Proceedings of the 1997 Winter Simulation Conference
ed. S. Andradóttir, K. J. Healy, D. H. Withers, and B. L. Nelson

A SIMULATION ENVIRONMENT FOR THE COORDINATED OPERATION OF MULTIPLE AUTONOMOUS UNDERWATER VEHICLES

João Borges de Sousa

DEEC, Faculdade de Engenharia
Universidade do Porto
R. dos Bragas, 4099 Porto Codex, Portugal

Aleks Göllü

Dept.of Electrical Engineering and Computer Sciences
University of California at Berkeley
Berkeley, CA 94720, USA

ABSTRACT

A simulation environment of the coordinated operation of multiple Autonomous Underwater Vehicles (AUVs) is presented. The primary application of this simulation environment is the specification and analysis of an innovative approach to coastal oceanography based on the Generalized Vehicle [GV] concept. A Generalized Vehicle is a group of vehicles whose spatial and logic organization is coordinated in such a way that the group behaves as a single entity. The simulation environment was developed in SHIFT, a new specification language for describing dynamic networks of hybrid automata. These constitute the most adequate modeling formalism for this problem domain. The expressive power of SHIFT provides a compact notation for modeling spatial and logical relationships in a dynamic environment and for modeling and analyzing control strategies governing object interactions.

Keywords: General Applications, SHIFT Language, Hybrid Systems

1 INTRODUCTION

This paper presents a simulation environment of the coordinated operation of multiple Autonomous Underwater Vehicles [AUV]. The primary application of the simulation environment is the specification and analysis of the design of the Generalized Vehicle [GV] architecture (Sousa and Pereira 1996). This architecture, that is based on the GV concept, is at the heart of an innovative approach to oceanographic field studies proposed under the project *Multiple Autonomous Underwater Vehicles for Coastal Oceanography.*

The requirements of this simulation environment include: 1) Complete 6 degree-of-freedom [dof] dynamic models of AUVs; 2) Description of oceanographic phenomena; 3) Models of the interactions between models of the vehicles and the environment; 4) Models of sensors, actuators and communication devices; 5) Detailed description of the coordinated operation of AUVs in a finite number of formations; 6) Models of alternative control architectures.

The simulation environment was developed in SHIFT (Deshpande, Göllü and Semenzato 1997), a new programming language for describing dynamic networks of hybrid automata. Networks of hybrid automata constitute a powerful modeling framework for this problem domain that is suitable for incorporating the discrete event and continuous dynamics that arise in the coordinated operation of multiple AUVs. The expressive power of SHIFT enabled the mapping of the application domain description into a compact representation, and the modeling of complex dynamic interactions among modeled objects.

This paper is organized as follows. In Section 2, we present the motivation for this work by describing the GV concept and the project *Multiple Autonomous Underwater Vehicles for Coastal Oceanography.* In Section 3, we describe the control architecture in order to state the simulation requirements. In Section 4, we present the structure of the simulation environment and the design methodology that we used, emphasizing the specific aspects of the SHIFT language that proved invaluable at this stage. In Section 5, we present some simulation results. In Section 6 we discuss the conclusions.

2 MOTIVATION

The sampling requirements of oceanographic and environmental field studies are becoming more and more demanding. The requirements of the next generation of oceanographic field studies, collectively described as real-time oceanography (Curtin et al. 1993), include: 1) obtain spatially distributed, temporally correlated measurements; 2) respond in a timely fashion to episodic events; 3) obtain time series of spatially distributed phenomena; and 4) in-

teract with measurement platforms in the course of observations. It becomes clear that traditional sampling techniques involving moored instrumentation or towed bodies from ships are no longer adequate: 1) Arrays of moored, and therefore static, instrumentation can provide simultaneous time series, but spatial resolution is typically poor due to the high cost; 2) Towed bodies can provide quasi-synoptic two-dimensional sections of the evolving fields. The limitations of traditional approaches are mainly related to the absence of adaptive spatial and temporal sampling capabilities. These capabilities, enabling an economic and detailed characterization of oceanographic phenomena, are within the possibilities of an emerging technology: AUVs (Healey, Pascoal and Pereira 1995). The technical challenge is then to manage the complexity of the operation of AUVs in an adverse environment, the ocean, where navigation and communications have severe limitations.

