[*]An Application of Artificial Intelligence for the Safety in the Neighbourhood of Airport Runways

Christophe Blaess
Cabinet Conseil Blaess
132, avenue Guy Mocquet
94 340 Joinville-le-Pont, France.
ccb@club-internet.fr

Claude Tsiampalidis
Aeroports de Paris
OA.Z.NA - Orly Sud 103
94 396 Orly Aerogares Cedex, France.
(+33) 01-49-75-72-04

Jean-Charles Vallee
Services Techniques de la Navigation Aerienne - 3RA
1 avenue du Dr. Grynfogel
ZAC de Basso-Cambo, BP 1084
31 035 Toulouse, France.

Abstract

With the traffic density augmentation, the importance of surface movements on the airport runways and in their neighbourhood is always increasing. The probability of collision, and near-collision in this area is also getting higher. Most of the national civil aviation agencies take part in the SMGCS project to improve safety while aircrafts land, takeoff on runways or simply run on taxiways. An experiment is performed at this time on the two main French airports, using an expert-system with real-time abilities to prevent them from dangerous situations. We used a pragmatic approach to the problem, setting more focus on effective implementation and operational features than on truly intelligent capabilities of the knowledge base processing. The expert-system has been built after several previous experiments with the help of Air Traffic Controllers from several airports. The knowledge base is analysed and compiled first in order to provide an efficient system, able to cope with the deadlines of real-time requirements. The real-time features of the system have been verified, and the present experiment will validate the logical aspect of the knowledge base acting on real operational data.

[*]This paper appears in the proceedings of the IEEE ICTAI'96 Workshop on Artificial Intelligence for Aeronautics and Space, November 16th 1996, Toulouse France.

1. Introduction

We will introduce, in this paper, an experiment running at present at the Roissy-Charles-de-Gaulle and Orly airports (Paris, France). The aim of this project is the improvement of the safety on the airports runways, and in their neighbourhood, especially during landing and takeoff phases. Several previous experiments led us to use technics deriving from Artificial Intelligence Science, in order to detect potentially dangerous situations, and warn the air controllers in the nick of time.

We will first present the needs for runways safety improvement, and the integration of this problem in the worldwide SMGCS project (Surface Movement Guidance and Control System). Then we will deal with the historical background of our project, and the previous experiments. Finally, we will detail the present experiment, the architecture of the actual system, and the first validation results.

2. The need for runways safety improvement

The most critical phases during an aircraft flight are takeoff and landing processes. Because of traffic density, and dependencies to visibility conditions, pilots and drivers need a reliable assistance from the surface controllers. Because of increasing number of simultaneous events, due to traffic augmentation, the work of the Air Traffic Controllers get more and more difficult, increasing the probability for errors and mistake [7].

In the USA, a statistical analysis of F.A.A. (Federal Aviation Administration) data, from 1975 through 1989, shows that surface collisions occured three times as often as those in midair. [21] One of the worst disaster in aviation history took place on the runway of Tenerife Airport, on March 1977, with the collision of two Boeing 747, killing 583 persons. FAA statistics also shows that while actual runway collisions don't happen often, there are an increasing number of near-collisions.

The main problem is the increasing number of runway incursions [20]. In order to avoid runway intrusion, ICAO (International Civil Aviation Organization) defines a general framework to help taxiing on the airports. This system is called SMGCS (Surface Movement Guidance and Control System) and is widely developped on international airports [11]. The SMGCS features includes painted marks, signs, stop bar light... A lot of national civil aviation agencies tries to extend SMGCS features, especially for automatic detection of runways incursion, in order to warn surface controllers.

3. Historical background of our project

3.1. The first basis of SALADIN

In 1987, ADP (Aéroports de Paris — Airports of Paris) decided to begin a study which preliminary aim was the detection of runways incursions. This project called SALADIN (Systeme d'Aide à LA Détection d'INtrusion — Helping system for intrusion detection). set focus on detection and localization of any mobile — aircraft or vehicle — in the neighbourhood of the runways. The problem at this time was distinction between authorized mobiles, which have received clearance from the controllers to enter the runway, and unauthorized intrus.

