Control System Architecture for Unmanned Land Vehicles

Sandor Szabo, Karl N. Murphy, Harry A. Scott, Steven A. Legowik and
Roger V. Bostelman

Systems Integration Group
Robot Systems Division
National Institute of Standards and Technology
Gaithersburg, MD 20899

Overview

A second major effort is currently underway toward the development of vehicles with autonomous capabilities. The Autonomous Land Vehicle and Advanced Ground Vehicle Technology programs highlighted the first efforts in autonomous vehicles. The second push is fueled by the Office of the Secretary of Defense's (OSD) Unmanned Ground Vehicle (UGV) program. The Surrogate Teleoperated Vehicle and Demo I programs have placed current technology into the soldiers' hands. Demo II seeks to advance the state-of-art in multiple autonomous military vehicles.

In the commercial arena, the effects of increased traffic congestion and competition from abroad have pushed the U.S. Department of Transportation to institute the Intelligent Vehicle Highway System (IVHS) program as an instrument to advance the nation's highway systems into the next century. A major thrust of IVHS is also increased vehicle automation.

While such government supported research has produced a plethora of component technology (i.e., software, hardware and systems that satisfy various requirements for autonomous vehicles), it has become increasingly more difficult to evaluate the technology to determine its worth. In addition, managing the technology development and guiding the participating industries, both crucial to an efficient program, are becoming as difficult as controlling our nation's highways.

The role of the National Institute of Standards and Technology (under the Department of Commerce) becomes apparent under such conditions. Its role serves two purposes. First, NIST, serving as a technical resource, is often called upon by other Government agencies to evaluate various technologies that could improve the competitiveness of the United States. Second, in the cases where the required systems and components do not exist, NIST works with industry and academia in developing the necessary technology.

NIST's role in autonomous vehicle development started with the U.S. Army LABCOM techbase program. This program became part of OSD's Robotics Master plan and culminated in Demo I. NIST's responsibility included development of an architecture for UGV's which would support integration and evaluation of various

component technology. The methods used by NIST to develop the architecture, the current status of the program and future directions are presented in this paper.

Approach - the RCS Methodology

Background

NIST started exploring the use of robots and automation in the late 1970's. One of the early results was the discovery of how complex it is to build "intelligent machines". Also, there was a gap between the AI and the controls communities in how best to develop these machines. NIST decided to take the best of both communities and to unify them into a "control system architecture". The architecture also incorporated early work in structured systems: modularity, loose coupling between and tight coupling within systems and incremental development. Including all of these attributes within a single architecture was necessary for NIST to meet its goal of evaluating various component technology.

A product of this work was the formalization of the *hierarchical control system* [Al 92]. One of strongest attributes of this architecture is its acknowledgment of how time affects the performance of machines. Thus, early implementation of the control systems were called real-time control systems (RCS). These RCS's served as the platforms for research in many application domains related to mobile robots [Sz 84, Al 88, Hu 90, Sz 92]. Examples of RCS applications in manipulators can be found in [Mc 88, Fi 92].

The RCS methodology is based on structured engineering principles. The methodology consists of an analysis and design phase (design) and a development and test phase (implementation). The methods will be covered briefly.

Top down design

The first step in developing a system is to establish the goals. The goals can be expressed as a written document stating the functional and operational requirements of the system. A scenario detailing various actions performed by the system is another mechanism for describing the requirements. Often missing from a requirements document are the criteria that will be used to judge whether the system is a success.

The next step is to analyze the scenario to determine what tasks must be performed by which systems. In RCS, this form of analysis is called task decomposition. The goal of task decomposition is to produce an inverted tree structure where high level modules decompose high level functions into increasingly simpler functions for successively lower level modules. This process starts to define the functionality of system modules and the interfaces between them. Different criteria may be used to determine how a task decomposes. Examples include system parallelization, temporal resolution and spatial resolution. A benefit of following common reference model guidelines is that systems will appear similar, thus, improving comprehension, and software is easily shared among systems reducing development costs. Not all systems need follow the reference model. Legacy (i.e., pre-existing) components can be incorporated as long as they are modular and have clearly defined interfaces.

