High Level Mobility Controller for a Remotely Operated Unmanned Land Vehicle

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

Systems Integration Group

Robot Systems Division

National Institute of Standards and Technology

Gaithersburg, MD 20899

Abstract

The U.S. Army Laboratory Command, as part of the Department of Defense Robotics Testbed Program, is developing a testbed for cooperative, real-time control of unmanned land vehicles. The program entails the development and integration of many elements which allow the vehicles to perform both autonomous and teleoperated functions. The National Institute of Standards and Technology (NIST) is supporting this program by developing the vehicle control system using the Real-time Control System (RCS) architecture. RCS is a hierarchical, sensory-based control system, initially developed for the control of industrial robots and automated manufacturing systems. NIST is developing the portions of RCS that control all vehicle mobility functions, coordinate the operations of the other subsystems on the vehicle, and communicate between the vehicle and the remote operator control station. This paper reviews the overall control system architecture, the design and implementation of the mobility and communication functions, and results from recent testing.

This work is funded by the U.S. Army Laboratory Command. This paper was prepared by U.S. Governme employees and is not subject to copyright. Equipment listings do not imply a recommendation by NIST.

Introduction

Artificial intelligence, control technology, and vision processing capabilities have not reached the point where robotic land vehicles can function autonomously in off-road tactical situations. Therefore, the vehicle control systems must support a mix of capabilities ranging from master-slave teleoperation to autonomous control. The development of such a control system is underway as part of a program initiated by the U.S. Army Laboratory Command. This program, called the Robotics Testbed for Unmanned Ground Vehicles (RT), is a joint effort between several U.S. Army organizations, national laboratories, and commercial contractors. The goal of the program is to develop and demonstrate technology for use on tactical vehicles in the areas of mobility, sensory processing, target detection and tracking, and communications.

In the RT scenario, humans remotely operate several Robotic Combat Vehicles (RCVs) from an Operator Control Unit (OCU). Each vehicle contains: actuators on the steering, brake, transmission, etc.; an inertial navigation system (INS); a mission package which performs target detection, tracking, and laser designation; and data and video communication links. The OCU contains controls and displays for route planning, driving, operation of the mission package, and control of the communication links.

A typical mission includes a planning phase where the operator plans a route using a digital terrain data base. The operator then remotely drives the vehicle to a desired location as the vehicle records the route using INS data. The operator activates the mission package for automatic target detection, and when a target is detected, the mission package designates the target with a laser. The vehicle then automatically retraces the route, a process termed retro–traverse.

The RT control architecture follows the Real-time Control System (RCS) methodology developed at National Institute of Standards and Technology (NIST) [1-3]. The application of RCS technology to the RT program is presented in Reference [4] and the control system architecture developed for the RT program is presented in Reference [5]. The RCS architecture is a hierarchy of control modules in which each module controls one or more modules at the next lower level. High-level commands are decomposed into simpler commands as they pass down through the levels. Each control module is composed of task decomposition, world modeling, and sensor processing. Task decomposition implements real-time planning, control, and monitoring functions. It decomposes tasks both spatially and temporally. The sensory processing modules detect, filter, and correlate sensory information. The world modeling modules estimate the state of the external world and make predictions and evaluations based on these estimates.

This paper focuses on the implementation of the mobility RCS on the vehicle testbed and presents results of mobility tests performed. The first section provides an overview of the control system architecture for the entire system. Next, the design sections of this paper present the details of the mobility and the communication subsystems. The implementation section describes the current state of the mobility and communication control systems. The final section presents initial results obtained from testing performed at the U.S. Army Aberdeen Proving Grounds, Aberdeen, Maryland.

Control Architecture

The RCS architecture for RT is an inverted tree structure containing several levels. Figure 1 shows the control modules that reside on each RCV and Figure 2 shows the modules that reside in the OCU. The highest control module on each vehicle is the RCV Supervisor. It coordinates three subsystems: mobility, mission package, and communication.

The mobility subsystem drives the vehicle and gathers sensory information for the operator in the OCU. At the Servo level, the Vehicle Platform Controller drives independent actuators that position the steering wheel, accelerator pedal, brake pedal, and other linkages. The Primitive (PRIM) level Mobility Platform Controller generates smooth trajectories in a convenient coordinate frame based on either goals from the operator or from prerecorded paths. The Elemental (EMOVE) level Mobility Subsystem Controller computes paths that are clear of obstacles. Although the current goals for RT do not include automatic generation of obstacle free paths, future developments in this area would occur at this level.

The automatic target acquisition (ATA) control modules in the mission package subsystem automatically detect, locate, and laser designate moving tanks and other targets [6]. The ATA functions are divided among control modules for processing imaging sensors, infrared detectors, and nonimaging sensors, as well as modules for controlling leveling and aiming platforms.

The communications subsystem controls data and video communications between the RCVs and the OCU. The data communication modules provide data paths between modules on the vehicles and the OCU using a single radio link. The video modules utilize compression techniques to reduce the radio link bandwidth requirements while maintaining sufficient feedback to the operators for teleoperation.

