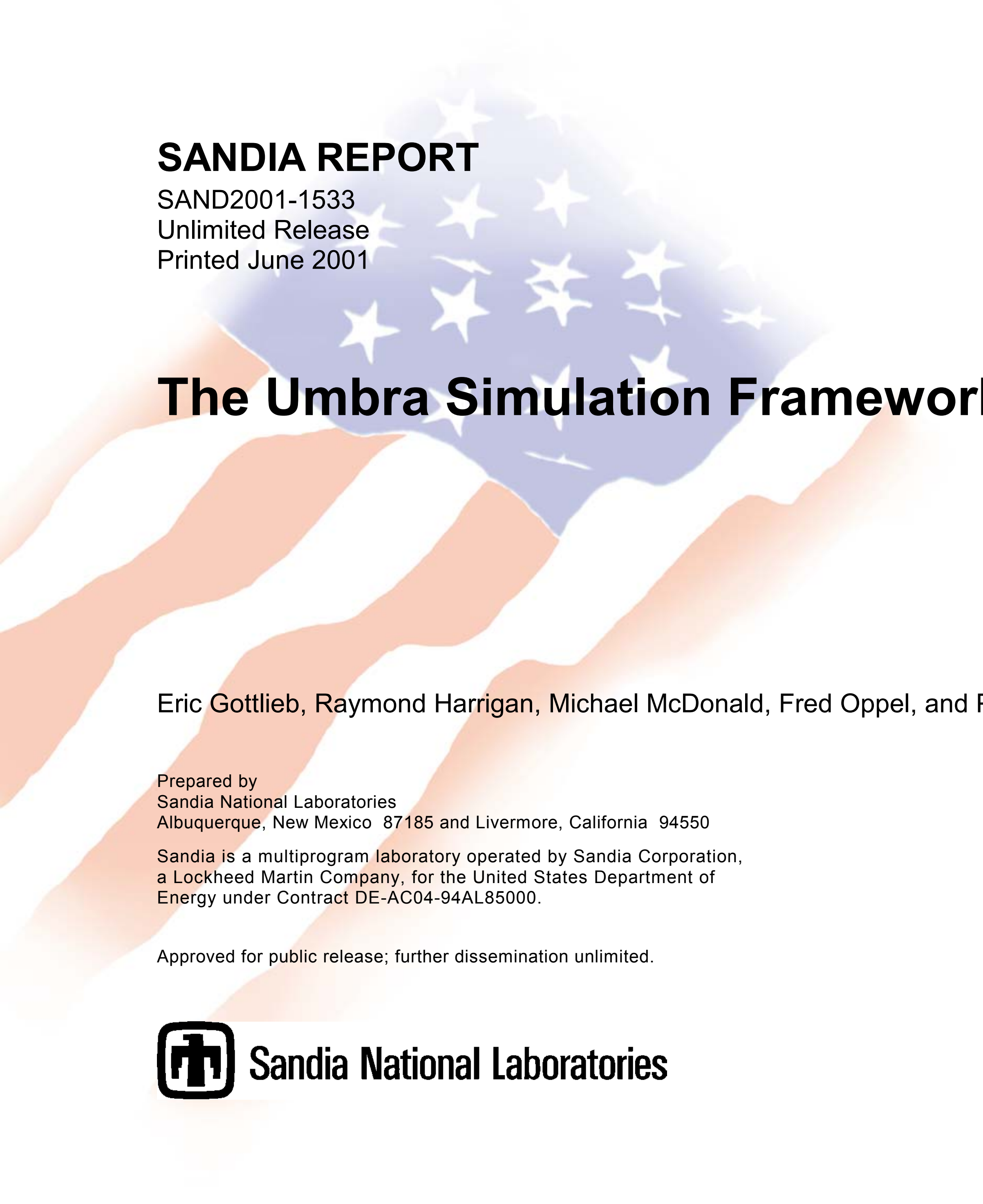

# SANDIA REPORT

SAND2001-1533
Unlimited Release
Printed June 2001

# The Umbra Simulation Framework

Eric Gottlieb, Raymond Harrigan, Michael McDonald, Fred Oppel, and Patrick Xavier

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation,
a Lockheed Martin Company, for the United States Department of
Energy under Contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865)576-8401
Facsimile: (865)576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.doe.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800)553-6847
Facsimile: (703)605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/ordering.htm

SAND2001-1533
Unlimited Release
Printed June 2001

# The Umbra Simulation Framework

Eric Gottlieb, Raymond Harrigan,
Michael McDonald, Fred Oppel, and Patrick Xavier
Intelligent Systems and Robotics Center
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-1004

## ABSTRACT

Umbra is a new Sandia-developed modeling and simulation framework. The Umbra framework allows users to quickly build models and simulations for intelligent system development, analysis, experimentation, and control and supports tradeoff analyses of complex robotic systems, device, and component concepts. Umbra links together heterogeneous collections of modeling tools. The models in Umbra include 3D geometry and physics models of robots, devices and their environments. Model components can be built with varying levels of fidelity and readily switched to allow models built with low fidelity for conceptual analysis to be gradually converted to high fidelity models for later phase detailed analysis. Within control environments, the models can be readily replaced with actual control elements. This paper describes Umbra at a functional level and describes issues that Sandia uses Umbra to address.

