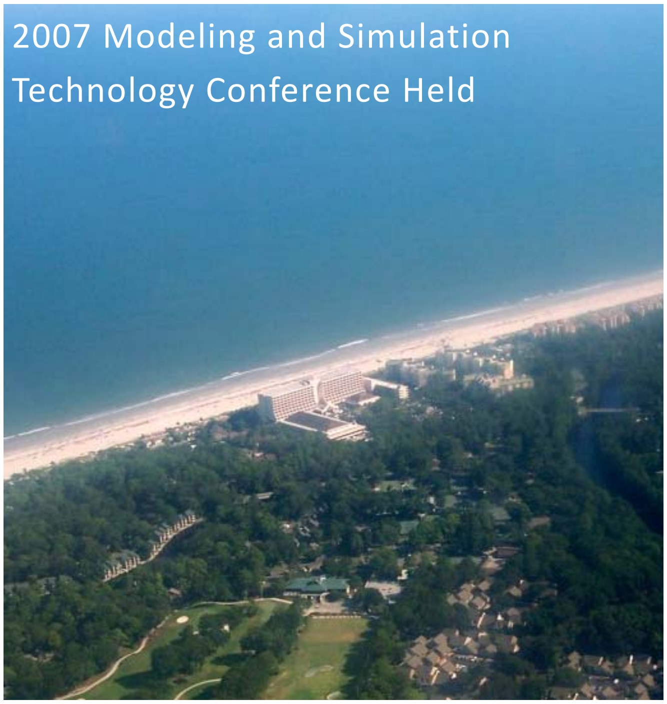
Modeling & Simulation

A Publication of the AIAA Modeling & Simulation Technical Committee

Volume 1, Issue 1 January 2008

2007 Modeling and Simulation Technology Conference Held

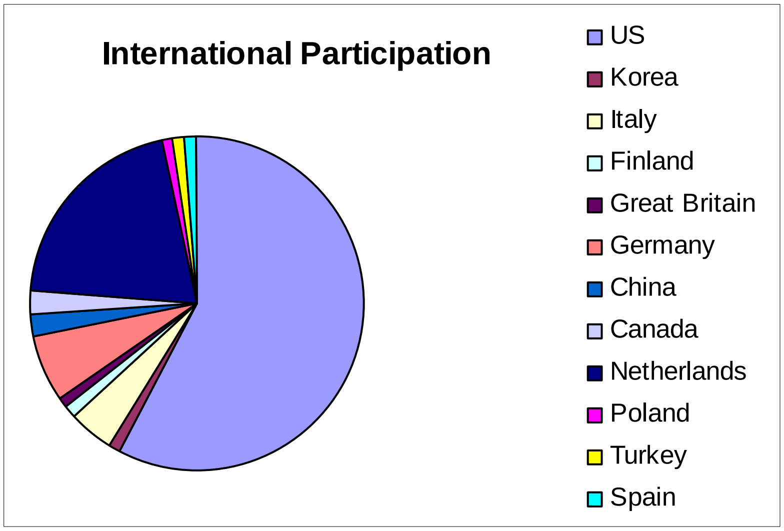


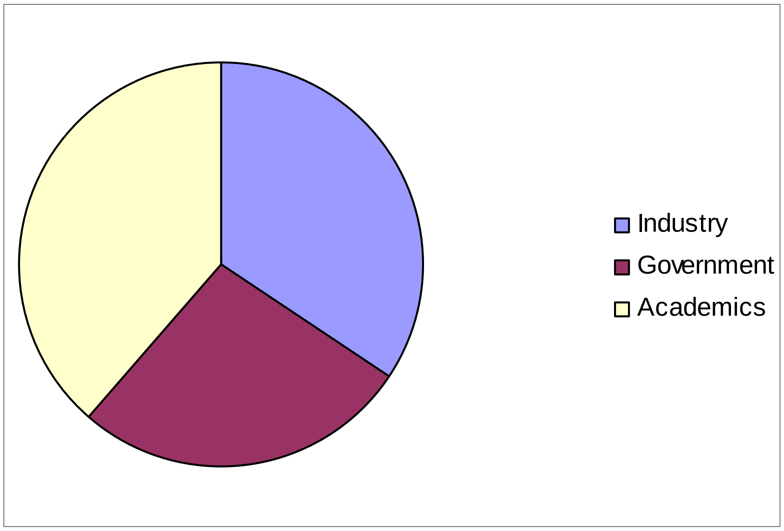
The conference center at Hilton Head, site of the 2007 conferences.

A very successful AIAA Modeling and Simulation Technologies Conference was held at Hilton Head, South Carolina on August 20-23, 2007. Terry Burress of Lockheed Martin Aeronautics served as the Technical Chair and Brent York of Indra Systems, Inc. served as the General Chair. Attendees presented ninety-seven papers distributed among eighteen sessions. Ruud Hosman of Aerospace Man-Machine Systems Consulting chaired a Motion Working Group Panel. The agenda of the Motion Working Group is to gather all known and verified facts about motion perception, its influence on pilot's control behavior, and its application to simulation motion cueing. The MST plenary session hosted Michael Conroy, a NASA simulation manager at Kennedy Space Center, who gave a presentation entitled "Simulation in NASA Space Exploration".

Two panel sessions were well-attended and received good reviews. Edward Burnett of Lockheed Martin Aeronautics Company presented "Modeling, Tuning and Optimizing Multi-Loop Control Systems in Simulink". Ravindra Jategaonkar of DLR, German Aerospace Center and David R. Gingras, Bihrle Applied Research, Inc., presented "Aerodynamic Database Validation and Update."

Next year's conference will be held at the Hawaiian Convention Center in Honolulu, Hawaii on August 18-21. The general chair will be Feri Weller of Northrop Grumann Corporation and the technical chair will be Matt Duquette of the U.S. Air Force Research Laboratory.