The Multiple AUVs for Coastal Oceanography project from Porto University (Sousa et al. 1997) envisages the demonstration of an innovative approach for oceanographic field studies. The demonstration unit consists of AUVs of the REMUS (Remote Environmental Measuring Units) class, developed by the Woods Hole Oceanographic Institution (Alt et al. 1994). Two field demos (Pereira et al. 1996), involving CTD [Conductivity-Temperature-Depth] measurements in a pre-defined path are scheduled for 1997. The project proposes a systems perspective, based on the concept of the Generalized Vehicle [GV], involving the coordinated operation of multiple vehicles within an environment where the maximization of the synergistic interactions with other devices and vehicles is sought. A GV is a group of vehicles whose spatial and logic organization is controlled in such a way that the group behaves as a single entity. A framework governing the coordinated management of resources distributed among several vehicles has several applications: 1) Endows the operation of small, and relatively simple, vehicles with sophisticated behavior that does not have a parallel in the operation of more complex vehicles, either in isolated operation or coordinated by humans. 2) Enables the specification of new vehicles and systems whose features are designed for the operation in the GV framework. In conclusion, the whole concept shifts mission complexity from the mechanical world to the control architecture.

3 REQUIREMENTS

The primary application of the simulation environment is the specification and analysis of the design of the GV architecture. The description requirements are very complex since the whole system includes both continuous activities and discrete-event features (i.e., constitutes a hybrid system) evolving in several time scales. The dynamic nature of the problem stems from the existence of multiple vehicles whose roles, relative positions, and dependencies change during underwater operations. To meet these complex system description requirements, the GV architecture is modeled as a dynamic network of hybrid systems using the SHIFT formalism and specification language.

SHIFT was the specification language since it provided the most adequate modeling formalism for this problem domain. Furthermore, its expressive power in terms of the coordination and control of multiple vehicles had been successfully illustrated by the PATH project (Varaiya 1993).

The structural simulation requirements are better described in terms of the GV control architecture that is organized in five layers:

The physical layer. For each vehicle, this layer consists of actuators, sensors and communication devices. The vehicle dynamical models are given in terms of nonlinear ordinary differential equations.

The abstraction layer. This layer defines a uniform interface to the functional layer by encapsulating each component of the physical layer in a transition system providing adequate abstractions of the underlying behavior. Each abstracted component also handles local fault analysis and diagnosis.

The functional layer. The components of this layer are virtual sensors and systems that basically provide the motion, guidance and navigation operators:

1) The positioning system, involving the integration of data from different sensors and external communication;

2) The guidance system, with two major classes of operation, pre-defined paths or phenomena-guided motion (eg. gradient following);

3) The motion control system, where different controllers are required not only by the complex dynamics of an AUV, but also by the phenomena-guided activities;

4) The power system, that monitors and manages a crucial resource for autonomous operation thus constraining the set of available modes of operation. Each system has several modes of operation and different combinations of the operation modes from each system are required for the GV operation.

The functional layer implements these systems and fault management. Each system is organized as a set of components characterized by discrete and contin-

uous dynamics. The corresponding transition structures are defined by the internal objectives, by commands generated by the coordination layer and by events generated by the physical environment.

Examples of the functional layer activities corresponding to some modes of operation emphasize and illustrate specific aspects of the simulation requirements. In the bottom-following operation, the control loop is closed with measurements from an altimeter while the AUV maintains a legal position, in terms of safety rules and of the operational constraints of the current positioning system mode. A motion coordination operation requires inter-vehicle communication to close the control loop. The communication processes add complexity to the above requirements by introducing delays and communication errors that involve additional control modes. This represents a novel hybrid control configuration, where several hybrid automata involved in guidance and maneuver control of individual vehicles interact through asynchronous message passing.

The coordination layer. There are two levels of coordination. 1) *Intra-vehicle* coordination dynamically selects composition rules for the systems composing the functional layer of each vehicle according to its role in the GV. This role is defined by the Inter-vehicle coordination layer and by the environment constraints. For example, the navigation and control pairs are selected according to the motion strategy defined in the mission specification. 2) *Inter-vehicle coordination* schedules and supervises the execution of maneuvers corresponding to the spatial and logic organization modes defined by the organization layer. Adequate abstractions are required since the continuous evolution of each vehicle will be constrained and controlled by the logic and spatial organization of the GV. The GV spatial organization can be fixed, time-variant or, possibly, adaptive. A fixed spatial organization enables time-correlated simultaneous measurements to be taken by each vehicle. Spatial organization is planned according to the characteristics of the sampled phenomena. Adaptive spatial organization enables active search (e.g. gradient following) where no *a-priori* path is defined, and the mission is described by a search strategy.

