Presented at NATO Symposium,
Applied Vehicle Technology Panel,
Ankara, Turkey,October 2000-08-29

Olivier Doucy*, Donald Brutzman+, Anthony Healey+

(*)Sirehna, SA Nantes France
(+)Naval Postgraduate School, Monterey, CA,

Operation in shallow waters under wave disturbance is a critical task for an autonomous underwater vehicle (AUV) mapping minefields or taking video snapshots. The design of an operational station keeping system requires an efficient compensation of first order wave disturbance. This paper describes also the development of shallow waters station keeping control schemes for the former NPS AUV, Phoenix in a simulation. A key feature is the use of the AUV virtual world and the real time simulation architecture based on IEEE Distributed Interactive Simulation (DIS) protocol for the validation and post-testing analysis of the control design. The virtual world developed at NPS allows real-time 3D visualisation and sophisticated performance analysis. It is an important complement to real world experimental missions, which are time consuming and dangerous for the vehicle at software design stage. Part of the work described here has been interfacing the virtual world with traditional control design tools such as Matlab and Simulink. The result is a user-friendly environment for simulation and control design of AUVs and other vehicles. Station keeping is based on previous studies at NPS on surge control extended to other degrees of freedom of the AUV. A short presentation of the last generation of Naval Post Graduate School AUV, the ARIES is also given here.

This paper focuses on the development of a simulation tool for an AUV, that could be easily used to assess vehicle systems design and implementation with particular emphasis on control. In the AUV development process, simulation has been a necessary tool since the early days especially because of risks involved by launching invalidated vehicles and control implementations. Nevertheless, besides the efforts required developing models of underwater vehicles, of the surrounding environment and of the embedded systems (sonars), the use of simulation has been slowed down by two factors: The difficulty to interpret a full 6 d.o.f. behaviour with the sole use of 2D plots and the software design and programming effort required from non computer literate people (e.g. the naval architects, mechanical and control engineers) usually involved in a robotic vehicle design. This paper describes an attempt to solve those two problems through the development of an open simulation environment. This product is based on traditional standard tools of AUV developers interfaced with a real-time 3D virtual reality representation. It is also a necessary tool for post-mission analysis as the AUV is totally invisible during its mission. For this 3D representation, our work is based on the use on open standards and software freely available on the Internet. The simulator and 3D viewer run on low cost hardware, which is a contrast with other simulators with real-time 3D virtual representation. Those are usually based on supercomputers and costly proprietary standards. The use of the simulation environment for the station keeping of a small AUV is described here as an example of application. Future real world implementation is envisaged on the ARIES Vehicle developed at NPS.

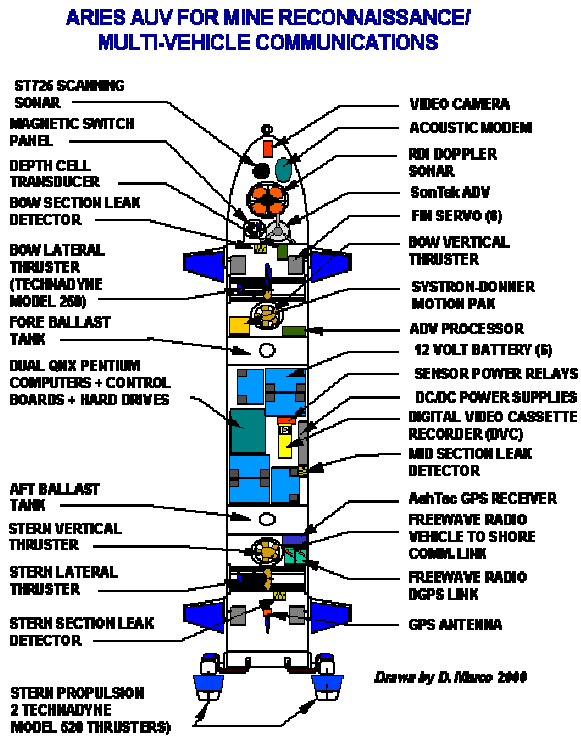

Fig 1 : Drawing of the ARIES vehicle

The Center for AUV Research at NPS, Monterey has developed AUVs and underwater vehicle navigation and control schemes since 1986. ARIES

(Acoustic Radio Interoperative Exploratory Server) is a third generation vehicle. The vehicle is intended for multi-mission objectives :

- •Network Server for Multi-Vehicle Cooperation
- •Oceanographic Survey/Sampling
- •Target acquisition/reacquisition

The vehicle went for the first time in the water in November 1999 and performs weekly missions in the Monterey bay. Mains specifications are the following :

- •Dimensions & Hardware:
  - •Length: 3 m, Width: 0.5 m, Height: 0.4 m
  - •Weight 230 Kg, Neutrally Buoyant
  - •Twin Main Propulsion Thrusters, 2 Lateral and 2 Vertical Cross- Body Thrusters
- •Power System:
  - •6 12 Volt Rechargeable Lead Acid Batteries
  - •4 Hour Endurance at Top Speed, 20 Hour Hotel Load Only
- •Top Speed 3.5 Knots

