SKY	ADDING-TO-CLIQUE	INCONSISTENT-WITH-CLIQUE
83	SKY	ADDING-TO-CLIQUE ∧ CLIQUE-CONTAINS(sky)	MISMATCHED-BRIGHTNESS
84	SKY	SPARSE-RANGE-IS-AVAILABLE	MUST-NOT-HAVE-FINITE-RANGE
87	GROUND	CLIQUE-CONTAINS(complete-sky)	PARTIALLY-ABOVE-SKYLINE
88	GROUND	ADDING-TO-CLIQUE	INCONSISTENT-WITH-CLIQUE
89	GROUND	DENSE-RANGE-IS-AVAILABLE	SLOPE-TOO-STEEP
91	FOLIAGE	ADDING-TO-CLIQUE	INCONSISTENT-WITH-CLIQUE
93	COMPLETE-GROUND	ADDING-TO-CLIQUE	INCONSISTENT-WITH-CLIQUE

Table 6: Type III Context Sets: Consistency Determination

Evaluator	909	910	911	912	914
ABOVE-HORIZON	1.00	1.00	1.00	1.00	1.00
BRIGHT	0.44	0.71	0.94	0.76	0.67
UNTEXTURED	0.19	0.67	0.52	0.50	0.36
NEAR-TOP	0.51	0.79	0.37	0.73	0.66

Table 7: Initial evaluation of sky candidates


(a) sky


(b) ground

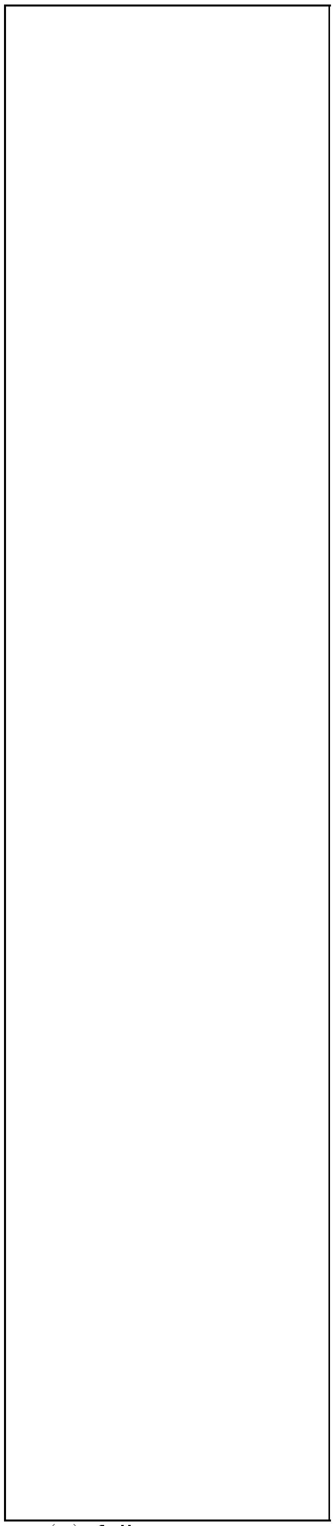

(c) foliage

Figure 5: Some candidates generated by Condor

Examining the table reveals that candidate 910 was scored at least as high as candidate 909 by every evaluator. Therefore, 910 is preferred over 909 as a sky candidate. Other unanimous preferences are

$$910 \succ 914, \quad 912 \succ 909, \quad 912 \succ 914, \quad \text{and} \quad 914 \succ 909 \; .$$

These relations are assembled into a partial order and displayed in Figure 6, after removing transitivities. Candidate 909, which roughly delineates the trees, is at the bottom of the partial order as one would hope. Candidates 910, 911, and 912, were found to be equally promising sky regions.

Figure 6: Partial order of candidates for sky

Figure 7: Partial order of candidates for ground

Figure 8: Partial order of candidates for foliage

The partial orders generated for ground and foliage are shown in Figures 7 and 8. At this point the active recognition vocabulary is

{sky, ground, foliage, geometric-horizon, complete-sky, complete-ground, skyline}

and Condor proceeds to generate candidates and partial orders for the remainder of these classes[3].