With the help of surface controllers a minimal set of rules has been defined to identify unauthorized mobiles and trigger an alarm. The potentially dangerous situations were described with the formalism of production rules because of the ability of this declarative knowledge to be understood by non computer specialists especially the Air Traffic Controllers. At this point, we didn't think that an expert system would have been necessary, the number of rules (about 30) allowing a classical programming approach. A first prototype of the kernel of SALADIN was developped, and tested on simulated data coming from ideal perfect sensors. The prototype was limited to a single runway.

3.2. Improvement of SALADIN

The first prototype worked fine with ideal information about positions of each mobiles in the neighbourhood of independant runways. But this first set of rules was poor with global airport traffic such like Orly platform with crossed runways (runways 2 and 3, see figure 2).

In 1989, a refinement of the rules was done, leading to a bigger base, which needed a real expert system kernel, build at this time in Prolog. While the first knowledge base were only 0 order predicates, for example ”an_aircraft_land_on_runways”, or ”a_vehicle_in_the_90_meter_area”, the second set of rules (about 100) relied upon 1 order rules, using variables such like ”landing_on_runway (aircraft, runway)”, ”in_90_meter_area (vehicle, runway)” or ”in_front_of (aircraft, vehicle)”.

A second phase of tests has been run with simulated ideal data, taking account of the whole airport, including crossed runways.

3.3. First validation of the set of rules

The second set of rules was validated by the air controllers, but we had in 1993 to set focus on the real-time aspect of the project. The deadline reaction time was decided to be one second between data acquisition and alarm trigering. The first system built in Prolog was rather fast, but with backward chaining used at this time, we couldn't control precisely the maximal reaction delay of the system. We decided at this time to built our own inference engine using only manually compiled rules, in order to be sure of the maximal delay for alarm trigering.

The second system was built in C++ using a forward chaining process, in which the selection and ordering of the rules was fixed at compile time. The results allowed us to validate the timing aspect of the system. The input data were still simulated, but they were closer of real data format, and come from a graphic simulator through a serial link, with same rythm than real available information. Obviously the problem with a manually compiled knowledge base was the difficulty to modify the rules, and to be sure at each modification that we had not inserted loop.

Simultaneously, an adaptation of the set of rules to the Toulouse-Blagnac Airport has been conducted by ENAC ingeneers [10], thus proving the portability of the basical rule set to any other airports. They also have shown that an accident such as those of Tenerife would have been detected by the knowledge base, allowing the Air Controllers to act on one of the two boeings to avoid collision, or at least reduce the impact speed.

3.4. Present project : real-time and real-data processing

In 1995, our present project begun, with two main focus: using real information coming from external sensors (radar, radio-localization systems, flight plan databases, ...) and allowing easy modifications in the knowledge base while keeping real-time computing features.

In order to allow easy modifications in the rule base, we designed a simple language to express rules, and built a compiler wich translate these rules in C procedures. The rules ordering and analysis is performed at compile time, in order to eliminate any risk of loop in the reasonning. The compilation gives a set of C source files, ready to be compiled and linked with all the data acquisition and preprocessing modules.

This project will be detailed in the next section. We have already tested intensively the compiler, the data acquisition module, and the global features of the system. The present experiments will validate the knowledge base of the system (about 150 rules) with real data .

4. Presentation of our system

4.1. Global overview of the project

The expert system is integrated in a global system including a lot of different tasks (Fig 1).

First a data acquisition is performed from several redundant sources allowing mobile localization (terminal radars, surface radar, radio-positioning system). Then these data are processed together with information coming from other sources (flight plans,...) to perform a data fusion which delivers ethernet frames including identity, position, status,... concerning every mobiles on the airport and every planes in approach. The frames are sent using a specific category [14] included in the general Asterix european protocol [8]. These data are received by HMI (human-machine interface) system, displaying all the synthetical information for the controllers.