- •Computer System:
  - •Two Independent Cooperating Pentium II processors + PC104 Data acquisition boards
- •Embedded Sensors:
  - •RD Instruments Navigator DVL
  - •Tritech ST725 and ST1000 Scanning Sonars
  - •Systron Donner 3-axis Motion Pak
  - •Magnetic Compass
  - •Video Camera
  - •DGPS
- •Navigation:
  - •8 State Extended Kalman Filter using Sensors above with Periodic DGPS Updates for Global Position Correction
- •Communications:
  - •Ethernet While in the Lab Environment
  - •Radio Modems while Deployed and Surfaced
  - •Acoustic Modem While Submerged

Multi-low cost Vehicle Fleets have been identified as potential valuables solutions for search and reconnaissance tasks. Those fleets composed of vehicles with reduced individual capabilities, need a communication server to act for vehicle tasking/retasking and communication relay. The use of a mobile vehicle as a server has many advantages such as avoiding buoys deployment and optimising bandwidth and reliability as the server can be in close proximity to the search vehicles. Being a server vehicle is the main objective of ARIES. More details about ARIES and its missions can be found in Marco2000.

NPS has long been convinced that the use of simulation was necessary for the development of an AUV (Brutzman94). To diminish launching risks or to demonstrate mission feasibility, vehicle and environment models have to be as close to real world behaviour as possible. A real-time 3D representation is also necessary to analyse complex situations encountered for example in obstacle avoidance and relative to feature navigation (Marco 96). NPS Center for AUV research has used a virtual world for AUV for many years (Brutzman94). This simulation environment had extended possibilities, including “hardware in the loop” simulation where the virtual world actually simulated sensors responses and was interfaced to the real robot software and hardware. Nevertheless, this early simulation environment had many important drawbacks. First of all, it was written in C and was platform specific. The 3D representation was programmed in OpenGL and only ran on costly Silicon Graphics workstations. It was therefore an expensive tool to implement and maintain in an era of cost reduction. Moreover customisation and testing of specific mission or additional system or algorithm required the intervention of a Computer Science specialist with previous experience of the code. It is also required to use the entire robot code which was an additional difficulty to isolate failures and bugs.

The first of the drawbacks was recently fixed with the development of the 3D representation under VRML 2.0 (Virtual Reality Modelling Language) (Brutzman97) as a networked viewer. The use of open standards for networking (DIS) and 3D representation (VRML) has tremendously driven down the cost of software and allows the simulation to run on different platforms including low-cost PCs within a standard web-browser (Netscape 4.7). Slight loss of performance is not an issue as the exponential growth of computing power at constant price erases this loss in a few months.

Nevertheless, giving direct access to the simulation to a large audience of naval architects and mechanical engineers was still an issue. Within the community, for a few years Matlab® and its temporal domain simulation environment Simulink® have become standard tools. The most efficient way to provide a simulation environment for AUV design and assessment, feasibility testing, was to use those software tools. The intention was to build a powerful and user-friendly tool. The environment had to be completely open for the addition of new simulated systems and control schemes. Besides this, interface to powerful plotting functions, Computer Aided Publishing software and a real-time virtual reality representation were needed. Plotting functions and

interfaces to publishing tools are provided by Matlab®. The interface to NPS virtual world had to be built based on the standards previously used.

VRTP is a transfer protocol developed at NPS to build Large Scale Virtual Environment, that is to say 3D graphics Virtual Environment distributed over the Internet where thousands of users (players) interact in real-time. VRTP was designed aiming at virtual battlefield applications (Brutzman97). VRTP is based on three main features which are open standards: VRML and X3D (next generation for VRML) for virtual scene description and animation, DIS (Distributed Interactive Simulation) protocol (IEEE 1278) for multi-user application level communication and Mbone to provide multicasting streaming capabilities over Local Area Networks and the Internet. We will now give more details about VRML and DIS as they have actually been used for the development of the simulation environment. Mbone is a much more in depth layer, which has only been seen as user point of view.

VRML was designed as a language for 3D virtual scene description tailored for Web applications. As VRML 1.0 standard (Bell95) was a core set of object-oriented graphics constructs augmented by hypermedia links, all suitable for cross-platform 3D scene generation. VRML 2.0 (Carey96) extends 3D scene interactivity by defining language interfaces for JavaScript (Netscape96) and Java (Sun96). Java is much more powerful tool as a compiled language that permits direct network connections using sockets and therefore allows more functions such as animating shapes from the network or interacting with distributed applications. VRML scenes are usually viewed directly inside 2D browsers such as Netscape enhanced with a specific "helper" application called a "plug-in", "Cosmoplayer". All these tools are freely available on the Internet.

