Learning Aircraft Behavior from Real Air Traffic

Arcady Rantrua, Eric Maesen, Sébastien Chabrier, Marie-Pierre Gleizes

Learning Aircraft Behavior from Real Air Traffic

Arcady Rantrua[1,2], Eric Maesen[1], Sebastien Chabrier[1], Marie-Pierre Gleizes[2]
{firstname.lastname}@soprasteria.com
{firstname.lastname}@irit.fr
[1] R&D Dept, Sopra Steria, Toulouse, France
[2] SMAC Team, Paul Sabatier University, IRIT, Toulouse, France

Abstract. What if we could observe the real world to teach our simulation how to work? There would be no need for physics computation, no need to describe what kind of entities exists in the world. Everything would be observed, learned and made usable for simulation. This is the goal of EVAA.

1. Context and Problems

The process of creating a scenario for an Air Traffic Generator (ATG) is often a tedious task. Many parameters have to be entered manually in a very iterative and long process. Moreover they heavily rely on the flight plan to generate the trajectory of the plane. But, after taking off, the plane's trajectory rapidly differs from its original flight plan. It may be because of the weather, because the controller gave an ATC order to separate aircrafts, or because he or she gave a clearance to take a more direct route because the traffic was light. For those reasons we propose a new way of generating traffic based on behavioral learning.

The behavior of an aircraft is difficult to simulate because it's defined by many parameters related to the aircraft and to its current environmental conditions. The information necessary to realistically simulate the flight of an aircraft is often either insufficient or unavailable. In addition, we said above that the flight plan might be changed during the flight, the whole flight is full of changes, change of speed, change of altitude, … ATCO (Air Traffic COntroller) orders are given and actions are taken. For any action of the plane there can be many reasons and we cannot discriminate between them. For these reasons any machine learning method based on a complete knowledge of the environment is not applicable.

With EVAA we present a learning algorithm able to use incomplete data using cooperative multi-agent systems [1] to produce autonomous and self-adaptive behaviors for aircraft in a simulated environment. Through a large volume of real flight data we build a network of agents, each tasked to learn a piece of the aircraft's behavior. Those agents communicate with each other to build the global behavior that can be used later for a simulation.

2. Learning on Real Flight Data

When we observe real aircrafts flying, they emit through their ADS-B transponder, at any time, a set of parameters. Those parameters match real percepts like latitude, longitude or speed for example and each of them has a value.

Change	Parameter	Example of value
*	Time	26 Nov 2014 12:07:06
	Callsign	AF263PE
*	Latitude	45.66
*	Longitude	-0.3073
*	Altitude	25700
*	Heading	130
	Departure airport	BOD
*	Destination airport	ORY
	Type of aircraft	A321
	Registration number	393320 F-GMZA
*	Ground speed	425
*	Vertical speed	1664
	Transponder ID	4e6f657
	Squawk	1000
	Radar code	F-LFCH2

Table 1 - Observable parameters

The Table 1 shows the exhaustive list of observable parameters in our system. Some of those values are static and cannot be changed during the simulation, others can and are marked with a "*" in the first column. The last column shows an example of correct value for each parameter.

While we observe the real traffic we can capture the value of each of those parameters in a snapshot that we call a situation. An aircraft will fly through many different situations. Each situation is linked to its previous one (temporally speaking) creating multiple situation vectors.

The Figure 1 represents the results of the learning on one aircraft where each situation encountered by the aircraft is linked to its following and so on until the last situation which gives us a "unary tree".

Nevertheless, it would not be realistic to hope to create a graph with every situation encountered by the plane. Many sections of "unary graphs" can be simplified by removing intermediary node if the changes described between the first node and the last node of the section is linear (like an aircraft moving in a straight line). The remaining situations are called situations of interest.

Every observed aircraft gives us one "unary tree". Since we observe multiple aircrafts, some node (or situation) can be merged. The merging process is based on the proximity between the situations, if the distance between nodes is below a given threshold they merge with each other. When all the relevant nodes are merged we are left with a directed graph able to guide an aircraft.

