# PROCEEDINGS

Of SPIE-The International Society for Optical Engineering

Volume 485

# Applications of Artificial Intelligence

John F. Gilmore
Chairman/Editor

May 3-4, 1984
Arlington, Virginia

Proceedings of SPIE—The International Society for Optical Engineering

Volume 485

# Applications of Artificial Intelligence



**John F. Gilmore**
*Chairman/Editor*

**May 3-4, 1984**
**Arlington, Virginia**



*Published by*
**SPIE—The International Society for Optical Engineering**
**P.O. Box 10, Bellingham, Washington 98227-0010 USA**
Telephone 206/676-3290 (Pacific Time) • Telex 46-7053

SPIE (The Society of Photo-Optical Instrumentation Engineers) is a nonprofit society dedicated to advancing engineering and scientific applications of optical, electro-optical, and optoelectronic instrumentation, systems, and technology.

Please use the following format to cite material from this book:
Author(s), "Title of Paper," *Applications of Artificial Intelligence,* John F. Gilmore, Editor, Proc. SPIE 485, page numbers (1984).

Library of Congress Catalog Card No. 84-50820
ISBN 0-89252-520-7



Printed in the United States of America.

*APPLICATIONS OF ARTIFICIAL INTELLIGENCE*

Volume 485

**Conference Committee**

*Chairman*
**John F. Gilmore**
Georgia Institute of Technology

*Co-Chairmen*
**Andrew J. Spiessbach**
Georgia Institute of Technology

**David Y. Tseng**
Hughes Artificial Intelligence Center

**Tice F. DeYoung**
U.S. Army Engineer Topographic Laboratories

*Session Chairmen*
Session 1—Expert Systems
**John F. Gilmore**, Georgia Institute of Technology

Session 2—Knowledge-based Systems
**Andrew J. Spiessbach**, Georgia Institute of Technology

Session 3—Autonomous Vehicles
**Tice F. DeYoung**, U.S. Army Engineer Topographic Laboratories

Session 4—Image Understanding
**David Y. Tseng**, Hughes Artificial Intelligence Center

*APPLICATIONS OF ARTIFICIAL INTELLIGENCE*

Volume 485

# INTRODUCTION

To the surprise of many, Artificial Intelligence (AI) is a discipline with its initial roots in the 1950s. The majority of work in this area since that time has been dedicated to research in the areas of problem solving, knowledge representation, heuristic search, machine vision, knowledge-based systems, predicate logic and natural language. In the 1970s a culmination of these efforts led to the development of expert systems, thus providing a technical foundation for the application of artificial intelligence to a wide variety of real-world problems.

The goal of the Applications of Artificial Intelligence Conference is to annually present a technical forum in which applications work can be presented. As the focus of the National AI Conference is primarily restricted to academic research, the SPIE forum is dedicated to presenting high quality applications work and identifying the contributions these programs are making to technology. To these ends, this year's conference consists of four sessions: expert systems, knowledge-based systems, autonomous vehicles, and image understanding.

The expert systems session reviews applications in the area of conflict avoidance, data interpretation, information fusion, scene analysis, decision aids, and fault diagnosis. In addition, it also includes presentation of a new expert system building tool developed by Lockheed.

Knowledge-based systems are intelligent programs that differ from expert systems in that their knowledge does not include expertise extracted from an expert in the application problem domain. Efforts presented in this area include consulting systems, intelligent optical design programs, decision support systems and image analysis.

Though originally planned well over a year ago, the inclusion of a session on autonomous vehicles is very timely in light of the recent military activity in this field. Several active programs in this area are presented including research work performed at the Georgia Tech Engineering Experiment Station, Hughes Aircraft, Honeywell, and the University of Florida.

Image understanding is the final session of this year's program. Advanced vision systems for target acquisition as well as efforts in the area of motion estimation, spatial reasoning, and image analysis constitute the body of this session.

I would like to thank my co-chairmen who have aided me greatly in the preparation of the technical program, and made it possible to assure the success of the first annual Applications of Artificial Intelligence Conference.