## Background

Robotic, or more inclusively, intelligent machine (IM) system technologies are growing in capability and the degree and scope with which they can be integrated. Whereas factories once used robots in cells of integration, modern factories themselves have begun to exhibit characteristics of integrated, distributed, multi-user IM systems. Likewise, new military concepts are emphasizing network-centric warfare with distributed sensing, computer-assisted decision making and automated response over earlier concepts that use discrete robots within military systems. Such new military systems will themselves behave like integrated, distributed, multi-user IM systems. The world is moving from systems with robots to systems that are robotic.

At the same time, rapidly accelerating computer capabilities are making computationally intensive automated planning and programming (AP&P) practical in both IM system design and use. Automated planners can customize aircraft painting programs at a click of a mouse. The graphical programming technologies, characterized by user interfaces with 3D simulation environments and AP&P [SMALL-97], [CHRISTIANSEN-90] have reached production status for applications including refurbishing fighter jet stealth-coatings. Likewise, military operations are benefiting from AP&P. For example, modern military terrain analysis tools [CAMPBELL-96] allow military planners to conduct terrain reasoning and mobility calculations to accurately predict and optimize troop and vehicle movement speeds while maximizing concealment. As graphical programming did with AP&P and simulation, these tools will likely find their use in field command and eventually vehicle navigation and control.

Within this context, in the mid-1990s Sandia foresaw the need for better modeling and simulation resources to fully address its system design and development needs. On the systems analysis side, Sandia needed tools to support decision-making when addressing difficult system-level design questions. For example:

- What is the optimal system configuration?
- What can be automated?
- How do specific technologies impact overall system performance?
- What human interfaces are effective?
- What human to machine communication is needed?
- How can you project hardware test data to new environments?
- How will increased computing impact system performance?
- Which emerging technologies will have the highest payoff?
- What are the system-level failure modes?
- What are the effects of mapping and sensing uncertainties?
- What is the impact of communication disruptions on system performance?
- What countermeasures might be effective against a given system design?

On the system development side, Sandia needed tools to better support both motion and system-level AP&P development and be more usable in control applications than existent tools. In addition, it was believed that better analysis of the systems issues would lead to new classes of AP&P solutions.

The result was Sandia's development of a modeling and simulation environment or framework[1] called Umbra. The remainder of this paper describes that new modeling and simulation environment.

## Description of Umbra

Umbra, Sandia's new modeling and simulation framework, links together heterogeneous collections of modeling tools to allow tradeoff analyses of complex systems concepts. The Umbra framework allows users to quickly build models and simulations for intelligent system development, analysis, experimentation, and control. The models in Umbra include 3D geometry and physics models of robots, devices and their environments. Model components can be built with varying levels of fidelity and readily switched to allow models built with low fidelity for conceptual analysis to be gradually converted to high fidelity models for later phase detailed analysis.

Umbra simulations typically model devices and the environments within which they operate. These devices are modeled in Umbra as embodied agents (defined below), and fine and coarse-grained physical effects models are combined to represent interactions among devices and the physical world. Three-dimensional graphics displays are used for visualization. Umbra can also be used to model disembodied agent systems to support basic research in distributed intelligence.

A key attribute of Umbra is its ability to correctly model the topological structure of integrated systems. For example, robots are typically modeled with behavior, effectors and sensors with separate computational modules. (Here, effector models typically include vehicle motion as well as radio transmissions and other effects modules. Sensor modules typically include geometric sensors such as touch and proximity sensors as well as radio receivers and chemical sensor models.) These effector, sensor, and behavior modules are then configured into meta-modules that are connected in the same way that real sensors and effectors are connected to robot controllers.

Umbra fills a unique niche in modeling and simulation in how it addresses the "middle layer" of simulation between high-level mission analysis tools such as typical DIS & HLA-based military simulators and the low-level engineering analysis tools used for modeling physical phenomena such as MATLAB and ADAMS. As a result, Umbra bridges between low-level engineering and constructive-level scenario simulation environments.

Generally, Umbra incorporates the following capabilities:

---

[1] Frameworks are reusable designs described by a set of abstract classes that constrain the shape of a design while enabling the user to customize the details for a particular application [James O. Coplien. *Advanced C++ Programming Styles and Idioms.* Addison-Wesley Publishing Company, Reading MA, 1992.]. A framework user supplies the code to fill in the abstractions making it a complete system that solves a problem. When many problems need to be solved in a domain, frameworks can be cost-effective. They represent an excellent example of code reuse and programming to an interface rather than an implementation.