The nodes in the final graph represent the points where the aircraft has acted. It's close to the notion of navigation point and we could expect those points to match the waypoints of

Figure 1 - "unary graph", the results of observing one aircraft

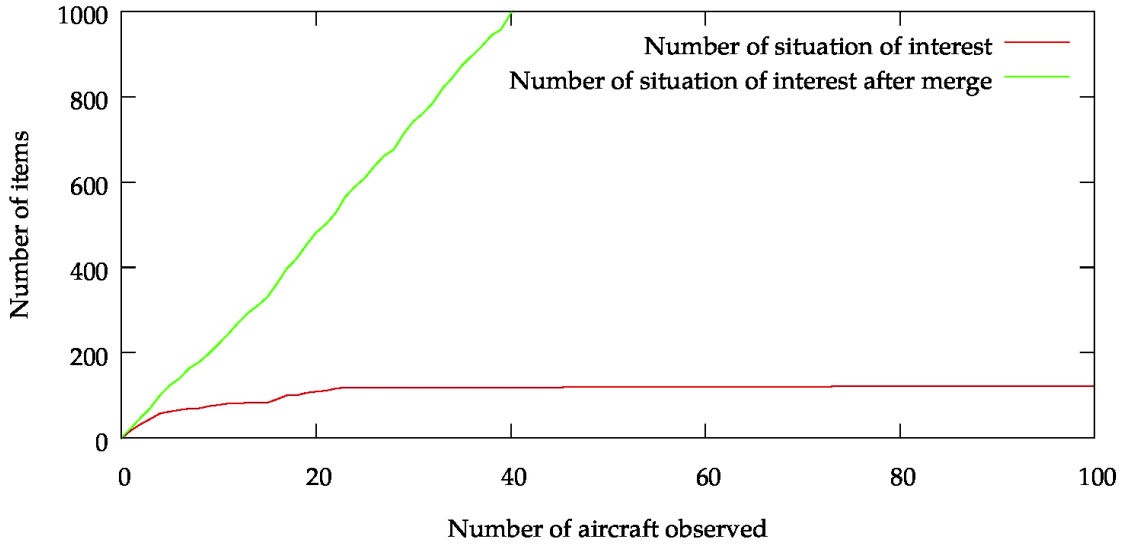

Figure 2 - Situations over number of aircrafts observed

the flight plan but our results disprove this hypothesis by showing that many aircrafts take shortcut multiple time during the flight. Hence, there is no perfect matching between waypoints and nodes.

The scalability and usability of this method entirely depend on learning. If learning is not doable on big samples of diverse data then it cannot simulate diverse data and EVAA is not a realistic simulator. The graph in Figure 2 shows that the number of situations of interest increase linearly with the number of aircrafts observed. It could be a problem but the merging of situations which enable us to merge redundant information and increase the speed of the learning process.

3. Adaptive Traffic Generation

A traffic generator must be able to compute the location of an aircraft over time. With the situation vector mechanism we are able to compute a set of future locations for the aircraft: if the current location of the aircraft matches the initial situation of a vector it means that the terminal situation of this vector is a potential future situation.

The traffic will be generated by "agentifying" the aircrafts. These aircrafts start with a specific situation which can be the departure airport (defining latitude, longitude and altitude) and ready to take off (speed is null, callsign is set …) or at a specific 3D position as if it were already flying. From their starting situation a plane can find what its next action will be by comparing its current situation with the initial point of every situation vector in the area. All of those who are sufficiently close in a Nth dimension comparison (N being the number of parameters) are candidates and the best situation vector among the candidate is chosen. Now the plane knows exactly what it should do next:

- It knows where it should go by looking at the latitude, longitude, altitude of the terminal situation of the situation vector.

- It knows at which speed it must go there by looking at the time difference between the initial and the terminal situation vector.

```
distance(current, initial):
        dist = 0;
        for p in parameters:
                if isNumeric(p):
                        dist = dist + scale(p, | p(initial) – p(current) | )
                else if p(initial) != p(current):
                        dist = dist + 100;
        return dist;
```

Algorithm 1 – Distance between two situations

This method, applied on multiple aircrafts and on long recording of flight, gives us the basis for learning a realistic behavior.

An agent is launched for every existing situation vector. Those agents are geographically located on the map. Any simulated aircraft start with an initial situation which is used to find the first objective of this aircraft. An objective is a situation in which the aircraft "wants" to be. The first step is to find this objective. EVAA provides a way to send a message to any agent in a specific radius of a location. The aircraft sends an objective request, a message containing its current situation, to any situation vector in a radius R around itself. Each vector has to decide if its initial situation matches the current aircraft's situation. This is done with the Algorithm 1 which is able to compute a numerical distance between two situations. The scale function put the difference between p(initial) and p(current) on the same scale between 0 and 10. It is necessary because a difference of 1 unit of heading is not much whereas 1 unit of latitude/longitude is huge.

Then if the distance if less than a defined threshold the vector decides that it matches the current situation and send its initial and terminal situation to the aircraft.

