A collision detection and resolution multi agent approach using utility functions

George Valkanas, Pantelis Natsiavas, Nick Bassiliades

Department of Informatics
Aristotle University of Thessaloniki
Thessaloniki, Greece
{gbalk, natpan, nbassili}@csd.auth.gr

Abstract— **Free Flight is the concept introduced by NASA and FAA in order to change the aviation of the 21st century, allowing for pilots to choose dynamically ("on the fly") their nominal paths. Despite its many advantages as opposed to today's situation, the free flight concept raises many new issues that need to be addressed before being applicable, with the main interest focusing on conflict detection and resolution (CD&R), in order to ensure the safety of the aircrafts. In this paper we present a decentralized CD&R approach using agents, as well as a general multi-agent framework where not only the proposed but new agent-based approaches may be implemented and tested.**

Index Terms— ***Collision detection and resolution, Multi-Agent systems, Utility functions***

I. Introduction

The Free Flight concept was introduced by NASA and FAA as a means to address the increasing demand of air traffic, by utilizing the airspace more efficiently. Free flight, as introduced in [1] is "a safe and efficient flight operating capability under instrument flight rules, in which operators have the freedom to select their path and speed in real-time". Therefore, the trajectories selected by crews would be a lot shorter, nearly optimal, by taking advantage of favorable winds, thus increasing environmental and economic benefits by consuming less fuel per aircraft (green aviation). Delays will also be minimized in most, if not all, aspects of aviation (during flight, at terminal stations, etc.). A decentralization of control could also be achieved by transferring portion of the required computations to the aircraft side, reducing the workload of the Air Traffic Control (ATC) centers, therefore requiring less computational power at the ATC end and removing the risk of a central point of failure. Given the latest technological advantages in Global Positioning System (GPS), data-link and powerful on-board communication as well as Traffic Alert and Collision Detection Systems, the Free Flight vision seems to be feasible.

Despite these obvious advantages, a lot of security risks arise as every airplane is free to choose its course, increasing the aircraft density near popular destinations. Until today, predefined airways are used to restrict airplanes' courses, maintaining the proprietary density and making the air traffic easier-to-control by airports personnel. Giving so many degrees of freedom to airplane routing demands that safe separation of aircrafts is taken into account under a more thorough perspective. Two aircrafts are said to be safely separated if they are 5 mi apart horizontally when flying at the same altitude (3 mi near airports), or 2000 ft vertically if they have the same latitude and longitude.

In this paper we present the *Navigating Aircrafts Multi-Agent (NAMA)* framework, a general platform oriented to agent based Free Flight implementations. From our perspective an agent is an autonomous software component, supposed to run on and control an aircraft. We also propose a decentralized CD&R approach following the Free Flight initiative, implemented on NAMA.

This article is organized as follows. In the second section, we briefly present some of the related work done on this domain. In the third section, we introduce the general NAMA platform architecture, while in fourth section we present our CD&R strategy. In the fifth section we conclude and describe some future work planned to the same direction.

II. Related Work

A lot of work has been done in simulating airplane navigation and air traffic control, using different approaches, including centralized and decentralized ones.

A pure agent based approach has been followed in [4] where software agents running on each aircraft negotiate using the Monotonic Concession Protocol, in order to compromise and resolve a possible conflict of interest. A similar strategy is followed in [5], where a more thorough investigation on negotiation techniques for this purpose is presented. However, these negotiation-based approaches cannot guarantee that an agreement between agents will be reached within an adequate time interval. This is highly demonstrated by the fact that these systems handle non-agreement using a worst case scenario.

In [8] a conflict resolution protocol is presented. In [10] satisficing game theory is used, where agents cooperate in decision making by taking into account the preferences of others. In [9] a non-cooperative model is introduced, where agents act in a discrete, shared resource environment. However, in all of these approaches a 2D simulation environment is considered, disallowing agents from altering their speed or altitude, restricting their maneuverability. Moreover, in [9] a non-real time planning scheme is employed, where a central node informs agents about collision-free trajectories, further restricting the Free Flight paradigm.

CD&R is addressed in [6] either as a single or as a multi objective optimization problem and relevant solving techniques using sequential quadratic programming (SQP) are proposed. While this mathematic approach seems rather efficient, it is not based on the agent programming paradigm and, furthermore, it is strictly focused on collision resolution. Finally, there are no references regarding inter-airplane communication, lacking one of the most important aspects of decentralization and reality.