**John F. Gilmore**
**Georgia Institute of Technology**

*APPLICATIONS OF ARTIFICIAL INTELLIGENCE*

Volume 485

## Contents

*APPLICATIONS OF ARTIFICIAL INTELLIGENCE*

Volume 485

# Session 1

## Expert Systems

*Chairman*
**John F. Gilmore**
Georgia Institute of Technology

# A Context Dependent Automatic Target Recognition System

J.H. Kim, D.W. Payton, K.E. Olin, and D.Y. Tseng

Artificial Intelligence Center
Hughes Aircraft Company
23901 Calabasas Road
Calabasas, CA 91302

## Abstract

This paper describes a new approach to automatic target recognizer (ATR) development utilizing artificial intelligent techniques. The ATR system exploits contextual information in its detection and classification processes to provide a high degree of robustness and adaptability. In the system, knowledge about domain objects and their contextual relationships is encoded in frames, separating it from low level image processing algorithms. This knowledge-based system demonstrates an improvement over the conventional statistical approach through the exploitation of diverse forms of knowledge in its decision-making process.

## Introduction

The capabilities of current generation ATR represent a significant technical advancement towards the development of systems that can detect and classify targets automatically.[1,2] Unfortunately, there are a number of serious deficiencies associated with the basic approach used in these first generation systems, and these cannot be overcome by continuing the current line of development. The basic approach used in all of the current systems is to first extract feature information from imagery and then classify it using statistical techniques. The first generation ATRs developed this way are all highly data dependent in their performance. Noise and variations in the sensed image readily influence system performance, and they need frequent adjustment of parameters to produce good results. At the same time, no one fully understands how to set the parameters to yield the best result. The systems are also highly dependent on fixed design paramaters, such as segmentation window size, causing another type of performance sensitivity to the input data. Also the current approaches use a fixed processing sequence to collect feature information. There is no feedback mechanism to either change the type of information collected in response to the type of evidence needed by the high level decision making system.

This paper describes a new approach to ATR systems, that do not suffer from the above severe problems. The system, which is implemented on the Symbolics LISP Machine, is knowledge-based, and exploits various types of contextual (temporal, global, structural and non-imagerial) information in the detection and classification process. The exloitation of the contexutal information contained in the scene provides not only improvement of detection and classification performance but also reduces processing requirements. The additional information results in enhanced classification accuracy and ensures robust performance with scene variability, yet permitting opportunistic, requirement-driven, processing.

The system achieves synergistic integration of diverse forms of contextual information through symbolic representation and reasoning known as an object-oriented approach[3,4]. All of the information about a particular object is put together, or encapsulated, into one location, rather than scattered around in an unarraged manner. This object-oriented knowledge representation is sufficiently flexible to permit representation of diverse types of knowledge and yet sufficiently modular to enable easy modification and extension. Object classes form a hierarchy with super/subclass (generic/specific) relationships in that "children" represent specific classes of their "parent". All of the parent's attributes are inherited to its children as a default. With this inheritance mechanism, complex concepts are encoded parsimoniously in such a way that characteristics common to all the children are stored only in the parent's location and only the characteristics that make a child different from its siblings are encoded at the child's location. Model objects constitute a model data base in that all the object classes expected to be encountered in the problem domain are defined.

The inference and classification process proceeds by generating hypotheses about the existence of certain objects in the image, and then seeking the required evidence to satisfy those hypotheses. In doing so, new hypotheses may be generated, and existing hypotheses may be refined by obtaining new evidence. Although the initial hypotheses are generated through a bottom-up processing of the image, further hypotheses are generated from top-down, model-driven processing. This model-driven processing re-directs low-level image processing algorithms to gain new information from the image. Finally, each hypothesis is evaluated, and image regions are classified in an "unforced" way by subclassifying an object only when enough information is available.

## System overview

This ATR system takes a sequence of digitized images as input and produces descriptions of interesting objects and their behaviors. Our approach to target detection and classification is mainly based on region segmentation. Raw gray scale images are segmented into regions and the regions are described in terms of prestored object classes. Initially the regions are classified as the most generic class that subsumes all the object classes in the image domain. Then they are subclassified into more specific classes based on various feature values and contextual relationships among them. Although the ultimate goal of this system is to report locations of high-valued tactical objects, the entire regions in the scene are labeled so that contextual information can be utilized through a feedback process to detect high-valued tactical targets which are initially missed.

Like many other knowledge-based systems, this system separates domain specific knowledge from its control structure, and thus, image domain knowledge can be viewed as a distinct input to the inference engine as shown in Figure 1. The domain knowledge base provides domain specific rules and heuristics relevant to object detection and identification, and the inference engine interprets the rules and heuristics and performs specified tasks.

