A 3D Interface for an Unmanned Aerial Vehicle

B. Cervin [1], C. Mills[1], and B. C. Wünsche[2]

[1] Dept. of Software Engineering, University of Auckland, Private Bag 92019, Auckland, New Zealand.
[2] Graphics Group, Dept. of Computer Science, University of Auckland, Private Bag 92019, Auckland, New Zealand.
bcervin@gmail.com, cmil074@ec.auckland.ac.nz, burkhard@cs.auckland.ac.nz

Abstract

The Defence Technology Agency (DTA) of the New Zealand Defence Force is developing a remotely controlled aircraft to provide reconnaissance for units in the field. In this paper we describe the development of a real-time 3D interface for this aircraft which is used for remote control and mission planning. The interface uses a public domain flight simulator for 3D graphics display and integrates terrain data supplied by the Joint Geospatial Facility (JGSF) and Geographical Information Systems (GIS). Furthermore the design of the interface enables its integration into the Virtual Maritime System Architecture which will be used for the simulation of large scale interactive multi-user battlefield scenarios.

Keywords: Graphical user interfaces, flight simulator, military simulations, terrain visualization

1 Introduction

The Defence Technology Agency (DTA) of the New Zealand Defence Force is developing an Unmanned Aerial Vehicle (UAV) for deployment in New Zealand and abroad. The UAV will be used primarily for reconnaissance of surrounding areas. It utilises a camera that relays images back to its operators. The images not only provide reconnaissance information but are also used for controlling the aircraft remotely.

In order to improve the control of the UAV and in order perform simulations and training an interface is needed which displays the aircraft in a virtual three dimensional (3D) environment. Currently the DTA uses a roadmap-like 2D display to control the flight path of the UAV. This is not satisfactory for mission critical deployments.

In this paper we present an interface which visualizes the UAV in terrain generated from data provided by the Joint Geospatial Facility (JGSF) and Geographic Information Systems (GIS). Our tool makes it possible to view the terrain from the UAV camera position in order to match real images with the virtual terrain. The virtual terrain can be augmented with additional information such as roads and buildings which is important for simulating battlefield scenarios. In addition a general view of the UAV can be selected and positional and flight information is displayed in order to facilitate the remote control of the aircraft.

2 Flight Simulators

After consultation with the project leaders it was decided that instead of creating a 3D simulator from scratch it would more appropriate to use an already existing platform, such as a flight simulator, as a terrain engine. The decision was made to use a flight simulator rather than a game engine because flight simulators offer more advanced controls for aerial vehicles and popular flight simulators, such as FlightGear and Microsoft Flight Simulator, have scenery from all around the world available that can be used for training purposes.

The following constraints have to be met by the flight simulator: the UAV data needs to be plugged into the flight simulator so that the correct position and orientation of the aircraft can be displayed (including roll, pitch, yaw and altitude). It must be possible to convert real terrain data from the JGSF and GISs from their original form (i.e. Digital Elevation Maps (DEM) and Vector Product Formats (VPF)) into the terrain format used by the flight simulator. It must be possible to augment the terrain with features including roads, rivers, lakes and cities to provide a realistic 3D environment.

Initial research narrowed the possible choices for a flight simulator down to FlightGear and Microsoft Flight Simulator 2002 (MSFS). In making our choice we considered their usability, extendibility, graphics rendering quality, and tools available for scenery generation.

Microsoft Flight Simulator 2002 is a proprietary flight simulator developed by Microsoft Corporation. It is supported by commercial plug-ins that can be purchased online. The main interfaces that can be used for interaction use Direct X [1,2]. FlightGear is an Open-Source flight simulator developed in C/C++ under Linux and distributed to most systems, including Microsoft Windows [3].

We chose FlightGear because of its usability, specifically in terms of network play. It allows the slaving of an instance of FlightGear from another instance by sending packets of information using an exposed C++ interface class (FGNetFDM). By altering the fields in this class and then sending it to an instance of FlightGear we can control the aircraft.

A second advantage of FlightGear is that its graphics output is superior to MSFS and higher resolution terrains are available (3-arcsec scenery ≡ 100m resolution). Finally several excellent tools are available for FlightGear, specifically for terrain generation. In particular the open source project Terragear [4] was commenced for developing scenery for FlightGear. Scenery can be represented as Digital Elevation Maps (DEM) or by Vector Product Formats (VPF) which are supported by geographical associations around the world. In contrast the BGL terrain format used by the MSFS terrain engine is a proprietary format and no tools seem to be available for converting common terrain formats into BGL scenery files. Two other useful FlightGear tools for military simulations are Atlas, which provides a 2D map with moving map pointer for FlightGear; and the Flight Gear Scenery Designer project, which provides a way of generating terrain data from maps including placement of 3D objects and the use of photos for scenery.