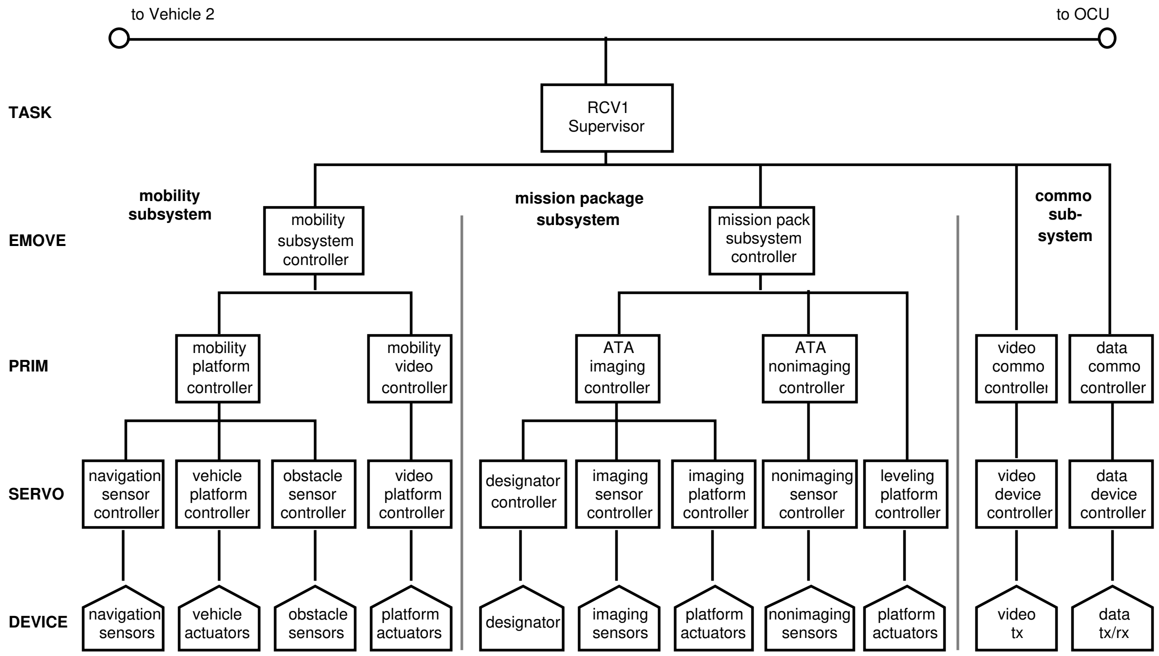

Figure 1. Control system architecture for RT Robotic Combat Vehicles.

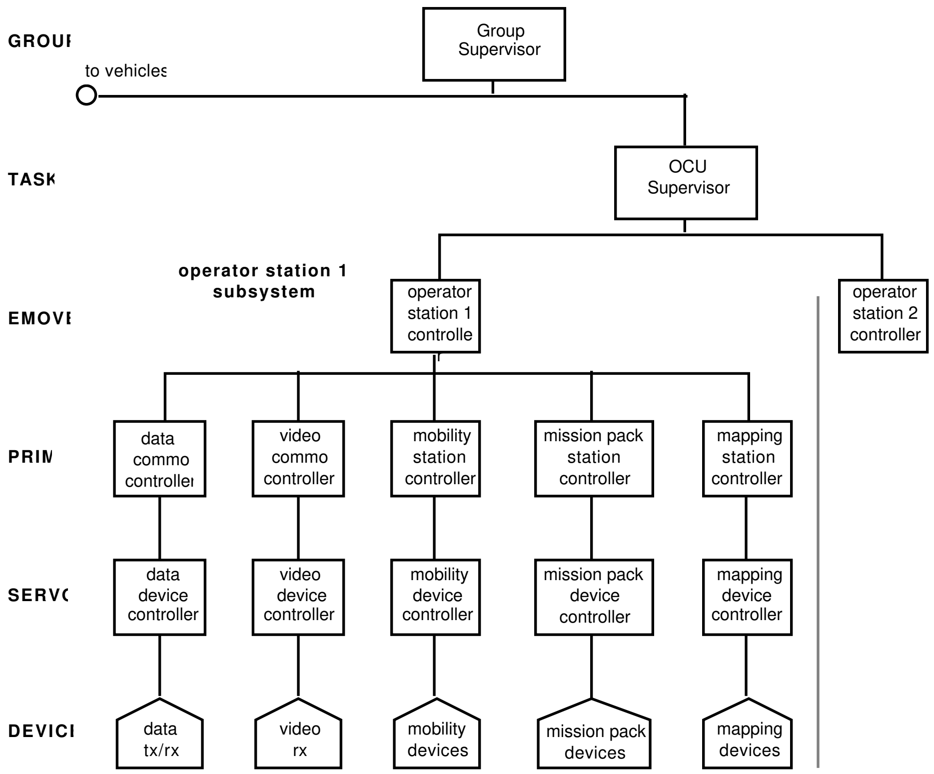

Figure 2. Control system architecture for Operator Control Unit.

The highest-level control module in the RT architecture, the Group Supervisor, resides in the OCU. This module coordinates activities that involve multiple vehicles such as convoy deployment and cooperative target detection. Beneath the Group Supervisor, the OCU Supervisor coordinates activities between two operator station subsystems. Each operator station subsystem contains joysticks, displays, video monitors, etc., that allow the operator to drive the vehicle and to control the mission package. Included in each operator station subsystem are the communication controllers required to maintain video and data communication with the vehicle being controlled.

Initially, the architecture is a tool for analyzing functions of the system and designing the various subsystems. The next step is to perform a detailed design of the modules that make up a subsystem. The guideline for detailed designs specifies further breakdown of control modules into sensor processing (SP), world modeling (WM) and task decomposition (TD) and establishing the flow of information between these sub-modules. The functions performed by the task decomposition sub-module is further divided to enable parallel operations between planning and execution. The Job Assigner (JA) spatially decomposes tasks between planners. Each planner (PL) decomposes tasks temporally in order to achieve goals within a planning horizon. Whilethe planner looks ahead in time, the executor (EX) works in parallel to achieve immediate goals. For further details on these modules see Reference [3]. The architecture and guidelines for development of an RCS were the basis for the detailed design efforts for the mobility and communications subsystems discussed next.