Then the aircraft receive responses to its request and do a certain amount of checks and verifications on every potential vectors. Those which don't pass those tests are discarded. An example of sanity check is: if a vector advice to go from 0 to 36000 ft in 1 second this message will be discarded. Also an aircraft will prefer to change it's heading than its altitude unless it's close to its arrival airport.

Among the remaining vectors, the aircraft will choose the one with the smallest distance to its current situation as its new objective. Once the objective is reached the process starts again with the new current position.

By following this process the aircraft is able to follow a realistic trajectory with realistic parameters in the virtual sky of EVAA without any human help whatsoever.

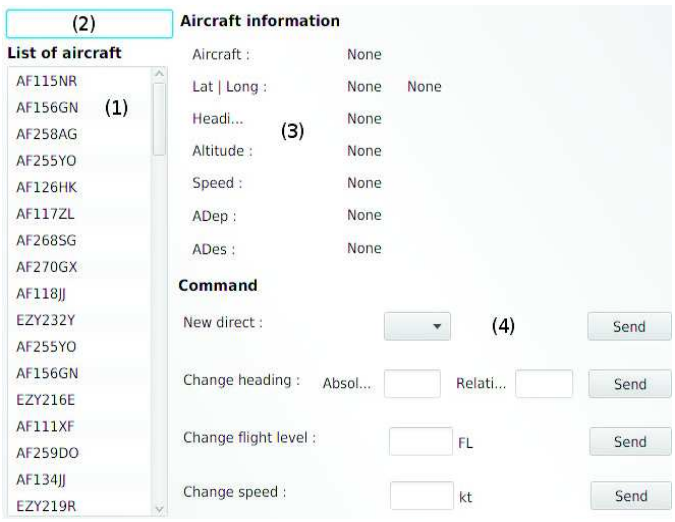

Figure 3 - Pseudo pilot interface

Figure 4 - Trajectories of real aircrafts

Figure 5 - Trajectories of simulated aircrafts

The Adaptive Multi-Agent System (AMAS) [2] [3] technology provides a way to deal with unpredictable events (like ATCO tactical orders) that aircrafts encounter during their flight. Those events are the reason very few aircraft follow their original flight plan. Using AMAS means that we have to follow a set of principles if we want to benefit from those advantages.

- Agent should be autonomous and the network between them should be self-organizing.
- Agents should base their decision only on local knowledge.
- Agents should cooperate with each other. Not to a point where they would be altruistic but they should try to help their neighbors if it improves the local state.

We saw that the learning phase uses the self-organization principle when situation vectors build their networks on the fly. We use the locality principle when an aircraft only asks its potential objective to the vectors in its neighborhood. The fact that situation vectors, when they receive an objective request, can judge themselves as non-pertinent (and do not send a response) shows cooperation.

4. Supervised Traffic Generation

When using traffic generation you might want the aircrafts to follow a specific route. In EVAA, aircrafts with their flight plan specified can switch between adaptive mode and flight plan mode with a single click.

This functionality is necessary because, in a controlled zone, every aircraft must follow its flight plan unless it has been said otherwise by the air traffic controller.

More over, any aircraft can be remotely piloted with high-level orders through a generic message-based API.

Figure 3 shows an interface we developed to show the capabilities of EVAA. In (1) you can see the list of aircrafts. If one of them is selected then its information are displayed in (3). You can also search a plane using the auto-complete field in (2) and then send it a control order with the panel (4).