Bottom up implementation

In classical design methodologies, the next step is to develop detailed specifications of the software modules using such tools as state diagrams, structured English and data flow. It has been discovered that while use of these tools may be

helpful, the key to engineering high risk software is rapid prototyping and testing under simulation. The hierarchical model is ideal in this environment. Since the system is modularized with defined interfaces, various modules and external devices can be simulated by developing software that meets portions of the interface specification.

Prototyping allows systems to be developed quickly in an incremental fashion from the bottom up. The prototyped modules help refine the system architecture. Modules that have undergone significant testing become "deliverable code" and are integrated with the system. Essential to this method is a good computing platform that supports multiprocessing and communications. Fortunately, there is a growing number of development environments that provide such a platform.

Architecture for Autonomous Vehicles

One of the first steps performed by NIST to support its evaluation of autonomous vehicle component technology, was to develop a reference model. The reference model describes what functions are to be performed and attempts to organize them based on a consistent set of guidelines. In all of our applications, we attempt to use this consistent set of guidelines in the hope of one day achieving a standard.

Figure 1 shows the reference model architecture for an autonomous land vehicle. Modules in the hierarchy are shown with Sensor Processing (SP), World Modeling (WM) and Task Decomposition (TD). The roles of these submodules are described in more detail in [Al 92a].

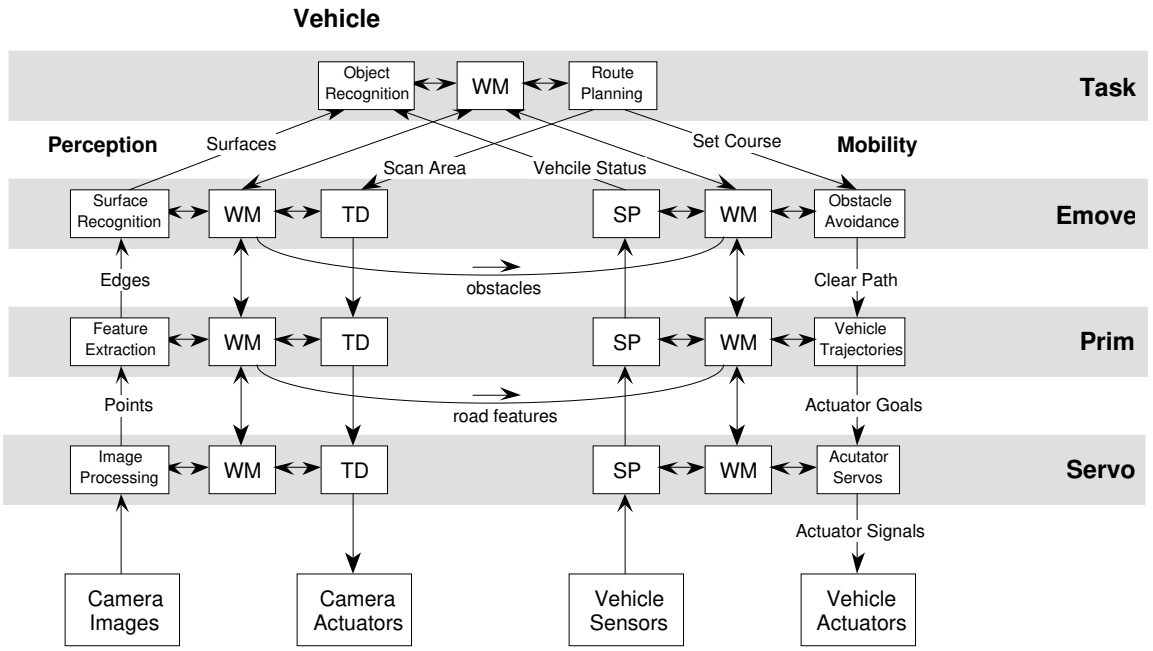

Figure 1. Reference model control architecture for an autonomous land vehicle.