Inside this issue:

Note from the Chair

Joe Nalepka

Before proceeding with a few words on the upcoming year, I would like to once again say thanks to our outgoing Chairman, Bill Bezdek, for all of the hard work he has put into the MSTC for the last two years. He has made following him as Chairman a hard act to follow. As we approach the 2008 U.S. Presidential elections, we are becoming overwhelmed with campaign slogans and promises from each of the candidates. Sticking with the spirit of this season, I have decided to create my own campaign slogan for my tenure as Chairman. I am going to "Put More Tech into the Technical Committee." Through our normal business meetings, I am going to experiment with different ways of bringing in technical presentations and discussions from our membership. This will not only give us an opportunity to better understand the research activities being conducted at the member organizations but also allow us to discuss technical issues/problems that may be of mutual interest to the MSTC. I would also like to bring guest speakers to our meetings to talk about a wider range of modeling and simulation topics. The sky is the limit in what we can do. I am always looking for suggestions and would welcome any inputs from the committee membership. I look forward to the challenge of Chairman for the next two years as well as the opportunity to work with each of you on committee related activities.

DAVE-ML Standard Progresses

Bruce Hildreth, Bruce Jackson

"The purpose of this standard is to make it easier to share simulation models between participating facilities"

The MSTC Simulation Standards subcommittee submitted draft 2 of the "Standards for the Exchange of Simulation Data" to the AIAA Committee on Standards on 12 October 2007. This document provides a recommended standard for encoding of data elements of a typical rigid-body aerospace vehicle flight dynamic model in a form suitable for transmittal between simulation facilities.

The purpose of this standard is to make it easier to share simulation models between participating facilities without imposing an undue burden on those facilities to adopt new software languages, methods, or data flows. Once import and export scripts or tools are developed or modified for each site, it should be possible to exchange models much more rapidly and precisely than current practice. This standard incorporates axis definitions (referenced from the Recommended Practice ANSI/AIAA R-004-1992), a suggested variable naming convention, and an XML-based encoding grammar (DAVE-ML) for static subsystem models, such as aero or inertial models, that includes data tables, equation buildups, confidence intervals, and verification checkcase data.

This encoding method has been used extensively by the Australian Defense Science Technology organization (DSTO) for several years; they have graciously provided a C library/API, Janus, to facilitate creation and parsing compliant models. A NASA open-source Java-based package, DAVEtools, is also available to convert a DAVE-ML compliant model into Simulink(R).

The draft standard and annexes are available for download from:

http://daveml.nasa.gov/AIAA_stds

More information on DAVE-ML and example models are also available at:

http://daveml.nasa.gov

This effort has been led by Bruce Hildreth of SAIC, former MSTC chair, and Bruce Jackson (NASA Langley). Geoff Brian (DSTO), Brent York (Indra Systems) and Bill Cleveland (NASA Ames), as well as many others helped in this work.

OpenEaagles, An Open Source Simulation Framework **Douglas Hodson**

OpenEaagles is an acronym for the Open Extensible Architecture for the Analysis and Generation of Linked Simulations. The “Open” prefix in the name has been included to emphasize that the software is freely available as open-source software. Furthermore, the “Open” prefix is also used to distinguish the open-source framework from its proprietary and copyrighted United States Air Force counterpart, Eaagles. The primary difference between OpenEaagles and Eaagles is that the former does not include a few components and functional capabilities. Otherwise the fundamental infrastructures offered by the two are nearly identical in every aspect. In other words, OpenEaagles is a proper subset of the proprietary Eaagles software.

OpenEaagles was open-sourced and released into the public domain in July of 2006. But the software it’s derived from, Eaagles has been in active development for over 10 years. This was done to facilitate usage among universities that had indicated an interest in using it as a teaching aid. Releasing the software has also enabled and encouraged software developers in the business of building simulation applications to leverage its capabilities.

As a framework, OpenEaagles provides a structure for developing simulation applications. It increases the productivity of software developers in the business of developing scalable, virtual, constructive, stand-alone, and distributed simulation applications. It leverages modern object-oriented software design principles while incorporating fundamental performance-oriented real-time system design techniques to meet human and hardware in-the-loop interaction/timing requirements. The primary emphasis of OpenEaagles has been to design and develop virtual simulations that interact with a human or hardware. A flight simulator involving a human pilot flying a simulated aircraft is a good example.

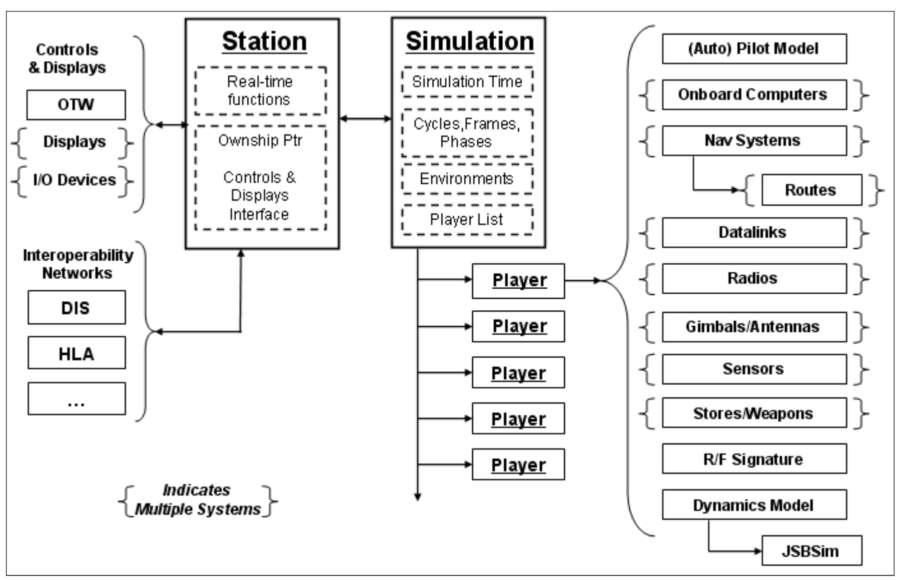
The framework provides a large, coordinated collection of C++ classes that provide platform-independent functionality for many commonly used constructs in a simulation. These classes are organized into a set of layered libraries, each provided specialized capabilities. At the foundation is the “basic” library. It provides a C++ system object and classes for storing and manipulating physical entities such as: decibels, Euclidean coordinates represented as latitude-longitude pairs, colors represented using different standards and even a parser for reading a simple input language. It also provides classes for commonly used data structures like stacks, queues, and tables.