Mobility Design Details

The RT program requires two modes of vehicle mobility: remote control and retro-traverse. In the remote control mode, the operator drives the vehicle from the OCU. In the retro-traverse mode, the vehicle automatically retraces a previously driven path using the position data from the inertial navigation system. The PRIM level modules responsible for these driving modes are the Mobility Platform Controller on the vehicle and the Mobility Station Controller in the OCU. Figure 3 shows the internal decomposition of the mobility modules. The arrows between the OCU and the RCV represent information flowing between the two world models. This world model registration is supported by communication modules discussed in the next section. The breakdown of the mobility modules' responsibilities for the various driving modes is discussed next.

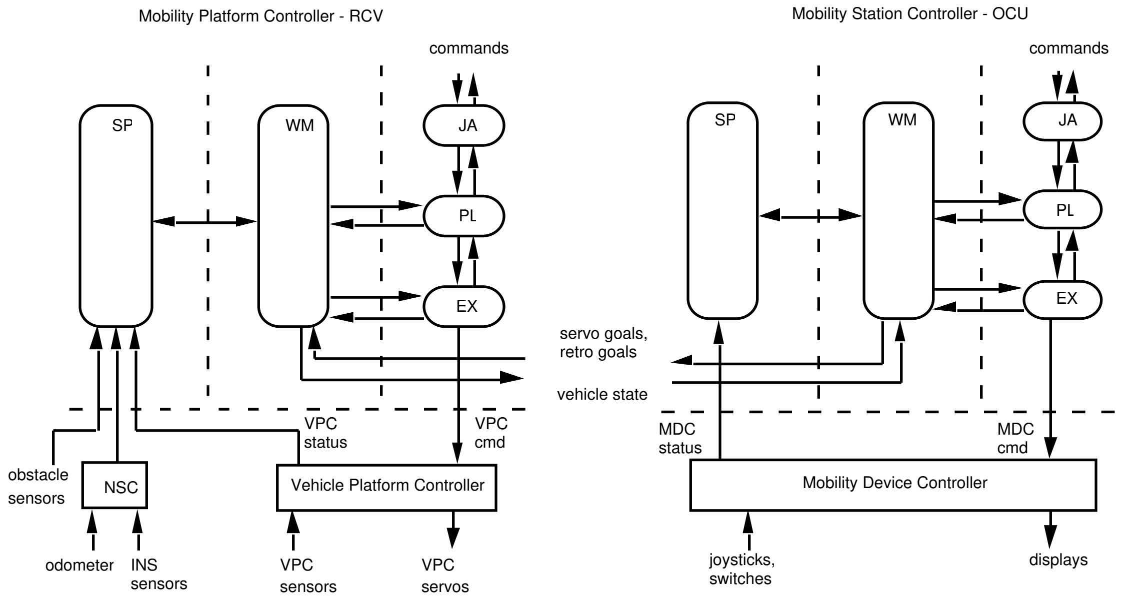

Figure 3. Mobility Prim Level: Command and data paths

Remote Control

The vehicle Mobility Platform Controller (MPC) sends servo commands to the Vehicle Platform Controller (VPC). During remote control the goals come from the operator. In this case, the communication system delivers the current state of the operator input devices (e.g., steering, brake, throttle, etc.) to the world model. The MPC executor retrieves the operator's goals, scales them, and sends them to the Vehicle Platform Controller. Certain reflex actions are also handled by the executor. For example, when information from the operator devices is not available due to a communications error, or when an obstacle is detected, the executor stops the vehicle. Various vehicle platform sensors provide information such as vehicle speed, engine temperature, and engine rpm. The MPC SP sub-module processes this information and places it in the world model where it is accessible from the OCU.

In the OCU, the MSC ensures that operator mobility goals are sent to the vehicle and that vehicle state information is displayed for the operator. The MSC SP sub-module reads the current state of the operator's drive devices and places the data in the world model. The state of the vehicle is retrieved from the world model by the MSC executor and becomes part of a display data command issued to the appropriate display device.

Retro-traverse

The retro-traverse mode of driving gives the RT vehicles some degree of autonomy during a mission. Initially, a path (i.e., a sequence of x-y positions) is recorded as the operator remotely drives the vehicle. Then, at a later time, the vehicle retraces the path without the direct control of the operator. This is analogous to teach programming and playback commonly used by industrial robots.

During the teach phase, the mobility control system functions in the remote control mode. In addition, the Mobility Platform Controller world model records the path based on navigation data acquired by the MPC sensor processor. The data may be produced by different types of navigation systems, but the source is transparent at this level in sensor processing. At the end of the path, the operator halts the path recording and specifies a turn around maneuver that avoids obstacles.

Later in the mission, the operator initiates the autonomous drive phase of retro-traverse. The MPC planner retrieves the turn around maneuver from the world model, selects the appropriate gear and commands the steering either hard left or right. Path following begins once the vehicle's approximate heading is close to the path. The MPC planner retrieves the path and the vehicle's current position and orientation from the world model. Using a PID controller, the planner continuously generates new throttle and brake commands to maintain the desired speed, similar to an automobile cruise control. Steering commands are calculated that drive the vehicle along the path using a "pure pursuit" steering algorithm [7].

The pure pursuit method is a simple yet very stable method for controlling vehicle trajectory. Each cycle the turn radius, and hence the steering angle, is selected to drive the vehicle over a point on the path a specified distance ahead of the vehicle. The relationship between the vehicle and the path is shown in Figure 4. No regard is given to the orientation of the vehicle at this goal point if the vehicle were to maintain the selected steering angle. During the next control cycle a new steering angle is calculated using a new goal point.