Simultaneously, the expert system receives and processes the ethernet frames coming from the Data Fusion Unit, the meteorological data concerning the runways, and information about stop-bars and lightning systems in order to send alarm or simple warning messages to the HMI system. At present the alarms and warnings are not displayed by the HMI system, and are just displayed on another terminal, but the effective integration of alarm messages will be done soon.

A study as been performed by ENAC engineers [10] to optimize the way to display alarms especially in the case of several simultaneous alarm messages.

The operational specifications of Saladin [1] tell that the system must be able to supervise 200 mobiles, with a maximal response-time delay of 1 second. The probability of accident caused by the system must be lower than 10^{-7}. The system must be up 24 hours a day with annual disponibility rate over $99,5\%$.

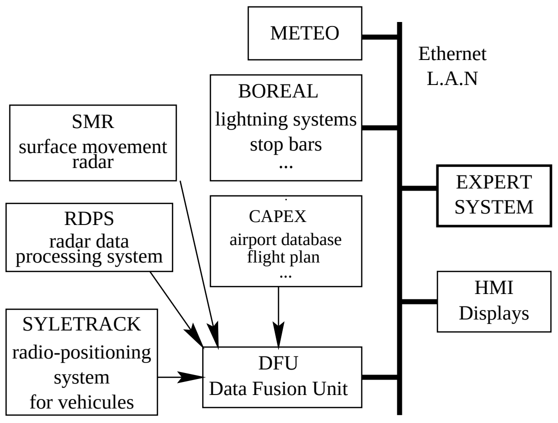

Figure 1. Overview of the whole system.

4.2. Knowledge base

The aim of the project was to improve the safety on the runways, in their neighbourhoods, and during final approaches. So we limited the influence of the expert system to an area of 150 meters around the runways, and to the last 10 NM (18 km) of approaches trajectories. The neighbourhood area are represented with dashed lines on figure 2.

We have have defined a minimal set of data needed to detect potential dangers. These data include:

- Position of each mobile on the supervised area. The required precision is about 50 meters during final approaches, and about 5 meters on the ground.
- Speed of the mobiles. A real instantaneous speed value would be very usefull, but is rarely available. Indeed we use generally a calculated value of speed, using the last positions of the mobiles.
- Nature of the mobile (aircraft, helicopter, cars, or ”unidentified”). The unidentified mobiles are for example technical vehicles which are not equiped with radio-positioning system. These vehicles are not allowed to circulate in neighbourhood of a runway when this runway is in use.
- Eventually we use some additional information (like acceleration to determine pull-up occurence).

The knowledge base of the expert system includes about 150 rules, and may be divided in three groups each containing about 50 rules.

- First, there ist a set of rules to analyse the data coming from ethernet frames, in order to complete the information concerning each mobiles. These rules are not concerned by the air controllers expertise, but it's important that this technical knowledge is expressed in a declarative form, to allow easy modification of the vision of the expert system. An example of such a rules is "*if* (mobile . main_data_source = SMR_RADAR) *and* (mobile . correlation_sources = SMR_AND_SYLETRACK) *then* mobile . nature = VEHICLE"