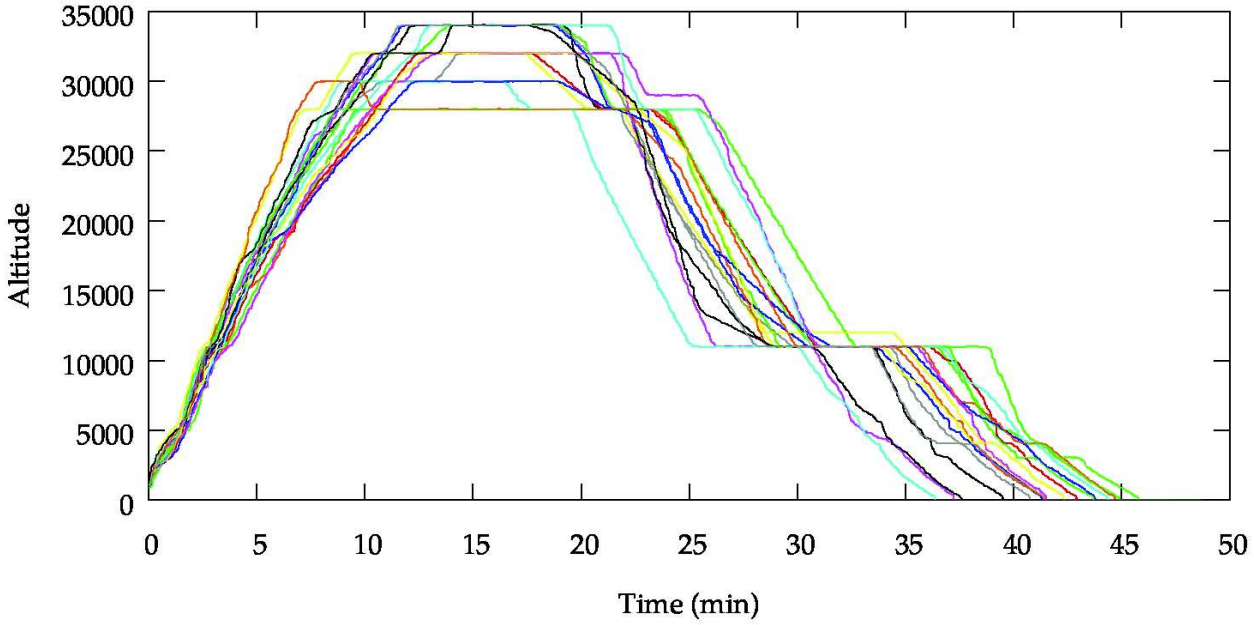

Figure 6 - Altitude profile of real aircrafts over time

5. Results

In this section we will compare the results of our simulation with the reality between French national airports.

The Figure 4 shows a set of 50 real trajectories (here Toulouse to Paris Orly) and the Figure 5 shows a set of 50 simulated flight for the same ADEP/ADES air line. We can see that the trajectories are very similar in shape and that many different kinds of trajectory are available in the simulation providing diversity and realism.

The Figure 6 shows the altitude profile of 25 real aircrafts between Toulouse and Paris airports. We see that the aircrafts start by climbing, then reach their cruise level, then descend to reach another flight level to finally land on ADES.

The Figure 7 shows the altitude profile of 100

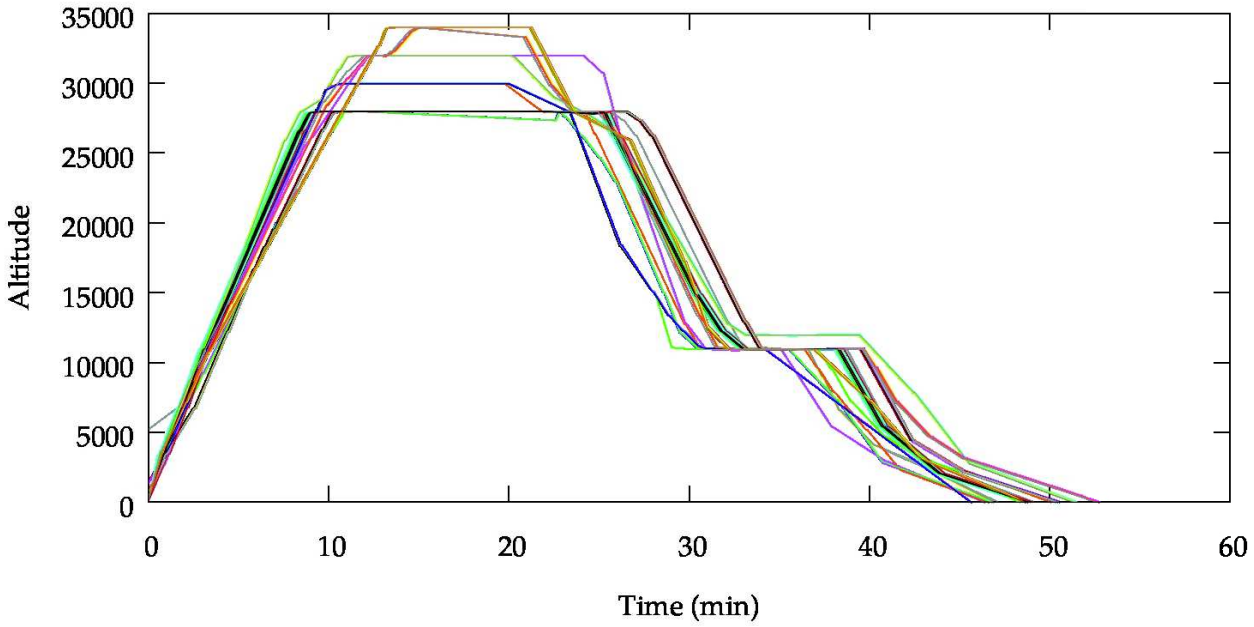

Figure 7 - Altitude profile of simulated aircrafts over time

EVAA ATG simulated flights. We can see the same succession of phases than in the real world with a little less variety.

