A SUBMARINE MANEUVERING SYSTEM DEMONSTRATION BASED ON THE NIST REAL-TIME CONTROL SYSTEM REFERENCE MODEL

Hui-Min Huang, Ron Hira, and Richard Quintero
Robot Systems Division
National Institute of Standards and Technology
Gaithersburg, MD 20899
email: huang@cme.nist.gov

ABSTRACT

The Robot Systems Division (RSD) at the National Institute of Standards and Technology (NIST) has been developing a generic reference model architecture, known as the Real-time Control System (RCS), for the last two decades. This paper demonstrates the application of RCS to the automation of submarine operations, which requires an enormous amount of intelligence be built into its control system. A summary of the reference model is given, followed by a description of the implementation process. The long term goal is to establish a generic development methodology for intelligent control systems.

1. INTRODUCTION

1.1 Objectives

Submarines often conduct missions in hostile and uncertain environments. A large volume of information, including sensory data, systems status, and a priori mission knowledge, must be fused in order to organize and communicate to support decision making in real-time. In today's submarines most of the work is manually performed at watch stations by crew members. As our subsystem technology has advanced, crew size and in turn submarine size have increased in order to handle the information processing load. Therefore, the challenge is how to process and communicate information more intelligently and efficiently without increasing crew size. The ultimate goal, crew size reduction, cannot be met without system operations automation.

Under the Advanced Research Projects Agency (ARPA) Maritime Systems Technology Office (MSTO) sponsorship, NIST RSD researchers have implemented a series of increasingly more complex submarine automation demonstrations utilizing sophisticated computer equipment and software. In this paper, we describe how RCS can be applied to the automation of a submarine maneuvering system. This work is also part of a long range RSD plan to develop a software engineering methodology and a software development environment which will enable systems engineers to efficiently apply the RCS technology to a wide variety of complex intelligent control systems applications.

1.2 The Reference Model Architecture and the Methodology

The Real-time Control System (RCS) reference model architecture has been a focus of research and development for the Robot Systems Division (RSD) at the National Institute of Standards and Technology (NIST). Numerous papers were published to describe our results, including [Al'92, Sz'92, Qu'92, Hu'91, Jo'91, Al'89]. Essentially, RCS is a hierarchical structure containing multiple levels of abstraction for describing a system (figure 1). Each level may contain multiple controllers (except for the highest level). Each controller contains sensory processing, world modeling, and behavior generation functions (figure 2). These functions allow the controller to exhibit intelligent behaviors and to coordinate with other controllers to achieve system goals.

A focus of the current research efforts within the NIST RSD is to summarize its two decades of experiences.
and derive a generic RCS methodology. [Qu 92] highlights our current achievement in this effort. This paper serves as an illustration to this methodology description. The following sections describe our implementation in detail.

2. SUBMARINE MANEUVERING PROBLEMS

The first and foremost task for building intelligent control systems is learning detail about the problem domain, which, in this case, is a U.S. Navy 637 class nuclear powered submarine. This section provides details of the demonstration submarine and introduces the mission scenario.

2.1 Maneuvering Mechanisms

A submarine is an extremely complex system; therefore, this demonstration is limited to selected maneuvering functions, namely depth, buoyancy, orientation, and speed. The selected mechanisms performing these functions (see figure 3) are described below. This is a complex and cross coupled multi-input multi-output (MIMO) control system. A particular mechanism may be used to control both the ship's depth and orientation.

The sail planes, used for depth control, are located on the conning tower. The stem planes, used for depth and pitch control, are located at the rear of the submarine. The rudder is used for steering the submarine left or right.

The turbine, which drives the propeller, is bi-directional. However, the reverse motion of the turbine is used only for emergency deceleration during forward motion of the submarine. The submarine never maneuvers astern.

The main ballast tanks are used for establishing the gross buoyancy for a submarine, particularly for submerging and surfacing. The variable ballast tanks are used for small adjustments in buoyancy and orientation.

2.2 Scenario

An initial step in the RCS design approach consists of developing scenarios to flesh out details of operation. Former submarine commanders provided detailed information on submarine operations.