The organization layer. This layer supervises the execution of a mission plan thus defining the temporal evolution of the spatial and logical organization modes of the GV.

The GV operation imposes complex requirements to the control architecture. Systematic procedures for the design of individual pieces of controllers or other components of the control architecture are available in the literature. However, there is no systematic procedure ensuring that a given architecture – a structure where these individual pieces are integrated – is the best solution in terms of the available resources and the stated requirements. The problem is further complicated since the different operational scenarios impose additional requirements for the architecture in terms of flexibility and readiness. In the absence of such systematic procedure, simulation plays an important role in the study and evaluation of different alternatives for the control architecture.

In conclusion, a domain specific micro simulation framework is needed to facilitate the timely and cost-effective analysis, design, evaluation and implementation of the GV framework. The framework provides an implementation of the overall GV control architecture as well as dynamical models of AUVs. It allows control and modeling engineers to easily specify, simulate, and sanitize their control, communication, and sensing algorithms before they start experiments with actual AUVs. In a later stage of development, planning of operational missions by oceanographers and engineers will be facilitated by added visualization and graphical interfaces (Brutzman 1994) supporting user-interaction with simulation runs.

4 THE SIMULATION ENVIRONMENT

SHIFT users define types (classes) with continuous and discrete behavior. A simulation starts with an initial set of components that are instantiations of these types. The world-evolution is derived from the behavior of these components.

A type consists of numerical variables, link variables, a set of discrete states, and a set of event labels. The variables are grouped into input, state, and output variables. The inputs and outputs of different components can be interconnected. Each discrete state has a set of differential equations and algebraic definitions (flow equations) that govern the continuous evolution of numeric variables. These equations are based on numeric variables of this type and outputs of other types accessible through link variables. The transition structure of the hybrid automaton may involve synchronization of pairs or sets of components.

The system alternates between the continuous mode, where the evolution is governed by the flow equations, and the discrete mode, where simulation time is stopped and all possible transitions, determined by guards on transitions and/or by event synchronizations among components, are taken. During a discrete step components can be created, interconnected, and destroyed. Currently the continuous mode is implemented by a fixed step Runge-Kutta in-

tegration algorithm and the step size determines the accuracy of the simulation.

4.1 The Main Entities

An object-oriented methodology was used in the process of mapping the simulation requirements into a SHIFT specification. The object-oriented features of the language were crucial to ensure the modularity and the re-usability of the code.

We developed a framework defining the objects in the world and the overall control architecture. Users will, within this architecture, provide individual control pieces, such as supervisors and feedback controllers, as plug-and-play components. The specification of the main entities evolving in the simulation environment resulted from mapping the major components of the Future Real World [FRW] onto the SHIFT types that compose the `world` type. The plug-and-play capabilities dictated the preservation, under this map, of the interfaces between models of physical devices and models of controllers.

The `world` containment hierarchy can be found in Figure 1 – the `Soundwave` type models the propagation of sound waves in the ocean. To model the underwater environment we used SHIFT functions which define a generic interface for querying underwater environment databases (Patrikalakis et. al 1995). In this simulation model, the (x,y,z) position of a component is described in a global reference frame.

The specification of the main types of the `world` type was followed by the definition of the structure of each of those types. In the FRW, instances of these types are generically composed of: 1) physical devices, and 2) the respective controllers, if any. Figure 1 presents the containment hierarchy of the `AUV` type. Physical devices, are mapped onto the `physical layer` type. Their models and interactions with the underwater environment are invariant with respect to the design of the control architecture.

In the ensuing specification step we developed models of the physical entities – physical layer – for each of the main types and models of the corresponding internal interactions. Figure 1 presents the containment hierarchy of the physical layer of the `AUV` type. The `Dynamic_Model` type describes a complete 6 dof hydrodynamic model of a rigid-body underwater body (Healey and Lienard 1993). This type can be configured by over one hundred parameters to model any submarine of interest to the project. The type inputs were standardized so that inheritance can be used to represent others types of vehicles (such as cars). Accordingly, the inputs are torque and forces that are generated by the corresponding actuator models. A library of sensors, actuators, power systems, navigation and communication devices enables to rapidly assemble different configurations of the physical layer. `Sensor_Env_P` is the unique type of this layer that was not directly mapped from a physical device. Instances of this type are attached to the corresponding sensor type instances (through input/output and synchronization links) in order to model the interactions with the rest of the world, thus preserving modularity and re-usability of the sensor models.