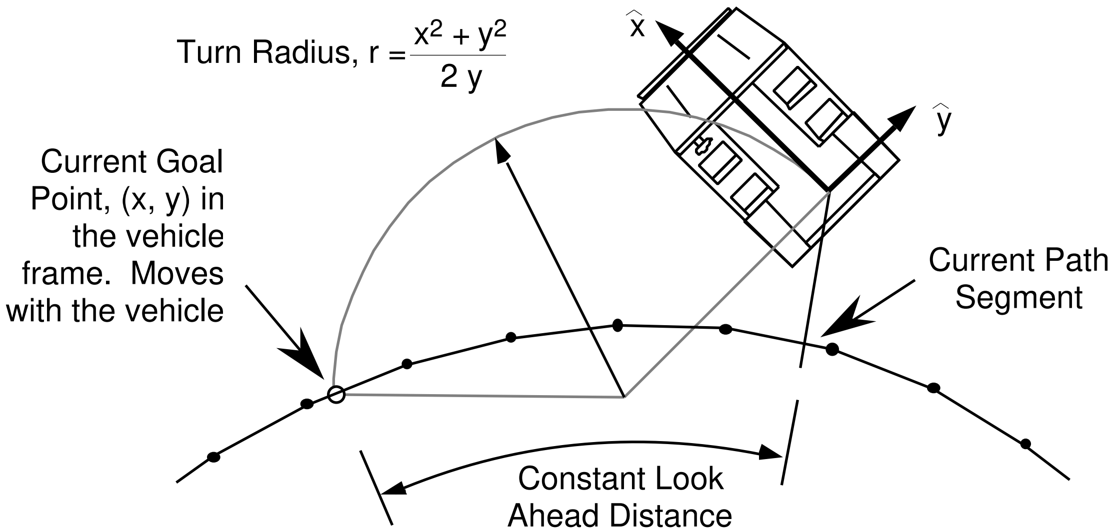

Figure 4. The goal point is the point on the prerecorded path a given distance ahead of the vehicle. The turn radius, and hence the steering angle, is selected to drive the vehicle over the goal point. The goal point, turn radius, and steering angle are updated each control cycle as the vehicle moves.

If the goal point is at (x,y) in the vehicle frame, as shown in Figure 4, then the center of the turn circle is (0, –r) and

$$r = \frac{x^2 + y^2}{2y}$$

The steering angle is inversely proportional to r. When r is negative the vehicle turns left and when it is positive the vehicle turns right.

Communication Design Details

Both video and data communication are required between the OCU and the vehicles. The transmission of video signals for driving requires high-bandwidth communications which is undesirable in tactical operations. Data compression techniques will be used to reduce the quantity of video data transmitted. Work in this area is presented in Reference [8]. For data communication, a single rf (radio frequency) link carries all information between the vehicle and OCU. Four categories of information are multiplexed on the single data channel. The categories are time-critical, emergency, time-noncritical, and development.

Time-critical data is transmitted frequently, often at a fixed rate, and between processes that cannot tolerate large transmission delays. This class includes servo goals used for driving (steer, brake, and throttle) and for manual control of the mission package platform. This class represents the most demanding communication occurring on the rf link.

Emergency messages are time-critical but are transmitted infrequently. These messages communicate emergency stop commands to the vehicle and emergency status from the vehicle.

Time-noncritical data is transmitted relatively infrequently and not with real-time urgency. Examples include file transfers, commands that set various vehicle modes, such as gear selection, and high level command and status exchanges.

The development data supports code development, downloading, process monitoring, and debugging. This information may be exchanged at any time, including during vehicle operation.

Figure 5 illustrates how the communication functions are integrated into the RT architecture. The Data Device Controllers serve all RCV-OCU data communications. Any relevant portions of the vehicle and OCU world models may be transmitted over the rf link. Several vehicle and OCU control modules will require this type of communication. Shown here is an example of information flow during the remote control driving mode. Vehicle state information flows from the vehicle Mobility Platform Controller to the RCV Data DeviceController, over the rf link to the OCU Data Device Controller, and finally to the OCU Mobility Station Controller. A similar path is used, in the opposite direction, to convey vehicle goal states from the OCU to the vehicle.

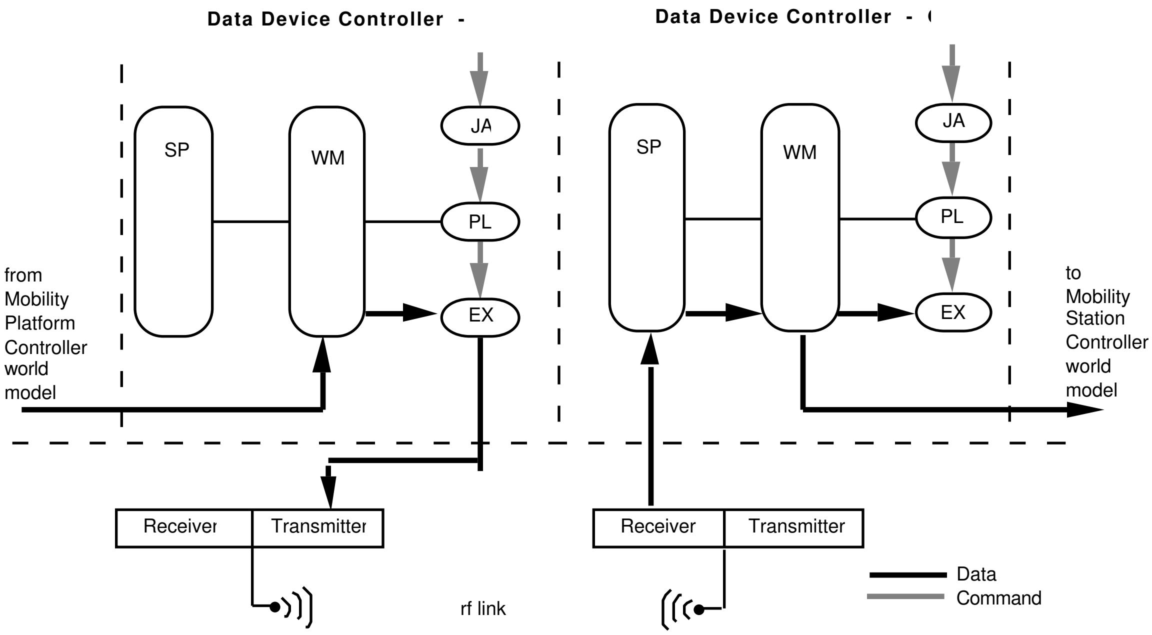

Figure 5. Data Communications between the RCV and OCU.