From this foundation, two distinct and independent class hierarchies are defined to support the development of simulations and operator-vehicle interface displays utilizing OpenGL-based graphics. The simulation side provides a wealth of capabilities including abstract classes for representing a variety of player or entity types such as aircraft, ships, tanks, ground vehicles, space vehicles and even lifeforms. To provide access to a large number of dynamics models, an interface class to fully utilize JSBSim is also supported. Also

(Continued on page 4)

“The primary emphasis of OpenEaagles has been to design and develop virtual simulations that interact with a human or hardware. A flight simulator involving a human pilot flying a simulated aircraft is a good example.”

OpenEaagles Simulation structure.



(Continued from page 3)

included in the simulation hierarchy is a complete radar modeling environment.

The graphics hierarchy provides a mature collection of classes that can be used to render instruments that are commonly used in operator-vehicle interface displays. A sampling of the instrument objects available include: analog dials for altimeters, dials for direction finders, speedometer dials, and even landing gear indicators.

For more information about OpenEaagles, see the web site at www.openeaagles.org

The framework provides sophisticated classes that can be used to generate three dimensional moving terrain maps and drive Out-of-the-window (OTW) visual systems. In addition, several convenient interfaces to interface third party software GUI tool kits such as: Fox, FLTK, and wxWidgets are available.

Building an OpenEaagles –based simulation application consists of linking the simulation models, to graphical views and input/output devices. In other words, the framework promotes the model-view-controller design pattern. Not only is this a good software programming practice, it also enables developed simulation models and graphical displays to more easily be used and interfaced to other simulation applications.

To demonstrate the flexibility of the framework beyond typical flight oriented applications, a simulation of the Apollo era Lunar Module performing a rendezvous with the Command Service Module in earth orbit (as with Apollo 9) was shown at I/ITSEC 2006.

From a simulation standpoint, OpenEaagles is a time-stepped system. As a real-time system, OpenEaagles is executed as a cyclic-based foreground/background system. The foreground processes the time-stepped mathematics of the represented models, while the background performs less time-sensitive processing, such as drawing graphics, saving data or interacting with a network interface. The simulation application creates the foreground and background threads as needed to take full advantage of multi-core/multi-CPU architectures.

Interest in OpenEaagles has been growing since its release. The Wright State Research Institute just published a modeling and simulation book based on utilizing the framework titled "Design & Implementation of Virtual and Constructive Simulations Using OpenEaagles." This book along with the supplied examples provides a unique venue to the practical issues of building simulation systems. More information about OpenEaagles can be found at: www.OpenEaagles.org. (The OpenEaagles project and website is maintained by a group of dedicated enthusiastic software developers. It is not sponsored by, nor receives funding from, the United States Air Force.)

2008 M&S Conference Abstract Submission Deadline Approaches

Submission Deadline:

4 February 2008

Prospective authors are invited to electronically submit abstracts of 500–1000 words via the AIAA Web site at **www.aiaa.org/events/mst**. This Web site is now open for abstract submittal. The system will allow for file upload in the following formats: Word, WordPerfect, Text, RTF, and PDF. All abstracts must be received no later than **4 February 2008**. Official notification will be sent via e-mail on or about **31 March 2008.**

Authors must submit their final manuscript electronically for publication no later than **11 August 2008.**

Technical areas and topics of particular interest for this year's forum include:

- **History and Future of Flight Simulation**
- **Future-Based Technology Challenges**
- **Next-Generation Training Concepts and Needs**
- **Unmanned Systems Modeling and Simulation**

(Continued on page 5)

(Continued from page 4)

- **Space Systems and Simulations**
- **Threat, Weapons, and Engagement Simulation**
- **Distributed Flight Simulation Technologies**
- **Mobile, Reconfigurable, and Embedded Flight Simulators and Training Aids**
- **Training System Acquisition and Program Overviews**
- **Flight Simulator/Flight Training Device Design, Integration, Acceptance Testing, Verification, Validation, and Qualification/Accreditation**
- **Enhancement of Pilot Decision-Making Skills via Flight Simulation**
- **Human Factors, Human Perception, and Cueing in Flight Simulation**
- **In-Flight Simulation, Virtual Reality, and Low-Cost/PC-Based Flight Simulation**
- **Stimulation vs. Simulation of Aircraft Components and Systems**
- **Flight Vehicle, Aircraft Systems, Environmental, and Hardware-in-the-Loop Modeling and Simulation**
- **Mission Planning (Flight Crew Training and Pre-brief/De-brief System Design with Enhanced Functionality and Capabilities)**
- **Flight Training Facility Issues and Requirements**
- **Simulator Requirements, Fidelity, Standards, and Reuse**
- **Flight Simulator Maintenance and Support**
- **Use of Commercial Off-the-Shelf (COTS) Software for Flight Simulation Development**
- **Advances in 3D, Synthetic, and Virtual Training and Simulation Environments**
- **Flight Simulator Subsystems and Architectures**
- **Simulation Quality Assurance: *Objectives and Requirements***
- **Economics of Modeling and Simulation**
- **Innovative Algorithm Applications: *Scenario Testing, Fitness, Validation, etc.***
- **Civil Aviation Regulatory Requirements, Challenges, and Issues**
- **Air Traffic Management (ATM) Requirements, Challenges, and Issues**
- **Role of Flight Simulators in Aircraft Accident Investigation**
- **Engineering Simulator Use and Techniques: Fire, Emergency Procedures, and Cabin Trainers**