- Complex, non-linear world modeling – Umbra models geometry, physics, control laws, sensors, communication, functional subsystems and environments in a modular fashion. This enables the use of models with asymmetric levels of fidelity.
- System-level modeling – modules are configured to mimic system structures.
- Embodied agent modeling – Entities modeled with behavior, geometry, sensing, and physics. Robots are typically modeled as embodied agents.
- Disembodied agent modeling – Entities predominantly modeled with behavioral, as opposed to physical aspects. Typically used to analyze large collective systems of computational agents [BASU-97].
- Encapsulation – Enables modularity and legacy code integration.
- Continuous time and event driven simulation – Allows combining realistic simulation of real-world physics and control laws with high-level commanded event responses.
- Computational steering – Allows users to interactively modify simulations to highlight effects that develop during analysis. Adding unexpected obstacles to terrain models to examine dynamic control response is an example.
- Rapid integration of terrain and feature data – Allows analysis of systems in outdoor terrains and urban environments. Feature data includes obstacle geometries, roads, and mobile vehicles as well as chemical plumes and other sensed physical features.

### How Umbra is Used

Umbra was developed in Sandia's ISRC to support algorithm development. Umbra is used to build virtual environments that can contain mobile robots, robotic arms, and cooperative robotic systems containing several to hundreds of robots along with the environments with which the robots operate. Researchers use these virtual environments to develop and tune new robot control and planning algorithms and to test new system theories.

In addition, Sandia uses Umbra for robot control. Its F-117 robotic aircraft painting system uses Umbra for Graphical Programming [SMALL-97] or simulation-based control. Its dexterous manipulation research platforms use Umbra for its systems integration capabilities. Finally, Umbra is used to simulate and test human interfaces for robot system control. Here, Umbra is used to provide a "virtual" environment that incorporates the same control algorithms and physical models as the actual robots use.

Umbra's ability to correctly model system structures makes it an ideal tool for analyzing system level performance, including failure and recovery modes, of complex robotic systems. For example, in mobile robot applications, robot mobility, sensor-based control and inter-robot communication can be modeled directly. Errors and failures can be injected, for example, in the communications stream both at the individual radio and environment levels to simulate radio failure and jamming. These errors would directly affect the data seen at the monitoring station models, thereby testing the system performance with degraded communications.

Umbra's modular structure and libraries of robots, sensors, and environments allow the rapid configuration of simulations to facilitate tradeoff analyses against mission success parameters. For example, Umbra has been used to analyze a diverse set of mobile robot systems and missions problems. Specific analyses include:

- Evaluation of high speed robot system collision avoidance approaches.
- Technologies required for automatic route planning on both roads and off-road situations where terrain features strongly impact system performance.
- Novel mobility designs and their compatibility with mission requirements for speed, survivability, and supportability.
- Tradeoff analyses involving sensor integration and on-board computational requirements, modes of mobility and energy consumption in varying terrain.
- Tradeoff analyses considering mobile robots vs. fixed devices to create and maintain stable communication networks in rough terrain.
- Performance of robots during system degradation due to subsystem failures.

## Modeling in Umbra

IM systems are modeled in Umbra as sets of connected modules. Each module represents an individual component of the IM system. Robots are typically modeled with separate modules for the sensors, central computer, and actuating hardware. For example, the robot manipulator arm in Figure 1 might be modeled with the modules shown in Figure 2.

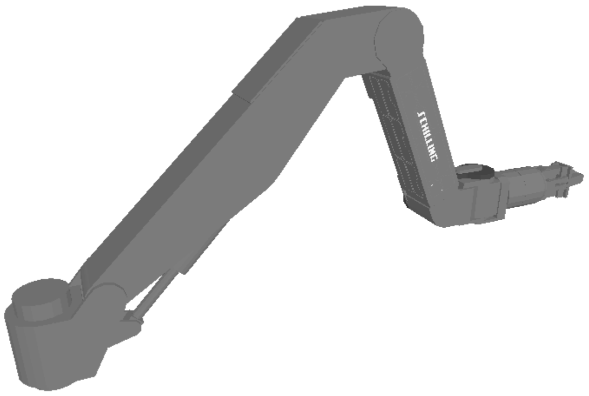

*Figure 1. Robot Manipulator*

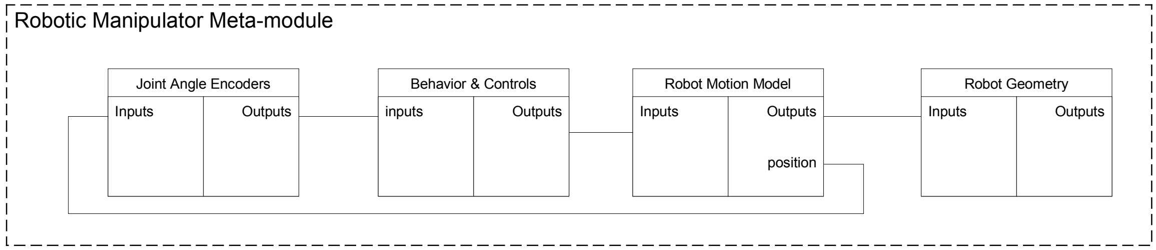

*Figure 2. Umbra Modular Representation of Robotic Manipulator*

Each of these modules encapsulates code, typically written in C++, that defines the module's behavior. Each module's code is written to execute once for each time step. For example, at each time step, robot motion models use dynamic and kinematic models to convert control inputs into robot motions while joint angle modules typically digitize joint position data (from the robot motion model) to generate a discretized representation of the robot's joint angles as read by the controller. At this level of modeling, robot behaviors typically focus on low-level control and reflexive behaviors, as opposed to high-level task behaviors.