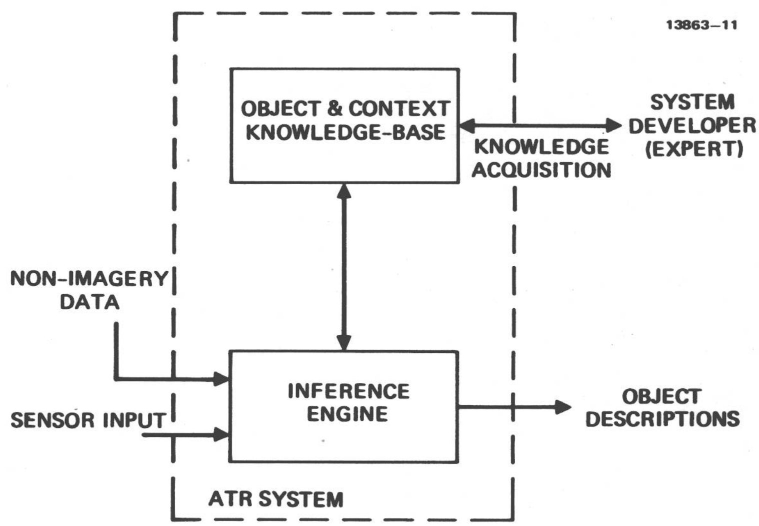



Figure 1: System Top Level

The knowledge base is a collection of model objects and their relationships. Each model object, which is realized as a frame data structure, represents a class of objects that are expected to be encountered during the processing of the scenes. Model objects form an object hierarchy that plays the role of the hierarchical classifier. An unknown object will match against each of the model objects and be classified to the best matching class. A model hierarchy used in a prototype is shown in Figure 2. In this hierarchy, the scene object is the most general class and subsumes all other object classes in the image domain.

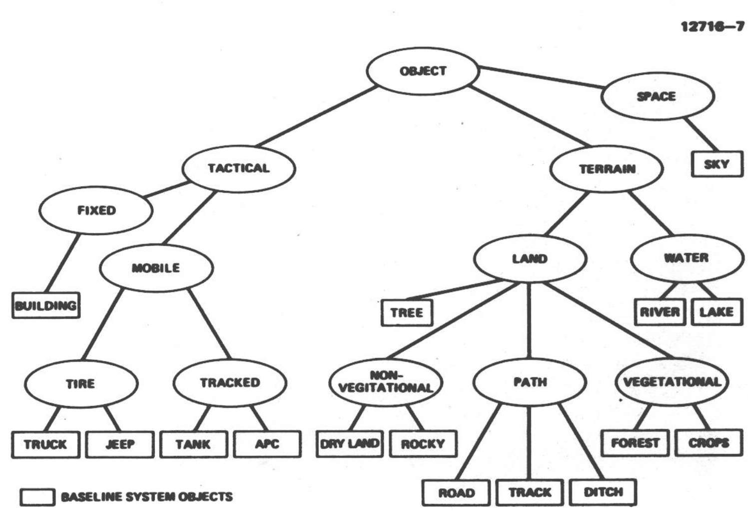



Figure 2: An Object Hierarchy

The inference engine of the system consists of a number of image processing modules, high level decision making modules and a frame interpreter. The image processing modules transform the raw gray level images into intermediate symbolic representations, i.e., in terms of lines and regions, etc. The high-level decision making components generate class hypotheses and evaluate available evidence to confirm or refute the hypotheses. The frame interpreter provides an interface between the knowledge base and the other components of the inference engine. It interprets the knowledge base and controls image processing and high level decision-making modules accordingly.

Both model objects and modules of the infernece engine are formulated as objects in this system. An operator that performs a specific function on given data is viewed as an expert that provides specific services upon request.[5] Invocation of expert modules is achieved by sending a predefined message to the expert. The expert, then, responds by sending back messages. Since passing messages is the only way to communcate with experts, a high degree of modularity is achieved. The overall system design and the functional realtionships among the major components are shown in Figure 3.

The spatial blackboard is a symbolic pixel array[6] which maintains all scene specific information preserving spatial relationships among scene elements. The information in the symbolic pixel array is accessible from all the components of the system and includes raw images, sensor characteristics, symbolic transformations, as well as classification hypotheses. The symbolic pixel array also possesses reasoning capabilitites about spatial relations such as ABOVE, CONTAINED, TOUCHING, etc. It reduces the burdens of image data management by doing all the required bookkeeping tasks automatically, and by providing a capability to handle a group of objects (such as pixels, lines, regions) either as a group or as individual objects.