In [7] a distributed algorithm for CD&R is presented, while [3] describes a multi-layered rule-based algorithm for collision detection and resolution. Both of these approaches are not agent-based. The latter ([3]) is also restricted on two airplanes scenarios, where one of the two is assigned to resolve the conflict.

Finally, AGENTFLY [2] concerns an agent-based platform for employing CD&R scenarios. Despite its advantages, which are mainly focused on visualization issues, the platform is developed on top of an agent-based platform that is not FIPA-compliant, requires a lot of input for each agent, such as velocity and points that must be visited, and implements specific CD&R algorithms, without proposing or encouraging the implementation of others. Another drawback of the underling agent platform is that when messages are delivered, the agent interrupts its execution to handle the incoming message, which is not an actual real time execution.

III. THE NAMA PLATFORM

The NAMA platform was designed to test if Free Flight, from airport to airport, without any centralized control is feasible. Software design followed the same purpose, trying to modularly organize every factor affecting an airplane's flight. This way we can easily extend NAMA in the future, experimenting with other more intelligent agent behaviors or modeling other aspects of aviation. NAMA is written in Java using JADE [11] as the underling agent framework, which in contrast with [2] is FIPA-compliant. Figure 1 shows the basic block diagram of NAMA, where the basic modules of our platform are modeled.

It should be made clear, that this is only a simulation model and does not by any way restrict the way that any of the CD&R approaches tested through NAMA would be implemented on the field. Therefore, the possible actual experimental implementations are not by any way restricted on using Java or JADE as software infrastructure.

Each aircraft is simulated by an *Autopilot Airplane Agent*, or 3A for short, which is a subclass of the JADE Agent class. Each 3A is responsible for complete handling of the aircraft where it (supposedly) resides. Although in a real-world example an agent would play a more advisory role, for simplicity we have assumed that the pilot's actions are identical with the 3A's handling propositions. This way we could evaluate the actual decisions made by 3A.

By extending the basic *Agent* class, each 3A may implement specific behaviors, which are the foundation for all 3As' activity. In our implementation, each 3A makes use of two basic behaviors: *Navigation Behavior* and *Communication Behavior*. The former incorporates the logic for navigating each aircraft while the latter is used for communication. Moreover, each of these behaviors runs on a separate thread, which approximates real-time scenarios in a better way.

Although the 3A class is responsible for both navigation and communication, status information about each participating aircraft is kept within the *Airplane* class, separating, thus, the way an aircraft navigates from the aircraft itself and allowing, for example, different types of aircrafts having the same navigational strategy and vice versa.

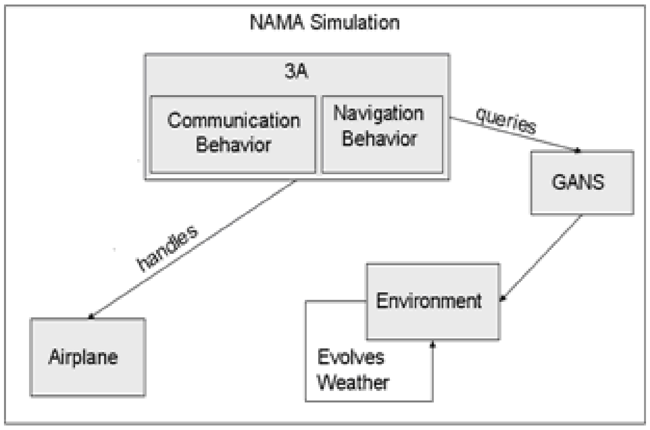

Figure 1. Basic modules of NAMA framework in block diagram

In order for the aircrafts to have knowledge of their surroundings so that they act in a context-aware manner, we have introduced the ***Global Aircraft Navigation System*** *(GANS)* module. GANS models all the supposed information sources for 3As like navigation maps, GPS, radar etc. GANS is not a central control unit (like a control tower); rather, it models in a software class all electronic devices that a real airplane is equipped with. The information that a 3A can retrieve from GANS includes other airplanes' location and weather condition in a specific distance – corresponding to common radar, GPS and sensor information. Therefore, GANS does not by any way affect NAMA's decentralized nature.