3 System Design

Figure 1 shows an overview of our system. The UAV interface receives flight data and images from the UAV via a wireless connection. Several instances of FlightGear can be connected to the interface showing different views of the data. For example, one instance can show the terrain from the camera position for use by military planners and strategists, whereas another instance can show a 3rd person view of the UAV in order to facilitate its remote control.

The design of the UAV interface is illustrated in figure 2. The "UAV Poller" handles the polling of the UAV (using self-defined packet-based communication), the processing of the packets and then the delivery of formatted data to FlightGear. Communication with the UAV is handled via serial port, but the code is modular enough so that any future communication medium (e.g. USB, Firewire) can be used. The self defined packets are small and contain a few ASCII characters that are used as values depending on the packet header. Packets also have checksum fields to test the integrity of the packet [5]. Faulty packets are detected and discarded. A more detailed description of the system design is found in [6].

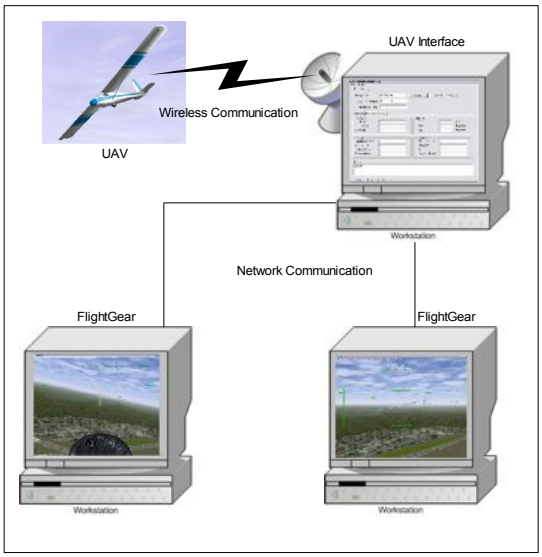

Figure 1: UAV System Overview.

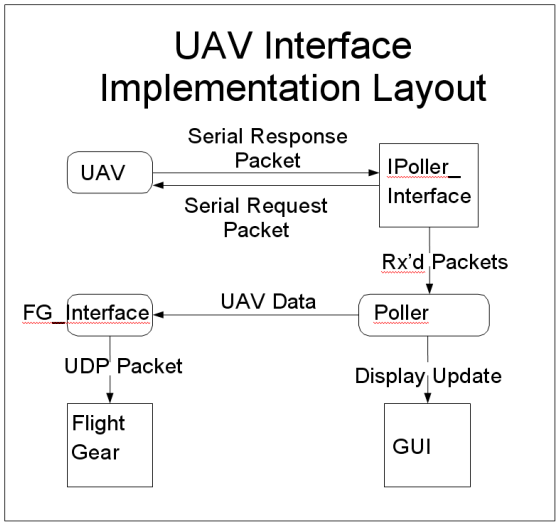

Figure 2: UAV Interface Design.

We use Terragear to compile terrain data into scenery files for FlightGear using the following steps:

I. Convert the raw data into ASCII files for processing.

II. Cut the resultant data into tiles and convert them into polygonal digital elevation maps (DEM) (we use the tools 'demcop' and 'hgtchop').

III. Generate the scenery objects and textures for the terrain. Using the tool 'tgvpf' we can generates different layers of scenery, such as landmass, farmland, towns, cities, lakes and roads.

IV. Combine the DEM, textures and scenery objects into tiles for FlightGear.

We found that the scenery generation takes several hours even for simple terrains. In future we will use a client-server architecture to create realistic and accurate 3D terrains of New Zealand and other operational areas from non-publicly available high resolution data provided by the Royal New Zealand Navy (RNZN) through the JGSF. The JGSF data has a resolution of 20m which is considerably higher than the 100m resolution of publicly available data.

4 User Interface

The graphical user interface (GUI) of the UAV interface displays information received from the UAV and sent through to FlightGear (see figure 3). The FGNetFDM interface class allows the updating of many variables within FlightGear. This includes altitude, speed, direction and of course location. We can also enter values for the battery capacity and other flight instrument data into FlightGear, but there is no specific way of displaying this information in FlightGear.

Figure 3: The UAV Serial Interface.