4.2 Dynamic Interactions

The simulation of the dynamic interactions among instances of the types described in the previous section is made possible by the SHIFT syntax and semantic model. Interactions between each instance of an `AUV` type and the underwater environment are modeled by `Sensor_Env_P` components. The interactions within an instance of the `world` type are expressed by the input/output connections among components and by the link variables that can change their value during a discrete transition. For example, the interactions among the internal components of an instance of `AUV` type may occur synchronously - in the control loops - or asynchronously - a power failure is modeled by a collective transition to the shutdown discrete state in all components that depend on that system. These interactions occur concurrently in the `world` type which, as depicted in Figure 1, is composed of several components evolving in the global simulation time.

Due to space limitations we briefly describe a simple example of the interactions between instances of `AUV` and `Transponder` types to illustrate the underlying hybrid behavior. One of the acoustic navigation systems modeled in this simulation environment uses a hyperbolic algorithm to determine the position of an AUV with respect to three transponders located in known positions. The master transponder `M` creates a `Soundwave` of frequency f1 every `Q` seconds. When this `Soundwave` reaches the second transponder `S1`, this, in turn, creates a `Soundwave` of frequency f2. The process is repeated with the third transponder `S2`. Each individual AUV mounts a `Hydrophone` type sensor to detect `Soundwave` components. By using the time intervals between two consecutive arrivals the navigation system determines the position of the AUV. A simplified 2D representation of this scheme is presented in Figure 2, where a series of snapshots describe the temporal and spatial evolution of `Soundwave` instances and the events triggered by this process.

The type `Soundwave`, whose instances are created

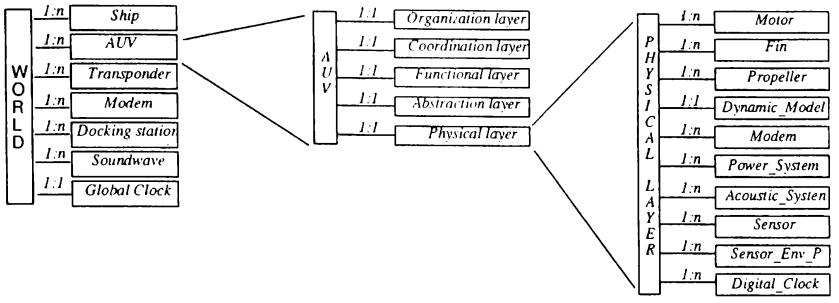

Figure 1: Simulation entities and containment hierarchies

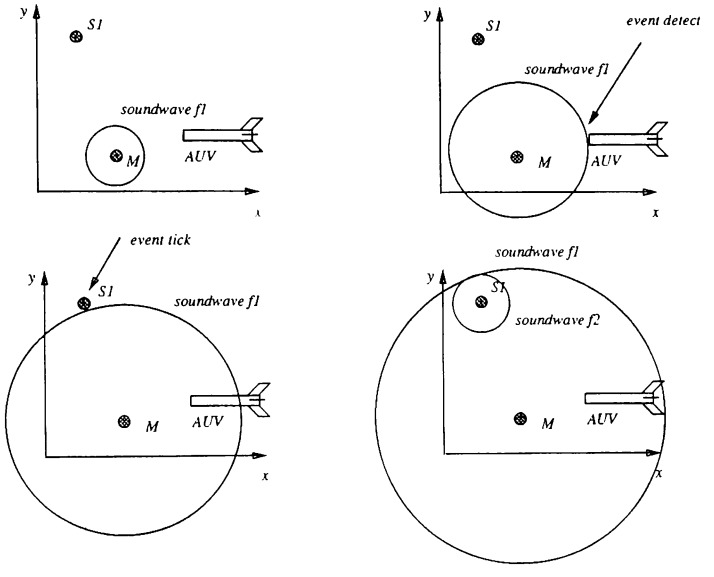

Figure 2: The acoustic navigation system scheme

by Tranponder components, models the spherical propagation of sound waves (that are used for navigation and communication). The corresponding flow equation is linear in the speed of sound in water (1500 ms-1). The outputs of a Soundwave component are the current diameter d and the coordinates (x0,y0,z0) of the corresponding Transponder component at creation time. Upon creation, each instance of Soundwave adds itself to a global set G_Waves. Fragments of the Soundwave type are.

```
type Soundwave { ...
   output continuous number x0,y0,z0,d,freq;...
   flow default d' = speed_of_sound; ...
   ...
   setup do {G_Waves := G_Waves + {self};};
.... }
```