As stated in (Brutzman97), the DIS standard describes the packet format for 27 protocol data on units (PDUs) specified for use in military combat. Of these only a few have general applicability. DIS embeds functions necessary for entity identification, exercise specification, event occurrences (such as firing a weapon) in compact form suitable for streaming and large size environments. We are most interested in the widely used Entity State PDU (ESPDU). Entity state in this context includes linear and rotational values for posture, velocity and acceleration in combination with efficiently designed algorithms for dead reckoning and track smoothing. The ESPDU is well suited for relaying physically based model information among numerous interacting participants in real time. In clear, DIS describes the messages sent over the network by the distributed components to describe the dynamic behavior of entities present in the virtual environment.

Components of VRTP are shown on next figure.

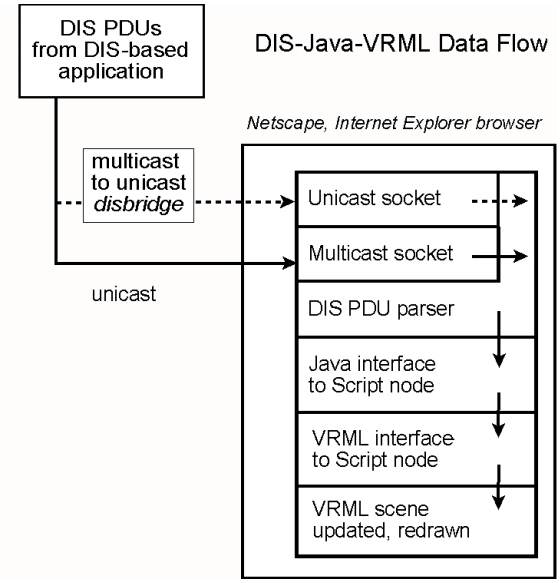

figure 2 : Implementation of VRTP

We have used all components of VRTP as developed at NPS, allowing our simulation environment to be part of a large-scale distributed simulation in a mid-term perspective.

The purpose of our work was to build a high-resolution dynamic simulation of the AUV with real-time and faster than real-time capacities. Environment simulation had to include wave and currents. Modular and open architecture was a must be since future functionalities are planned to be added. Additional functions could be for example surface buoyancy efforts, sonar and terrain models, additional AUVs…

Physical behavior of the AUV is based on NPS dynamic model (Healey92), used in Jeffrey Riedel's work (Riedel99). The dynamic equations are written in appendix 1. For full nomenclature definition and model definition refer to Riedel99. Wave forces are included in those equations. It is assumed here that the AUV body is small compared to the wave length and that it is totally submerged. Therefore, only excitation forces (Froude-Kriloff) from incoming waves are taken into account. Linear damping due to radiation is small and only viscous damping (Morison term) is taken into account. The interest of the model is that it can be easily generalized to any AUV as soon as the hydrodynamic derivatives that appear in the

equations have been determined through experiment or theoretical evaluation. Our simulation environment is not restricted to NPS AUV, but can be used for any vehicle.

In order to compute wave efforts incoming wave induced fluid velocities and accelerations have to be simulated. It is well known that those are random sequences. They are usually modeled by their energy spectra. Several representations have been proposed throughout the decades based on experimental data. Our simulation uses the well-known representation of Pierson-Moskowitz (Lewis89) . This spectrum equation is:

$$S(\omega) = \frac{AH_s^2}{T_s^4 \omega^5} e^{-(B/T_s^4\omega^4)} \quad (1)$$

with A=0.0081g^2 and B=0.74g^4, ω the wave pulsation in rad/s, H_s the significant wave height and S the spectral density of wave height at ω. The simulation also includes a directional spreading of the spectra based on a cosine repartition. Up to three spectra can be simulated at the same time for multi-directional seas.

But, more than the actual dynamic model of the vehicle and its environment the architecture of the simulation environment was of importance here. As a consequence it has been designed using a modular approach allowing easy addition and replacement of functionalities. On the vehicle systems simulation side, the main modules have been designed to respect the general architecture of the AUV system architecture, reproducing the Sense-Decide-Act internal control loop. This has been achieved through three main modules :

- Sensors: contains all embedded sensors. Different level of model can be introduced (perfect measurement of the physical quantities they measure, introduction systematic and non-systematic errors, introduction of delays, failures….)
- Control: This module includes the simulation of the control scheme embedded in the vehicle. This the place were usesr would put the control algorithm they have under development and want to simulate. The modular approach allows easy replacement and partial testing of specific functionalities. Again different levels of detail can be introduced (full continuous control, discrete control, introduction of strategic and tactical control level…). Depending on their abilities, users can program with Simulink® graphical language or Matlab® programming language or use C language. The latter requires Computer Science literate people but allows testing a source code close to what will actually be implemented on the onboard computer. A mid-term perspective is to allow hardware in the loop simulation by interfacing this control module to the actual robot computer. This would require the Real-Time Toolbox® which is another Matlab® product.
- Inspector includes the display and 2D plots of all simulation variables grouped by modules and also storage functions. Those allow analysis of the simulation while it is being performed.

