Combining Human Assessment and Reasoning Aids for Decision-making in Planning Forest Fire Fighting*

Paolo Avesani and Francesco Ricci and Anna Perini
I.R.S.T. - 38050 Povo (TN), Italy
{avesani,ricci,perini}@irst.itc.it

Abstract
In this extended abstract we shall illustrate a work in progress aimed at developing an AI system for planning first attacks to a forest fire. It is based on two major techniques: case-based reasoning and constraint reasoning. This planning component is embedded in an Intelligent Decision Support System aimed at supporting the user in the whole process of forest fires management. The novelty of the proposed approach is mainly due to the use of advanced techniques for lazy learning, the application of the case-based paradigm to the planning of the first fire attack and the integration of the case-based reasoner with a constraint solver, mainly in charge of temporal reasoning.

determining a large variety of world states. Data are always incomplete and uncertain, unless, in some cases, totally absent.
In this framework the role of past knowledge is extremely important, sub optimal solutions are often adopted, because of the bias in the decision process. The management of forest fire, as in general environmental emergency, involves organisations which have decisional and operative centers distributed on the territory. So managing a forest fire can require several centers to cooperate. Moreover complex coordination problems can arise when resources from different organisations, like forestry people, police, ambulances are employed.
These are all general features of the fire domain. To be more specific, the management of forest fires follows a precise operational workflow that is typical of each fire fighting organisation. We shall concentrate on the work organisation in an Italian Provincial center.

Introduction
In this extended abstract we shall illustrate a work in progress aimed at developing an AI system for planning first attacks to a forest fire. It is based on two major techniques: case-based reasoning and constraint reasoning. This planning component is embedded in an Intelligent Decision Support System [Ricci et al., 1994] aimed at supporting the user in the whole process of forest fires management. The novelty of the proposed approach is mainly due to the use of advanced techniques for lazy learning, the application of the case-based paradigm to the planning of the first fire attack and the integration of the case-based reasoner with a constraint solver, mainly in charge of temporal reasoning.

1 The fire domain.
The complexity of the forest fire domain comes from features which are typical of environmental problems. Fire is a dynamic phenomenon whose evolution is determined by weather conditions, in particular wind intensity and direction, by humidity, by fuel type, those parameters which usually change rapidly and sometimes in an unpredictable way. Operational constraints impose often quick decisions that drastically limit the possibility to build a plan [Ricci et al., 1993]. Moreover relevant fire events can happen on very different time and spatial scales, from seconds to days, from meters to kilometers, 2 The operational context.

In order to better understand how the user can be supported in planning an intervention plan we illustrate the operational context in which the system is to be deployed.
The user of the system is the controller based in a provincial centre. His tools are: a workstation, a dedicated line to acquire data from infrared sensors and meteo sensors, a radio, a fax, a telephone and a printer. The system running on the workstation comprises a Geographic information System, a graphical simulator of the fire evolution, tools for territorial, meteo and resource assessment and a module for supporting the intervention planning and control.

When a new fire is reported, the alarm is promptly validated and the situation assessed by the user possibly running the fire spreading model. On the screen the operator can look at the output of the fire spreading model and access, through a graphical interaction, information on the graphical symbols showed by the map. At the end of this phase the operator has acquired enough information for drawing on the map a number of lines sectors that subdivide the original fire front.
This operation, called sectorization, is one of the more crucial decisions because it involves the strategic choices of the fight to the fire: which fire front to take into account in detailing a tactic definition, how far from the
fire epicentre to locate the resources, what kind of supply needs a particular attention (for example inhabited houses, railway paths, etc).

Once the sectors have been identified on the map the operator is now looking for a plan to fight the fire in each sector. The plan may use air forces and/or ground means, and they have to adopt a specific scheme of work. Searching in a data base of past sector plans, the system retrieves a set of plans that in a similar context worked successfully. Follows a modification phase to fit these plans to the current situation. The plans are showed to the operator by means of a predefined form. After this phase of sector plan evaluation the operator may choose to repair a plan editing some subpart. Otherwise he may propose a new one, that seems applicable, based on his experience. The system will check the numeric consistency of the repaired plan: that is verify the temporal constraints, water quantity and resources availability.