When new vehicle state information is detected in the Mobility Platform Controller world model, it is passed to the vehicle Data Device Controller world model. An inter-process communications (IPC) mechanism, which will be discussed later, is used to facilitate communication between processes. The executor obtains this data, prepares the appropriate message structure for it and executes the control functions required to transmit the rf message. It also monitors feedback from the radio that indicates its performance.

In the current design, the job assigner and planner establish the remote operation mode, which results in the information flow described above. In other hardware configurations the job assigner would distribute information over several separate radio equipment interfaces and the planner would select schemes to optimize communications.

The information transmitted from the vehicle radio is detected by the OCU receiver and retrieved by the OCU Data Device Controller sensor processor. The executor evaluates the reception and, as a result of certain error conditions, may report failures to the Prim level Data Communications Controller. The vehicle state information is then sent to the Mobility Station Controller world model via the IPC mechanism.

Inter-Process Communications

The inter-process communication mechanism is a set of utilities supporting mailbox-type communication between processes in an RCS application. It provides a consistent user view for communications, while hiding the various underlying mechanisms employed. Process-to-process, board-to-board, and subsystem-to-subsystem communications benefit from a standard communication interface. The underlying communication techniques include socket-based and common memory-based methods.

The IPC user can connect to a mailbox using a symbolic name, send variable length messages to a mailbox, and synchronize control processes through appropriate use of blocking reads. A read operation may be directed to block (with a timeout period) for synchronization, or may be non-blocking. Double buffering and internal semaphoring free the control processes from concern about overwriting a previous message. These capabilities provide a communication mechanism particularly well suited to an RCS environment and their ease of use encourages modularization of the system code.

Implementation

Several organizations are involved in the development of the RT control system. An important consideration in such an endeavor is integration of the individual subsystems that function in both the vehicle and the OCU. The system architecture, if properly designed and implemented, assists in the integration and serves as a useful tool for managing the testbed. Figure 6 shows the portions of the control system (highlighted) that were implemented for the U. S. Army Materiel Command (AMC) Technology Symposium at Aberdeen Proving Grounds on October 1-4, 1990. Demonstrated at the Symposium were teleoperation of the vehicle from a remotely situated driver station, recording vehicle trajectories, displaying vehicle trajectories on a map display, and executing retro-traverse.