Aircrafts are viewed as point-mass models, acting in a bounded, discrete, flat terrain world, shared-resource environment. We term each discrete unit of the world a "block" and the block an aircraft is currently at is its position. Unlike [9], agents are allowed to communicate, if they wish, by making use of the JADE communication protocol, thus resulting in a cooperative model. Dynamic weather conditions over a user-specified area are also modeled within NAMA, varying from sunny and rainy to stormy and snowy weather. Apart from conventional console output, plus JADE's tools for message sniffing, NAMA provides a 2D graphical representation of the world (altitude is suppressed in visualization), displaying both airports and aircrafts locations periodically. Figure 2 displays a screenshot of the 2D graphical representation for a 4 vs 4 conflict resolution scenario.

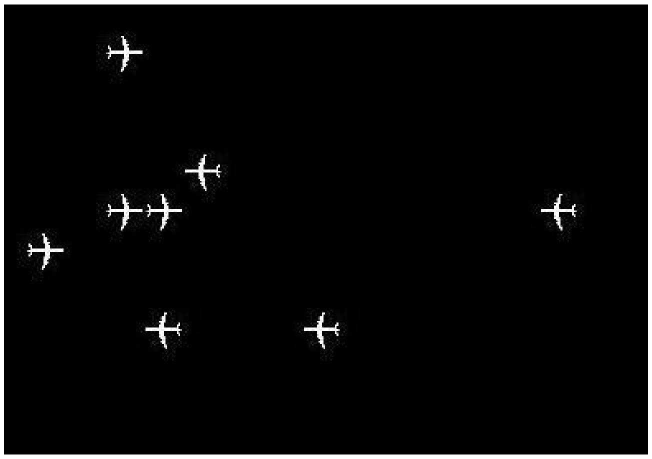
Figure 2. Graphical representation of a 4 vs 4 conflict resolution scenario

The software model described above allows NAMA to work as a general Free Flight simulation platform. Extending or altering the already described classes is relatively easy, allowing for someone to test new-multi agent aviation approaches. For instance, testing a scenario where centralized control would define airplane routing, a change in GANS and in 3As' behaviors should be enough. Furthermore, NAMA could easily be used to test and compare the behavior of different airplane types, just by extending the *Airplane* class defining special parameters for each airplane type like minimum allowed landing length, fuel consumption coefficient, maximum flight height, maximum speed etc.

IV. Utility Functions for Conflict Resolution

Although NAMA is a generic platform, qualifying for various experimentations concerning free flight, it was built in order to test a specific multi-agent strategy, containing both a CD&R approach and a scheduling mechanism at the terminals. The basic concept behind NAMA is to completely eliminate any central control, relying only on the communication of the aircrafts, as the real power of the multi-agent paradigm is the distribution of the computing tasks and the emergent behavior of the agents' interaction.

In our approach, agents (namely the 3As) communicate directly to each other through broadcasting. Communication between 3As takes place only when the airplanes are inside communication range, which is an aircraft parameter (each aircraft may have its own range). The information broadcasted includes the status and short-term location of the corresponding airplane. Every other agent inside the communication range will receive this information and adjust its navigation behavior accordingly. Each 3A decides only for the airplane it handles what it should do in order to arrive safely at its destination using the shortest possible route.

We avoid any kind of negotiation (as in e.g. [4], [5]), due to the nondeterministic nature of the negotiation procedure, and the variability in time these kind of processes usually have. In extreme situations, like a possible collision, the collision resolution strategy should always result in a safe outcome and within a fixed time interval.

Our approach relies on a **Route Utility Function** (RUF), for each possible route that the airplane can choose, leaning towards a pure agent-based approach. Supposing the airplane is at a specific location, the 3A quantifies its environment, evaluating its *k*-next possible locations using a **Point Utility Function** (PUF), for each one of them. Location evaluation is based on information like weather condition, current and future positions of other airplanes, distance to destination, etc. retrieved directly from GANS and the messages broadcasted from other 3As in the area. The RUF sums up, for each route, the values returned by PUF for the locations that fall in the specific route. To simplify matters, we assume that the information gained by GANS is always valid (i.e. the location of nearby aircrafts can be determined with high precision from a radar or the weather can be continuously updated via an appropriate weather report feed for farther locations and via sensors for immediate locations).

Finally, 3A selects the best available course (the one with the highest score), which it will consequently follow, until reaching its destination. In this sense, every 3A employs hill-climbing while searching for the most suitable route to follow. Hill climbing allows for fast computation which is desirable for a real-time system, and since the computation is performed periodically, in small intervals, it encapsulates the dynamicity of the environment. RUF can also be used to restrict the aircraft from inspecting certain locations, where it may not go (for example a mountain or another airplane in close vicinity).