The AIAA Modeling and Simulation Technologies (M&ST) Technical Committee has once again developed a compelling technical program designed to address the challenges being faced today with an emphasis on the changing requirements resulting from military operations including urban combat, while addressing the design and development of flight simulation hardware, soft ware, systems, innovative approaches, applications, and relative advances that keep modeling and simulation tools a viable, effective, and efficient engineering tool. The M&ST Conference serves as a community catalyst that brings together a diverse collection of professionals, including flight simulation and subsystem design engineers, manufacturers, functional experts, operators, training specialists, researchers, and aviation regulators to collaborate, leverage, and continue building on past experiences that will result in better understanding industry and customer needs, while further enhancing the overall capabilities of aviation flight simulation.

The Delta 3D Open Source Game Engine **Perry McDowell**

> *In recent years, the military training and operational industries have begun to use solutions from the entertainment industry, most especially games. However, many of the proprietary solutions in the military game and visual simulation industry do not meet the needs of all developers.*

Delta3D is an open source (Lesser GNU Public License (LPGL)) game engine created at the MOVES Institute, part of the Naval Postgraduate School in Monterey. While originally designed for military game-based simulations, Delta3D can be used as the underlying architecture for a wide range of game applications.

Delta3D Philosophy

In recent years, the military training and operational industries have begun to use solutions from the entertainment industry [1], most especially games. However, many of the proprietary solutions in the military game and visual simulation industry do not meet the needs of all developers. Licensing costs and restrictions make it difficult to produce the large number of applications required for the myriad of training applications the military requires [2]. Delta3D was created to meet the need for an open source commodity game and simulation engine. In examining the problems or proprietary solutions for this space, we came up with a four-part philosophical credo upon which we based building our game and simulation engine:

1. Maintain openness in all aspects to avoid lock-ins and increase flexibility.
2. Maintain the capability to support multiple genres, since we never know what type of application it will have to support next.
3. Build the engine in a modular fashion so that we can swap anything out as technologies mature at different rates.
4. Build a community (or leverage existing ones) so the military does not have to pay all the bills.

The first of these, Maintain openness in all aspects to avoid lock-ins and lock-outs and increase flexibility", addresses three of the problems in the current paradigm. By keeping everything open, no vendor would be able to lock the military into its technology. This would allow any follow-on applications to be bid on by multiple companies, with the resulting competition reducing the costs. Also, lock-outs occur when a vendor decides to no longer support a product; a good example of this is some of the earlier Windows operating systems, which are no longer supported by Microsoft. If a customer had an application which doesn't operate under a new version of Windows, they cannot continue to use the software after support has ended, since they don't have access to the Windows source code. Additionally, because the tools are open, developers have access to the source code. This means that if the tools don't meet the developers' requirements, the developers can change the tools as needed for their applications without waiting for a vendor to decide to do so.

The second principle, "Make it multi-genre since we never know what type of application it will to have to support next", is designed to ensure that Delta3D can meet the developer's needs, whatever they are. Within just one of the military services, the Navy, the number of training applications is immense. When he was the commander of the Navy Education and Training Command (NETC) in 2004, Vice Admiral Alfred Harms estimated that he would need approximately 1,500 training games to meet his requirements of performing all individual training (as opposed to team training) within the Navy [3]. There is no single genre of games that will be able to meet all those requirements, which are just a small portion of all those in the military, not to mention the entire spectrum outside the military. While traditionally game engines have been built for a single genre or even a single game, that model would not work for the military and those interested in a commodity solution. By having one engine which can meet all requirements it is easy to standardize the production pipeline and reuse content for multiple applications, thus reducing the cost involved.

(Continued on page 7)

(Continued from page 6)

The third of these tenets, "Build the engine in a modular fashion so that we can swap anything out as technologies mature at different rates", will allow the engine to be state of the art for a long period of time. Each of the various elements of the engine consists either of an open source library or code developed in house. In either case, we have kept the different modules as separate as possible. Therefore, if one of the modules making up Delta3D is surpassed by another open source project and is no longer the "best of breed", it is possible to replace that module with the better one. This can continue with only minor modifications to the Delta3D API, thus allowing the engine to remain current significantly longer than most existing game engines. While longevity was the primary motivation for Delta3D's modularity, it has led to a near-term advantage as well. In some cases, the commodity solution provided by the open source Delta3D module does not meet the needs of the developer. For example, in a close quarters battle simulation, it might be required to use more advanced avatars than those provided by Delta3D and the developer may wish to use a proprietary solution, such as Boston Dynamics'™ DIGuy ™. Because Delta3D is licensed under the LGPL, such proprietary solutions can be integrated without the requirement of being released as open source, such as would be required under the GPL or other viral licenses. This flexibility gives the developer the freedom to choose the best solution in creating an application.

The final part of the credo, "Build a community (or leverage existing ones) so the military does not have to pay all the bills", is another factor driving us towards an open source solution. The power of open source projects is that the energy of a huge development team can be brought to bear upon problems without actually employing such a large team. By building a well designed system that people are interested in using for their own applications, they will also add improvements to the original system. Over time, these may add up to have significantly more value than the original system. However, building such a community takes a great deal of time. Leveraging existing open source communities by incorporating current open source projects with large developer bases into the engine creates a built-in group of developers. The advantages this accrues will be discussed more below.