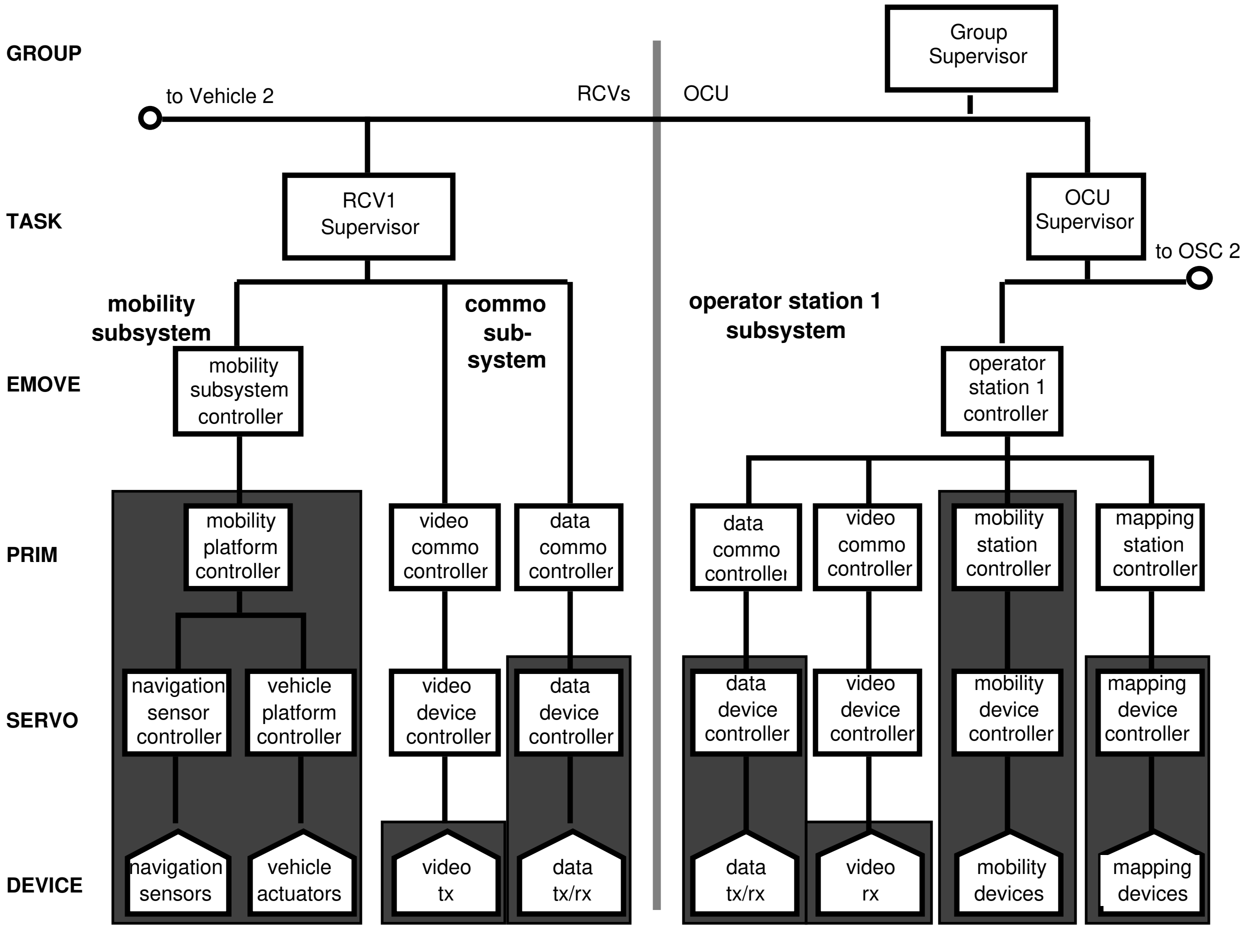

Figure 6. Portions of control system implemented for teleoperated driving and retro-traverse.

Several modules shown in Figure 6 exist as commercial products, thus eliminating the need for in–house development. The Vehicle Platform Controller module and vehicle actuators were developed by Kaman Sciences. Two types of navigation sensors are used in the testbed. The U.S. Army Modular Azimuth and Position System (MAPS) uses accelerometers, an odometer, and optical ring laser gyros to sense vehicle inertial positions and orientations. Asecond navigation system, the Radio Frequency Navigation Grid (RFNG) provides vehicle position data during testing and development. The RFNG employs radio beacons placed at the corners of the test area which emit timing pulses received by the vehicle. Differences in phase information from each transmitter are used to compute vehicle position. Velocity and heading are derived bydifferencing position. For information on both MAPS and RFNG see Reference [9].

The data radio system is composed of components developed by Telesystems SLW Inc. The radios employ a direct sequence, spread spectrum modulation technology. They operate at 915 MHz, with a spread of 26 MHz and a maximum power output of 1 watt. The system is configured as nodes in a network with eight nodes on the vehicle connected to eight nodes in the OCU via the single RF link. Each node supports error-free communication over anasynchronous RS-232 serial interface at individually configurable rates of up to 19.2 kbaud. A single component, known as a network unit, is used at the vehicle and at the OCU to provide the eight nodes. One of the eight vehicle-OCU connections is used to connect the an ethernet local area network (LAN) on the vehicle to a LAN on the OCU by way of the Serial Line Interface Protocol (SLIP). This essentially provides a bridge function between the two LANs. The network unit satisfied the requirements of the Data Device Controllers (see Figure 6) by providing such functions as multiplexing of different information categories over the single RF link, error detection and retransmission, and control of the data radio transceiver.

Analog video information is provided from a vehicle camera and sent through a Coherent video transmitter and is received by a common VHF receiver in the OCU for display on a monitor.

In lieu of a fixed-base driver station, NIST developed a portable unit called the Mobility Control Station (MCS). The MCS provides the functionality of the Mobility Devices and Mobility Device Controller modules specified in the OCU portion of the system architecture (see Figure 6). Shown in Figure 7, the MCS consists of a steering wheel, a joystick for brakes and throttle (and for use as an alternate steering device), several switches for controlling ignition, transmission, transfercase, etc., and a display for vehicle status. The display also provides a touch screen which offers flexibility in programming various switches, and in alternating between remote driving, retro–traverse and debug modes. A small, stand-alone data acquisition computer inside the MCS handles data acquisition of the mobility devices and controls the mobility display.

The vehicle used for the Robotics Testbed is the U. S. Army's High Mobility Multipurpose Wheeled Vehicle (HMMWV). The HMMWV is a one and one quarter ton, four wheel drive truck. The vehicle is shown in Figure 8. A sealed, passively cooled enclosure houses a shock mounted 19-inch electronics rack containing the control system computers and the radios.

Figure 7. The NIST developed portable Mobility Control Station.

Figure 8. The Robotics Testbed vehicle, a standard U.S. Army HMMWV.

The high-level computing environment selected to implement the control system utilizes a real-time, multi-tasking, UNIX-like operating system, VxWorks (Wind Rivers Systems), for the target hardware, and a separate UNIX system (Sun Microsystems) for software development. The target hardware is based on Motorola CPUs interconnected by VME backplanes. Individual backplanes are typically connected using an ethernet local area network (LAN). Code is developed on Sun workstations and downloaded to a battery backed RAM disk over the LAN, or over the radio using the Serial Line Interface Protocol. The RAM disk allows the on-board computers to boot without connecting to the Sun's or requiring that application code be stored in PROM.

This environment is highly suited for an RCS application. For example, referring to the detailed design presented earlier, the Mobility Platform Controller is implemented as several modular tasks running on a single board computer as shown in Figure 9. The tasks interfacing to the Navigation Sensor Controller modules continuously update the world model estimation of the vehicle position. The task containing the job assigner, planner, and world modeling is responsible for recording and planning vehicle trajectories used during retro-traverse. The final task interfacing to the Vehicle Platform Controller is responsible for generating the servo level commands for the Kaman Sciences vehicle controller. These goals are derived from the operator during teleoperation or from the planner during retro-traverse. Note the distribution of sensor processing, world modeling and task decomposition between several tasks as opposed to a single task for each. This approach is taken to maximize software performance and to hide the details of interfaces to external equipment.