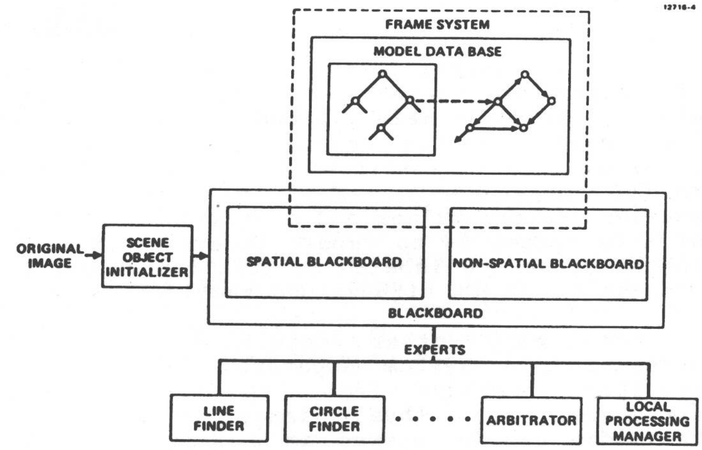
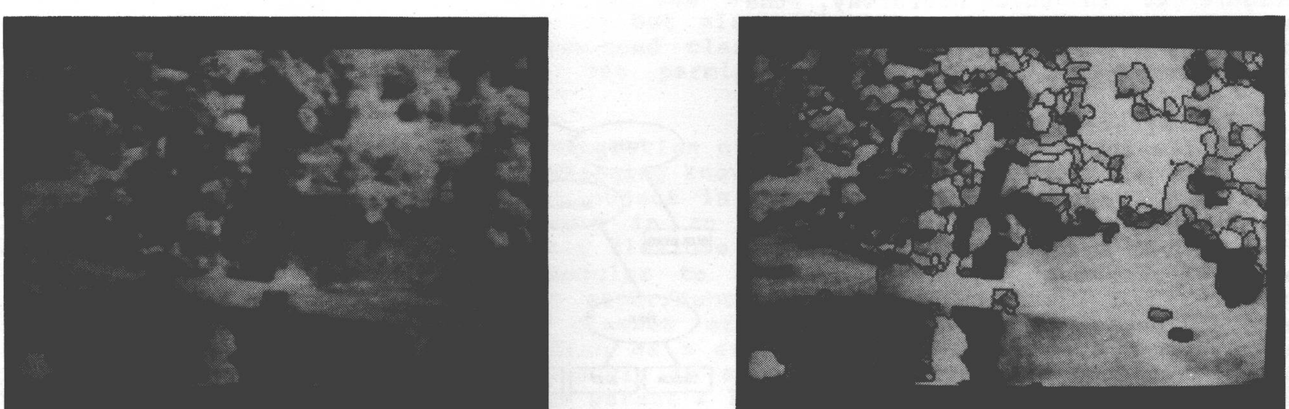



Figure 3: System Overview

Various features, either characterizing a single scene element or describing relationships among relevant objects, are computed by calling various expert modules as needed. These feature measurements are used for resolving uncertainties in the determination of class labels. In this computation, model objects provide instructions for when and how to call expert modules and how to use the feature measurements to determine object classification.

A model object can be classified, i.e., described, at multiple levels of abstraction because it is a generic description of more specific objects and, at the same time, is a specialization of its more generic classes. For example, a tank is a tracked object, and at the same time it is a mobile object and a tactical object. Therefore, a scene object which is known as a tank can be placed into classes defined at different levels of abstraction, such as tactical object, mobile object, tracked object, and tank. However, a more specific description of an object requires more information. Intelligent beings, such as humans, would refuse to classify an object into a specific class unless sufficient information were available. For example, we might say "That is definitely a moving object but I don't know whether that is a tank or truck". The classification algorithm used in this system models this behavior by producing multiple class descriptions incorporating the object hierarchy.



original　　　　　　　　segmented

Figure 4: Initial Segmentation Result

## Symbolic transformation

Raw gray-scale images are transformed into a number of symbolic representations such as edges, lines, vertices, and regions. These symbolic representations are recorded on the spatial blackboard and made available to the high-level decision making modules that utilize them throughout the detection and classification process. In general, these symbolic transformations require well selected processing parameters that are critical to the performance of the entire system. In our system, however, the parameters are dynamically determined through a feedback process to yield maximum performance and computational efficiency. For example, if a road is found in the initial analysis of a scene, a region segmentor is then applied with a lower threshold in the vicinity of the road to find tactical objects that may have been missed in the initial analysis. Figure 4 shows a typical imagery we are working on and its region segmentation results.

