FLIGHT SIMULATION ENVIRONMENTS APPLIED TO AGENT-BASED AUTONOMOUS UAVS

Ricardo Gimenes
Polytechnic School - University of São Paulo, Av. Prof. Luciano Gualberto, trav. 3, n. 158
CEP 05508-010, São Paulo, Brazil

Daniel Castro Silva, Luís Paulo Reis, Eugénio Oliveira
FEUP - DEI / LIACC, Rua Dr. Roberto Frias, s/n 4200-465, Porto, Portugal

Keywords: Simulation Environments, Multi-Agent Systems, UAVs.

Abstract: Developed countries have made significant efforts to integrate Unmanned Aerial Vehicle (UAV) operations in controlled aerial space due to a rising interest of using UAVs for civilian as well as military purposes. This paper focuses on looking for reliable solutions and a way to validate an autonomous multiagent control system through a different variety of Flight Simulations. This study has two main lines, the first being the use of multiagent systems in UAVs, aiming at a fully autonomous control. The second and focal line is a survey about the variety of simulation systems dedicated to aerodynamics and aircrafts, comparing them and their feasibility to validate the developed multiagent control system. One critical factor is hazard situations, like emergency landing without runway or equipment failure, which should be predicted in an automated system. Most flight simulators are not realistic enough to validate a multiagent algorithm in hazard situations. At the same time, it is impossible to predict every type of failure in real world. The boundaries of simulation should be very well enclosured in order to present results using simulation.

1 INTRODUCTION

Ever since the construction of the first vehicles, men have dreamed of automating its operations. While this autonomy has been restricted to limited functions, it is desirable to elevate it to a full scale, allowing vehicles to be fully autonomous. When full autonomy is reached, vehicles will have to cooperate and coordinate their actions with one another, in order to ensure both security and an optimal use of resources. The choice of an agent-oriented approach to control autonomous vehicles is intended to save both time and resources, enabling the vehicles to communicate, and make their own mission planning and real-time decisions.

In order to better develop and test such agent-oriented approaches and coordination methods, a simulation environment must be used. Such environment also saves time and resources, since it would be prohibitory to test these systems using real vehicles.

This paper focuses on several existing simulation environments that can be adapted to serve the objective at hand.

1.1 Unmanned Aerial Vehicles

UAV, acronym for Unmanned Aerial Vehicle, is considered, in this paper, as a vehicle able to fly autonomously without any external remote control.

There are already some examples of civilian UAV usage, either operating or in developmental stage, namely those related to surveillance or in need of access to risky areas (Air & Space Europe, 1999). These do not have significant influence on the air traffic system, since the UAVs fly over clearly defined areas at low altitude and speed.

The use of UAVs still involves a few uncertainties and legal limitations as to where and how they can be used. The majority of projects involving UAVs are strictly military. Opposing to the military vision, crewless civilian passenger aircraft raise a completely diverse paradigm in terms of reliability. Because of this, new hurdles come in the way of a future categorization of passenger carrying UAV, as well as of non-passenger carrying UAV sharing airspace with conventional aircrafts.

Thus, operating UAV on international airspace

Gimenes R., Castro Silva D., Paulo Reis L. and Oliveira E. (2008).
FLIGHT SIMULATION ENVIRONMENTS APPLIED TO AGENT-BASED AUTONOMOUS UAVS.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - SAIC*, pages 243-246
DOI: 10.5220/0001710802430246

must entail two closely related conditions (Allouche, 2000): the UAV must be safe and reliable enough to fly over densely populated areas and must be safely operated through airspace.

1.2 Multiagent Systems

A multi-agent system (MAS) can be seen as a system where entities are represented by independent agents, which in turn communicate with each other, coordinating their activities.

The UAV operation can surely be considered a MAS in which every aircraft is an agent with its own goals (destination, time frame of arrival, service standards, etc.), independent of the goals of other aircraft. With this approach, it is possible to apply negotiation techniques, which will allow the aircraft to cooperate in solving airspace conflicts. Answers involving artificial intelligence software or MAS have been researched on, in the search for decision-making systems capable of avoiding aircraft encounters in a given airspace (Lian and Deshmukh, 2006).

In section two of this paper, some of the requirements that were taken into account when evaluating the simulation environments are briefly described. Section three presents some of the analyzed simulators, game engines, flight simulators and middleware engines. In the last section, some conclusions are withdrawn and future work is outlined.

2 SIMULATION ENVIRONMENT REQUIREMENTS

The choice of a simulation environment is dependent on the objectives of the project. In this section, some key features that may be desirable in the simulation environment are briefly described. These features can be grouped into categories, such as graphical, physical or openness characteristics, among others, for a better analysis.