A. *Point Utility Function*

Based on the above, it is evident that the Point Utility Function affects the actual aviation of the airplane. Consequently, PUF should return values responding to the specific blocks position, weather condition and other aircrafts status that allow 3A to follow a safe and optimized route to the destination airport. The rules that apply in PUF should be well defined and have safety as a priority.

It should be made clear that PUF values are based on parameters that are arbitrarily chosen. Altering these values allows us to experiment with other possible aviation approaches as they change the overall 3A behavior. What would be most interesting is to understand the impact of these values to the aircraft's navigational behavior both while flying alone and when interacting with others. Below, we present the rules that we applied in our experimental strategy:

1. PUF value is decreased when the block tested is near other airplanes

This rule is used for avoiding collision between airplanes. Whenever two airplanes approach each other, and loss of separation is imminent, PUF is decreased.

2. PUF value is decreased when the block tested is in or near bad weather

This rule is used for avoiding flying through bad weather when this is evitable. Its priority is lower than rule 1. Moreover, the decrease depends on the type of bad weather, since rainy weather is not as bad as stormy or snowy weather.

3. PUF value is increased when the block is higher but not higher than the airplane's maximum allowed height

Fuel consumption is generally minimized when flying at higher levels. Therefore, increasing the value when at a higher altitude optimizes the selected course in terms of fuel consumption. The height limit of the specific airplane should also be taken under consideration, decreasing the value when the tested block is above the maximum allowed height, to avoid a safety violation for that aircraft.

4. PUF value is increased when the block tested is closer to the destination

This rule allows 3A to optimize the airplane course, preferring the shortest available path to the destination airport. The advantage of using utility functions in such a way is that user preferences are taken into consideration all at once.

Furthermore, additional preferences or more advanced techniques for calculating these specific preferences may be encoded by altering the PUF code, while the priority of these rules may be changed by changing the corresponding value. PUF is the core of the route grade calculation and changing the effective coefficients of the respective rules would lead to a whole different CD&R strategy. The selection of the optimal PUF coefficients combination is an issue of experimentation and still under research.

B. Collision Resolution

Given that we consider the airspace a shared-resource environment, one of the airplanes must occupy the resource of space earlier than others, otherwise a collision will occur. For this reason, a hierarchy scheme for taking into consideration other agents' intentions is used. While executing its RUF, each agent takes into consideration received messages of higher priority than its own according to a ranking criterion, disregarding the ones with lower priority.

In our CD&R experiments, the first scheme tested lacked any kind of hierarchy, making agents use all available information. This is not an efficient approach as the geometry of the problem is neglected. From our point of view, the geometry is related to the current position and heading of all aircrafts within a specific range, which – for simplicity sake – we have chosen to be equal to the communication range. A hierarchy scheme that relies on the conflict's spatial geometry may lead to more efficient utilization of airspace.

In a more sophisticated scheme tested, each airplane first calculates the centroid of the locations of the airplanes inside its communication range. Such information, which depends on the geometry of the problem can be easily calculated by each aircraft with very small deviations between them, while being objective and thus cannot be exploited, in case an agent is malevolent. The criterion used is the cosine of the angle between the target location, the centroid and the airplane. Each agent calculates this cosine for itself and for every aircraft in its communication range from the messages received and forms its hierarchy in descending order. Intuitively, the criterion used gives higher priority to the aircrafts that will fly closer to the centroid, in which case they are more liable to collisions.

Table I demonstrates the averaged minimum, average and maximum distances, in nautical miles, of all airplanes within all time slots during conflict resolution, for various scenarios. In each scenario, *n* airplanes move in the opposite direction against *m*, forming an *n* vs *m* conflict area. We assume that safe separation is maintained for distance greater than or equal to 3 nautical miles. Distances are scaled, so that the recorded minimum is 3 (nautical miles), thus giving a scaling factor *f*. Then, all other distances are multiplied with *f* and min, avg and max are consequently computed. Distances between aircrafts that are greater than 2**k***f*, where *k* is the number of positions that an aircraft checks for future moves, are not taken into account when averaging. The reason for this is that 2**k* is the number of positions evaluated by RUF in our implementation, when other aircrafts are within communication range.

Our current experiments emphasize on preserving safe separation of the aircrafts. This is the basic reason why no results of time-related measurements are currently presented. Furthermore, there are not any widely accepted baseline approaches with regards to time and / or space measures to compare with. It is for this reason that other techniques presented in the related work, as well as ours, do not compare with one another.