## Classification

The most distinguishable characterisitic of the intelligent classification process is probably its ability to adaptively determine the appropriate levels of description based on the amount of information available. People describe an object in a specific level only when sufficient information is available. For example, when a low resolution image is given to a person and he is asked to tell what's in the image, it is quite common to hear such a response that "it is probably a vehicle, but I don't know whether it is a jeep or truck." The computational scheme employed in this system models this kind of intelligent classification reasoning normally employed by human experts.

Once through the initial segmentation step, an input image is represented as a group of regions. Each region is repesented as a frame with its own values for various region attributes such as area, centroid, and primary and secondary axes. These regions are subject to classification in terms of the classes encoded in the model object hierarchy. Each region is initially classified as a scene object which is the most generic object class that subsumes all object classes in the image domain. Note that the classification of a region as a scene object does not involve any ambiguities or uncertainties. As the inference process proceeds, each object is hypothized as each of its subclasses and evidence is gathered to confirm or refute these hypotheses. This cycle of hypothesis generation, evidence gathering, and confirmation or refutation is repeated for each confirmed hypothesis until each region is classified as a terminal objects class of the object hierarchy, or no further subclassification is possible because of low confidence for the current classification.

An object's class hypothesis is represented as an instance of the class model. Therefore, generation of a hypothesis corresponds to instantiation of the class model. Hypotheses may be generated in either a data driven or a model driven manner. In the data driven form, hypotheses are generated in an attempt to subclassfy 'established' objects into their subclasses. In the model driven form, hypotheses are generated from the need to find new evidence for existing hypotheses. For example, based on the heuristic rule that vehicles are ususally found on roads, vehicle hypotheses may be

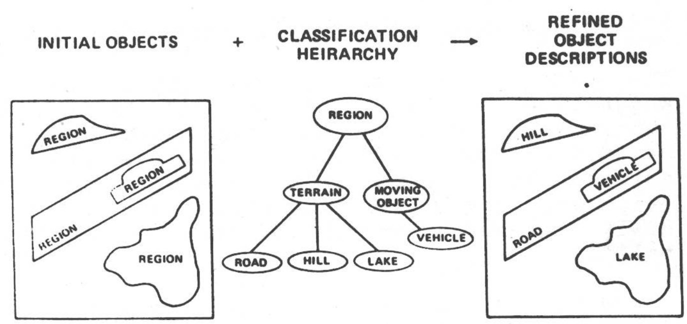



Figure 5: Data-driven Hypothesis Generation

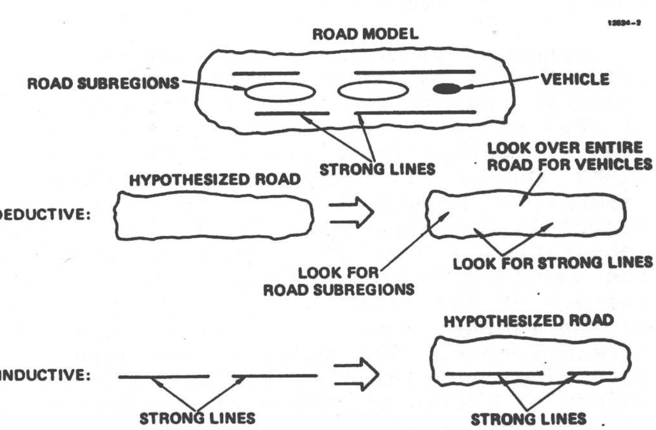



Figure 6: Model-driven Hypothesis Generation

generated in an attempt to gather evidence for a road hypothesis. Figures 5 and 6 illustrate the data-driven and model-driven approaches for hypothesis generation.

Associated with each hypothesis is a confidence factor that represents the degree of the likelihood of a hypothesis being true by a value between -1 and 1. Upon instantiation, the confidence factor is set to zero to represent the neutral belief state. Generated hypotheses are subject to passing a series of extensive testing to confirm or refute their class membership. Central to the confirmation and refutation process is the integration of various evidence and updating of confidence factors to lead to a class decision. To achieve efficiency, we proceeded with two computational steps. In the first step, hypotheses are filtered by computationally simple tests. Highly unlikely hypotheses are quickly eliminated by simple binary, true/false tests. Only the hypotheses that pass this screening test will receive further evaluation. The second step is very complex and time consuming. Each piece of evidence is carefully weighed, integrated into the existing body of evidence, and confidence is updated. The final class label for an object is determined through consideration of the confidence scores accumulated on all hypothosized classes for the object.