- The second kind of rules are those allowing an analyse of global situations by incorporating local information about mobiles. These rules concerne the status of the runways, the kind of movement done by the aircraft (landing, taxiing, taking-off), and relations between several aircrafts (landing-after, ...). For example such a rule could be: "*if* is_landing (aircraft_1, runway) and in_final_approach (aircraft_2, runway) *and* (aircraft_1 . Qfu = aircraft_2 . Qfu) *and* (aircraft_1 . Speed > 5) *and* (distance_from_offset (aircraft_1, runway) > 2500) *and* (runway . meteo = METEO_FOR_LAND_AFTER) *and* (runway . status = DRY) *then* is_landing_after (aircraft_2, aircraft_1, runway)"
- The third group includes the rules determining danger occurences on the runways, neighbourhoood and final appproaches. These rules have been totally defined by the air controllers and are obviously the main expert part of the knowledge base. One of these rules could be "*if* is_landing (aircraft, runway) *and* in_90_m_area (vehicle, runway) *and* in_front_of (aircraft, vehicle) *then* alarm (1, aircraft, vehicle, runway)" This specific rule seems to be obvious, but most of them are not, especially those including several aircraft landing/taking off on different runways.

As it can be seen above, the rules include 1 order predicates with variables. A carefull analysis of the knowledge base after definition by the air controllers showed us two important points:

- Each second, we start with some known facts, deduced from the ethernet frames which are processed by the first set of rules to have a correct view of each mobile, then we can apply the second set of rules, building a global appreciation of the situation on the airport, and finally we can perform the last deductions about eventually dangerous events, with the third set of rules. This structure led us to use a forward chaining system, applying each of the selected rules to perform a saturation of the knowledge base.
- In each of the three groups of rules, we saw that we can totally avoid loops. This means that we could formulate all the knowledge to avoid any situation like two rules "*if* A *and* B *then* C" and "*if* C *and* D *then* A". This point has been very important during the analysis of the real-time aspect of the project.

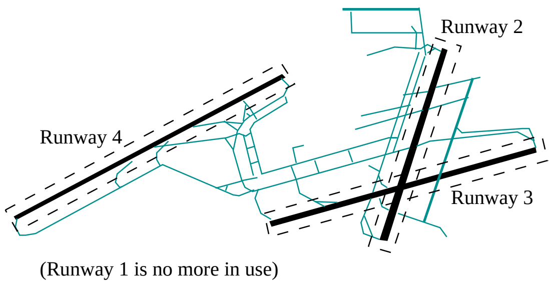

Figure 2. Runways of the Orly Airport

4.3. Rules compiler

The real-time aspect of the system implies some specific adaptations of classical knowledge based technics. A real-time system has to respect predefined deadlines during his computations [19]. Moreover for A.I. based real-time systems, the main requirement is not a fast computation, but a *predicabily* fast system. We will first set focus on the "fast" aspect, then on the "predicably" aspect of the system[17].

Several options are allowed to incorporate A.I. tools within a real-time system [4][9][18], and we choosed what seemed to be the easiest way: reducing the AI performances, while keeping all the tasks of the search under timing deadlines.

To do this we have first decided to compile all the rules of the knowledge base into efficient C procedures. We build a compiler using classical Unix tools (Lex and Yacc) [15]. The compiler take a knowledge base expressed as simple declarative statements, and produce a C source code for each rules (see figure 3). The output source code is designed to be human-readable [3] [13], in order to validate the compiler features.

The knowledge base uses a simple object representation (no inheritance) for the main concepts of the expert domain (aircraft, vehicule, runways,...). Some predicates are directly provided each second to the system by the preprocessing of the data (for example "in_150_m_area (FLYCO_8, RUNWAY_4)"), while the others are deduced from the rules.

Some predicates are just internal representation of the situation (for example "has_just_landed (AF1415, RUNWAY_3)", and some others are used for interfacing the knowledge base with the external world ("alarm (CASE_5, RUNWAY_3, FLYCO_8)"). This predicates are used to call external C procedures provided in a library, and perform desired actions (in this case send an alarm message to the controller).