At this point the operator has to assign means to actions. He can normally schedule actions or take advantage of an automatic resource allocator. The resource allocator starting from the resource information provided by the assessment looks for a suitable solution that respects the constraints formulated by the intervention plan. If the resource allocator can not find a feasible schedule the operator can manually perform a partial resource allocation.

The operator finally takes a decision: selects a plan and his related schedule and sends the appropriate orders to the bases all around.

3 The general architecture

The general architecture of the Charade decision support system is mainly divided in three subsystems. In figure 1 both functional modules and data types are shown.

An architecture for a decision support system must enable the co-operation of man and machine capabilities. In Charade the focus of this co-operation between man and computer leads to a task-oriented approach to system design, where analyses of the operator tasks are conducted, and task models are built and exploited in the design process. The architecture of the Man-Machine Interface subsystem uses this model of the operator-system activity (notion of task-model) to drive and control the overall dialogue; in particular the activation of planning and situation assessment computer functions is realized from the task co-ordinator contained in the MMI subsystem.

The situation assessment subsystem roughly implements a blackboard architecture. Different knowledge sources (fire spreading model, spatial reasoner, etc.) collaborate to obtain a global description of the problem domain that is contained in the Dynamic Data Space. The system activity task model included in the MMI can be seen as the main control that activates the different knowledge sources.

Regarding the Intervention Planning subsystem an architecture which integrate constraint reasoning and case-based reasoning is proposed [Avesani et al., 1993]. The case-based reasoner plays the role of assumption maker, suggesting old solution to similar situations, and constraint reasoning filters those assumptions in a feasible solution. The user is always called to an active role, for example refining an hypothesis proposed by the case-based reasoner or browsing the constraint network. The following section gives a more detailed view of the Intervention Planning subsystem components.

4 Intervention Planning

The intervention planning subsystem develops an hybrid architecture for planning that integrates a Case-Based Planner and a Constraint Reasoner.

4.1 The Plan Reasoner

The Plan Reasoner is based on the idea of reusing the previous expert experience in planning initial attacks in order to build a plan for the new situation. This idea yields that we don’t need to know the principle of fire fighting planning but we have to focus on recall of past intervention plans and on their adaptation.

Usually a case-based system starts from an initial assumption on the similarity metric and its accuracy is related to the number of instances stored in the case memory. But increasing the collection of cases doesn’t affect the definition of a similarity metric. In this perspective developing a similarity metric is an empirical task strongly related to the domain knowledge [Kolodner, 1993].

Standards approaches to case-based reasoning adopt a generalization of the Euclidean metric, where the squared differences of two points components is multiplied by a real positive factors called weights, i.e.,

\[
d(x, y) = \sqrt{\sum_{i=1}^{N} w_i | x_i - y_i |^2}
\]

These metrics are also called global metrics as they are defined uniformly on the whole input space.

The set of these weights in a case-based reasoning are normally derived from domain knowledge. Such kind of metrics have many limitations, for example is not possible to assert that the relevance of a feature \( f_i \) depends on the values taken by another feature \( f_j \).
We have introduced a special kind of metric that we called AASM (Asymmetric Anisotropic Similarity Metric), that can lead to a well fitted similarity definition. It is based on two basic assumptions. The first one (anistropic) states that the metric is defined locally: the space around a trial case is measured using the metric attached to that case. The second one (asymmetric) states that the distance between two points in a continuous feature space $F_t$ is not symmetric, i.e., $d_t(x_i,y_i) \neq d_t(y_i,x_i)$. In fact we use two different weights for the "left" and the "right" directions.

We can define an anisotropic and asymmetric metric on $\mathcal{C} \times \mathcal{C}$ (where $\mathcal{C}$ is the case space and $\mathcal{C} = \{x_1, \ldots, x_M\}$), which we call $\delta$, such that:

$$\delta(x_i,y) = \sum_{j=1}^{N} w_{ij} d^p_j(x_{ij}, y_{ij})^{1/p},$$

for all $i = 1, \ldots, M$, and $y \in C$, where

$$w_{ij} = \begin{cases} p_{ij} & \text{if } x_{ij} \geq y_{ij} \text{ and } F_j = [0,1] \\ q_{ij} & \text{if } x_{ij} < y_{ij} \text{ and } F_j = [0,1] \\ q_{ij} & \text{if } F_j \text{ is finite} \end{cases}$$

In [Ricci and Avesani, 1995] we proved that AASM can lead to a significant reduction of the number of stored cases still maintaining good accuracy and therefore to a sensible speed-up in run time performances.