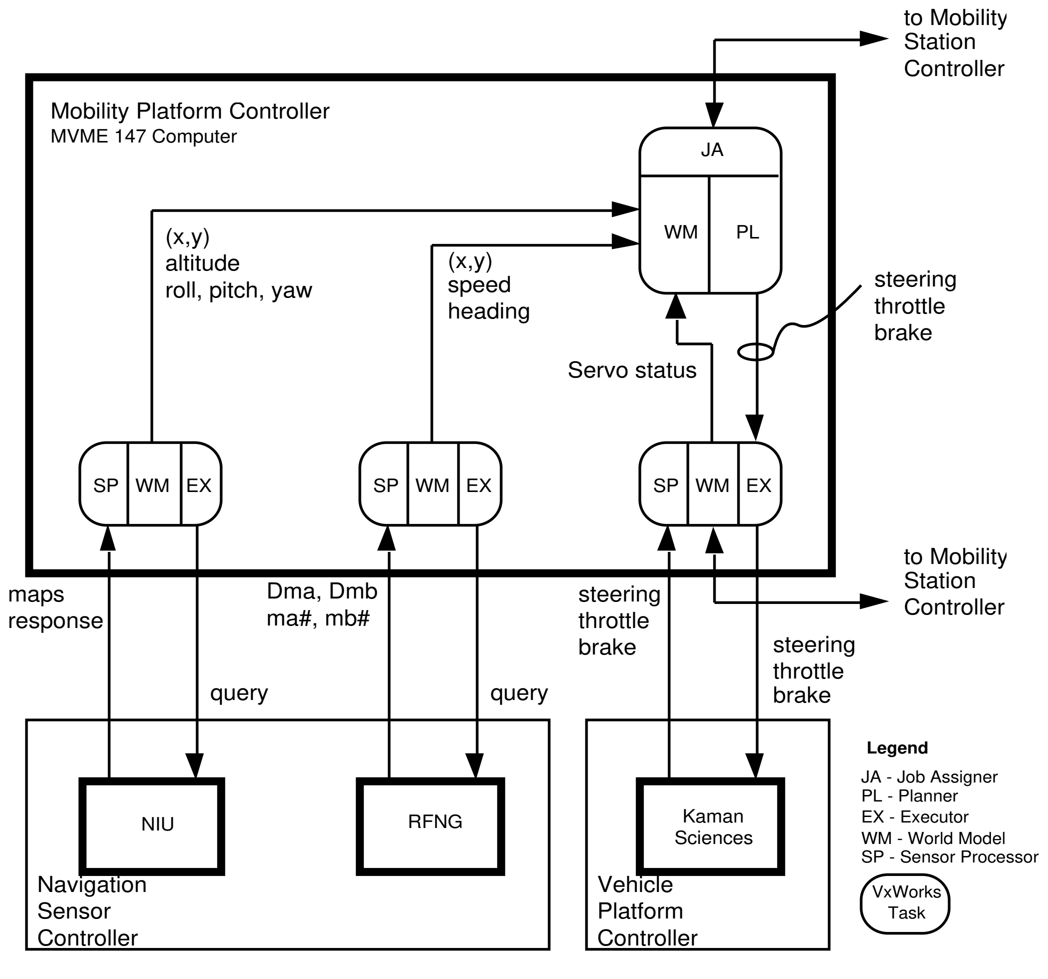

Figure 9. Mapping the mobility platform controller into a multi-tasking operating system.

Results

Both remote control and retro-traverse functions were demonstrated during the Technology Symposium at Aberdeen Proving Grounds and additional testing is ongoing. The performance characterization of remote control was limited to measuring delays in the control system and determining limitations of the communication system. The Kaman Sciences controller accepted actuator commands at a 60 msec update. The MCS was capable of acquiring the state of the operator devices and displaying the status of the vehicle at a 60 msec update. The radio link experienced delays of 30 msec in each direction when operated multiplexed with the other serial ports. The radio link automatically attempts retransmission upon errors which adds significant delays. Initial testing indicates that the transmission characteristics of this radio system provided range up to 1.5 km, and at ranges of a few kilometers, exhibited little performance degradation when obstacles such as clumps of trees or small buildings were in the direct line of sight. The effective update rate for vehicle teleoperation, including optimization whereby communications and display updates are done in parallel, totaled 160 msec. Subjective opinions by several drivers indicated that the control system was sluggish in avoiding obstacles at speeds in excess of 32 kph (20 mph). Several approaches to minimizing the delays are presently under investigation. More controlled testing will be conducted jointly with the Human Engineering Laboratory in the future.

The effectiveness of retro-traverse depends on the repeatability of the navigation system and the performance of the control algorithms. The MAPS, and an associated Navigation Interface Unit (NIU), provides 3-D position and orientation at 25 Hz with drift of 0.1% of distance traveled. The RFNG system provides position updates at 20 Hz with 0.1 meter repeatability (no drift).

The pure pursuit method was used to steer the vehicle during retro-traverse as described in the Mobility Design Details section. With this method, the vehicle followed the path accurately and was stable at speeds up to 64 kph (40 mph) during straight stretches of road with good surfaces. The goal point's look-ahead distance was varied to determine the optimum value. Large values caused the vehicle to cut corners, while small values caused the controller to become unstable at high speeds. A look-ahead distance of 5 meters is used for speeds below 24 kph and 10 meters for speeds up to 64 kph.

Results for a test run are shown in Figure 10. The look-ahead distance is 5 meters and the speed ranges from 8 kph in the sharp turns to 24 kph on the straighter portions. Note the vehicle cuts the sharp corners slightly and is 0.5 meters off course during the sharp turns. The average distance that the vehicle is off the path is 0.1 meter RMS.