On the world (environment and dynamics) side, a mirror architecture to this loop has been established by building also three main modules :

- Environment: This module is responsible for the simulation of environment variables. For instance, it includes current and waves. Our wish list of additional functions includes terrain model to simulate sonar responses or collisions with the ground.
- Dynamics: performs the actual simulation of the vehicle dynamic equations in their continuous form. Once again a modular approach has been chosen: Different efforts, Newton law integration and kinematic equations integration have been separated in modules allowing for easy change of a specific model (thrusters or fins forces, high resolution buoyancy, specific external fluid velocity field…).
- 3D Views: gives access to the virtual world interface. The user launches the virtual interface and configures DIS parameters.

The structure of the simulation environment is shown on the following figures.

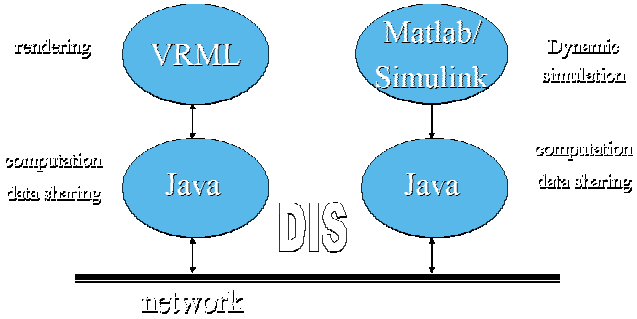

figure 3 : Communication Architecture

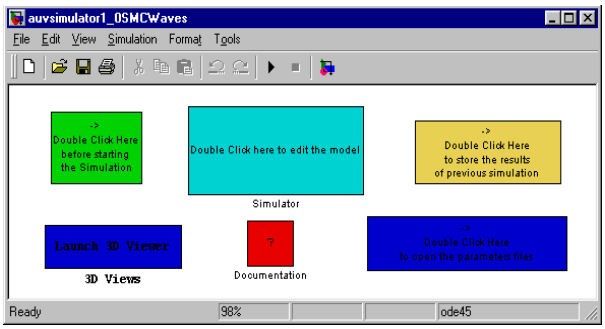

figure 4 : Simulator main window layout

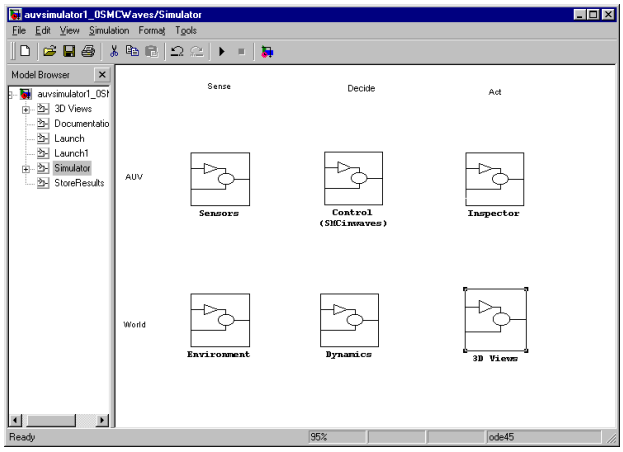

figure 5 : Simulator model architecture

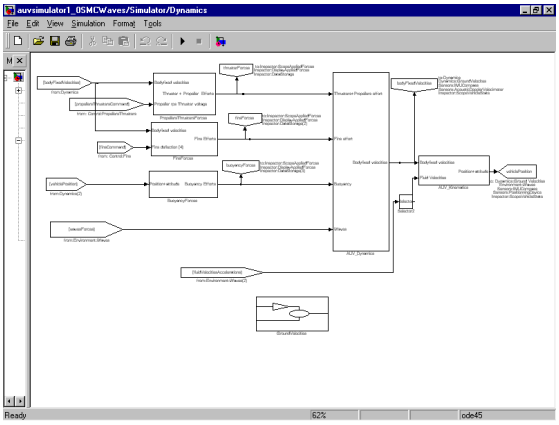

figure 6 : Simulator Dynamics module Architecture

This standard environment has entirely been developed in C, which gives the fastest simulation. This is not a limitation since all customisation can be made using one of the three different methods: Simulink® graphical language, Matlab® programming language or C language.

Besides specific components as described above, the user is able to access simulation parameters (Wave description, Current scenario, AUV Physical parameters, simulation initial conditions, control scheme tunings and set-points…). Those are accessed and initialised through Matlab script files.

As stated above a 3D-virtual real-time representation is necessary for the analysis of the AUV behavior. The method chosen was to interface the simulation environment with the already existing VRTP (DIS-Java-VRML) architecture. For this purpose, a specific module has been added. Java object, executed from Simulink implement the DIS protocol and send the ESPDU described in 3.2. Dynamic simulation and viewers can be both executed on a single powerful PC (Pentium III, 700 MHz, 3D Labs 32 Mo graphics card) or distributed between two less powerful machines. The architecture allows our environment to be a player in a multi-user distributed simulation. Within a mid-term perspective, players could interact with it over the network.