Moreover our approach in the similarity metric design would take advantage of the interactive nature of the cbr system. After each retrieval step the user can be more or less satisfied with the case proposed by the system. The user evaluation could be an useful feedback to the system in order to modify his behaviour, that is his similarity metric. In this context to update the similarity metric means to modify the feature weights. A good cbr system should evolve towards a similarity metric that meets the user expectation.

In CHARADE we are going to merge a machine learning technique suitable to develop an adaptive retrieval method. The basic idea is to build a learning automata model [Narendra and Thathachar, 1989; Kokar and Rev elitios, 1993] that modify itself starting from the feedback received by the user interaction. It decreases (reinforcement) the distance between an input case $c$ and the nearest neighbour $nn$ if the user fruitfully can use the past plan; whereas the distance between $c$ and $nn$ is increased (punishment) if the user substantially revises the past plan. This kind of general scheme is called reinforcement learning.

4.2 The Constraint Reasoner

The Constraint Reasoner manages the the constraints defined on a plan, i.e. the temporal and domain constraints on actions and respectively on the associated resources [Perini et al., 1994; Perini and Ricci, 1994].

Actions and plans are characterized by a start time and an end time that define the time interval during which the action (plan) is performed.

From a user point of view, the temporal structure of a plan is represented with the temporal relations that can be defined between those time intervals in the form of a set of Allen’s interval relations [Allen, 1983]. That is, given two plan components, a conjunction of a set of the thirteen Allen’s temporal relations can be stated for them.

From the constraint reasoner point of view an Allen’s relation between intervals is translated into bounded difference constraints between time points.

A bounded difference constraint between two time points $x$ and $y$ is the inequality $y - x \leq a$. Qualitative relationships can be represented using infinity values as distance bounds. For example “$x$ before or equal to $y$” is expresses constraining the distance between $x$ and $y$ to between zero and infinity ($0 \leq y - x \leq \infty$).

Action’s durations are also represented with bounded difference constraints. These bounds comes from evaluation of the resources’ capacity and data on fire evaluation. So estimates on minimum and maximum action duration bound the distance between the start and end times of the action, while a sector deadline gives an upper bound to the distance between the start and end times of the sector plan. Note that all this information can be represented with bounded difference constraints.

A set of bounded difference constraints on a set of time points defines a CSP problem on continuous domains [Dechter et al., 1991]. Different computations can be performed on constraint network, for instance finding the minimal network (computing the Floyd-Warshall all-pair shortest path algorithm), or checking the consistency of a constraint network (computing the Bellman-Ford’s single-source shortest path).

For example each time a user adds a new constraint or attempts to modify the definition of a previously installed constraint, the constraint network is checked against possible inconsistencies and the new domain extensions for the temporal variables are computed using shortest path algorithms. In such a way the user is enabled to incrementally define a plan, by adding or removing actions and constraints, while the system checks the correctness of these operations with respect to the temporal dimension.

Acknowledgments

Charade Project has been managed by Alenia. The other partners are Alcatel (Software Platform), Thomson CSF/SDC (Man-Machine Interface), Insel (Situation Assessment), Italsoft (Resource Manager and Resource Allocator), Thomson CSF/LER (cartographic module).

References


the experimental knowledge will be modelled by means of cases or experiences. This cooperation tries to get benefit from the advantages of both kinds of knowledge, and to cope with typical shortcomings either from knowledge-based systems: do not learn from experience, the knowledge acquisition problem, the brittleness; or from automatic control systems: complexity of the processes, ill-structured domains, non-numerical or qualitative information, uncertainty or approximate knowledge. This paradigm combination integrates in a single architecture some cognitive processes as knowledge-based reasoning, case-based reasoning, learning, knowledge acquisition, problem solving [Plaza et al., 1993].

2 General expert knowledge

In this paradigm, the knowledge about the domain is modelled with inference rules. The process of extracting these rules (knowledge acquisition) from experts—with interactive sessions where the experts try to make explicit their knowledge and reasoning processes—is quite often described as a bottleneck. Therefore, much effort in AI has been addressed to overcome it. In this case, the knowledge acquisition process have been done with Linneo$^+$ [Béjar, 1995] (a semi-automatic classification tool). The main objective of Linneo$^+$ is to build classifications for ill-structured domains; where much imprecise information exists, it is assumed that observations vary in their degree of membership with regard to each class. Bearing all this in mind, the use of the conventional concept of distance as a fuzzy similarity value is used.