A Tcl-based scripting interface is used to direct Umbra to construct individual modules and to configure them into a model, which is also known as a meta-module. In this way, model construction is conceptually similar to other module-based simulation environments in that the modules are connected through an interpreter. Unlike most modular simulators, Umbra uses efficient non-interpreted mechanisms to facilitate data exchange between modules. The result is a very low module overhead.

Just as Umbra models are constructed by combining modules, multi-agent systems are modeled by creating multiple meta-module sets into Umbra's run-time environment. Each meta-module models a single agent. Agents that interact with physical environment models are called embodied agents. Embodied agents typically interact with one another through physically simulated communications links.

Figure 3 shows a representation of a mobile robot modeled as an Umbra embodied agent meta-model. A behavior and control module is at the center of the model. At each update cycle, this module executes code that a typical robot would execute within its main control loop. Typically, this involves investigating sensor and communication inputs along with stored state information and then adjusting control outputs and passing messages. The left of figure xx shows three input modules, a Range Sensor, a Radio Reciever, and a Position (e.g., GPS) Sensor. Typically, sensor and communications input modules like these reference terrain and other geometric models as well as other environment models to compute a simulated sensed value. For example, radio receiver modules can check their position against the position of radio transmission sources and identify transmission obstacles to determine whether a given signal will reach the receiver. Range sensor models typically use geometric analysis techniques to determine whether, for example, an ultrasonic range sensor cone would detect an obstacle. The right of figure xxx shows vehicle physics, radio transmitter and vehicle geometry modules. Vehicle physics modules typically reference terrain database modules to compute how the vehicle would move under its commanded signal. The radio transmission module is used as a sending point for other vehicle radio receivers. The vehicle geometry module is provided to drive the visualization.

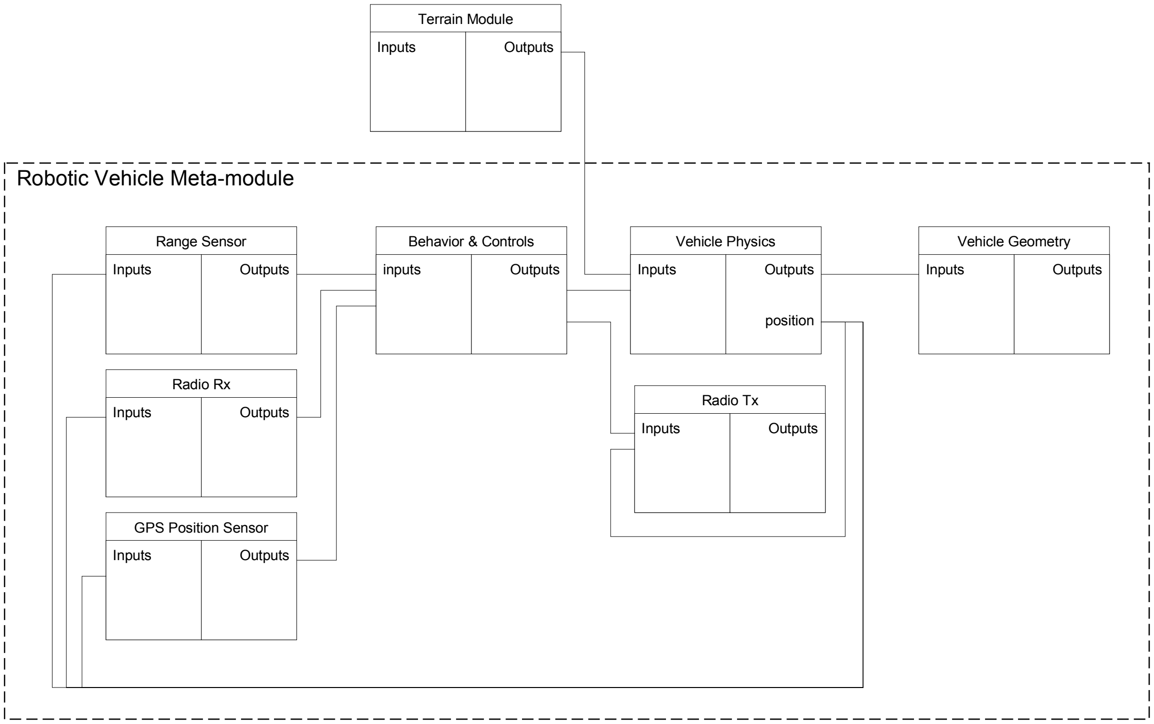

*Figure 3. Umbra Modular Representation of Robotic Vehicle*

Multiple copies of identical robot agents are used to model homogenous robot swarms. Because the modules are configured through a scripting language, code to construct large swarms is straightforward to implement. Because each module encapsulates its local state data, modules remain distinct and are treated the same way whether used in collections or in isolation.