Task Level

Mission Planning - The highest level of command for an individual vehicle, the Task level module, is responsible for executing mission tasks phrased in symbolic terms, such as: *Drive to Rendezvous Position Alpha* or *Prepare for Convoy Operation.* A

vehicle may be equipped with several subsystems, such as navigation, perception and mission modules, which are directed by the Task level to achieve certain phases of the task. In this paper, the navigation and perception aspects of a task are addressed.

Given a task requiring the vehicle to drive to a location, the Task level uses symbolic maps to plan a route. The maps contain the latest estimation of the world but at a limited resolution. At some point, the vehicle must move, using its mobility subsystem, and acquire current estimates of the world using its perception subsystem. Information such as road conditions, proximity to other vehicles and street signs are used by the Task level to fill in maps and plan appropriate courses of action. For navigation purposes, the waypoints of a route are passed to the Elemental level (Emove) Mobility module every few seconds.

Object Recognition - The Task level sensor processing function is object recognition. Model based recognition uses internal 3D models and attempts to match projections of a model with labeled features such as edges and surfaces. These features are derived from the lower levels of the perception subsystem.

Emove Level

Obstacle Avoidance - The Emove Mobility module for driving accepts commands such as *Drive to Coordinates XY*. The module uses information from the world model to calculate motion passageways free of obstacles. Each pathway is sent to the Primitive (Prim) level. The Emove module produces new goals every few hundred milliseconds.

Surface Recognition - This Emove sensor processing module is responsible for surface recognition. Surfaces can be extracted by examining the edge boundaries produced by the Prim level sensor processing. Laser scanners can provide direct depth information to individual surfaces. Surfaces that are identified as obstacles are used by the Mobility module to plan avoidance trajectories.

Prim Level

Vehicle Trajectory - This Prim level module accepts commands such as *Goto XY*. The module assumes that the path is free of obstacles and is only concerned with generating a smooth vehicle trajectory. The output of the module is produced every few tens of milliseconds.

Feature Extraction - The Prim module for a perception hierarchy extracts feature data such as curves, corners, and surface patches from a stream of images. Methods to perform this include road model matching, stereo and motion analysis.

Servo Level

Actuator Servos - Mobility actuators include steering, brake and throttle. Vision actuators consist of camera controls, pan/tilt and stabilized platforms. The modules perform low level servos and generate actuator control signals every few milliseconds.

Image Processing - This module is the lowest level in the perception hierarchy. It performs functions such as filtering, image enhancement, boundary detection and region growing. An example output of the module (up the hierarchy) are the edge points in an image.

Current System Description

The goal for the LABCOM techbase program is to develop a testbed vehicle to explore various military applications such as navigation and surveillance. Related issues

include human interfaces and communications. Several organizations contributed in developing a driving package for the testbed vehicles. The vehicle used was a successor to the jeep, the HMMWV. Besides NIST, they included the Tooele Army Depot, the Harry Diamond Army Lab and the Human Engineering Army Lab. The culmination of the first phase was Demo I conducted in April and May of 1992 at the Churchville site of Aberdeen Proving Grounds, Maryland.

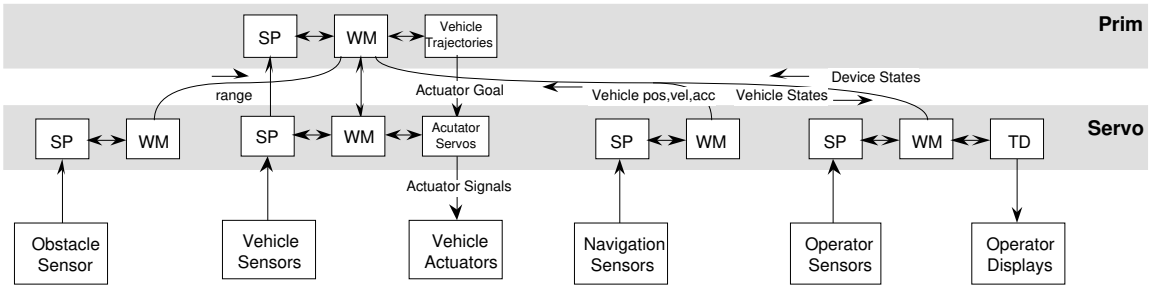

Figure 2. Architecture for Demo One Mobility Controller.