5.3 Clique formation

When all satisfied context sets for classes in the active recognition vocabulary have been employed, Condor begins to build cliques of mutually consistent candidates. The candidates at the tops of the four partial orders are eligible to be introduced into an (empty) clique. The choice of which candidate to nominate first is made with the aid of a heuristic that chooses on the basis of the reliability of the operator that generated the candidate, the desirability of adding the candidate's class to the clique, the nearness of the candidate to the camera, and the size of the candidate. If this choice is made poorly, it may lead to a small clique and more cliques will have to be generated before a large, mutually coherent clique is constructed.

[3]These are of no special interest and are not shown.

According to the heuristic, the geometric-horizon candidate is chosen first and added as the sole candidate in Clique 1. This tentative conclusion constitutes new context, albeit for Clique 1 only. All Type I context sets are reevaluated to see if any new candidates are generated, and all Type II context sets are reevaluated to update the partial orders. The only new candidate that is produced is a complete-ground candidate generated by context set 21. Type II context set 66 is now satisfied and adds BELOW-GEOMETRIC-HORIZON to the list of evaluators for ground candidates. Its use happens to cause no changes in the ground partial order.

Condor continues to test candidates for inclusion in the clique, adding those that are consistent and pruning those that are not. After each addition to the clique, the context sets are reevaluated to determine whether additional candidate generation operators or evaluation metrics have become applicable in the new context. The partial orders are updated after each change and processing continues until no candidates remain to be tested.

Figure 9 shows the complete sequence of nominations to the first clique. The composite labeling of the image that results from those that were accepted is given by Figure 10. A total of 36 candidates were generated for this clique, of which 18 were accepted in the clique, 10 were found to be inconsistent, and 8 were pruned without testing.

5.4 Clique selection

In this case, the first clique generated did a good job recognizing the target vocabulary, but Condor has no definitive way of knowing this. Condor generates additional cliques to see if its interpretation can be improved. In this experiment, six additional cliques were generated, but none of them exceeded the reliability and coverage of the components of the first clique.

As a result, Condor selects the first clique as its best interpretation and stores its results in the CKS database to be used as context for future reference. When range data is available, it is used to position the objects in the world. Without range data, Condor uses the image location of detected objects along with a digital terrain model stored in the CKS to constrain the possible locations of each object. This updated database is then used by Condor during analysis of subsequent images as context to aid interpretation.

6 Evaluation

We are currently conducting an extensive series of experiments to test the validity of our ideas and to explore the limits of the implemented system. For purposes of this experimentation, we have concentrated on tailoring the context-set knowledge base to the task of recognizing natural objects in ground level images obtained from a two-square mile portion of the foothills behind the Stanford University campus. Our ultimate goal is for Condor to be able to understand the scene in any non-degenerate image acquired in this area. The natural objects occurring in this environment consist of trees, bushes, rocks, trails, and grass in addition to the sky and the ground.

Figure 9: Sequence of candidates nominated for inclusion in Clique 1

Figure 10: Composite labeling found by Clique 1

Our intent is to develop a capacity for recognition that is on a par with that exhibited, say, by a rabbit, which inhabits the same environment. While the representations and cognitive processes employed by a rabbit are undoubtedly different than those reported here, we attempt to provide the same information that is available to a rabbit and to strive for the same level of recognition. This requires the ability to recognize scenes under many conditions including variations due to sun angle, weather, seasonal changes, normal plant growth, and discrete changes such as when a tree has fallen or a new trail has been blazed. A key issue in our research has been determining what contextual information should be recognized and stored to enable robust recognition under such a diversity of conditions.

We have taken over 100 photographs at the experimental site of which approximately 30 have been digitized and analyzed by Condor. This data includes monochrome as well as color imagery, and range data obtained from automatic compilation of binocular stereo pairs. A digital terrain model of the area and the information appearing on a USGS map provide initial context.