Heterogeneous collections of robot agents are used to model diverse collections of cooperating robot systems. Because modules are written without reference to the devices to which they attach, robot meta-modules in heterogeneous systems often share many vehicle modules. For example, radio transmitters, vehicle sensors, and sometimes even scalable vehicle physics models are used by otherwise distinct agents.

In a similar fashion, networks of unattended sensors as well as intelligent higher-order sensing devices are modeled with physical modules that characterize sensor performance within a complex environment such as a militarily important terrain, behavior or control modules that characterize how the sensors transmit sensed information and communications or radio models that link to sensor monitoring station models. The monitoring station models likewise include communications and behavior modules to receive and process the sensor data.

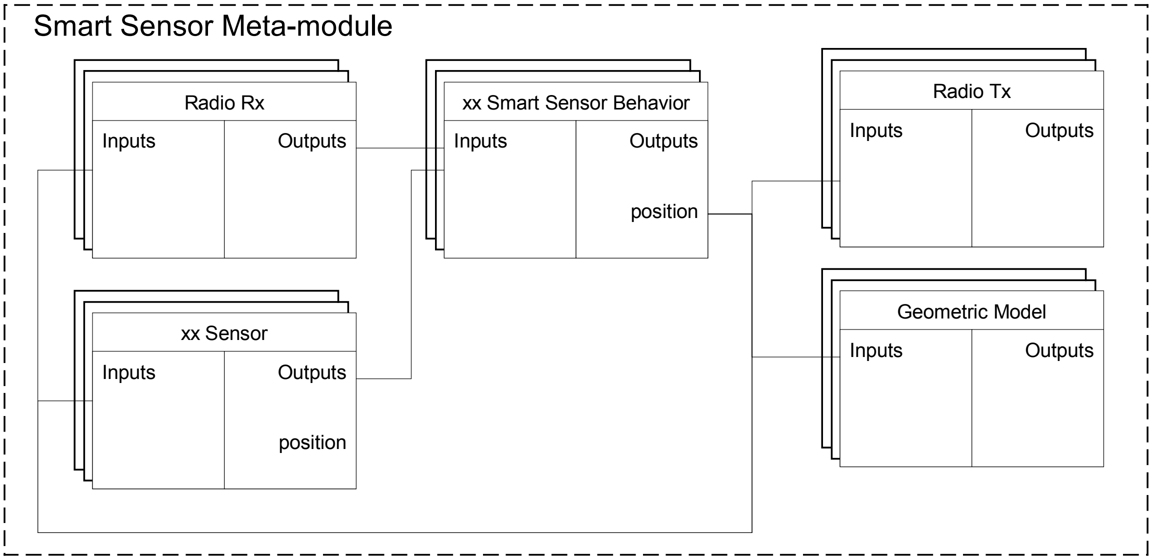

*Figure 4. Umbra Modular Representation of Smart Sensor*

Figure 4 shows a conceptual Umbra model of such a smart sensor meta-module. Here, the xx Smart Sensor Behavior module is like a robot's behavior and control module, the sensor like the robot's sensor modules, and the radio modules like the robot's radio models. Here, however, the sensor behaviors are typically built to process or collect sensor data and then communicate significant information changes rather than to control the motion of a robot. Fixed sensors would likely sense different types of effects than would robot-mounted sensors. For example, a smart seismic sensor might need to be modeled by referencing the world model to determine the position, size, and speed of all surrounding vehicles and to compute a simulated seismic value for the sensor. A high fidelity module might also consider terrain effects in propagating the seismic events.

The introduction of fixed sensors to mobile robot simulations illuminates an additional advantage of the Umbra modular framework. For example, while fixed sensors might communicate with mobile platforms, more computationally efficient modules might be built for fixed sensors to minimize the number of times that radio disturbance effects need to be computed. Umbra's modular framework allows these more efficient modules to readily interoperate with the more complex mobile sensor models.

Here, parallel simulations can be implemented by evaluating the computational load of each module and distributing the modules across computational resources via statically-scheduled, coarse-grained load balancing. In order to support future large-scale, high-fidelity simulations, Umbra was designed with MPI protocol to enable efficient distribution of the computing load across multiple processors. The success of this approach to parallel computing requires careful matching of compute module timing and code granulation.

## Integrating and Federating Umbra with Other Tools

Umbra was designed for integration with other tools. Integration can be achieved by encapsulating other tool libraries as collections of modules or by providing communications or database linkages to other codes. (The latter approach is often called federating.)

As noted, Several of Umbra's libraries are a result of integrating Umbra with pre-existing libraries. For example, Sandia has invested heavily in the Sandia Modular Architecture for Robotics and Teleoperation (SMART) system [ANDERSON-96]. SMART uses a patented process to guarantee stability of modular software control system. While SMART is appropriate for robot control, its modules are also useful in a variety of systems where guaranteed control is not required. With approximately 200 modules currently available, the SMART library provides a significant base from which to build. For example, its kinematics computational modules can be used to simulate robot manipulators and its user input device modules can be used in training simulator interfaces. To provide Umbra with this large database of user interface and robotics control and mathematics modules, special Umbra modules were built to directly use SMART modules.