A crucial point for the WWTP supervision is the concept of working situations. A situation is an operational working state of the plant, described by measures of the relevant attributes of the process. The experts characterized 20 working situations of a plant using 23 attributes [Serra et al., 1994]:

- bulking-non-filamentous
- bulking-sulphures
- bulking-not-enough-oxygen
- bulking-too-much-oxygen
- bulking-F/M
- bulking-toxic-substances
- high-plant-inflow
- bad-carriers-operation
- overloading
- bad-primary-settlers-operation
- bad-wasting
- in-plant-overloading
- toxic-substances-loading

Each situation could be defined in terms of raw descriptions and relationships. For example, the bulking-non-filamentous situation was defined as follows:

- Outflow-COD $\rightarrow$ High
- Sludge age $\rightarrow$ Old
- Filamentous $\rightarrow$ Normal
- SVI $\rightarrow$ High
- Volatile SS-recircul. $\rightarrow$ Low
- All-other-attr. $\rightarrow$ Nought-value (don't care)

The results of the classification process with Linneo$^+$ provide intensive and extensive description of the generated classes (situations) and a fuzzy membership matrix that relates observations (data) to generated classes. At this point, inference rules can be generated. This process is depicted in figure 1. Bearing in mind the prototype of a class and the superclassification structure, inference rules that lead the diagnosis process in the target system, can be derived. These rules identify to which class (set of situations) belongs a given observation (data). Identifying rules characterize the values that descriptors of new data must show for being member of a class. For instance a rule as the following could be generated:

\[
\text{(if } \text{(COD-EXIT HIGH SVI HIGH SS-RECIRCULATION LOW SSV-RECIRCULATION LOW) VERY-POSSIBLE (INFER CLASS-3A))}
\]

Other discriminating rules must be provided to discriminate to which situation—with a class—belongs a given observation. For this task is taken into account the fuzzy membership matrix. For example:

\[
\text{(if } \text{(CLASS-3A SLUDGE OLD FILAMENTOUS NORMAL) ALMOST-SURE (INFER BULKING-NON-FILAMENTOUS))}
\]

![Figure 1. General expert knowledge](image-url)
This process is semi-automatic, so that requires the experts final validation. All these rules are analyzed (subsumption detection, synonymy analysis, etc.) and, afterwards, they are organized in a hierarchical way to guide the diagnosis process. The results are made known to the experts who can accept and confirm them, or have the chance to go back to the classification process.

When all these actions are over, rules can be validated using new observations not previously included in the classification sample. As soon as rules are accepted by experts, they can be incorporated into the KB’s of the distributed architecture. These rules capture the subjective domain knowledge of the experts in their daily work at the WWTP. From the several KB’s can be done the diagnosis process that leads to identify the generic working situation(s) of the plant.

3 Specific experimental knowledge

In this approach, the knowledge about the practical problem solving in the domain is represented by means of cases or experiences (in this case situations), which are organized in the Case library. This Case library contains information about previously detected situations and solutions given to them as well as their efficiency (specific experimental knowledge). A Case-based reasoning can be performed in order to get benefit from these past experiences and cases. The Case library will be modified accordingly with the new information [Kolodner, 1993; Schank and Slade, 1991; Riesbeck and Schank, 1989].

Cases denote a working situation of the plant. These cases are previously experienced situations, which have been captured and learned—in such a way— that they can be reused in the solving of future situations. The reasoning process in the Case-Based Reasoning and Learning agent is performed by a general cycle described by the following steps:

- Retrieving the most similar case(s) (previous working situations) by means of some heuristic functions or distances, possibly domain dependent.
- Adapting or reusing the information and knowledge in that case to solve the new problem (the current working situation of the plant).
- Evaluation of the proposed solution. Usually, it is performed by simulation, by questioning to human oracle or by future checking of effectiveness.
- Learning the parts of this experience likely to be useful for future problem solving. The agent can learn both from successful solutions and from failure ones. This retaining is made by updating the Case library accordingly.