There were several forms of navigation activities performed by various vehicles. One form, called Retro-traverse, was used to teach the vehicle a path during teleoperation and to autonomously return along the path. This form of navigation allowed the vehicle to lay a smoke screen and travel through the smoke without the operator in the loop. Retro-traverse navigation requires only the Prim level functionality as described above. Figure 2 shows the control system architecture.

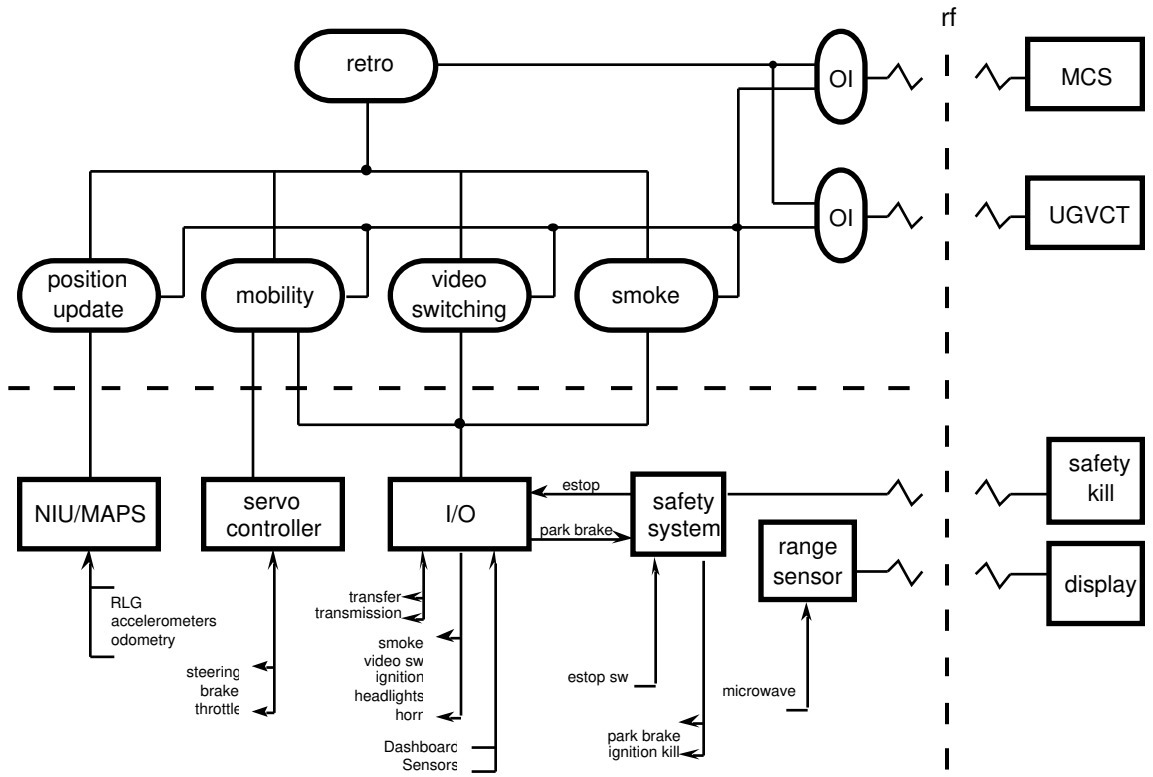

Figure 3. System diagram for Demo One Mobility Controller.

Figure 3 shows the system diagram of the control system. The rounded rectangles are software tasks developed by NIST to perform various functions. The rectangles on the right side are the operator interface units that were used for Demo I. The rectangles at the lower left represent separate pieces of equipment integrated by NIST into the driving package. In general, the system diagram reflects the architecture, but some components such as I/O and communications boards do not map directly.