## Confidence Evaluation

The computation of the confidence score for a hypothesis must take into account all the evidence computable from the corresponding model object as well as from the relationships with other hypotheses. Each model object contains information of how attribute values impact the hypothesis in terms of IF-THEN rules. The IF part states conditions in terms of various attribute values and the THEN part is a confidence factor which states the amount of confidence to be added if the condition is fully satisfied. Following MYCIN's convention[7], the confidence factor simply represents the change of belief status upon observing the satisfaction of the conditions. For example, consider the following road object rule:

**IF (highly-elongated SELF) THEN** 0.5 .

The rule is read as "if the object under investigation is highly elongated, there is suggestive evidence(0.5) that it is a road." A negative confidence factor represents subtracted confidence when the condition is satisfied. When the conditions are not fully satisfied, the confidence factor is proportionately reduced according to the degree of satisfaction. Like MYCIN, supporting evidence is maintained separately from refuting evidence. Pieces of supporting or refuting evidence are combined repectively such that a newly acquired piece of evidence is applied proportionately to the remaining disbelief or belief.

With the same rule format, context relations can be represented. For example, a rule for tactical objects,

**IF (near SELF ROAD) THEN** 0.4 ,

represents that an object near a road is likely to be a tactical object. In the evaluation of the rule, the inference system selects the best road among candidates satisfying user specified conditions and uses it to compute the tactical object's confidence score.

Since an object may have several different hypothized classifications, the confidence score for a hypothesis should be consistent with that of the other hypotheses for the objects. Supporting evidence for one object classification is also support generalization of that classification. On the other hand, refuting evidence for a given classification is also refute specializations of that classification. Furthermore, a hypothesis of a generic class should be always more confident than the hypotheses of its subclasses. As an example, an object likely to be a truck is more likely to to be a vehicle since vehicle is a generic description of truck. The confidence score of a given hypothesis is computed by integrating all of the supporting evidence to its descendant class hypotheses and all of the refuting evidence to its ancestor class hypotheses. After computing the confidence factors of all established hypotheses, the object is described as the class corresponding to the highest score subclass at each branching point, traversing from the root node to a terminal node. The nodes on the path represents the most likely classes for the object. Note that the selected classes are not mutually exclusive, but they describe the same object at different levels of specification.

## Distributed control

The control machanism of this system is object-oriented, i.e., distributed over the objects in the system. Each classification hypothesis is associated with a process which controls its evidence gathering and creation of new hypothses as required. When a hypothesis is created, the associated process executes procedures to gather evidence and evaluates rules to update the confidence factor to confirm or refute the hypothesis. Once the hypothesis is confirmed , it spawns a process for each of its subclass hypotheses. Control knowledge for each process is contained within the associated object model, allowing the evidence gathering behaviors to be generic to their models, but specific to their local context. This distributed control mechanism, which combines synergistically model-driven and data-driven control strategies, can be naturally implemented in parallel processing architectures.

## Summary

A knowledge-based system for context dependent ATR has been presented. Exploiting various types of contextual information contained in the scene, the system represents an improvement over the conventional approach, and paves the way for the next generation ATR system using artificial intelligence techniques. This system's skeleton, knowledge representation mechanism and inference engine, can be easily applied to the other classification problems by supplying different domain knowledge.


## Acknowledgments

The authors would like to thank Bruce Bullock, Gary Edwards, David Keirsey and Felicia Vilnrotter for their discussion throughout the system development.


## References


1. Rubin, L.M. and Tseng, D.Y., "Automatic Target Cueing," Proceedings of the Fourth International Joint Conference on Pattern Recognition, Kyoto, Japan, p 774, 1978.
2. Sevigny, L., et al., "Discrimination and Classification of Vehicles in Natural Scenes from Thermal Imagery," Computer vision, Graphics, and Image processing, Vol 24, p 229-243, 1983.
3. Brachman, R.J., "On Epistemological status of semantic Networks," in Associative Networks: Representation and Use of Knowledge by Computers, N.V. Findler, ed.,Academic Press, New York, pp 3-50, 1979.
4. Goldberg, A. and Robson, D., Smalltalk-80, the language and its implementation, Addison Wesley, 1983.
5. Hewitt, C., "Viewing Control Structures as Patterns of Message Passing," Artificial Intelligence, 8, p 323 - 364, 1977.
6. Payton, D., "A Symbolic Pixel Array for Representation of Spatial Knowledge," Proceedings of the Third Annual Conference on Computers and Communication, Phoenix, Arizona, March 1984.
7. Shortliffe, E., Computer-Based Medical Consultations: MYCIN, Americal Elsevier, 1976.