![Diagram of Case-Based Reasoning and Learning](image)

The initial Case library is fitted with some situations obtained by Linneo+ classification, from a real data stream of 521 data (days) corresponding to the period 1990-1991. Each data is described by means of the daily mean of 39 variables. That study [Sánchez et al., 1994a] provided a classification of the real specific working situations of the concrete plant. It is interesting to notice that with these data Linneo+ discovered that it exists four subtypes of normal situations (usually, not considered by the experts) and revealed that about 20% of the variables provided by the plant's operators were not relevant for the characterization of situations. The outstanding situations were:

- Toxic substances loading
- Normal (4)
- Primary-treatment problems
- Solid’s shock
- Plant problems
- Storm
- Secondary-treatment problems

So, the Case library contains a set of experienced situations of the plant (specific situations). It evolves from initial contents and captures the experimental knowledge of the concrete plant under control, learning either from its successfully solved situations (plans) or from its failed ones (objective knowledge). All this process is shown in figure 2.
4 The integrated and distributed supervisory multi-level architecture

The proposed architecture (called DAI-DEPUR [Sánchez et al., 1994b]) is a distributed and integrated supervisory multi-level system (see Figure 3). It is a distributed architecture so that is formed of several interacting subsystems (agents) that can be executed in parallel. For instance, the supervisory agent, the case-based reasoning agent, primary settler-KBS agent, biological reactor-KBS agent, etc. Distribution criteria are based on spatial and semantic distance [Bond and Gasser, 1988]. The main reason to choose a Supervisory Distributed AI System is because for WWTP there is a set of fixed abnormal situations as Storm, Bulking, Toxic load, etc., that may be treated with a predetermined plan or strategy, in a more efficiently way than other types of DAI architectures as Blackboard Systems or Contract Nets (see [Sánchez et al., 1994b] for a more detailed description).
DAI-DEPUR is an integrated architecture because of joining in a single system several cognitive tasks as learning, reasoning, knowledge acquisition, problem solving, etc. Moreover, focusing on the expertise level [Steels, 1990] – that is the aim of this paper –, there is the integration of the two paradigms of knowledge modelling.

Also, the architecture is multi-level, and provides independence to all the levels. Taking into account the domain theory (models), it can be structured as a four-level architecture: 1) data level, 2) expertise level, 3) situations level and 4) plans level.

Data level
It is formed of on-line data gathered from sensors and off-line values provided by the operator as laboratory analysis, subjective information, etc.

Expertise level
Modelled by the two paradigms or approaches explained in this paper: general expert knowledge and specific experimental knowledge.

Situations level
The global operating situation of a plant is obtained by combination of its several subsystems' local situations.

Plans level
At this level, the identified whole situation, some previous solved similar situations as well as predefined (canned) plans are taken into account to propose a first solution, that has to be validated against the operator, who can modify the proposed plan. Then, an arranged plan can be executed to cope with the actual operating situation of the plant. Plans are a sequence of actions to be taken in order to restore the plant performance.

By the other hand, considering the processes acting over the models (methods) the architecture can be decomposed in a six-level processes or phases: 1) evaluation level, 2) adaptation level, 3) diagnosis level, 4) supervision level, 5) validation level and 6) actuation level. The system activates a new supervisory cycle at fixed intervals of time.

Evaluation process
For this purpose it is necessary to know some values for certain variables of the process. All this data can be extracted from the evolutionary Data Base, fitted either with the on-line sensors values coming from the data collecting systems or with some other features provided by the operator (like a laboratory analysis, qualitative observation, etc.).

Adaptation process
This is a process that is sometimes performed either by dynamic learning from past proposed solutions and its efficiency – that can update the Case library – or by acquiring some new knowledge from (new) experts or (new) sources through classification techniques.

Diagnosis process
In a new cycle the Supervisory agent activates the Knowledge-based agents to diagnose the state of the different subsystems of the plant by means of rule-based reasoning. At the same time in the diagnosis phase the Case-Based Reasoning and Learning agent (CBRL) is activated to retrieve similar cases recorded in the Case library. This means concurrent execution of all agents involved. Next, is updated the most similar one in order to adapt it to actual situation of the plant. For this task, needs to access to Data Base. The results are communicated to the Supervisory agent.

Validation process
The system can be inquired by the operator in some ways as asking for explanations, retrieving certain values, etc. The Supervisory agent waits for the operator's validation of actions to be taken in order to update the current working state of the plant.