Umbra has also been integrated with the University of North Carolina (UNC) at Chapel Hill's V-Collide [LIN-96] and Sandia's C-Space Toolkit [XAVIER-97], collision detection libraries for large environment intersection analysis. Unlike the SMART integration, these tools were integrated at the "worlds" level. Here, the geometric computation worlds use a database of scene and other geometry to compute intersections between geometric objects. V-Collide is used for its high speed analysis in static environments. For example, non-contact sensor simulation modules compute whether cones that represent the sensed region intersect with the surrounding environment. The C-Space Toolkit uses high speed analysis of swept volume intersections. An example is rapid motion planning for articulated robot manipulators.

In federation activities, specific communications interface modules are typically developed to support federating Umbra with other simulators. Existing communications modules using networking protocols including CORBA, MPI, or TCP/IP can be leveraged where possible to simplify integration. (These same protocols and multiple world model architectures are exploited to provide coarse-grained parallel computation to support large high fidelity simulations as well as to integrate Umbra with operational robot equipment.) An ongoing effort is being executed to provide Umbra with an HLA interface.

## The Umbra Module Library

Several types of modules have been developed to support Sandia's analysis and algorithm development for mobile robot systems. These modules generally fall under the categories of Environment, Sensor, Communication, and Vehicle Mobility models. In addition, Behavior and Automated Planning modules have been developed to test various high and low level robotic behavior algorithms.

Environment models typically provide a geometric and related datasets that other modules utilize for sensing and physical effects simulation.

- Geometry models are used in conjunction with V-Collide and CSTC to perform polygon-to-polygon intersection tests to determine whether two geometry sets collide. Various Line of Sight modules are used this way to determine whether two points in space are obscured by any geometry.
- Static and dynamic plume modules have been developed to represent Chemical and other sensed plumes spread over terrains. Dynamic and static plumes have been implemented.
- A variety of utilities have been developed to import USGS, SAR, and DEM terrain datasets and to create fictitious terrain models. Elevation and image texture mapping are supported.

Sensor models typically investigate the environment dataset to simulate sensor values upon which robots. Sensor noise modules can be inserted to study noise effects.

- Obstacle detection sensors have been modeled where triangles or sets of cones representing laser stripes and infrared proximity detectors are tested for collision to simulate the response of photo detector-based sensors. (The V-Collide library is used for collision computation.)
- A distance measurement transmitter-receiver module pair has been developed to model Sandia's novel ultrasonic measurement system. (Sandia's ultrasonic transmitters emit an ultrasonic pulse that is heard by receivers on other vehicles. A radio transmission is made at the same time the pulse is sent, allowing each vehicle to measure ultrasonic time of flight and thereby distance from the sending unit.) Ultrasonic receivers modules provide chirp results as time of flight and power level and are modeled by testing line of site collisions (via the V-Collide library.) Power is blocked by obstacles but otherwise decreases with a $1/r^2$ law.
- GPS and compass sensors have been developed to provide a device position in space discretized to simulate GPS and digital compasses at various levels of resolution.
- Plume sensors have been developed to measure local values of plumes (see environment model). Response delays are modeled to reflect the long delays in many chemical sensors.

Communications models are important for modeling distributed multi-robot systems using radios or other communications networks to communicate. For radio communications, line of site and distance are measured between geometric nodes. The module returns a power function that drops as $1/r^2$ for line of sight and $1/r^4$ if an obstacle has been detected. Multiple transmitter and receiver modules can use same network frequency. Packet collisions are tested as a function of simulated baud rate and overlapping transmissions are scrambled. Multiple data types are supported.

Vehicle Mobility models typically move a vehicle across a terrain at an ideal bearing and speed determined by its control inputs. For wheeled vehicles, the bearing and speed are typically reduced from the ideal by referencing terrain mobility characteristic (modeled through the environment) and height, pitch, and yaw are clamped to the terrain.

- Skid-steered vehicle motion (crawlers/tanks) is computed as a function of commanded right/left side wheel velocities.

- RATLER™ Motion is computed like skid-steered vehicle. Height, pitch, and yaw clamped to terrain using central pivot in body to simulate movement of Sandia RATLER™ vehicle [KLARER-93].
- Ackerman vehicle motion (cars/trucks) is computed as a function of commanded right/left side wheel velocities.
- Hoppers motion is computed by projecting a ballistic travel trajectory based on jump angle and power. Hopper to ground impact is simulated with bouncing, angle of impact and energy dissipation. A mobility-like environment characteristic is used to locally modify energy dissipation rates.
- UAV motion (winged/hovering) is computed by considering commanded speed and turning angle constraints. The current UAV model provides low fidelity simulation support.

Key aspects of the mobile robotics research have been in the development and testing of Behavior and Planning Modules. Umbra's architecture, combined with the strategies described herein for modeling the topological structures of integrated systems, enables researchers to readily port these modules to operating hardware platforms. Because many behaviors are designed hierarchically, they can be readily combined to address new problems. A few behaviors and planning modules are listed below.