Symetrically, there is a way to incorporate in the rules the results of some external C functions for several purposes

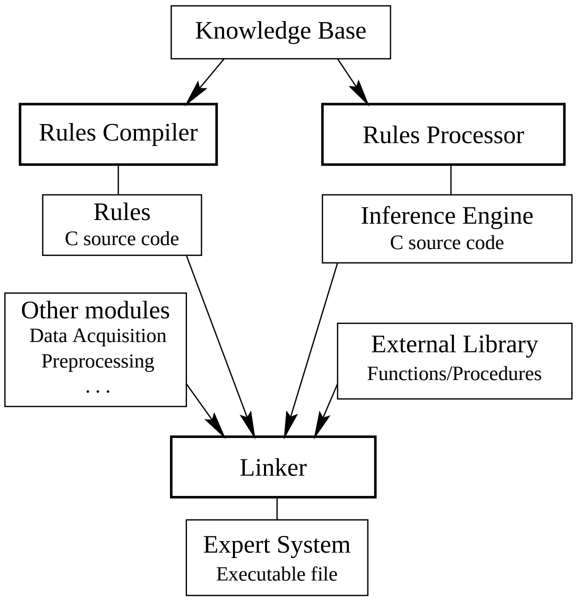

Figure 3. Expert system compilation.

(providing mathematical computation, acquiering data from future external sources,...).

Finally we have a set of C procedures, each of them corresponding to an initial rule.

4.4. Rules ordering

The second important point in our system was to avoid any loop in the reasonning process, this means beeing able to provide knowledge base saturation each second, with one pass over the set of rules. We had to dramaticaly limit the possibilities of the expert system [12], but it was necessary to guarantee predicability of the timing behavior.

After analysing the set of rules defined by the air controllers, we decided that the system will not accept that a rule using a predicate ”X” (or the value of an object attribute) to produce a predicate ”Y”, coexists with a rule using the ”Y” predicate to produce ”X”. We also ensure that this situation could not appear through a chain of rules.

This means that all the rule base can be represented by a graph whose nodes are the predicates (or object attributes setting), and whose edges are the rules. The graph is layered, and the edges can only be used in a bottom-up way. The bottom level of nodes is constitued from basical predicates and mobile attributes values which are set during the acquisition and preprocessing of data, the intermediate levels contain the internal predicates used for knowledge expression, and the upper level contains the output predicates, in our case trigering of alarm messages or information warnings.

One must not forget that the nodes of this graphs represent predicates acting on variable objects, and not only propositional predicates. Otherwise a connectionist approach would probably have been more indicated for the problem.

While compiling the rule base, the system also analyses the graph, eventually aborting compilation if loops possibilities occurs. Then it can order the rules in the best way to produce saturation of the knowledge base in one pass. Optionally it can do optimization, by avoiding the test of a whole set of rules if it relies upon internal predicates produced by a set of rules that have all failed.

The complexity of each rule depends on the number of variables used in its declaration and is generaly $O(n)$ or $O(n^2)$, n beeing the number of mobiles in the supervised area. With a one-pass processing of the rule base, the whole system keeps running under a polynomial complexity.

5. First results

The first test phase set focus on the computational features of the system. We had to verify the functionalities of the compiler, the effects of rule ordering, and the correctness of the other modules (data acquisition, preprocessing, alarm message transmission...). Then we ensure the real-time aspect running the system with the whole set of rules and an average load of the airport traffic.

This results shows us that the computational aspect of the expert system was satisfying. The response time is clearly under the one second deadline. The system has been tested on a i486DX4/100 computer running under Linux 2.0.0.

The second test phase set focus on the rule base, and improves the set of rules in order to cope with real data. This phase is running at present. We have already defined the rules of the first group (those who process incoming data to compute local information about mobiles). The second group of rules still needs some refinement, in order to avoid some problem with noisy information (echo, double positioning of mobiles, delay with data coming from primary radar). The third group (dangerous situations detection) will be validated and enhanced with the help of air controllers.

In order to validate completely the rule base, we are actually building a system which allows recording of real-data, and edition of this data to add for example simulated trajectories within the real situation. This system will also allow the validation of the real-time aspect of the expert system with a maximal load of the airport traffic.