Actuation process
The Supervisory agent recognizes situations and uses the right strategy or plan, in order to keep the process controlled or if normal situation has been detected, then, the automatic numerical control is maintained or activated. If there are on-line actuators, the plant can be automatically updated through the Actuator system. If not, manual operation is required.

5 Preliminary Results and Evaluation
The knowledge-based paradigm (expert knowledge) is already built-up. Knowledge-based agents are implemented using the G2 shell (in prototype status). They are incrementally validated with some real data stream taken from the plant, against the experts' opinion and some simulation studies, yielding good results [Serra et
The case-based reasoning approach (experimental knowledge) is under development. The Case library is being implemented as a prioritized discrimination network, where the priority of node-attributes network is obtained from experts' opinion and from an inductive learning method (such as ID3, etc.). Cases retrieving and adding tasks are designed and, now, we are working on adaptation and evaluation steps. This agent is implemented in Lisp, such other agents (supervisory, etc.).

Data collection module is also developed. It gathers the values from status of turbines, pumps, automatic grids, etc., from the control panel of the plant under supervision, with 6 PLC's (SISTEL 8512). These 96 digital signals are transmitted through RS-422 to the monitoring computer (PC) for evaluation process. It also receives 9 analogue signals (inflow, wasting flow, recirculation flow, biogas produced, DO-line-1, pH, temperature-digester-1 and temperature-digester-2) converted by an AD/DA card. It will be connected to the main computer (SUN Sparc station) where are running all the other agents and processes.

After the validation on the simulation of the plant, the prototype system is currently being validated—in an incremental way—in a real plant under contract between our universities and "Junta de Sanjament de la Generalitat de Catalunya" as the responsible organism of wastewater treatment plants management in Catalonia. Until now, the results obtained with the system are promising.

6 Summary

From what has been previously described, one can state that the combination of both paradigms at the expertise level, let the system to model the subjective knowledge (supplied by the experts) as well as the objective knowledge (supplied by the real operation of the concrete plant under control)\(^1\). This integration presents some advantages that are the addition of the own ones from knowledge-based reasoning, from case-based reasoning, from dynamic learning and from semi-automatic knowledge acquisition, that are the methods acting at the expertise level:

\[ \text{It makes possible to reason in a poor understood and ill-structured domain, where other kinds of reasoning like model-based reasoning or algorithmic reasoning could not be possible.} \]

\[ \text{• Let the system to learn from previously solved problems and to adapt the available experimental knowledge over the domain (dynamic learning environment).} \]

\[ \text{• Overcoming the brittleness of KBS' in coping with unforeseen situations (not previously considered by the general expert knowledge), trying to solve them by means of the most closely situation in the Case library.} \]

\[ \text{• Capturing the knowledge provided by the experts (knowledge acquisition) which is very important—although subjective— to get a central corpus of knowledge about the domain.} \]

\[ \text{• Dealing either with prototypical situations (general knowledge) or with idiosyncratic or exceptional ones (specific knowledge).} \]

\[ \text{• Due to the dynamic learning environment, the} \]

\[ \text{system is able to adapt itself to different wastewater treatment plants, making the system to be exportable to any plant with some minor changes. It is only needed to fill the Case library with an initial set of specific cases (operating situations of the concrete WWTP), which can be obtained semi-automatically from real operational data.} \]

All these facts make it more powerful than other single technologies applied to wastewater treatment plants as knowledge-based approaches [Lapointe \textit{et al.}, 1989; Maeda, 1989], statistical process control techniques [Novotny \textit{et al.}, 1990], fuzzy controller methods [Czoaga and Rawlik, 1989; Alex \textit{et al.}, 1994], etc., as well as to other complex ill-structured domains. With this approach, the plant can be controlled in normal situations (mathematical control), in abnormal usual situations (expert control) and in abnormal unusual situations (experimental control).

Acknowledgements

Currently, it is been partially supported by the "Junta de Sanjament de la Generalitat de Catalunya" and the Spanish CICYT project RO89-679 and EEC project VIM ERCHXCT 930401. Also the authors wish to acknowledge the cooperation of Ricard Tomàs, manager of the Manresa's wastewater treatment plant.

References

Intelligence in Real Time Control (AIRTC'94), pages 75-80, València, October, 1994.