The Figure 8 is a box plot of the time it takes for an aircraft to fly from its departure to its arrival (The time scale for simulated flight has been multiplied by a coefficient to fix a problem with our platform. The coefficient is the same for each route). Those calculations have been made on 500 flights (50 flights for each route). We compared multiple air routes and the fact that the box (difference between first quartile and third quartile) is smaller in simulation than in reality shows a lack of diversity in the flights produced by EVAA. Nevertheless, we can see that our simulation always respects the minimum and maximum boundaries of travel time.

In most cases the statistical distributions of simulated flight are included into and statistically close from its real flight counterpart.

6. Conclusion

EVAA is using machine learning, multi agent systems and real trajectories observation to generate behaviors of aircrafts. Since those behaviors are based on what happen in the real world, the resulting trajectories will be very realistic. It is also possible to simulate supervised aircrafts, which will follow their flight plan (or the orders of a pseudo-pilot).

This enables EVAA to be usable in many situations such as pilot and controller training, generating autonomous surrounding traffic generation, fully human controlled traffic or any combination you can imagine.

7. References

1. L Panait, S Luke “Cooperative Multi-Agent Learning : The State of the Art”, Autonomous Agents and Multi-Agent Systems 11.3, p. 387–434, 2005.
2. G Di Marzo Serugendo, M Gleizes, A Karageorgos, “Self-Organisation and Emergence in MAS: An Overview”, Informatica, p. 45–54, 2006
3. Jean-Pierre GEORGÉ, Marie-Pierre GLEIZES, Pierre GLIZE (2003). “Conception of adaptive system with emergent functionality: The AMAS theory”

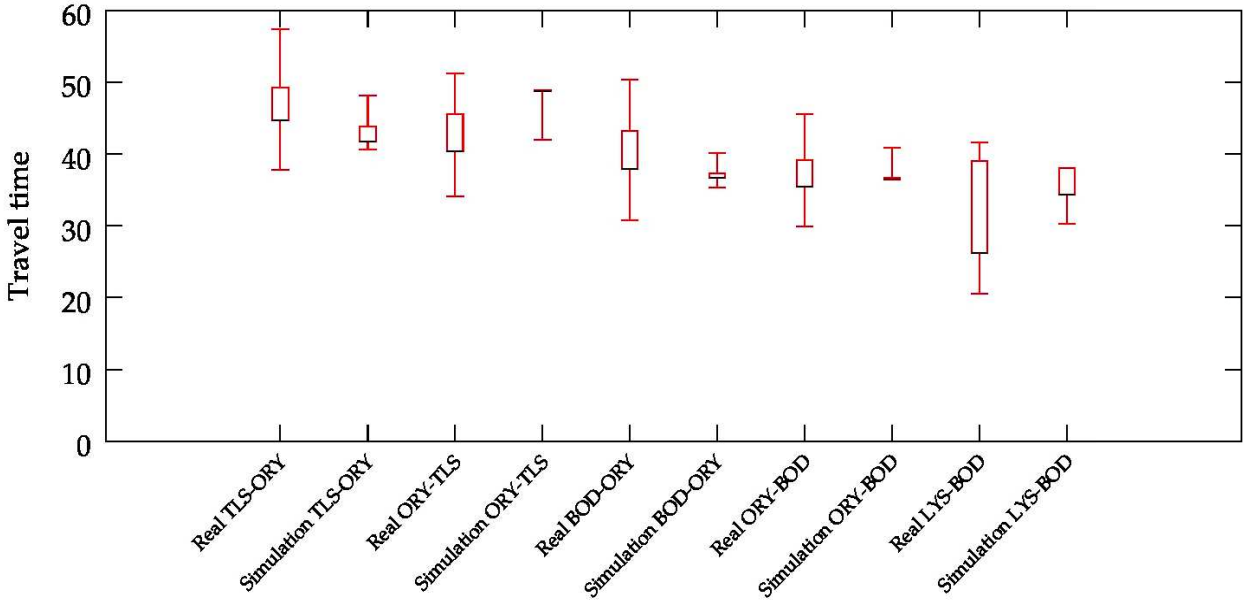

Figure 8 - Statistical distribution of travel time for aircrafts on multiple air routes in reality and in simulation