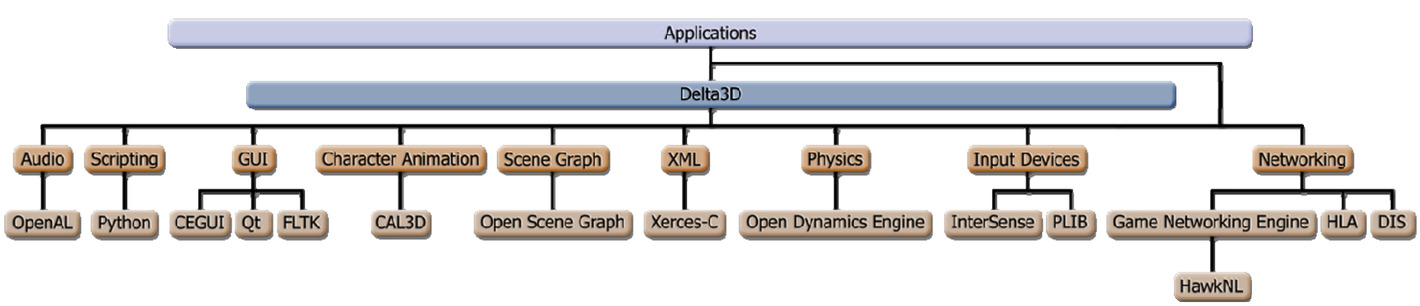
Delta3D Architecture

While building the Delta3D game engine, we had to determine how and what functionality we needed to add to the engine. Our preferred method of adding functionality was using other open source projects which met our requirements and used a license compatible with the LGPL. For example, we used OpenSceneGraph for Delta3D's rendering, Open Dynamics Engine for its physics, etc. Often there were multiple projects meeting these criteria, and in these cases we evaluated the projects for inclusion using two criteria: a project's technical merits and its user support base. The rationale for choosing projects upon their merits is obvious and considering a project's base has allowed Delta3D to gain many "indirect developers", i.e., developers who im-

(Continued on page 8)

"The final part of the credo, "Build a community (or leverage existing ones) so the military does not have to pay all the bills", is another factor driving us towards an open source solution."

Figure 1. Delta3D Architecture



"Delta3D has been used as the underlying architecture for several games and simulations, most of them in the serious games arena.

(Continued from page 7)

prove Delta3D by improving one of the underlying open source projects. Additionally, projects with large user bases are more likely to remain current and state of the art than those with only a small base, reducing the likelihood of needing to swap a module. This allowed us to build a robust game engine while minimizing the work required to be done in house.

As far as determining what to add to the engine, we have tried to keep Delta3D extremely lean and have added only those features which are required for the majority of applications. For a more in-depth discussion of the process of creating Delta3D, see McDowell et al. [4].

The initial modules using open source projects, along with the specific projects used for that module, are shown in Fig. 1.

Delta3D Applications

Delta3D has been used as the underlying architecture for several games and simulations, most of them in the serious games arena. Here is a short description of a couple to demonstrate the scope and capability of Delta3D.

1) Cleared Hot!: Cleared Hot! is a simulation created by USMC students at the MOVES Institute to train FAC(A)'s (Forward Air Controllers, Airborne). It is extremely true to the operation, with correct adherence to procedures required for successful completion of the tasks, and it is being adapted for training use in the military. Cleared Hot! has many characteristics of a game, such as a background story and the ability to move throughout the environment and explore; nonetheless the design gave very little thought to making the game fun for the player. Therefore, Cleared Hot! cannot be considered a true game. A screen shot of Cleared Hot! is shown in Figure 2.

2) HUMVEE Simulation. One of the most recent, and

(Continued on page 9)

Figure 2. Cleared Hot! A forward air controller, airborne, trainer

(Continued from page 8)

most game-based, trainers built using Delta3D is the HUMVEE driving simulation, built by the BMH Operations of Alion Science and Technology as part of the VIRTE Project. This app allows the user to drive around an environment modeled on Twenty Nine Palms Marine Corps Base in California. Additionally, the user can fire the vehicle's .50 caliber machine gun, exit the vehicle, engage night vision, and use many other features. A screenshot is shown in Figure 3.

References

[1] M. V. Capps, P. L. McDowell, & M. Zyda, A future for entertainment-defense research collaboration. IEEE Computer Graphics and Applications 21(1), 37-43.

[2] R. P. Darken and P. L. McDowell, "Open Source Game Engines: Disruptive Technologies in Training and Education", Proceedings of the 2005 Interservice/Industry Training, Simulation and Education Conference (I/ITSEC). Orlando, Florida: National Defense Industrial Association. November 28 – December 1 2005

[3] A. G. Harms Jr., "Investing in people", presentation given April 6, 2004, Pensacola Fl.

[4] P. L. McDowell, R. P. Darken., R. E. Johnson, & J. A. Sullivan, Delta3D: a complete open source game and simulation engine for building military training systems. Proceedings of the 2005 Interservice/Industry Training, Simulation and Education Conference (I/ITSEC). Orlando, Florida: National Defense Industrial Association. November 28 – December 1 2005.

Figure 3 – HUMVEE driving application

For more information about Delta3D, see the web site at www.delta3d.org

Using Google Earth's KML to Visualize Trajectories

Jon Berndt

For more information on Google Earth, see:

http://earth.google.com/

For more information on KML, see:

http://code.google.com/apis/kml/documentation/

The Google Earth tool—which is freely downloadable for personal evaluation and use—can be a useful visualization tool for simulated trajectories . The tool can import files in KML format, where KML stands for Keyhole Markup Language.

The typical structure of a KML file consists of information about the graphical elements (line color and style), and the trajectory data itself. The trajectory data consists of comma-separated data:

- *longitude* ≥ -180 and <= 180
- *latitude* ≥ -90 and ≤ 90
- *altitude* values (optional) are in meters above sea level

[Embedded spaces are not allowed.]