The developed simulator can therefore animate any VRML scene with connectivity nodes and the correct address settings. At this stage, the AUV behaviour could be seen in already existing scenes. Nevertheless to add realistic features, it was necessary to add a representation of the waves disturbing the AUV. A specific networked virtual scene was built under VRML. This scene includes the AUV, the bottom and the sea surface. The sea surface is animated to represent the wave propagation. The virtual representation chosen is a semi transparent surface and a mesh of perpendicular lines that give better appreciation of volumes. To avoid annoying the operator, we need to synchronize waves and the AUV motion. That is to say that not only we have to simulate the same spectral density but also at each time step the wave height on the entire surface has to corresponds to the one at the corresponding time step in the dynamic environment.

For good spatial resolution our surface uses 2600 points. Transferring their coordinates over the network in real-time with a reasonable (10Hz) update rate is simply not feasible nor desirable if we consider a multi-user distributed simulation. The chosen solution was therefore to simulate the wave motion also on the virtual scene side. Sea state parameters and other variables making the Pierson-Moskowitz spectra a deterministic temporal sequence are transferred at the beginning of the simulation. Timestamp of validity of AUV motion are also transferred with the corresponding ESPDU. It is used as the synchronising variable for the wave surface animation. The following figures show the AUV in the virtual scene with the animated wave surface. Yellow cones show the action of thrusters during station keeping.

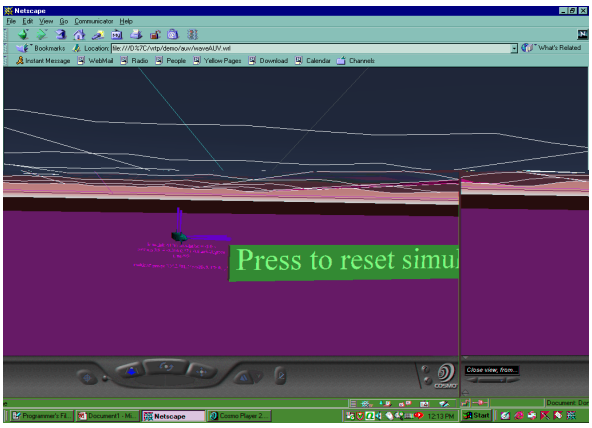

figure 7 : AUV in the virtual world with animated waves

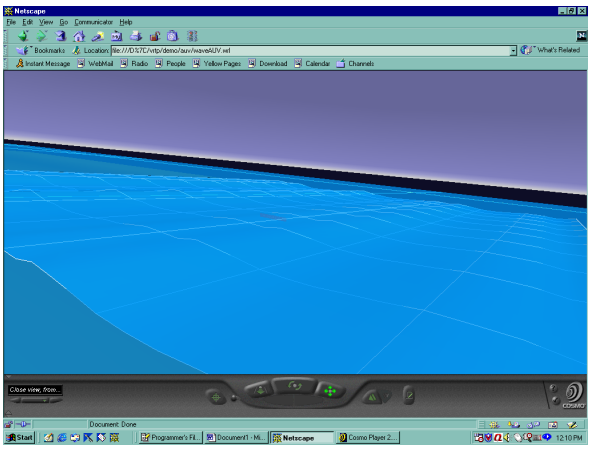

figure 8 : Virtual animated sea surface

For several years now, low cost AUVs have been identified as promising agents for mine warfare. With the emphasis put on very shallow waters (VSW) (ONR98) operations, a typical mission requires a vehicle to follow the bottom at a desired altitude, while being disturbed by incoming waves. Since small AUVs are sensitive to wave induced motions (An96), AUVs in VSW will perform better if they are able to compensate for those disturbances.

Then arises the question of how can be achieved control for disturbance rejection. This problem is well known and has a rich history of study. In previous studies (Riedel99), NPS has shown that knowledge or direct measurement/estimation of these disturbances improves control performance while keeping the control loop stable. This was demonstrated for the surge control of the AUV Phoenix in waves.

This is a rather unusual Dynamic Positioning System (DP) as there are currently no DP for small AUVs and most applications are for surface vessels. In most current applications, wave perturbations are not measured. Their effects are estimated through Kalman estimators and filtered out of vessel position measurement as only low frequency components are rejected.

This paper details work that has followed Jeffrey Riedel studies and that extends his results to other degrees of freedom: heave, pitch, yaw and sway.