- WayPoint Following: Individual vehicle commanded to move along a designated route. Supports localized obstacle detection/avoidance behaviors. Supports air and ground vehicles.
- RF-Coordinated Designated Formation Control: Team of vehicles commanded to move in a designated formation (e.g., line, box, phalanx, etc.). Supports localized obstacle detection/avoidance behaviors.
- RF Swarm Formation: Team of vehicles commanded to move in algorithmic formation (e.g., cluster at given density using potential field theory). Supports localized obstacle detection/avoidance behaviors.
- Chemical Plume Swarm: A variety of cooperative behavior algorithms have been implemented and tested to measure chemical or sensed signal intensity to locate plume or signal source.
- Autonomous Communications Network Deployment: Vehicles move along route. Vehicles volunteer to become transponders whenever they lose communications with base.
- Indoor Wall-Following Behavior: Vehicles search out and follow walls to use as navigation within building with local obstacle detection.
- Vehicle Route Planning: Utilizes constraint-based task planning to determine vehicle route. Plans move vehicle from start point to goal point or high-level goal area (i.e., observe or position vehicle for direct fires). Constraints including terrain slope, visibility from observation point, distance, terrain side slope, communication and maintaining communication are considered during route planning.
- Robot Arm collision free motion planning: Monte Carlo technique combined with smoothing algorithms rapidly compute collision free motions for robot arms. Similarly, Sandia's SANDROS [WATTERBERG-97] planner was utilized in Sandia's F-117 painting system.

## Performance Tests

We have successfully simulated low fidelity systems with over $10^4$ disembodied agent models and high fidelity embodied agent systems with 300 modules at 15 Hz on 1 GHz Pentium III class machines. To understand this performance, it is illustrative to review these two very different test cases. The first is of a disembodied agent simulation that was developed to determine module performance overhead in a large networked agent simulation. The second is of an embodied agent simulation. Here, a swarm of robots are programmed to collectively establish a communications network and search a building [SCHOENWALD-01].

Agents in the disembodied simulation had the following features. The agents were connected into a module network. Each node accepted input values, performed a minimal amount of computation on the input and passed the modified value to its output nodes. Developers monitored agent behavior by placing them on a 2D plane that approximated the network and representing state with color. The number of nodes was increased in successive runs. On a 1 GHz Pentium III computer, approximately $10^4$ nodes were simulated at 15 Hz. These tests demonstrated that the Umbra framework introduces very low overhead for agent scheduling and communication.

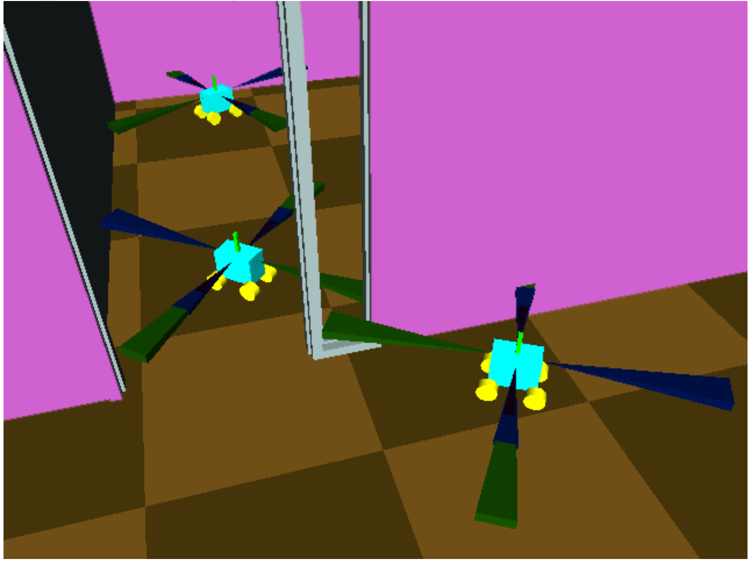

*Figure 5. Embodied Agent Simulation. Image shows three vehicles (with sensor cones rendered) exploring room and hallway.*

The embodied simulation simulates a team or swarm of small mobile robots operating within a building (Figure 5). Each robot has four proximity sensors, ultrasonic sender and receiver pairs, and radio transmitter and receiver pairs. The vehicles use skid steering and the terrain is flat. Each proximity sensor is modeled with three sets of cones representing long, mid, and close range sensing. Ultrasonic and radio transmission models described above are placed on each vehicle. The building environment is modeled with approximately 1300 triangles. Each vehicle is modeled with approximately 400 triangles.

Collision detection consumes the largest amount of computation. At each update, each of the 12 proximity sensor cones and the vehicle body are tested for collision. In addition one vehicle transmits an ultrasonic chirp and radio message on each simulation update. Here, a polygon is created and tested for collision between the transmitting and each potential receiving agent. Thus, each update requires computing 14 collision detection calls for each vehicle. Using a 1GHz Intel P-III-based computer, the simulation runs at approximately 25 Hz with 15 vehicles (210 collision calls), 15-20 Hz with 20 vehicles (280 collision calls), and 10-15 Hz with 25 vehicles (350 collision calls). (Update variation is a result of variable collision detection costs against specific geometric conditions.)