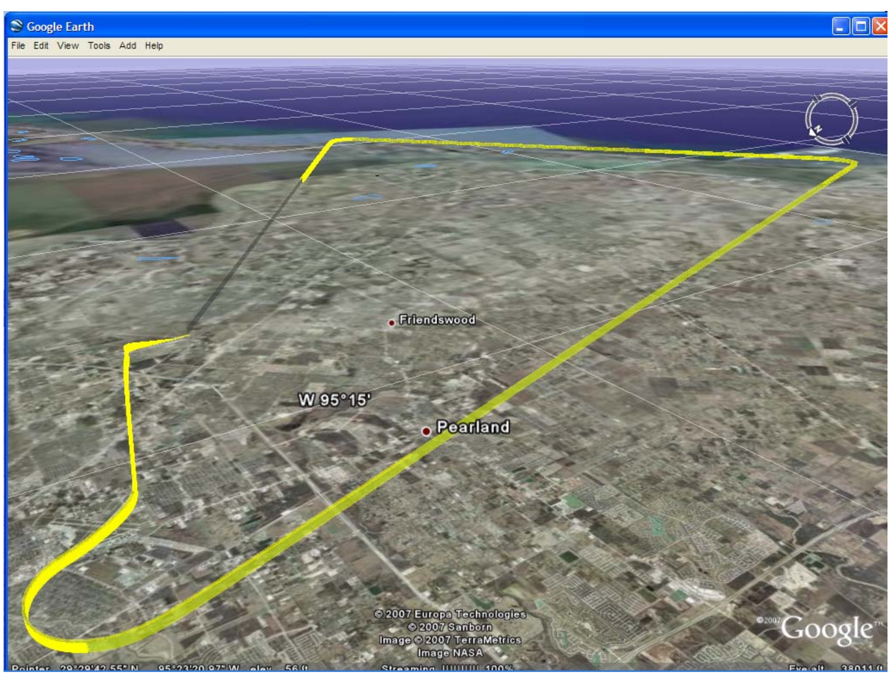
Here is a sample of what a KML file looks like:

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Document>
<name> Paths </name>
<description> Examples of paths. </description>

<Style id="yellowLineGreenPoly">
  <LineStyle>
    <color> 7f00ffff </color>
    <width> 4 </width>
  </LineStyle>
  <PolyStyle>
    <color> 7f00ff00 </color>
  </PolyStyle>
</Style>

<Placemark>
  <name> Flight Path </name>
  <description> Yellow flight path </description>
  <styleUrl> #yellowLineGreenPoly </styleUrl>
  <LineString>
    <extrude> 1 </extrude>
    <tessellate> 1 </tessellate>
    <altitudeMode> absolute </altitudeMode>
    <coordinates>
      …
      Longitude, Latitude, Altitude (data goes here)
      …
    </coordinates>
  </LineString>
</Placemark>