The DP control uses main propulsion (ducted propellers) and lateral and vertical cross body thrusters. As shown in Appendix 1, the AUV dynamic model is fully non-linear. Riedel obtained good performance in simulation and experiments while its DP design was based on Sliding Mode Control (SMC). For more details about SMC, one can refer to (Slotine85). The benefits of SMC are its ability easily treat non-linear and multi input/output systems and to take into account uncertainties on the dynamic model. The basic idea behind sliding mode control is to drive the system tracking errors on a stable trajectory with low-pass dynamics chosen by the designers. This trajectory called the “switching surface” or “sliding surface” specifies relations between the different dynamic states of the process. The control input result directly from the non-linear equations of the model and the sliding surface. To ensure that the states are driven down to the sliding surface, a discontinuous term is added to the control inputs. This discontinuous term accounts for disturbances and model uncertainties. In actual implementation, the discontinuous term is smoothed to avoid chattering. An hyberbolic tangent function is usually used to achieve this. There is a “boundary layer” around the sliding surface. Moreover, using SMC, it is possible in case of a multiple d.o.f. problem to build a set of low order controllers, while still explicitly accounting for coupled terms through the embedded model (Slotine86).

For surge control Riedel designed a SMC with wave disturbance feed-forward. The disturbance was determined on the basis of the measurement of the fluid velocities relative to the vehicle body in its vicinity. This controller showed perfect cancellation of the perturbation in the ideal case and gave the best results compared to LQR approaches with or without feed-forward. To extend to other degrees of freedom, we have built two uncoupled controllers, one for the vertical plane motions and one for the horizontal plane. Internal model is a simplified version the one detailed in appendix 1. The wave disturbances have been accounted for at the kinematics level only. Froude-Kryloff efforts are considered as unmodeled uncertainties. The equations of the controllers are

$$F_{cont} = \eta \tanh(\sigma / \Phi) + M(\lambda \dot{\tilde{Z}} + \ddot{\tilde{Z}})$$

where:

$\tilde{Z} = DesiredPosition - EstimatedPosition$

$\dot{\tilde{Z}} = DesiredAbsoluteVelocities$

$- EstimatedAbsoluteVelocites$

$\ddot{\tilde{Z}} = DesiredAccelerations$

$- EstimatedAccelerations$

The estimations are based on measurement and a simplified internal dynamic model.
M is the mass matrix.

$\sigma = \tilde{Z} + \lambda \dot{\tilde{Z}}$

σ is referred as the sliding surface error represents the gap between the vehicle state and the desired sliding trajectory.

$\eta \tanh(\sigma / \Phi)$ is the control switching term where η is gain related to the assumed importance of perturbations and model uncertainties and Φ is the boundary layer thickness.

The computed control efforts have to be produced by the combination of thrusters. Since, there are only two thrusters per side, the effort allocation is straightforward as long as there is no saturation. In case of saturations (under-dimensioned thrusters), the allocation scheme has to be studied further and depends on the objectives of the DP.

Inputs of the control scheme are:

- Vehicle position relative to the desired set point. It can be measured using altimeter, and acoustic positioning. The high output rate and the accuracy required for performance is achievable through hybridation with a dead reckoning navigation based on a bottom locked Doppler Velocity Log. In the case of Phoenix and ARIES a unit manufactured by RDI, San Diego, CA was used.
- Vehicle attitude (heading, pitch, roll) and rates measured by an IMU (Inertial Measuremant Unit) coupled with the onboard magnetic compass for a low cost implementation.
- Vehicle velocity relative to fluid and fluid absolute velocity (reconstructed from relative velocity measurement). Relative velocity measurement is performed by an acoustic velocimeter made by Sontek.

Some results are detailed hereafter.