The image depicted in Figure 11 is indicative of the level of complexity inherent in the scenes being analyzed by Condor. With this image and a digital terrain model of the scene, Condor is able to correctly locate the ground, the sky, the grass, and six of the trees that are present in the scene. Figure 12 shows wire frame models that have been constructed for each of the recognized trees. An absolute size is given to the tree models by retrieving the depth to each tree trunk from a range image constructed by binocular stereo. Reprojection of the wire frame model from another vantage point yields Figure 13.

We are conducting a series of experiments to demonstrate the competence of the system and the value of contextual information during recognition. The preliminary results from these studies are summarized below:

Experiment 1: A single image is analyzed using only the initial context obtained from the map. Upon completion, Condor stores its recognition results in the CKS and reanalyzes the same image or a similar but different image of the same scene. In many cases, the recognition result is improved by using the newly acquired context. Repeated analysis of the image set leads to both faster recognition of known objects as well as detection of some previously unrecognized objects.

Experiment 2: A sequence of imagery collected during a simulated traverse of the terrain is analyzed by Condor. The results of processing each image are stored in the CKS and made available as context for analyzing subsequent images. The temporal continuity provided by the information in the CKS allows Condor to improve the results it would have obtained without this additional contextual information. Upon completion of the sequence, Condor has built a 3D model of the objects visible during the traverse, and has annotated each with information that aids its recognition.

Experiment 3: A collection of images from a restricted area but varying widely in viewpoint, scale, time-of-day, season, and sky conditions are analyzed by Condor. In most

Figure 11: A natural outdoor scene from the experimentation site.

cases Condor obtains a consistent recognition result, demonstrating that the context-set knowledge base is insensitive to these types of change.

7 Complexity analysis

In the region-based approach to machine vision, an image is partitioned into r disjoint regions and a program must decide which of l potential labels to assign to each region. Because these assignments cannot be made independently, there are l^r potential labelings of the image from which the program must select the best.

In the model-based approach the regions associated with each model class are to be determined. Given l model classes and r possible locations of each model instance, there are r^l potential configurations of model instances in the worst case. (See Tsotsos [1988] for further elaboration.)

Most, if not all, of the existing systems for recognition can be viewed as strategies to explore either of these exponential search spaces. In contrast, Condor defines an entirely different search space — one that is polynomial in both the number of regions and the number of labels being considered — by identifying and exploring only the most promising portions of the space.

Figure 12: Result of analyzing Figure 11.

7.1 Computation time

To compute the computational complexity of the Condor architecture, it is convenient to characterize the algorithm as repeatedly testing candidates for consistency with a partially instantiated clique. At each stage, Condor must generate new candidates, update the partial orders, select a candidate for inclusion, and test it for consistency with the clique. In practice, Condor rarely needs to generate many new candidates after the initial iteration, but for analyzing worst-case complexity, we will assume that it does. Let

$$\begin{array}{rcl} l & = & \text{the number of labels in the recognition vocabulary} \\ c & = & \text{the number of candidates for each label} \\ r & = & \text{the number of candidate regions in the largest clique} \\ q & = & \text{the total number of cliques constructed.} \end{array}$$

At most, Condor must construct a total of lc candidates. Completely rebuilding each partial order requires c^2 operations, so lc^2 operations are required for partial order construction in the worst case. Selecting a candidate from the tops of the partial orders is no worse than linear in the number of candidates and testing for consistency could require as many as r tests. The maximum number of operations required for one complete iteration is

$$2lc + lc^2 + r \quad . \tag{1}$$

Figure 13: A perspective view of the 3D model produced from the analysis of the image shown in Figure 11.

This cycle must be repeated for each of the r candidates introduced into the clique. Completely repeating the entire process for q cliques is not necessary, but would require

$$(2lc + lc^2 + r)rq \tag{2}$$

operations. Therefore, the worst-case complexity for analyzing one image is

$$\mathcal{O}(qr^2 + lrqc^2) \ . \tag{3}$$

Formula 3 gives the total time complexity for analyzing one image and yields two important observations:

- Despite the combinatorics inherent in the recognition problem, our approach has no exponential behavior. The complexity is only quadratic in the number of regions to be recognized. This behavior is attributable to the fact that Condor constructs a fixed number of cliques and does not exhaustively search the exponential recognition space. While there is no guarantee that Condor will find the optimal clique, the context-based generation and relative ordering of candidates ensure that only good cliques are generated early in the search.