The openness of the software is an important aspect. Openness can be defined in terms of the possibility to view and alter the source code, as well as expansion capabilities, such as the possibility to develop add-ons, external modules and different agents that can be linked to the environment, via defined APIs, interaction models or communication protocols, the accessibility and format of the data that can be input to the environment, and the output data format as well.

Accurate physical simulation is very important, when dealing with robotic mobile agents, such as UAVs. The following subsections explain in more detail the importance of a correct physical modeling and simulation of the aircraft and the environment.

2.1 Specific Aviation-Related Requirements in Simulated Environments

Realistic simulation of robotic sensors and actuators in a complex, unstructured, dynamic environment, such as fluids, constitutes a research challenge. Most simulators consider only the four basic vectors that compose a flight: lift, weight, drag and thrust. (Schiff, 1978). However, in real flight, an aircraft has to deal with numerous factors, not only pertaining to the aircraft itself, but also external, environmental factors.

This way, to choose a flight simulator system, it is very important to define what is more relevant in the research. Each simulator has its characteristics suitable (or not) with a specific kind of research. Considering developing an autonomous multiagent environment as the main goal, the relevant parameters in the flight system can have three major lines: Acceptable flight model, considering the application under investigation; Flexibility to interact with the agent via programming interfaces; Possibility to have the flight model or simulated elements changed by external software.

Agent algorithms which have to control an aircraft will need a flight simulator model as real as possible. As the number of simulated elements offered by the flight simulator increases, so does its adequacy to serve as a suitable tool for simulated validation of the agent-based UAVs.

2.2 Fault Injection in Flight Simulation Models

The firm safety requirements related with aviation require the studies of any kind of flight simulation to keep in mind failure considerations. Fault injection is a complex research line to reach a reliable and safe flight model in flight simulation.

A fault injection method is essentially hardware fault injection. In a flight simulation, faults can be injected into different inputs or outputs of the aircraft. Considering the UAV completely controlled by software, the range of failure possibilities determines the way the faults have to be injected. The quantity of injected faults is a quality parameter of any fault injection method. Basically, the fault injection module is a software module which interrupts the original inputs and outputs of the simulated aircraft.

A critical problem for fault injection using COTS

flight simulators is how they are opened to interact with their simulated hardware of an aircraft. Thus, open source flight simulators or open API from proprietary software should be analyzed as decisive factors in a flight environments choice.

2.3 UAV Flight Simulation

The way the flight simulation is developed determines its possibilities and, consequentially, the possibilities and limitations the research has to deal with. Real-time visual simulation is not necessarily a relevant parameter. Simulations without visual interface offer the possibility to produce a quantity of different situations quicker than the visual simulation does.

The simulation of physics, kinematics and aerodynamics is fundamental in a reliable flight simulation. In contrast, structural forces are not relevant in UAV flight model studies.

3 COMMON SIMULATION ENVIRONMENTS

There are two main simulator categories: Game Engines and Flight Simulators.

In game engines, the most important aspect is an appealing visualization. Flight Simulators have the main focus in the flight simulation itself. There is some focus on the visual aspect as well, but the main efforts are dedicated to aerodynamics and flight factors present in real world.

3.1 Game Engines

First-person games offer a possibility to researchers play, test, and evaluate their robots with a different range of sensors. An accessible 3D environment with physics and kinematics opens new robotic research perspectives. Conversely, when aerodynamics is the main focus, game engines normally do not attend the requirements (Lewis and Jacobson, 2002).

3.2 COTS Flight Simulators

There is a great variety of COTS Flight Simulators. The graphics in modern flight simulators are very realistic. However, scientific research has to worry about the flight dynamics of an aircraft. Therefore, an evaluation on a few low cost COTS flight simulators is presented below.

3.2.1 Microsoft Flight Simulator

The current version of the Microsoft Flight Simulator series (FSX) is probably the most realistic on the market, in graphical terms, in part thanks to the use of the new DirectX 10 technology. Its flight model is based on a set of tables, and is independent of the visual model (Stock, 2007). The failure-modeling includes equipment failure, but without variation of the manner in which the equipment will fail.

FSX presents the SimConnect API, an FSUIPC-like access to functions and variables, which allows developers to create new add-ons to add or replace functionalities, as well as monitor activities (Microsoft Corporation, 2006).

3.2.2 X-Plane

X-Plane uses the geometric approach and its model is defined as structural. An engineering process called ”blade element theory” calculates the flight model, breaking the aircraft down into many small elements and then finding the forces on each element, which are then converted into accelerations, velocities and positions (Laminar Research, 2007). X-Plane also has failure-modeling, but as in FSX, it is not allowed to vary the way the equipments fail (McManus et al., 2003).

A creditability factor about X-Plane is its professional use and approval by the FAA for training towards airline transport certificate. However, Stock (Stock, 2007) compared X-Plane and FSX, and the results show how some conceptual flaws, and how they have serious impressions in their models.