The master transponder M creates a Soundwave every Q seconds. The creation is triggered by an event tick that is synchronized with a digital clock. In the case of transponders S1 and S2, the same event is synchronized with the event detect from a Sound_W_SEP component (to be described next).

```
type Transponder {
   input continuous number u_xp, u_yp, u_zp, u_f;
...
   output Sound_wave sw; ...
   discrete stop, send; ...
   export tick;
   transition send -> send {tick};
   do
   {
      sw := create(Sound_wave,
         x0:=u_xp,y0:=u_yp,
         z0:= z_up, freq:=u_f);
   }
}
```

On the AUV, the Acoustic_System is responsible for detecting instances of Soundwave. It decomposes the functionality into two types. The Sound_W_SEP type is a simulation entity that encapsulates interac-

tions with Soundwave in the simulation. In essence, this type provides an ideal sensor that will catch each Soundwave exactly. The Hydrophone type then uses this information and models any sensor noise that may be particular to the actual physical device. The Acoustic_System connects the two components and ensures that the detection of a Soundwave in a Hydrophone and Sound_W_SEP are synchronized on their detect event.

```
type Acoustic_System {
    state Sound_W_SEP SEP; Hydrophone H; ...
    setup connect  { SEP:detect <-> H:detect; ...
} ...
}
```

The Sound_W_SEP component maintains a set L_Waves of the Soundwave components that have been detected. The event detect is triggered when there is a Soundwave component in G_Waves that is not in L_Waves such that the corresponding sphere intersects the Sound_W_SEP location. In this case the new Soundwave is added to the local set L_Waves.

```
type Sound_W_SEP { ...
    input continuous number x,y,z,f;
    export detect; ...
    discrete track_env; ...
    transition track_env -> track_env {detect}
    when exists p in (G_Waves - L_Waves):
            ((dist(x0(p),y0(p),z0(p),x,y,z)
            <= d(p)) and f=freq(p))
        do {L_Waves = L_Waves + p};
.... }
```

5 SIMULATION RESULTS

The simulation environment consists of about 4,000 lines of SHIFT code that are translated to about 14,000 lines of C code. In the tests carried so far, the functions describing the underwater environment were quite simple. For example, the bottom of the ocean was modeled by the product of two sinusoids.

We simulated the operation of six Phoenix AUVs (Healey and Lienard 1993) with a complete 6 dof model, in an underwater environment where 3 fixed transponders, placed at appropriate locations, are used for acoustic navigation. The hydrodynamic model is quite complex involving more than one hundred parameters. The state variables are defined in local coordinates and the global position of each submarine is obtained by a standard coordinate transformation. Each instance of an AUV type involves over fifty SHIFT types ranging from the hydrodynamic model and models of altimeters and accelerometers, to types that encapsulate control laws and navigation algorithms or supervise the respective invocations. The calculation of the integration step size is determined by the evolution model of a SHIFT system. In this particular application, the size of the integration time step is determined by the accuracy of guard-crossings related to the acoustic navigation algorithm. This is based on the detection of sound waves. The flow equation of the Soundwave type is linear in the speed of sound in water (1500 ms-1). The maximum acceptable size for the integration time step was, therefore, 0.001 seconds. With this time step, the simulation time was about 3 times slower than real time. These experiments were performed on a SUN ULTRA-1 workstation with 64 MB of memory and Solaris 5.1 operating system. Our analysis indicate that a substantial amount of the simulation time is spent in the calculation of the AUV's trajectories, as expected due to the complexity of the AUV's dynamic model that we used. The structure of this problem is amenable to optimization of the simulation time since a reasonable accuracy of the integration of the dynamic model is achieved with a time step of 0.01 seconds.

6 CONCLUSIONS

This paper presented a simulation environment for the development and testing of an innovative approach to real-time oceanography based on the coordination of multiple autonomous underwater vehicles. The unique features of the SHIFT language, particularly the hybrid automata model, the dynamic reconfiguration of the model, the powerful synchronization constructs and the operations on sets were instrumental in the process of modeling the problem domain and in the specification of a structure were control architectures can be tested.

The modular specification and the development of libraries of actuators and sensors enabled to reuse most of the SHIFT code to simulate the operation of multiple industrial autonomous mobile platforms.

ACKNOWLEDGMENTS

This simulation environment was developed while João Sousa was a Visiting Scholar at the University of California, at Berkeley. The author is deeply grateful to his host Prof. Pravin Varaiya and to members of PATH. The authors also thank Akash Deshpande, Daniel Wiesman and Marco Antoniotti for stimulating discussions and valuable comments, insights and contributions. Joao João was in part funded by Fundacao Luso-Americana para o Desenvolvimento and by Direccao Geral do Ambiente, project PEAM/P/TAI/254/93. Aleks Göllü was in

part supported as part of the California PATH program of the University of California, in cooperation with the State of California Business, Transportation, and Housing Agency.