## Conclusions

Sandia has over 20 years of experience in developing and realizing robot system concepts. We have invested significant resources to develop robot system analysis tools (collectively called Umbra) to assist in the evaluation of complex robotic systems and as an environment to collect and deliver robotics expertise. Even the most experienced robot system designer cannot readily predict the system level impact of coupled dynamic robotic devices and subsystems. The Umbra environment allows experts to model and analyze these problems to rapidly address key scientific questions about the performance of new systems in dynamic unstructured environments.

## Bibliography

[ANDERSON-96] Anderson, R. J., *A Modular Approach to Robot Control*, **1996 IEEE International Conference On Robotics and Automation (ICRA)**, Minneapolis, MN, 1996, pp. 699-705.

[CAMPBELL-96] Campbell, L.; Richbourg, R.; Ray, C., *Terrain based mission planning for computer generated forces in a distributed interactive simulation*, **Military, Government and Aerospace Simulation**, New Orleans, LA, 8-11 April 1996.

[CHRISTIANSEN-90] Christiansen, B. K. and L. M. Desjarlais, *A Graphical Interface for Robotic Remediation of Underground Storage Tanks*, **First IEEE Conference on Visualization – Visualization '90**, San Francisco, CA, October 23-26, 1990.

[LIN-96] Lin, M; D. Manocha, J. Cohen and S. Gottschalk *Collision Detection: Algorithms and Applications*, **Proc. of Algorithms for Robotics Motion and Manipulation**, pp. 129-142, 1996 eds. Jean-Paul Laumond and M. Overmars, A.K. Peters (invited submission).

[SMALL-97] Small, D. E.; McDonald, M. J, *Graphical Programming of Telerobotic Tasks* **ANS 7th Topical on Robotics and Remote Systems**, Augusta,GA, April 1997

[KLARER-93] Klarer, P. and Purvis, J., *The Optical Configuration Analysis for the Robotic All Terrain Lunar Exploration Rover*, Sandia National Laboratories, 1993.

[WATTERBERG-97] Watterberg, Peter; Xavier, Patrick; Hwang, Yong, *Path Planning for Everyday Robotics With SANDROS*, **IEEE International Conference On Robotics and Automation**, Albuquerque, NM, April 20-25, 1997.

[XAVIER-97] Xavier, Patrick, *Fast Swept-Volume Distance for Robust Collision Detection*, **IEEE International Conference On Robotics and Automation**, Albuquerque, NM, April 20-25, 1997.

[SCHOENWALD-01] Schoenwald, D. A.; Feddema, J. T.; Oppel, F. J., *Decentralized Control of a Collective of Autonomous Robotic Vehicles*, 2001 American Control Conference, Arlington, VA, June 25-27, 2001.

DISTRIBUTION:

| | | |
|---|---|---|
| 1 | MS1002 | J. Langheim, 15200 |
| 1 | MS1005 | R. Shaum, 15201 (10) |
| 1 | MS1002 | P. Garcia, 15202 |
| 1 | MS1003 | R. Robinett, 15211 |
| 1 | MS1003 | J. Feddema, 15211 |
| 1 | MS1003 | B. Spletzer, 15211 |
| 1 | MS1004 | R. Harrigan, 15221 (5) |
| 1 | MS1004 | P. Bennett, 15221 |
| 1 | MS1004 | S. Gladwell, 15221 |
| 1 | MS1004 | C. Jones, 15221 |
| 1 | MS1004 | M. McDonald, 15221 (10) |
| 1 | MS1004 | E. Gottlieb, 15221 |
| 1 | MS1004 | F. Oppel, 15221 |
| 1 | MS1004 | R. Peters, 15221 |
| 1 | MS1004 | B. Rigdon, 15221 |
| 1 | MS1004 | D. Schoenwald, 15221 |
| 1 | MS1004 | J. Trinkle, 15221 |
| 1 | MS1004 | P. Xavier, 15221 |
| 5 | MS1004 | ISRC Library, 15221 (10) |
| 1 | MS1010 | M. Olson, 15222 |
| 1 | MS1010 | D. Small, 15222 |
| 1 | MS1125 | K. Miller, 15252 |
| 1 | MS1006 | L. Shipers, 15272 |
| 1 | MS1170 | R. Skocypec, 15310 (10) |
| 1 | MS1188 | P. Gray, 15311 |
| 1 | MS1188 | E. Parker, 15311 |
| 1 | MS1188 | M. Trahan, 15311 |
| 1 | MS1188 | J. Wagner, 15311 |
| 1 | MS0830 | C. Forsythe, 15311 |
| 1 | MS1176 | R. Cranwell, 15312 |
| 1 | MS1176 | T. Calton, 15312 (10) |
| 1 | MS0780 | S. Jordan, 05838 |
| 1 | MS1137 | O. Bray, 06534 |
| 1 | MS0318 | M. Boslough, 9212 |
| 1 | MS0188 | C. Meyers, 01030 |
| 1 | MS9018 | Central Technical Files, 8945-1 |
| 2 | MS0899 | Technical Library, 9616 |
| 1 | MS0612 | Review & Approval Desk, 9612 For DOE/OSTI |