3.2.3 FlightGear

FlightGear is a free, open-source, multi-platform, cooperative flight simulator development project, with the goal of creating a sophisticated flight simulator framework for use in research or academic environments, once it allows access to a very large number of internal state variables (FlightGear, 2007). It allows to remotely control FlightGear from an external script and to use an external flight dynamics module (including hardware autopilot) (Aeronautical Development Agency, 2001).

FlightGear allows the users to choose between three primary Flight Dynamics Models. More than this, it is possible to add new models or even interface to external flight dynamics models source, such as from Matlab. The three available models are JSBSim, YASim and UIUC (based on LaRCsim originally written by NASA (Jackson, 1995)).

3.2.4 Piccolo

Piccolo is a known auto-pilot system for small aircraft, by CloudCap Technology (CloudCap Technology, 2007). The main product is commercialized as hardware components that can be mounted on a small aircraft, allowing autonomous flight. There is also a software release of a simulation environment, for software-in-the-loop tests. The graphical component is minimal, but in terms of flight simulation it is very realistic, simulating many of the forces involved in a flight. It can also, by default, output its data to Microsoft Flight Simulator and FlightGear.

3.3 Dedicated Middleware Engines

There are hundreds of middleware engines which could be used in scientific research. But, considering the UAV scenario controlled by multi-agents algorithms, two are highlighted in this paper.

OpenFlight, MultiGen-Paradigm's native 3D content, is the leading visual database standard in the world and has become the standard format in the visual simulation industry (MultiGen-Paradigm, Inc., 2007). AeroSim is a Matlab block library which provides components for development of nonlinear (with six degrees of freedom) aircraft dynamic models (Unmanned Dynamics, 2007).

4 CONCLUSIONS

Since the choice of an appropriate environment depends on specific needs, no single environment is elected, but the choices of a suitable environment have been defined concerning specific characteristics chosen according to the objective in sight.

Some conclusions can be withdrawn from the analysis made to the simulating environments in the previous section, where characteristics like simulation of graphical presentation, kinematics, physical interpretation of the object, weather simulation and simulation cycle method were considered. In terms of software openness, each presents its advantages, FSX presenting for the first time an open API, and other products with their long-time known interfaces. This flexibility allows researchers to choose the combination of simulator modules that best suits their interests, thus being able to focus on the development or improvement of either the graphical content, the airplanes' model, or, in our case, the creation of automated aircraft able to communicate with one another and coordinate actions.

A variety of software able to simulate dynamics and graphical interfaces challenges the researchers in order to choose the best solution to their investigations. Further works need to better related and parameterize the solutions in defined scientific parameters in order to assist choices in new researches.

REFERENCES

Aeronautical Development Agency (2001). Pcs in flight simulation research the lca (navy) experience.

Air & Space Europe (1999). Civilian applications: the challenges facing the uav industry. *Air & Space Europe*, 1(5):63–66(4).

Allouche, M. (2000). The integration of uavs in airspace. *Air & Space Europe - Operations and Safety*, 2:101–104(4).

CloudCap Technology (2007). Piccolo system software. Web Page. Available online at http://www.cloudcaptech.com/resources_autopilots.shtm.

FlightGear (2007). Flightgear official site. Web Page. Available online at http://www.flightgear.org/.

Jackson, E. B. (1995). Manual for a workstation-based generic flight simulation program (larcsim), version 1.4. Technical Report NASA-95-tm110164, NASA.

Laminar Research (2007). X-plane flight simulator. Web Page. Available online at http://www.x-plane.com.

Lewis, M. and Jacobson, J. (2002). Game engines in scientific research - introduction. *Communications of the ACM*, 45(1):27–31.

Lian, Z. and Deshmukh, A. (2006). Performance prediction of an unmanned airborne vehicle multi-agent system. *European Journal of Operational Research*, 127(2):680–695. Available online at http://ideas.repec.org/a/eee/ejores/v172y2006i2p680-695.html.

McManus, I. A., Greer, D. G., and Walker, R. A. (2003). Uav avionics 'hardware in the loop' simulator. In *10th Australian International Aerospace Congress, Proceedings of, Brisbane, Queensland, Australia*.

Microsoft Corporation (2006). Simconnect sdk reference. Web Page. Available online at http://www.fs-seine-75.com/SDK/Corennect.htm.

MultiGen-Paradigm, Inc. (2007). Openflight official site. Web Page. Available online at http://www.multigen-paradigm.com/products/standards/openflight/ index.shtml.

Schiff, B. (1978). *Flying, A Golden Science Guide*.

Stock, C. (2007). Flight dynamics: Fsx and x-plane battle it out. Internet-based Magazine. Available online at http://www.simpilotnet.com/index.php?option=com_content&task=view&id=20&Itemid=9.

Unmanned Dynamics (2007). Aerosim blockset. Available online at http://www.u-dynamics.com/aerosim/default.htm.