REFERENCES

Alt, C., B. Allen, T. Austin and R. Stokey. 1994. Remote Environmental Measuring Units. In *Proceedings of IEEE Symposium on Autonomous Underwater Vehicle Technology (AUV'94)*, Cambridge, MA, USA, 19-20 July 1994. New York, NY, USA: IEEE, 1994. p. 13-19.

Brutzman, D. 1994. A Virtual World for an Underwater Vehicle. *Phd. Thesis.* Naval Postgraduate School, Monterey, California.

Curtin, T., J. Bellingham, J. Catipovic and D. Webb. 1993. Autonomous Oceanographic Sampling Networks. *Oceanography*, Vol. 6, (no. 3): 86–94.

Deshpande, A., A. Göllü and L. Semenzato. 1997. The SHIFT Programming Language and Runtime System for Dynamic Networks of Hybrid Automata. *California PATH Research Report UCB-ITS-PRR-97-7.*

Healey, A. and D. Lienard. 1993. Multi-variable Sliding Mode Control for Autonomous Underwater Diving and Steering of Unmanned Underwater Vehicles. *IEEE Journal of Oceanic Engineering*, July 1993, vol.18, (no.3):327-39.

Healey, A., A. Pascoal and F. Lobo Pereira. 1995. Autonomous Underwater Vehicles: An Application of Intelligent Control Technology. In *Proceedings of the American Control Conference - ACC'95*, WA, USA, 21-23 June 1995, Evanston, IL, USA: American Autom Control Council, 1995. p. 2943-9 vol.5.

Patrikalakis, N. M., C. Chryssostomidis, S. T. Tuohy, J. G. Bellingham, J. J. Leonard, J. W. Bales, B. A. Moran and J. W. Yoon. Virtual Environments for Ocean Exploration and Visualization. 1995. In *Proceedings of Computer Graphics Technology for the Exploration of the Sea 95*, Rostok, May 1995.

Pereira, F. L., J. Borges de Sousa, C. Gil Martins, E. Pereira da Silva. AUV System Requirements for Coastal Oceanography. 1996. In *Proceedings of Symposium on Autonomous Underwater Vehicle Technology*, Monterey, CA, USA, 2-6 June 1996,. New York, NY, USA: IEEE, 1996. p. 399-406.

Sousa, J. B. and F. Lobo Pereira. A Generalized Vehicle Control Architecture for Multiple Autonomous Vehicles. 1996. In *Proceedings of Symposium on Autonomous Underwater Vehicle Technology*, Monterey, CA, USA, 2-6 June 1996,. New York, NY, USA: IEEE, 1996. p. 223-230.

Sousa, J. B., N. Cruz, A. Matos and F. Lobo Pereira. 1997. Multiple AUVs for Coastal for Coastal Oceanography. To appear in *Proceedings of the OCEANS'97 Conference.*

Varaiya, P.1993. Smart Cars on Smart Roads: Problems of Control. 1993 *IEEE Trans. Automatic Control*, Feb. 1993, vol.38, (no.2):195-207.

AUTHOR BIOGRAPHIES

JOÃO T. F. BORGES DE SOUSA is a Ph.D. student in the Department of Electrical Engineering at Porto University, in Portugal. He received the Engineering Degree from Porto University in 1987 and he received an M.S. degree in electrical engineering and computer science from the same university in 1992. He worked as a consultant in the area of manufacturing systems and he participated in several projects funded by the EU and NATO. His research interests emphasize hybrid systems, motion coordination of multiple vehicles, optimization, autonomous underwater vehicles, robotics and manufacturing. He is a member of APCA.

ALEKS GÖLLÜ received his B.S. in Electrical Engineering from Massachusetts Institute of Technology in '87 and his M.S. and Ph.D. in Electrical Engineering and Computer Science from UC Berkeley in '89 and '95 respectively. He was a Systems Engineer and Project Manager at Teknekron Communications ('90-'92) Systems and a Software Engineer at Oracle ('89-'90). Currently he is a Research Engineer at PATH/UC Berkeley where his research interests include simulation, modeling, real-time control of hybrid systems. His domain expertise includes telecommunications, power distribution, highway automation, large-scale software development, database management systems, and control technologies.