One of the main requirements for the expert system is to maintain a very low rate of false alarm, while keeping danger detection as high as possible. This is a major need to obtain a real credibility for the system with a daily use in operational environment.

Some enhancements we can foresee concern final approaches processing (when the aircraft is doing bayonet for example), and some dynamical anticipation of movements on taxiways.

References

[1] ADP. Saladin, spécifications opérationnelles. Technical report, Aéroports de Paris ADP/OAZCA, February 1995.

[2] F. Balcells. Guidance and control on the A-SMGCS. *Airport / Air Traffic System Interface Workshop*, April 1994.

[3] M. Brader, D. Kepple, H. Spencer, L. Cannon, R. Elliot, L. K. J. Miller, R. Mitze, E. Schan, and N. Whittington. Recommended C style and coding standards. Technical report, AT&T Bell Labs, 1991. also known as Indian-Hill specifications.

[4] I.-R. Chen and T.-W. Tsao. A simulation methodology for the response time distribution of AI-embedded real-time systems. Technical Report MS 38677, University of Mississippi - Dept of Computer and Information Science, 1992.

[5] I.-R. Chen and T.-W. Tsao. A reliability model for real-time rule-based expert system. Technical Report UMCIS-1994-06, University of Mississippi - Dept of Computer and Information Science, 1994.

[6] DGAC. Réglementation de la circulation aérienne. Technical report, Direction Générale de l'Aviation Civile, April 1992.

[7] F. Dominguez. The Spanish A-SMGCS Concept. *Airport / Air Traffic System Interface Workshop*, April 1994.

[8] Eurocontrol. Proposed eurocontrol standard for radar data exchange. Technical Report 005-1-93, EuroControl, February 1995.

[9] M. J. Fasciano. Real-time case-based reasoning in a complex world. Technical Report TR-96-05, University of Chicago - Computer Science Dept, 1996.

[10] J. Fournier and D. Reitz. *Adaptation des regles de prévision de collision de Saladin á la plate-forme de Toulouse Blagnac et conception d'une maquette d'interface homme-machine d'un SMGCS pour grande approche*. Ecole Nationale de l'Aviation Civile, Juillet 1996.

[11] ICAO. *Manual of Surface Movement Guidance and Control Systems*, 1st edition, 1986.

[12] H. Kauz and B. Selman. A general framework for knowledge compilation. *Proceedings of the International Workshop on Processing Declarative Knowledge*, July 1991.

[13] B. W. Kernighan and D. M. Ritchie. *Le langage C*. Masson, 2nd edition, 1993.

[14] E. Kjeilen and A. Gilbert. Target tables asterix format cat 081. Technical Report SA-0168/A/29-Mar-96, Kongsberg Norcontrol AS, March 1996.

[15] J. R. Levine, T. Mason, and D. Brown. *Lex & Yacc*. O'Reilly International Thomson, 1995.

[16] D. J. Musliner, E. H. Durfee, and K. G. Shin. Integrating intelligence and real-time control into manufacturing systems. *SIGMAN Workshop on Intelligent Manufacturing Tehnology*, July 1993.

[17] D. J. Musliner, J. A. Hendler, and A. K. Agrawala. The challenges of real-time AI. *IEEE Computers*, 28(1), 1995 January.

[18] D. J. Musliner, K. G. Shin, and E. H. Durfee. Automating the design of real-time reactive systems. *Proceedings of Symposium on AI in Real-Time Control, Valencia Spain*, October 1994.

[19] F. Panzieri and R. Davoli. Real time systems: A tutorial. Technical Report UBLCS-93-22, University of Bologna - Laboratory for Computer Science, October 1993.

[20] E. H. Phillips. FAA program aims to reduce runway accidents. *Aviation Week and Space Technology*, April 1995.

[21] L. Reingold. Dangerous ground. *Air Transport World*, September 1991.

[22] P. Winston. *Artificial Intelligence*. Addison-Wesley, 2nd edition, 1984.