# Computer Understanding of Air Traffic Control Displays

Stephen E. Cross

Artificial Intelligence Laboratory, Department of Electrical Engineering
Air Force Institute of Technology, Wright-Patterson AFB, Ohio 45433

## Abstract

A human air traffic controller perceives possible aircraft collisions from a display of many aircraft. In the work described here, the computer understands the displayed aircraft conflict data by processing a global, semantic representation of the displayed data. Understanding implies that the computer can represent and interpret the displayed data in a manner suggestive of an experienced human controller. The representation is called the conflict structure. The paper describes the conflict structure and its use by an expert system that performs the enroute air traffic control task. An example is taken from a 'live' air traffic control training problem.

## Introduction

Human air traffic controllers recognize elegant plans for controlling aircraft. The plans may be original or suggested by another controller. The elegance of a plan is dependent on the recognition, interpretation, and use of planning attributes such as conflict localization, constraint relaxation, and multiple goal satisfaction. The human air traffic controller's representation of the air traffic problem makes these attributes explicit. An expert system architecture influenced by such planning attributes has been implemented [1]. The overall performance is dependent on the knowledge representations employed by the expert system, its problem solving strategies (for problem definition, decomposition, and resolution), and its ability to reason qualitatively about plans. In this paper, the design and implementation of a capability that allows the computer to represent and interpret (i.e., to understand) displayed air traffic control data is discussed. The approach to computer understanding is script-based [2].

Visual perception is a critical aspect of the human controller's skill. He can perceive possible collisions from a display of many aircraft. The display may be a time-ordered list of 'flight strips' or a computer generated display from radar tracking data and aircraft transponder data. The computer understands the displayed aircraft conflict data by processing a global, semantic representation of the displayed data. This knowledge is called the conflict structure and is represented in a semantic network where nodes are aircraft and links are indicators of the types and relative location of conflicts (i.e., head-on, merging, above, west-of). Based on interviews with controllers from the Chicago Air Traffic Control Center and the FAA Academy, the conflict structure is a plausible cognitive representation of the controller's visual field as restricted to the conflict avoidance task. The structure is the input to the expert system that performs the enroute air traffic control task. The air traffic control expert system accomplishes the conflict resolution task as well as other tasks such as fuel efficient route selection and metering.

A bottom-up problem solving approach would form a solution set of two aircraft problems. The approach taken in this work is top-down. Decomposition strategies, suggested by human controllers, manipulate the conflict structure to form less complex subproblems. In many cases, the initial conflict structure is transformed into more general abstract structures where nodes represent abstracted aircraft. Detailed problem solving is then accomplished using the human controller's heuristics and automated control algorithms. The paper describes the conflict structure and its use by the expert system. Examples are taken from 'live' air traffic case studies which were identified as difficult by human controllers. It is shown that the problem decomposition performed by the expert system approaches that of the highly skilled human controller.

## Human Controller Planning Behavior

An aircraft conflict occurs when two or more aircraft violate another aircraft's airspace. Controllers perceive conflict situations fifteen to twenty minutes in advance, and typically issue resolution commands three to five minutes in advance. The commands are chosen to achieve the high level goals of safety and expediency, and form part of a dynamic 'bug-prone' plan. Each command is chosen based on stereotypical knowledge of past conflict situations. A proposed command is criticized by comparing it against known constraints, such as not causing another conflict during a brief look-ahead time period.

The controller's stereotypical knowledge is obtained during a lengthy training period. The initial training takes place at the FAA Academy in Oklahoma City and concentrates on equipment familiarization and the acquisition of domain rules and constraints. After the initial training, the human controller spends approximately four years in training at a control center before he is fully qualified. The education process is a combination of on-the-job training and supervised practice in the Dynamic Simulator (DYSIM). The later form of learning involves understanding and representing the advice of another expert. A few observations are significant.

(1). Controllers recognize elegant solutions to conflict resolution problems. In fact, they learn by the incremental acquisition of those solutions.