The scenario for the latest work is to navigate under ice through the Bering Strait in covert mode and to control for a sudden salinity change. Salinity gradients may occur from fresh water runoff, where rivers of fresh water cause the water density to drop suddenly. A drop in the density of the sea water will cause the ship to have negative buoyancy and the ship will begin to sink. Temperature fluctuations, common in the open ocean, can cause similar depth control problems.

Ice avoidance is also an important activity in this demonstration. The submarine is given a goal point to reach. Ice detection sonar on a 637 class submarine consists of fourteen forward looking beams, one downward looking, and one upward looking. If the current sonar returns show ice keels blocking the path of the current heading, then a new course is computed. Once the ice keels are cleared, a new course is computed to direct the submarine toward the goal point. This is aided automatically by a Cerebellar Model Articulation Controller (CMAC) neural network [Al 75], which stores a map of the ice encountered.

2.3 Depth Control and Signature Management

A submarine can control its depth in several different ways. All operations cause noise to be generated; however, a primary goal is to keep noise to a minimum.

Figure 2: An RCS Controller

Figure 3: Submarine Maneuvering Mechanisms
These elements form a knowledge structure for an intelligent system.

3.2 Depth Control Behavior

Upon receiving a goal, Ship Maneuver (labeled as Maneuver in figure 1) typically sends a COME_TO DEPTH command to the Depth controller (DP, see figure 4). The depth control behavior (figure 5) can be described as follows:

- Plan Activation: The depth control plan is activated when the Depth controller receives the command, shown at (*O) in figure 5.

- Normal Behavior: The Depth controller would normally be in (Sl). It selects the ASCEND/DESCEND commands for the Dive/Rise controller to achieve the desired depth. The MAINTAIN DEPTH command activates after the ship comes within the depth tolerance. The Depth controller continues executing the same COME_TO DEPTH command and the Dive/Rise controller remains in the state of maintaining the depth unless an error occurs.

- Error Handling: Errors reported to Depth from its subordinate, Dive/Rise, will be accounted for immediately regardless of the controller's current state. In this plan, these errors are described at (Evt1) through (Evt7). Each of them is preceded by a "don't care" state, (11) through (7). The occurrence of any of these errors causes the control to switch to the corresponding "don't care" state no matter what the controller's previous state was. The error compensation actions, represented by the jobs and commands listed in the corresponding boxes, are taken. All of these actions lead the controller to the error correction state, (S2).

Dive/Rise will report any errors beyond its authority of correction. The first such error is called ERROR_1, as described at (*2) and (Evl2). The UP_BUBBLE command would be selected for the Dive/Rise controller for the covertness reason. The Depth controller is now in the state (S2). If either ERROR_1 persists for a predefined time or the speed or depth errors exceed a second set of thresholds, indicating a more severe condition, ERROR_2 flag, as shown at (*3), will be reported, and the ASCEND command will be activated.

If ERROR_3 at (*4) is received, Depth reports DP_ERR_1 to Ship Maneuver. Ship Maneuver, within its authority, would relax the stealth/safety constraints...
4. SYSTEM IMPLEMENTATION

4.1 Overall Software Architecture

The RCS methodology has a generic software architecture to facilitate development of RCS applications. Research results show that such an architecture may include the following hierarchies: RCS controller, human control interface, simulation, human simulation interface, and animation hierarchies, shown in figure 6. A generic communication mechanism serves these hierarchies, partitioned knowledge bases and shared memory. In this implementation, we utilized a 386 compatible PC2 for control and simulation and a Silicon Graphics Incorporated workstation for animation.

4.2 Software Structure for the RCS Hierarchy

The software has two primary components, the main program or Real-Time Executive (RTE) and generic software templates for hierarchy control modules.

The main program, the RTE, organizes the execution of the control modules, simulation modules, debug displays, and animation communication.

The RTE sets the state clock, the sample rate of the computer controlled system. Every execution cycle is initiated at the rate of this heartbeat. It should be emphasized that every control module is executed once each cycle. This approach does not preclude the use of time intensive algorithms. These algorithms can run on their designated processors; however, they must be designed to execute as finite state machines producing an incremental output on each cycle. Interrupt servicing is normally turned off in an RCS implementation during RTE execution. Large complex systems are extremely difficult to manage and become non-deterministic when processing interrupts.

---

References to product or company names are for identification only and do not imply Government endorsement.