- The complexity is linear in the number of terms in the recognition vocabulary. Therefore, expanding the system by adding additional categories to be recognized results only in a proportional increase in run time. This behavior is important because it allows Condor to be expanded to recognize a broad range of categories without a prohibitive increase in computation. We know of no other visual recognition system that possesses this property.

7.2 The number of cliques

The key to achieving desirable computational complexity is to accept candidates into cliques with sufficient reliability that the best clique is found early in the search. How reliable must candidate acceptance be?

Let p be the probability that a candidate nominated for inclusion into a clique is a member of the best clique (i.e., the label associated with the candidate is correct).[4] The probability of constructing a clique with r valid regions is p^r. On average, it will be necessary to construct $q = \frac{1}{p^r}$ cliques before the best one is found. Thus, if the best clique is to be found within the first q cliques, it will be necessary that

$$p \geq q^{\frac{-1}{r}} \quad .$$

It is clear that candidate acceptance must be perfect if only one clique is to be generated. If 95% reliability is attainable, then 7 cliques would be required; if only 90% reliability were attainable, then 68 cliques would be needed.

Although most of the operations employed by Condor are individually unreliable, their collective use is highly reliable. For example, in the course of analyzing the image in Figure 4, candidates that were accepted into cliques were 98% correct, based on a subjective assessment of which candidates were valid. At other stages of the analysis,

- 53% of the candidates generated by the context sets were valid.
- 78% of the candidates at the tops of the partial orders were valid.
- 82% of the candidates nominated for inclusion were valid.
- 98% of the candidates accepted by the consistency checking context sets to any clique were valid.
- 100% of the candidates in the best clique were valid.

[4]We assume for this analysis that the probability is the same for all nominated candidates.

8 Conclusion

The Condor architecture embodies a new approach to visual recognition which

- integrates the information available in both 2D and 3D imagery;
- uses the consensus of many simple procedures to achieve reliable results;
- exploits contextual information to aid the recognition process; and
- augments a database of contextual information with its own recognition results to improve its performance incrementally over time.

Many computational vision systems have been devised to recover the three-dimensional location and orientation of surfaces from image data. However, shape recovery is only a part of the functionality that is required of a vision system for autonomous robots, for military surveillance, for space station assembly and repair, and so on. In order for these systems to interact intelligently with their environments, they must be able to recognize things in terms of physical attributes and semantic qualities, not just shapes. Condor implements a new theory of computer vision that eliminates the traditional dependence on stored geometric models and universal image partitioning algorithms, and provides a basis for semantic interpretation.

References

[1] Barnard, Stephen T., "Stochastic Stereo Matching over Scale," *International Journal of Computer Vision*, **3**(1), 1989.

[2] Barrow, Harry G., and Tenenbaum, Jay M., "MSYS: A System for Reasoning about Scenes," Technical Note 121, Artificial Intelligence Center, SRI International, April 1976.

[3] Barrow, Harry G., Thomas D. Garvey, Jan Kremers, J. Martin Tenenbaum, and Helen C. Wolf, "Interactive Aids for Cartography and Interpretation," Technical Note 137, Artificial Intelligence Center, SRI International, January, 1977.

[4] Bolles, R.C., Horaud, R., and Hannah, M.J., "3DPO: A 3D Part Orientation System," in *Proceedings 8th International Joint Conference on Artificial Intelligence*, Karlsruhe, West Germany, August 1983, pp. 1116–1120.

[5] Brooks, Rodney A., "Model-Based 3-D Interpretations of 2-D Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 5, Number 2, March 1983, pp. 140–150.

[6] Draper, Bruce A., Robert T. Collins, John Brolio, Allen R. Hanson, and Edward M. Riseman, "The Schema System," *International Journal of Computer Vision,* Vol. 2, No. 3, January, 1989, pp. 209–250.

[7] Fischler, Martin A., and Strat, Thomas M., "Recognizing Trees, Bushes, Rocks and Rivers," *Proceedings of the AAAI Spring Symposium Series: Physical and Biological Approaches to Computational Vision*, Stanford University, March 1988, pp. 62–64.