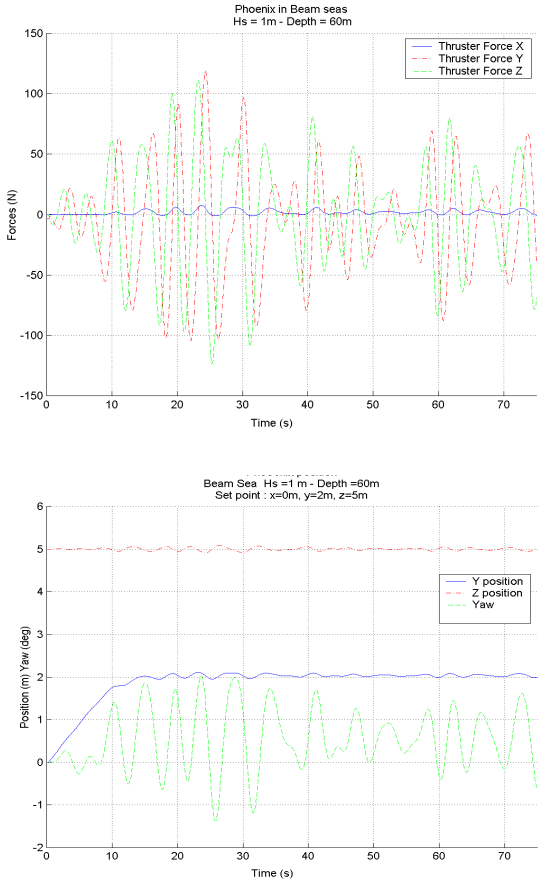

Fig 9 : Station keeping in open seas near the surface

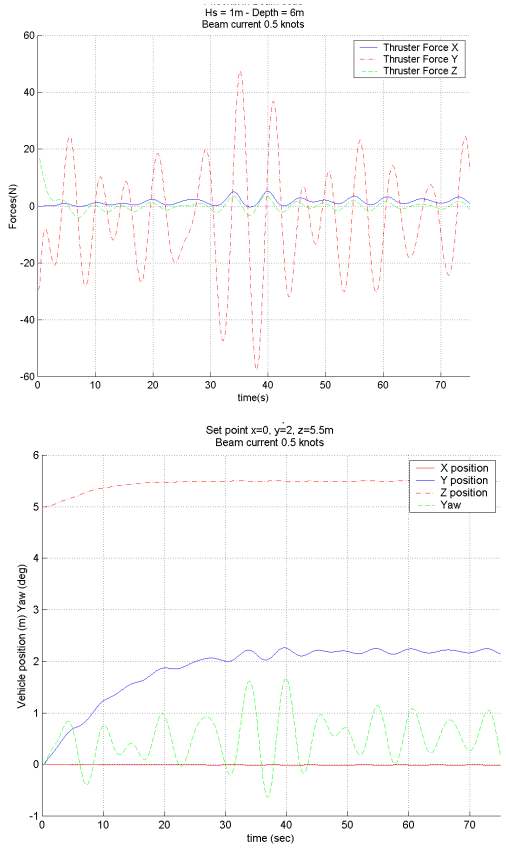

Fig 10 : Station keeping in shallow waters near the bottom

Those results are obtained in simulation. They are primary assumptions and demonstrate the feasibility of a multi d.of. station-keeping for an AUV. Besides, they will help to dimension cross body thrusters for the ARIES vehicle. Those will be installed in the following months. This will open the way for further development of the control algorithm (based on a more accurate model) and in water validation.

Besides the development of the ARIES AUV, NPS schedules to use it for an exploration mission in the Azores during summer 2001. This exploration is part of a collaborative project with the university of Lisbon (Portugal) and consist in the observation of vapour vents off shore the Azores. ARIES will collaborate with the Autonomous Surface Vehicle, CARAVELA.

NPS has also activities in mine warfare simulation and tactical assessment. NPS studies use statistical evaluation of minefield cleaning combined with 3D graphics virtual world simulation (Brutzman96).

Sirehna is currently developing an Autonomous Surface Vehicle. The vehicle has successfully passed integration tests. It is now planned to integrate control systems on a vehicle specifically designed for the intended missions.

An96: An, P.E., Leavitt, G., Smith, S.M., and Dunn, S.E.,"A Quantitative Measure of Sea-State Effect on Small Autonomous Underwater Vehicle Motion in Shallow Water," *Proceedings of Oceanology International 96*, Brighton, UK, March 1996, pp. 211-233.

Bell95: Bell, Gavin, Parisi, Anthony and Pesce, Mark, editors, "The Virtual Reality Modeling Language (VRML) Version 1.0 Specification," May 26 1995. Available via the VRML Repository *http://www.sdsc.edu/vrml*

Brutzman94: Don Brutzman, "A Virtual World for an Autonomous Vehicle", PhD Thesis, NPS, !994.

Brutzman96: Don Brutzman, Bryan Brauns. Paul Fleischman, Tony Lesperance, Brian Roth, Forrest Young, NPS Undersea Warfare Group, "Evaluation of AUV Search Tactics for Rapid Minefield Search using Analytic Simulation and a Virtual World".

Brutzman97: Don Brutzman, Mike Zyda and Kent Watsen, Mike Macedonia, virtual reality transfer protocol (vrtp) Design Rationale, *Workshops on Enabling Technology: Infrastructure for Collaborative Enterprises (WET ICE): Sharing a Distributed Virtual Reality*, Massachusetts Institute of Technology, Cambridge Massachusetts, June 18-20 1997.

Carey96: Carey, Rikk, Marrin, Chris and Bell, Gavin, editors, "The Virtual Reality Modeling Language (VRML) Version 2.0 Specification," International Standards Organization/ International Electrotechnical Commission (ISO/IEC) draft standard 14772, August 4 1996. Available via the VRML Consortium at *http://www.vrml.org*

Healey92: Healey, A.J., "Dynamics of Marine Vehicles," course notes for ME4823, Naval Postgraduate School, Monterey, CA, Fall 1992.

Lewis89: Lewis, E.V. "Principles of Naval Architecture", Vol III. SNAME, 1989.

Marco 96: Marco, D.B., *Autonomous Control of Underwater Vehicles and Local Area Maneuvering*,

Ph.D. Dissertation, Naval Postgraduate School, Monterey, CA, September 1996.

Marco2000: David Marco, A.J. Healey, NPS "Current Developments in Underwater Vehicle Control and Navigation: The NPS ARIES AUV", Oceans Conference September 2000.