The highest functional task, **retro**, is responsible for Retro-traverse. In Retro-traverse, the XY goal points do not come from a higher control level, but are retrieved from a list of goal points recorded earlier. The **retro** task is responsible for saving the points during a teach phase and tracking the points during a playback phase. A "pure pursuit" method [Om 90] is used to keep the vehicle on the path. The task performs velocity control which is set by the operator. The operator can also select automatic turnaround paths to reposition the vehicle at the beginning of a recorded path. The **retro** task outputs actuator commands to a **mobility** task which serves as an interface to the Servo level. The Modular Azimuth Position System (MAPS), an inertial navigation unit, is used to sense vehicle position and orientation. MAPS uses ring-laser gyros and accelerometers to determine vehicle motion. An interface board (called the Navigation Interface Unit) and software to integrate vehicle odometry with MAPS data was developed by Alliant Tech and used during Demo I. A **position** task maintains the vehicle position data for use by the rest of the system. Details of the navigation portion of the driving package are in [Mu 92].

Driving through a smoke screen rules out the use of a vision system by a remote operator, but some form of obstacle detection is useful in cases where vehicles or humans wander onto the path. A microwave sensor that would allow the vehicle to travel at slow speeds is being investigated. In addition, range information from this sensor can be used by the operator during close proximity driving.

Below the **retro** task, are the executor tasks responsible for low level control of various devices. A **smoke** task controls smoke generator functions such as on/off, type of smoke (visible, infra-red, microwave) and the intensity. The smoke can be controlled remotely by the operator or automatically at different points under autonomous navigation. Similar control applies to a **video switcher** task for determining which video signals are routed to the operator. The **mobility** task supports commands from the **retro** task and from the operator during teleoperation. It also gathers vehicle status information (similar to dash board status). A brake pressure interface (with associated control loop) is provided to increase the sensitivity of operator control. In addition, the **mobility** task performs certain safety functions such as bringing the vehicle to a halt in case of RF communication link failure.

A commercial servo control board is used to control the steering, brake and throttle actuators. A multipurpose I/O board handles analog and digital input and output. An isolated safety system is also part of the driving package. The system uses a separate radio link to execute an emergency stop, either by command of a safety observer or on loss of the safety radio link. The system applies the parking brake and turns off the engine.

Individual operator interface (**OI**) tasks serve communications to two operator control units. One is a suitcase controller developed by NIST for field testing called MCS (Mobility Control Station). The second operator station is the Unmanned Ground Vehicle Control Testbed (UGVCT) developed by FMC for the Tank Automotive Command. Each system allows the operator to control all mobility functions. High level

commands are issued using a touch screen display. A graphic display presents vehicle status to the operator.

A commercial RF data communication system was integrated for control and development purposes. A low latency data link allows the operator to control the vehicle at high speeds. An eight channel multiplexed link provides flexibility in interfacing to various devices. An ethernet link supports code development and testing. Full bandwidth video was carried on wideband RF links, while compressed video was communicated over the ethernet data link mentioned above.

Future directions

Current efforts involve extending the architecture to support future testing and evaluation of component technology. Initial algorithms for road following, developed for the Department of Transportation's IVHS program, have been tested and will be integrated into the control system this year. The architecture necessary for road following, shown in Figure 4, is derived from the two lowest levels of the generic vehicle architecture (Figure 1). The specific road feature of interest to the mobility control system is the coordinates of the lane of travel. The vision system uses a model of the road's edges to predict and track the curvature of the road. The coordinates of the center of the road are used by a task similar to **retro** to track the road instead of a list of predefined points. After the architecture is extended we hope to evaluate several strategies for road following.