(2). Elegant solutions are characterized by a controller's ability to localize conflicts, relax constraints, and recognize and plan for multiple goals.

a. Conflict localization is the ability to rapidly digest a traffic scenario and predict the conflicts. The controller knows which conflict pairs are related either spatially or by the similarity of the invoked commands.

b. Constraint relaxation refers to situations when a controller elects to 'break' the rules. For instance, a controller may elect to violate an aircraft's protection circle.

c. Multiple goal satisfaction refers to the controllers' ability to utilize a reduced command set to realize several goals. For example, a controller may elect to resolve several conflicts with one command.

(3). A human controller develops his own style of controlling aircraft, yet can understand the plans of another controller. Comprehension is demonstrated by the brief explanations that suffice when one controller 'hands off' his position to another controller.

Experienced controllers represent air traffic control problems in such a way that unneeded details are suppressed, the 'right' things are made explicit, and the resulting plans and their goals are easy to understand. Thus, the human controller's representation of a problem includes those attributes of elegance previously discussed. The purpose of this work, is to give a computer the ability to construct and process such a representation.

## The Conflict Structure

The conflict structure is a global, semantic representation of the aircraft conflict data. This data is derived from a processing algorithm which includes heuristic knowledge of ATC maps [3]. An example heuristic is 'two aircraft probably won't collide if their flight plans call for them to be on airways that don't intersect.' The algorithm produces a space of all conflict pairs during some specified time period. The conflict structure is created by generating a semantic network where nodes are the aircraft involved in conflicts and links describe the conflict type and relative position between aircraft. There are four types of conflicts in enroute air traffic control: head-on, crossing, merging, and overtake. Aircraft position is based on the following semantic descriptors: relative altitude (above, below, and same-altitude) and relative horizontal position (right-of, left-of, in-front-of, and in-back-of). An example conflict situation is shown in Fig. 1. The problem involves six aircraft and nine conflicts during a fifteen minute period. The same problem is used in the upgrade training of human controllers. The data processing algorithm describes each conflict pair in terms of the aircraft involved, the conflict time, and the miss distance. The flight plans of the respective aircraft are then examined to determine the semantic indicators of relative position, relative altitude, and conflict type.

The conflict structure is shown in Fig. 2. The links can be interpreted bidirectionally. For instance, ua86 is in a head-on conflict with aa83. aa83 is below and in-front-of ua86. This implies that ua86 is above and in-from-of aa83. The processing algorithm is also used to find the non-conflict aircraft neighborhood around each conflict pair. For instance, there may be aircraft to the left of ua86 which are not now in conflict. However, an avoidance maneuver to the left by ua86 may cause new conflicts. The neighborhood structure explicitly identifies this possibility. The neighborhood data are not shown in this example.

Figure 1. Typical Aircraft Conflict Problem

Table 1. Conflict Data From Figure 1

| time of conflict (hours) | conflict type | aircraft 1/2 | relative altitude | relative position | miss distance (nm) |
|---|---|---|---|---|---|
| 18.596 | head-on | ua86/aa83 | below | in-front-of | 0.00 |
| 18.644 | head-on | ua86/ua85 | above | in-front-of | 0.64 |
| 18.655 | head-on | aa83/ua90 | above | in-front-of | 0.00 |
| 18.677 | head-on | ua86/aa87 | above | in-front-of | 1.42 |
| 18.693 | crossing | ua85/n47v | below | left-of | 7.15 |
| 18.704 | head-on | ua90/ua85 | above | in-front-of | 2.38 |
| 18.715 | crossing | ua90/n47v | above | left-of | 9.37 |
| 18.727 | crossing | aa87/n47v | below | right-of | 3.56 |
| 18.735 | head-on | ua90/aa87 | above | in-front-of | 1.96 |

## Problem Recognition and Decomposition

Each link of the conflict structure specifies an aircraft conflict avoidance goal of the form:

```
(or
    (> (vertical-sep   ?ac1 ?ac2) *vert-sep-std*)
    (> (horizontal-sep ?ac1 ?ac2) *horz-sep-std*)))
```

The conflict structure can be decomposed into a list of two aircraft problems (a bottom-up approach to problem solving). A solution set of two aircraft problem solutions is formed. The set is then searched for common or contradictory plan segments. Wesson [4] used such an approach in his air traffic control planning system. It is interesting that novice controllers solve these types of problems in the same manner. However, an experienced controller takes advantage of the semantic descriptors to abstract common aircraft and thus form a new conflict structure prior to problem decomposition. For instance, the aircraft ua85 and aa87 are involved in head-on conflicts with ua86. Each is above ua86 at