---

Figure 5: A Depth Control Plan

and grant Depth permission to use larger control surface operation ranges (see (*5)). If the Dive/Rise ERROR_3 error persists, the Depth controller reports DP_ERRR_2 and DP_ERRR_3 up, shown as (*5) through (*7). Ship Maneuver will relax more stealth/safety constraints and will issue requests to the Propulsion controller to increase speed (see [Hu 93]). The system may re-enter (S1) from (S2) whenever the error is corrected and the normal depth control servo loop resumes.

Note that the CLOSE_TO_BOTTOM event, (Evt1) in the figure, takes even higher priority such that whenever it is detected, the error must be reported immediately. The Ship Maneuver controller will issue an emergency command to blow the main ballast tanks to surface the ship.

* Completion: Depth control would normally stay in (S1) to maintain the given desired depth and does not have a completion state.

All of the commands described in this plan, including ASCEND and UP_BUBBLE (see the tasks decomposed by DP in figure 4), are sent down the hierarchy to Dive/Rise and decomposed there as the Dive/Rise behavior. See [Hu 93] for detail.
4.3 Generic Controller Template

A generic controller template is used in coding control modules. The template is one of the keys to make RCS a robust, extensible, verifiable, and efficient software design methodology for large-scale automation projects. Once the basic submarine system was coded, extensive revisions were made to enhance functionality without scrapping existing code. There are several different functions contained within each controller.

The Sensory Processing and World Modeling (SP/WM) functions perform sensory data filtering and fusion. Processed data is used to update Global Memory and to determine the next appropriate action.

The Behavior Generation (BG) function selects the state table (plan) to execute. The state of the system calculated by the SP/WM functions determines which behavior is executed.

Debug and Performance Measure in the modules consist of tracking commands, status, and execution time.

4.4 Shared Memory Model for Communication within a CPU

An RCS controller is intended to be an independent closed-loop controller. Different types of communication are required between a controller and the rest of the system, as shown in figure 2. In this RCS application, the basic principle of improving data integrity during communication is realized through a triple copy mechanism. Three copies of all data are kept (writer local copy, GM copy, and reader local copy). User modules always have a local copy of the requisite data for control decisions. This mechanism ensures asynchronous execution (non-blocking). Real-time system control executes based on the most recent input data instead of having to pause for the receiving of incoming data. This communications model may be extended to integrate an RCS application in a heterogeneous environment. For example, an RCS hierarchy might be integrated with an expert system.

The same triple buffering concept also applies to the case of communication with other CPUs, although this is not the only method to perform such communication.

4.5 Multiple Mode Control -- The Automatic Mode and Interactive Mode Structures

The RCS architecture specifies that human operators should have the capability for interactive control of any module at any level. In addition, the system can alter environmental variables (see [Hu 93]).

4.6 Simulator Structure

The simulator software structure is also built in the form of an RCS hierarchy. Actuator controllers send commands to their respective simulators. The simulators compute the actuator movements accordingly and send the computed values back to the original controllers via the simulated sensors. At higher levels, the simulated ship state feeds back to higher level RCS controllers via simulated sensors.

4.7 Animation

Animation is a very powerful design tool used by the RCS methodology because it enables the user to visualize and debug the resultant actions of controllers.

A sample animation screen for the submarine maneuvering demonstration is shown in figure 7. Care was taken to maintain scale of dimensions for the submarine model. The ice keel and sea bottom profiles were fractally generated, which allowed a complex
5. CONCLUSIONS

We have described an RCS based automated maneuvering system for a 637 class nuclear submarine involved in under-ice transit in the Arctic Ocean. The control system is capable of maneuvering the simulated submarine toward its intended destination while using simulated sonar data to avoid dangerous ice keels and to maintain the submarine's ordered depth. It can operate autonomously or under human supervision. An operator is presented with all the important maneuvering data graphically in real-time. The system was developed using a set of generic C language controller templates as basic software building components. The RCS design techniques such as this are applicable to a broad class of both commercial and military control systems applications.

REFERENCES


ACKNOWLEDGMENTS

Dr. Anthony Barbera of the Advanced Technology and Research (ATR) Corporation, Laurel, Maryland provided invaluable technical insights to the development methodology. Philip Feldman, M.L. Fitzerald, Clyde Findley, and Nat Frampton of ATR and Ross Tabachow of NIST participated in various aspects of the development effort.