[8] Fischler, Martin A., and Strat, Thomas M., "Recognizing Objects in a Natural Environment: A Contextual Vision System," *Proceedings: DARPA Image Understanding Workshop*, Palo Alto, California, May 1989, pp 774 – 796.

[9] Fua, Pascal, and Hanson, Andrew J., "Using Generic Geometric Models for Intelligent Shape Extraction," *Proceedings: DARPA Image Understanding Workshop*, Los Angeles, California, February 1987, pp. 227–233.

[10] Hannah, M.J., "SRI's Baseline Stereo System," *Proceedings: DARPA Image Understanding Workshop*, Miami Beach, Florida, December, 1985, pp. 149 – 155.

[11] Hanson, A.R., and Riseman, E.M., "VISIONS: A Computer System for Interpreting Scenes," in *Computer Vision Systems*, Academic Press, New York, 1978, pp. 303–333.

[12] Hanson, A.J., and Quam, L., "Overview of the SRI Cartographic Modeling Environment," *Proceedings: DARPA Image Understanding Workshop*, Cambridge, Massachusetts, April 1988, pp. 576 – 582.

[13] Huttenlocher, Daniel P., and Ulman, Shimon, "Recognizing Solid Objects by Alignment," *Proceedings: DARPA Image Understanding Workshop*, Cambridge, Massachusetts, April 1988, pp. 1114–1122.

[14] Laws, Kenneth, I., "Integrated Split/Merge Image Segmentation," Technical Note 441, Artificial Intelligence Center, SRI International, July 1988.

[15] McKeown, David M., Jr., W.A. Harvey, Jr., and J. McDermott, "Rule-Based Interpretation of Aerial Imagery," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 7, No. 5, September, 1985, pp. 570–585.

[16] Ohta, Y., "A Region-Oriented Image-Analysis System by Computer," Doctoral dissertation, Information Science Department, Kyoto University, Kyoto, Japan, 1980.

[17] Rosenfeld, A., Hummel, R.A., and Zucker, S.W., "Scene Labeling by Relaxation Operations," *IEEE Transactions on Systems, Man, and Cybernetics*, Volume 6, Number 6, June 1976, pp. 420–433.

[18] Sloan, Kenneth R., Jr., "World Model Driven Recognition of Natural Scenes," PhD Thesis, Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia, Pennsylvania, June 1977.

[19] Smith, Grahame B., and Strat, Thomas M., "Information Management in a Sensor-Based Autonomous System," *Proceedings: DARPA Image Understanding Workshop*, Los Angeles, California, February, 1987, pp. 170–177.

[20] Strat, Thomas M. and Smith, Grahame, B., "A Knowledge-Based Information Manager for Autonomous Vehicles," Chapter 1 in *Image Understanding in Unstructured Environments*, edited by Su-shing Chen, World Scientific Publishing Co, 1988, pp 1-39.

[21] Strat, Thomas M. and Fischler, Martin A., "Context-Based Vision: Recognition of Natural Scenes," *Proceedings of the 23rd Asilomar Conference on Signals, Systems, and Man,* October, 1989, pp 532–536.

[22] Strat, Thomas M., "Natural Object Recognition," Ph.D. Dissertation, Department of Computer Science, Stanford University, Stanford, California, December, 1990.

[23] Tenenbaum, J. M. and Weyl, S., "A Region Analysis Subsystem for Interactive Scene Analysis," *Proceedings of the Fourth International Joint Conference on Artificial Intelligence*, Tbilisi, Georgia, USSR, September 1975, pp. 682–687.

[24] Tsotsos, John K., "A Complexity Level Analysis of Immediate Vision," *International Journal of Computer Vision,* Vol. 1, No. 4, 1988, pp. 303–320.

[25] Yakimovksy, Yoram, and Feldman, Jerome A., "A Semantics-Based Decision Theory Region Analyzer," *Proceedings of the Third Joint Conference on Artificial Intelligence*, Stanford, California, August 1973, pp. 580–588.