An additional interface is used to specify the socket configuration. Since the UAV is currently still in development we use a simulator (HawkSim) developed by the DTA. Figure 4 shows the output data of the simulator in raw form. In our simulations the HawkSim was queried every 20ms to obtain UAV data.

Figure 4: The UAV Simulator Interface.

Additional interfaces for selecting aircraft models and for loading terrains are available through the FlightGear simulator.

5 Results

Initial test results for our simulation interface are promising and we were able to simulate UAV reconnaissance missions over local terrain. Figure 5 shows the UAV over the Hobsonville Aerodrome (RNZAF) with the Auckland harbour in the background. Since the model of the UAV is classified we use instead a model of a Schweizer 2-33 which is most similar to the UAV in terms of size and shape. In our tests we queried the HawkSim simulator every 20ms to obtain UAV data. The FlightGear instance updated itself every 20 ms (50 Hz) however there was still some lagging on the display. This could be due to the UAV simulator, giving approximate data. It could also be attributed to slow hardware and network connections.

The purpose of this application of FlightGear is to enable a broad dissemination of UAV information to personnel in the navy. The information will be sent to battlefield commanders in order to facilitate the strategic planning and execution of missions.

6 Conclusion

Integrating the UAV into FlightGear was a success. The FlightGear instance flew according to the data that it received from the UAV and gave the correct position and orientation. The application is a small step towards a 3D environment to aid command and control. Further research needs to be carried out before commencing large scale projects; however, current feedback indicates that the New Zealand forces consider this research worthwhile.

Figure 5: The UAV flying over the Hobsonville Aerodrome (RNZAF) with the Auckland harbour in the background.

7 Future Work

At the moment the UAV is the only unit plugged into display software which enables commanders to get a picture of the battlefield. FlightGear was used in this application because the UAV is an aircraft and it was easier to use a flight simulator that already deals with aircraft rather than software that can generate troops, armoured vehicles and so on. Future projects will try to incorporate troop movements into this interface. Current research indicates that this could be achieved using the "external flight dynamics model" which provides multiple user support. Ground troops could hence be represented as aircrafts at zero altitude. Currently the implementation uses this feature to simulate shadows of an aircraft cast onto the ground.

In the next phase of this project we will investigate how the UAV interface can be integrated into the Virtual Maritime Systems Architecture (VMSA) which is used in virtual battleground scenarios conducted by New Zealand, Australia, Great Britain, Canada and the USA [7].

Computer graphics applications (specifically games) already deal with similar issues. As an example, Ground Control 2 [8] already has unit generation and map generation software (see figure 6). It also provides network interfaces so that multiple players can participate in a simulation. The ultimate aim is to achieve Network Centric Warfare (NCW) [9] where all data about units is centrally available and can be displayed within our simulator on demand.

8 Acknowledgements

We would like to thank Matt Hopkins from the Defence Technology Agency for his help and endless enthusiasm, Sally Garrett and the Joint Geospatial Facility for supplying us with terrain data, Microsoft Corporation New Zealand for donating a copy of the MS Flight Simulator and last but not least the developers of FlightGear and Terragear for their information and help.

Figure 6: Screenshot of the game 'Ground Control 2' [8].

9 References

[1] Microsoft Corporation, Microsoft Flight Simulator SDK, URL: http://www.microsoft.com/games/flightsimulator/fs2002_downloads_sdk.asp.

[2] T. Gregor and D. Gregor, Scenery Hall of Fame homepage – MS Flight Simulator 2002 tutorial, URL: http://www.scenery.org/tutorials_fs2k2_SDK.htm.

[3] FlightGear homepage, URL: http://www.flightgear.org.

[4] TerraGear homepage, URL: http://terragear.org/.

[5] Phil Strong, The Defence Technology Agency (DTA) UAV Uplink and Downlink Specification (classified information).

[6] B. Cervin and C. Mills, A 3D Interface for an Unmanned Aerial Vehicle, SE Stage 4 Project report, Department of Software Engineering, University of Auckland, Auckland, New Zealand, October 2004.

[7] Shane A. Canney, Virtual Maritime System Architecture Description Document Issue 2.00, Virtual Maritime System Document 00034, Defence Science and Technology Organisation, Australia, 2002.

[8] Massive Entertainment, Ground Control 2 homepage, URL: http://www.groundcontrol2.com/modules/news.

[9] David S. Alberts, John J. Garstka, Richard E. Hayes, David A. Signori; Understanding Information Age Warfare, Command Control Research Program (CCRP) Publications, United States of America Department of Defense, August 2001, URL: http://www.dodccrp.org/publications/pdf/Alberts_UIAW.pdf.