TABLE I. RESULTS COMPARING HIERARCHY WITH HIERARCHY-ABSENT APPROACH DURING CONFLICT RESOLUTION

	Hierarchy			**Hierarchy-absent**		
n* vs *m	***min***	***Avg***	***Max***	***Min***	***Avg***	***Max***
1 vs 1	21.73	25.69	33.32	21.73	25.69	33.32
2 vs 2	12.11	32.71	47.66	18.73	37.06	53.20
2 vs 3	15.19	37.45	58.98	18.73	33.32	49.04
2 vs 4	19.84	40.17	67.63	18.57	45.59	77.11
3 vs 3	11.45	34.22	56.93	19.04	39.40	58.55
3 vs 4	11.90	37.24	57.08	18.95	40.04	64.27
4 vs 4	13.39	39.84	72.56	18.80	39.97	71.11

Several observations can be made from the above measurements. First of all, the hierarchy scheme performs better, which confirms the aforementioned assumption concerning the geometry of the problem. Secondly, while the number of engaged airplanes increases, the average density remains roughly the same, as the average distance between aircrafts is maintained without loss of safe separation.

The maximum distance increases due to having more airplanes in the conflict area. A third remark is that the minimum distance can be greatly improved as it exceeds the safe separation threshold by far in all occasions. Specifically, in 1 vs 1 scenario both prioritization schemes performed in the same (bad) way, which means that PUF coefficients are solely responsible. Based on the first and third remark, both different PUF coefficients and alternative hierarchy schemes can be employed in order to optimize air-space utilization.

V. Conclusion

In this paper we presented NAMA, a multi agent platform allowing the implementation and testing of custom Free Flight strategies. We also presented a novel CD&R approach based on utility functions and compared two schemes, with and without hierarchy, defining how each agent's preferences are taken under consideration, in an attempt to utilize the airspace more efficiently. Overall, the proposed approach addresses the Free Flight issue in a simple, intuitive and flexible way.

Despite the positive results presented, our future work can focus on optimizing certain aspects of the domain. The accurate adjustment of the parameters used in Point Utility Function seems a good place to start. Furthermore, we could work on more efficient hierarchy schemes using the distance from the centroid, the airplanes' emergency status, flown time or their combination. Moreover, scenarios that simulate partial or lack of communication have to be thoroughly studied. Finally, enhancing NAMA with the definition of terrain peculiarities (mountains, cities etc) and increasing the degree of weather modeling are in our future plans.

References

[1] Federal Aviation Administration, "Advancing Free Flight through human factors", www.hf.faa.gov/docs/508/docs/freeflt.pdf, Accessed 1 August 2008, 1995

[2] Sislak D., Volf P., Kopriva S. Pechoucek M. "AGENTFLY: A Multi-Agent Airspace Test-bed", Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems: demo papers, 2005

[3] Erzberg H., "Automated Conflict Resolution for Air Traffic Control", Ames Research Center. http://ntrs.nasa.gov/archive/archive/nasa/casi.ntrs.gov/20050242942_2005243395.pdf, 2005

[4] Wollkind S., Valasek J., Ioerger RT, "Automated confilict resolution for air traffic management using cooperative multiagent negotiation", AIAA Guidance, Navigation and Control Conference, 2004

[5] Rong J., Valasek J., Geng S. Ioerger RT, "Air traffic conflict negotiation and resolution using an onboard multi agent system", Proceedings of the 21st Digital Avionics Systems Conference, 2002

[6] Menon PK, Sweriduk GD, Shridar B., "Optimal Strategies for Free Filght Air Traffic Conflict Resolution", 22:202-211, 1999

[7] Eby MS, Kelly WE, "Free flight seperation assurance using distributed algorithms", Proceedings of 1999 IEEE Aerospace Conference, 1999

[8] Hwang I., Kim J., Tomlin C., "Protocol-based conflict resolution for air traffic control", Air Traffic Control Quarterly 15(1):1-34, 2007

[9] Resmerita S., Heymann M., "Conflict Resolution in Multi-Agent Systems", Proceedings of the 41st IEEE Conference on Decision and Control, 2003

[10] Hill JC, Johnson FR, Archibald JK, Frost RL, Stirling WC, "Cooperative Multi-Agent Approach to Free Flight", Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems, 2005

[11] Bellifemine F., Poggi A., Rimassa G., "JADE – A FIPA-compliant agent framework", Telecom Italia internal technical report. http://jade.tilab. com/papers/PAAM.pdf, Accessed 1 August 2008, 1999