</Document>
</kml>
```



For single flight paths being plotted, the core of the KML file, above, can be used as-is, except for the data , which can be dropped in. It's not difficult to write a script that can automatically generate the above KML file, given the input data. The resulting KML file can be dragged and dropped onto the Google Earth application, resulting in a quick insight into the path being taken.

Similar to plotting continuous lines, discrete points can also be plotted.

I've used this approach to simultaneously visualize sets of trajectories for short rocket flights. For the average computer hardware I have, the application seems to have little trouble displaying in 3D several tens of traces, each consisting of many data points. The scene can be rotated and zoomed in on at will. Finally, the image can be written directly to an image file for inclusion in a presentation.

MSTC Member Close-up: Joe Nalepka

When I am not moonlighting as the MSTC Chairman, I serve as the Chief of the Advanced Structural Concepts Branch at the Air Force Research Laboratory. In this role, I oversee research activities focused on advanced structural technologies for current and future aerospace vehicles. The goal of this research is to maximize structural efficiencies, expand flight envelopes to hypersonic speeds, detect and assess the structural health of aircraft, increase the reliability and durability of airframes and enable new generations of smart air vehicles where the structures do significantly more than carry the load and provide aerodynamic shape. I have been in my current job since July2007 and have worked at the Air Force Research Laboratory for the past 20 years. I have been married for 17 years and have one child. I am an avid sports fan and enjoy playing baseball, softball, basketball, bowling, golf and "sampling" an occasional beer (or two)! One of our family traditions is to travel to a new baseball stadium each summer.

Between our family vacations and my business travel, I have been to half of the MLB stadiums. I am a graduate of the University of Cincinnati, University of Dayton and Wright State University. I also teach an eight grade Sunday school class and help lead a High School Youth Group.

MSTC Holds Spring 2007 Meeting — Terry Burress

The spring meetings of the AIAA Modeling and Simulation Technical Committee are held at a different location each year, with tours of simulation facilities in the area being taken at the same time. The 2007 spring meeting was held at Lockheed Martin Aeronautics in Fort Worth Texas on April 24th and 25th.

With the first day of the meeting being on April 24th, several of the attendees arrived early enough on the previous day to catch a Texas Rangers baseball game in the evening. The Rangers lost to the Seattle Mariners, but those who attended enjoyed the game anyway.

The technical committee's business meeting was held on the morning of April 24th in the Lockheed Martin simulation facility. Lunch of fajitas, a Texas specialty dish, was provided by Lockheed Martin.

Following lunch the attendees were provided with a presentation on the simulation facilities at Lockheed Martin Fort Worth followed by a tour of a part of the simulation facilities. The tour included some of the F-35 simulations with a tour of the hardware-in-the-loop simulations and the motion base system being used for development and evaluation of the F-35 control laws.

After the simulation tour, the attendees took a bus to the factory where they were met by a group of large golf carts. The golf carts provided comfortable transportation for the tour of the mile-long factory. This tour involved seeing the production of F-16s, F-22s and F-35s in progress. The tour included the state-of-the-art production line for the F-35 aircraft.

Lockheed Martin Fort Worth F-35 Vehicle Systems simulation cockpit

After a long day of meetings and tours, it was time to go to dinner. For dinner the group visited Cattlemen's steakhouse in the Fort Worth Historic

(Continued on page 12)

(Continued from page 11)

Stockyards. Luckily the tornadoes that were hitting North Texas at the time totally missed us and we had a great time.

Lockheed Martin Fort Worth Motion System

The following day everyone met at the CAE SimuFlite facility by DFW airport. In contrast to the research and development simulators shown at Lockheed Martin, this facility provides training to the business aviation center. We were shown simulators for an impressive variety of aircraft.. The group also was given a demonstration of the training software used to train the pilots in the classroom prior to using the simulation. This same software was shown in use for both cockpit training up to the full motion simulators. This provided a great overview of the full spectrum of training available at CAE SimuFlite, and the facilities required to host the wide variety of customers involved in business aviation.

At this point it was time to have lunch, and it seemed appropriate to have some Texas Barbecue. Spring Creek Barbecue provided a wonderful example of this food staple, and also provided a large room for the group to have to themselves. This worked out real well, allowing for us to wrap up a little business and have lunch at the same time.

To finish the spring meeting, the final tour was of the American Airlines Flight Academy. This provided a view of yet another sector of the flight simulation market, training for airline pilots and personnel. Though the tour was primarily of the flight simulators, the group was also provided a tour of the area where flight attendant training is performed as well as the facility for the training of emergency procedures, including using emergency exits into water provided by a large pool in the facility. We were also shown the wide variety of simulators available at the facility, which is used to train the American and American Eagle pilots for all the aircraft types flown. Several challenges in providing this level of instruction were discussed, including the maintenance of some of the older simulations to continue to provide training for some of the older aircraft in the fleet, as well as how aircraft software and hardware are used in the simulation to provide realistic avionics simulation.