A second steering algorithm was tested. In this case, the steering angle was set proportional to the perpendicular distance the vehicle was from the path, the speed that it was approaching the path, the heading error, and heading rate. This PD steering controller produced large errors in path following and proved unstable at speeds above 16 kph (10 mph). For a path similar to that in Figure 10, the average distance the vehicle was off the path was 0.8 meter RMS.

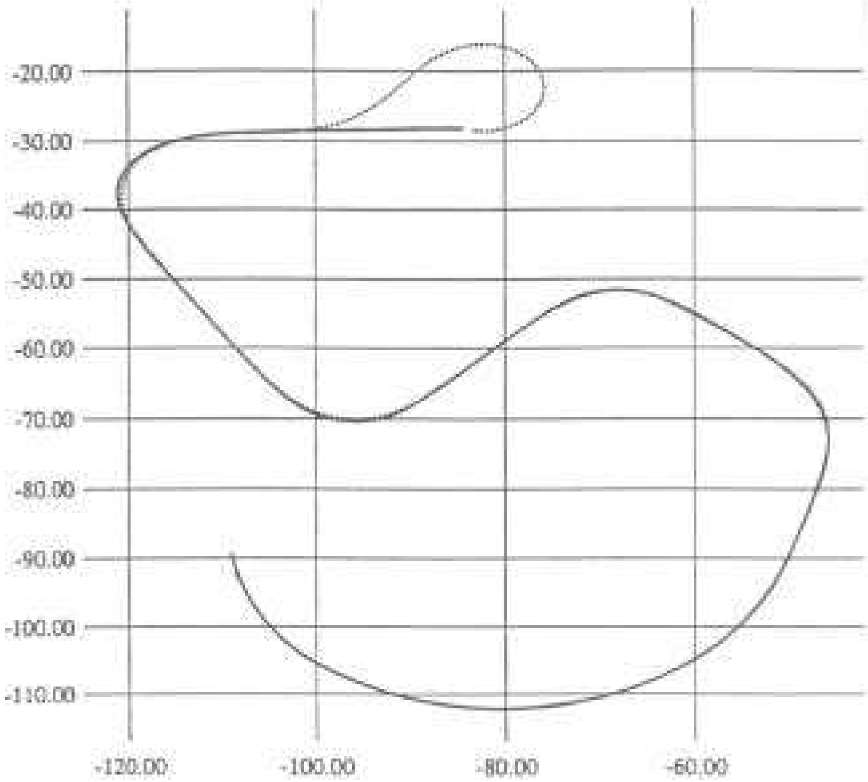

Figure 10. The "pure pursuit" steering algorithm with look ahead distance of 6 meters and speeds ranging from 8 to 24 kph. Dimensions are in meters.

Summary

The control system for the Robotics Testbed requires the development and integration of many elements which allow two vehicles to perform autonomous and teleoperated functions under the control of an operator from a remote site. NIST is supporting this program by developing the vehicle control system based on their Real-time Control System. The RCS is a hierarchical, sensory-based control system, initially developed for the control of industrial robots and automated manufacturing systems. In this application, the RCS controls all vehicle mobility functions, coordinates the operations of other subsystems on the vehicle, and communicates between the vehicle and the remote operator control station.

To date, the architecture has been defined in terms of the major functional modules. A high level computing environment supports development of the modules. The mobility and communications modules required to support remote control and retro-traverse driving have been implemented. Initial results indicate that remote control functions adequately at low speeds but future work is needed to improve the overall system throughput. Retro-traverse testing has shown promise in the ability to accurately control the vehicle based on inertial navigation sensors. Future work will concentrate on improving the accuracy and update rates of the control and navigation systems.

References

[1] Albus, J. S., Barbera, A. J., and Nagel, R. N., Theory and Practice of Hierarchical Control, *Proceedings of the Twenty-third IEEE Computer Society International Conference*, Washington, DC, September 15-17, 1981.

[2] Barbera, A. J., Fitzgerald, M. L., Albus, J. S., and Haynes, L. S., RCS: The NBS Real-Time Control System, *Proceedings of the Robots 8 Conference and Exposition*, Volume 2 - Future Considerations, Detroit, MI, June 4-7, 1984.

[3] Albus, J. S., McCain, H. G., and Lumia, R., *NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM)*, NIST Technical Note 1235, 1989 Edition, National Institute of Standards and Technology, Gaithersburg, MD, April 1989 (supersedes NBS Technical Note 1235, July 1987).

[4] Szabo, S., Scott, H. A., and Kilmer, R. D., Control System Architecture for the TEAM Program, *Proceedings of the Second International Symposium on Robotics and Manufacturing Research, Education and Applications*, Albuquerque, NM, November 16-18, 1988.

[5] Szabo, S., Scott, H. A., Murphy, K. N., and Legowik, S. A., Control System Architecture for a Remotely Operated Unmanned Land Vehicle, *Proceedings of the 5th IEEE International Symposium on Intelligent Control*, Philadelphia, PA, September 5-7, 1990.

[6] David, P., Balakirsky, S., and Hillis, D., A Real Time, Automatic Target Acquisition System, *Proceedings of AUVS-90*, Dayton, OH, July 30 - August 1, 1990.

[7] Omead, A., *Integrated Mobile Robot Control*, M.S Thesis, Carnegie Mellon University, May, 1990.

[8] Herman, M., Chaconas, K., Nashman, M., and Hong, T., Low Data Rate Remote Vehicle Driving, *Proceedings of the Third IEEE International Symposium on Intelligent Control*, Arlington, VA, August 24-26, 1988.

[9] Kraetz, W.F., *Application of Robotic Convoy Technology to Combat Training Centers - Phase I*, Report Number 7-90-3, Honeywell Defense Systems Group, Edina, MN, May 11, 1990.