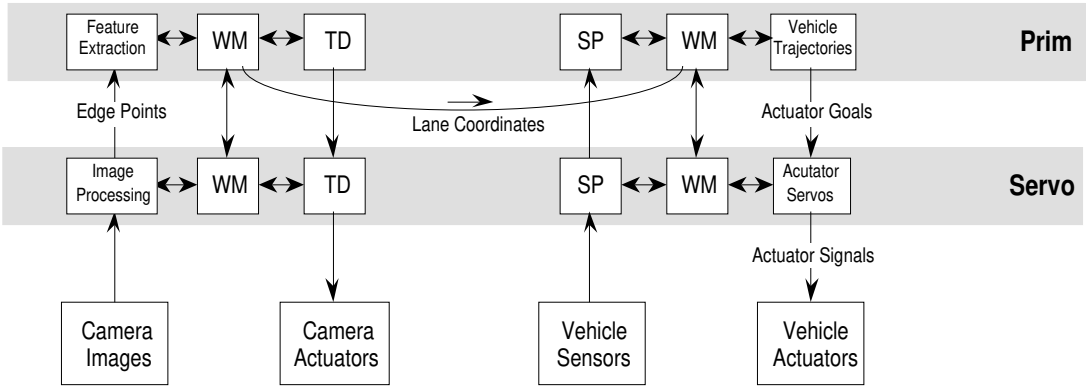

Figure 4. Extended architecture for lane following control system.

The next level of autonomous navigation involves obstacle avoidance. Initial work has been done using optical flow to detect and steer around obstacles. The use of stereo imagery for navigation in unstructured environments will also be evaluated.

Summary

NIST's roles are to evaluate component technology for autonomous vehicles and to work with industry and academia to advance the state-of-the-art. To perform such a task, an architecture has been developed that will allow incremental development of autonomous capabilities in a modular fashion. The low levels of the control system have been implemented to support the OSD Unmanned Ground Vehicle program. That system was demonstrated at the 1992 Demo I. The control system will be systematically extended to incorporate higher levels of autonomous capabilities to support further

evaluations and developments in conjunction with the OSD UGV and DOT IVHS programs.

References

[Al 88] Albus, J.S., "System Description and Design Architecture for Multiple Autonomous Undersea Vehicles", NIST Technical Note 1251, National Institute of Standards and Technology, Gaithersburg, MD, September 1988.

[Al 92] Albus, J.S., "A Theory of Intelligent Systems", Manufacturing & Automation Systems: Techniques & Technologies, Control & Dynamic Systems, Advances in Theory and Applications, Vol 45, 1992.

[Al 92a] Albus, J.S., Juberts, M., and Szabo, S., "RCS: A Reference Model Architecture for Intelligent Vehicle and Highway Systems", to be published in the Proceedings of ISATA 92, Florence, Italy, June 1992.

[Fi 92] Fiala, J., Wavering, A., and Lumia, R., "A Manipulator Control Testbed: Implementation and Applications", AAS Conf. on Guidance and Control, Keystone, CO, February 1992.

[Hu 90] Huang, Hui-Min, Hierarchical Real-time Control Task Decomposition for a Coal Mining Automation Project, NISTIR 90-4271, National Institute of Standards and Technology, Gaithersburg, MD, March 1990.

[Mc 86] McCain, H.G., Kilmer, R.D., Szabo, S., and Abrishamian, A., "A Hierarchically Controlled Autonomous Robot for Heavy Payload Military Field Applications", Proceedings of the International Symposium on Intelligent Autonomous Systems, Amsterdam, The Netherlands, December 8-11, 1986.

[Mu 92] Murphy, K.N., Navigation and Retro-Traverse on a Remotely Operated Vehicle, Proceedings of the IEEE Singapore International Conference on Intelligent Control and Instrumentation, February 1992.

[Om 92] Omead, A., Integrated Mobile Robot Control, MS Thesis, Carnegie Mellon University, 1992.

[Sz 84] Szabo, S., Juberts, M., and Kilmer, R.D., "Automated Guided Vehicles at the National Bureau of Standards Automatic Manufacturing Research Facility", Unmanned Systems, 3 (1), Summer 1984.

[Sz 92] Szabo, S., Scott, H.A., Murphy, K.N., Legowik, S.A., and Bostelman, R.V., "High-Level Mobility Controller for a Remotely Operated Unmanned Land Vehicle", Journal of Intelligent and Robotic Systems, 5 pp 63-77, 1992.