This concluded the tours, and with the last facility just south of DFW Airport, many of the attendees were able to head home that evening.

Lockheed Martin Electrical Power System Hardware use in the Simulation

MSTC Spring 2008 Meeting Planned: Corinne Ilvedson and Dan Cartmell

The 2008 MSTC Spring Facilities meeting, jointly hosted by Insitu, Inc and Boeing, will be held in Washington State May 6 -9.

Insitu is located in Bingen, WA, a small town in the Columbia River Gorge an hour east of Portland, Oregon. The rural Gorge is home to a technology cluster of high tech companies fostered by "lifestyle entrepreneurs", outdoor enthusiasts who want to work where they play. The Gorge offers world class white water kayaking, wind surfing, kite boarding, mountain biking, snowboarding, skiing, and more.

Insitu is an autonomous systems company producing a fleet of Unmanned Aircraft Systems that are low-cost, long-endurance, and have low personnel requirements. These UASs provide a no-runway launch, unprecedented stabilized day and night video for ISR, robotic flight control, and a no-nets capture. The ScanEagle UAV has a 10 foot wingspan, weighs just 35 lbs, has an endurance over 20 hours, and has logged over 60,000 flight hours since being deployed in 2004.

Boeing Commercial Airplanes is located in the greater Seattle area. Seattle is the urban center of a four-county metropolitan region with 3.5 million people and 1.7 million jobs. This area is a thriving business and commerce center with a diverse economy that spans aviation, information technology, health care, biotechnology, and foreign trade. The area is a major cultural and recreational center with numerous theaters and museums, as well as nearby mountains, lakes and the Puget Sound. Seattle is also the home to Mariner baseball, Seahawks football and Sonics basketball.

Boeing has been the premier manufacturer of commercial jetliners for more than 40 years. The main commercial products are the 737, 747, 767, and 777 families of airplanes and the Boeing Business Jet. New product development efforts are focused on the Boeing 787 Dreamliner, an all new super-efficient mid-size airplane, and the 747-8, a larger, more fuel-efficient and technologically advanced version of this highly successful product. Through Boeing Commercial Aviation Services, the company provides unsurpassed, around-the-clock technical support to help operators maintain their airplanes in peak condition. Boeing also trains maintenance and flight crews in the 100+ seat airliner market through Alteon, the world's largest and most comprehensive provider of airline training.

Schedule of Events

Tuesday, May 6

(Bingen, WA)

- Tour of Insitu UAS simulation activities and production areas followed by a ScanEagle flight demonstration.

Wednesday, May 7

(Hood River, OR)

- Tour of Cloudcap Technology in Hood River, OR. Cloudcap is the leader in UAV autopilots for unmanned, autonomous, micro UAV, mini UAV and micro UAS systems.
- Tour of ANPC (Advanced Navigation and Positioning Corporation) in Hood River, OR. ANPC provides airspace safety, efficiency, and capacity to users of aviation through state-of-the-art tracking and guidance.
- 3.5 hour scenic drive to Seattle, WA

Thursday, May 8

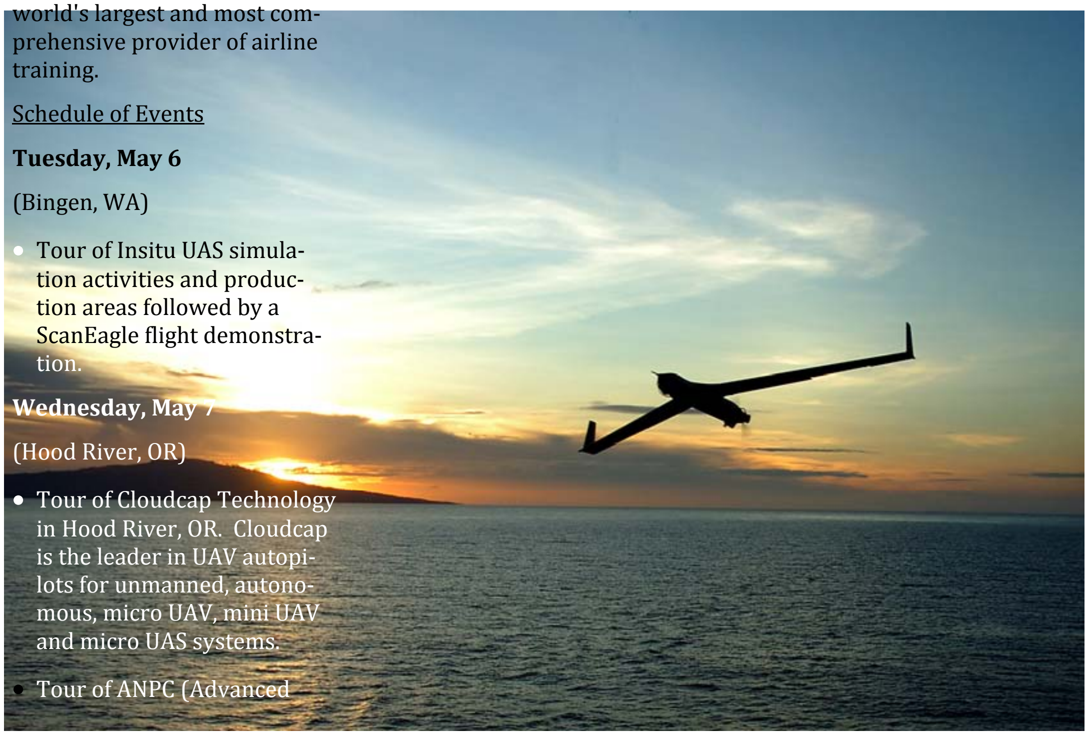
(Seattle, WA)

- Tour Boeing Integrated Airplane Systems Laboratory
- Visit other local companies involved with modeling and simulation
- Mariner's Baseball Game.

Friday, May 9 (morning only)

(Seattle, WA)

- Boeing Commercial Airline Factory Tour



The Insight/ScanEagle UAV. Photo courtesy of Insitu.

Lockheed Martin Team Opens Exploration Development Laboratory to Support NASA Constellation Program

Jon Berndt

The Lockheed Martin, United Space Alliance, and Honeywell jointly funded Exploration Development Laboratory (EDL) in Webster, Texas, was opened with a celebration on the morning of Wednesday, 12 December. In attendance were representatives from all three companies, NASA, local media, state and local government representatives, and many others (including this editor).

According to Lockheed Martin, the facility's capabilities are being designed "from the ground up as a set of distributed System Integration Labs (SIL), including a 10,000 square foot facility in Houston, TC, to join facilities in Denver, CO, Glendale, AZ, and Arlington, VA."

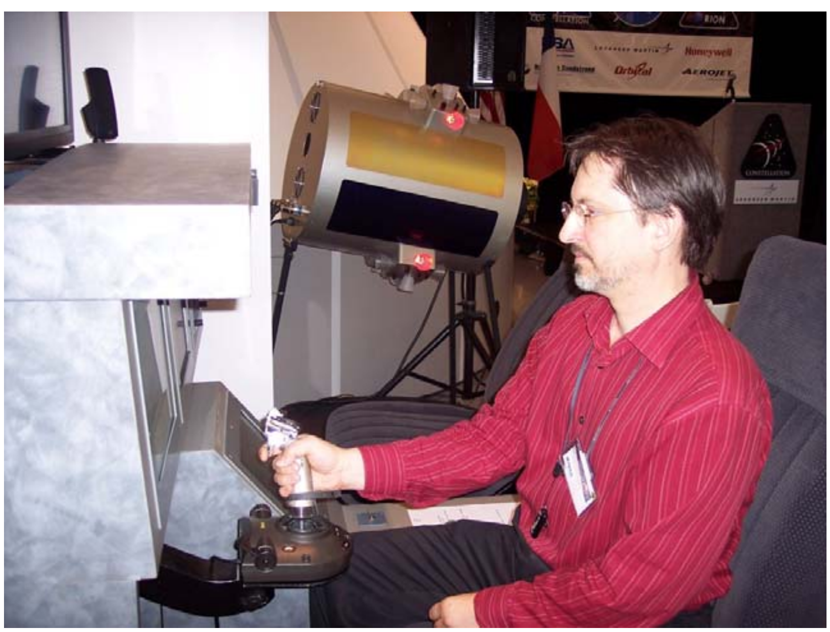
One of the goals of the facility is to reduce program risk by providing an early opportunity for testing avionics and software exactly as they are to eventually fly. Another goal is to enable increased collaboration between centers distributed throughout the U.S.

The EDL will focus particularly on these tasks:

- Human-rated avionics, software, autonomy, sensors and simulations using flight-like elements.
- System engineering, integration, and test.
- Human factors and interfaces.
- Constellation operations.

At the EDL opening, a mockup of the Orion Crew Exploration Vehicle (CEV) was open for viewing. The mockup was built during the proposal process, and externally it represents the 504 vehicle. Internally, the mockup has been updated to represent the more recent 606A iteration.

Adjacent to the CEV mockup was a pilot-in-the-loop / hardware-in-the-loop CEV simulator. The underlying core simulation is Lockheed's OSIRIS simulation. OSIRIS is built on top of the Johnson Space Center developed Trick simulation framework. Interfaced with the simulation are three Honeywell flight control modules of the type built for the Boeing 787. The FCM units connected to the simulator are not spaceflight qualified, but they nonetheless give an early capability in the simulation. Also interfaced with OSIRIS is a propulsion simulation device, which shows visually which reaction control jets are being fired at any time during a maneuver. I had the opportunity to fly a simplified docking with ISS, from a very short range. Translational control was provided through the joystick when a trigger was squeezed, otherwise, the stick acted as a traditional rotational hand controller.



At right, this editor flies a simple simulated docking approach. The cylindrical device in the background shows the jets firing (red lights).