Netscape96: Netscape Corporation, "JavaScript Introduction," Web page, October 1996. Available at http://home.netscape.com/,

ONR98: "VSW BAA," http://www.onr.navy.mil/02/baa/baa98008.htm, February 1998.

Riedel99: Riedel, Jeffrey Scott, "Seaway learning and motion compensation in shallow waters for small auvs", PhD Thesis, NPS, June 1999.

Sun96: Sun Microsystems Corporation, *Java* language Web page, October 1996. Available at *http://java.sun.com/*

Slotine85: Dana R. Yoerger, J.J. Slotine, IEEE Journal of Oceanic Engineering, VOL. OE-10 NO.4, October 1985

Slotine86: Dana R. Yoerger, James B. Newman, J.J. Slotine, IEEE Journal of Oceanic Engineering, VOL. OE-11 NO.3, July 1986

Surge Motion Equation:

$$m[\dot{u} - vr + wq - x_G(q^2 + r^2) + y_G(pq - \dot{r}) + z_G(pr + \dot{q})] - Z_{\dot{w}} wq + Y_{\dot{v}} vr$$
$$= X_{pp} p^2 + X_{qq} q^2 + X_{rr} r^2 + X_{pr} pr + X_{\dot{u}} \dot{u} + X_{wq} wq + X_{vp} vp + X_{vr} vr$$
$$+ uq(X_{q\delta_{bp}} \delta_{bp} + X_{q\delta_{sp}} \delta_{sp}) + ur(X_{r\delta_{br}} \delta_{br} + X_{r\delta_{sr}} \delta_{sr}) + X_{vv} v^2 + X_{ww} w^2$$
$$+ u|u|(X_{\delta_r \delta_r}(\delta_{sr}^2 + \delta_{br}^2) + X_{\delta_{sp}\delta_{sp}} \delta_{sp}^2 + X_{\delta_{bp}\delta_{bp}} \delta_{bp}^2 - (W - B)\sin\theta + F_{ls} + F_{rs} - X_{res} u|u|)$$
$$- \left(\frac{W - B}{g}\right)\left[\dot{T}_{11} U_f + \dot{T}_{12} V_f + \dot{T}_{13} W_f + T_{11} \dot{U}_f + T_{12} \dot{V}_f + T_{13} \dot{W}_f\right]$$

Sway Motion Equation:

$$m[\dot{v} + ur - wp + x_G(pq + \dot{r}) - y_G(p^2 + r^2) + z_G(qr - \dot{p})] + Z_{\dot{w}} wp + X_{\dot{u}} ur$$
$$= Y_{\dot{p}} \dot{p} + Y_{\dot{r}} \dot{r} + Y_{pq} pq + Y_{qr} qr + Y_{\dot{v}} \dot{v} + Y_p up + Y_r ur + Y_{vq} vq + Y_{wp} wp + Y_{wr} wr$$
$$+ Y_v uv + Y_{vw} vw + u|u|(Y_{\delta sr} \delta_{sr} + Y_{\delta br} \delta_{br} - \frac{\rho}{2} \int_{x_{tail}}^{x_{nose}} \left[C_{dy} h(x) |(v + xr)| (v + xr)\right] dx$$
$$+ (W - B)\cos\theta \sin\phi + F_{blt} + F_{slt} - \left(\frac{W - B}{g}\right)\left[\dot{T}_{21} U_f + \dot{T}_{22} V_f + \dot{T}_{23} W_f + T_{21} \dot{U}_f + T_{22} \dot{V}_f + T_{23} \dot{W}_f\right]$$

Heave Motion Equation:

$$m[\dot{w} - uq + vp + x_G(pr - \dot{q}) + y_G(qr + \dot{p}) - z_G(p^2 + q^2)] - Y_{\dot{v}} vp + X_{\dot{u}} ur$$
$$= Z_{\dot{q}} \dot{q} + Z_{pp} p^2 + Z_{pr} pr + Z_{rr} r^2 + Z_{\dot{w}} \dot{w} + Z_q uq + Z_{vp} vp + Z_{vr} vr$$
$$+ Z_{vv} v^2 + Z_w uw + u|u|(Z_{\delta sp} \delta_{sp} + Z_{\delta bp} \delta_{bp}) - \frac{\rho}{2} \int_{x_{tail}}^{x_{nose}} \left[C_{dz} b(x) |(w - xq)| (w - xq)\right]$$
$$+ (W - B)\cos\phi\cos\theta + F_{bvt} + F_{svt} - \left(\frac{W - B}{g}\right)\left[\dot{T}_{31} U_f + \dot{T}_{32} V_f + \dot{T}_{33} W_f + T_{31} \dot{U}_f + T_{32} \dot{V}_f + T_{33} \dot{W}_f\right]$$

Roll Motion Equation:

$$I_x\dot{p} + (I_z - I_y)qr + I_{xy}(pr - \dot{q}) - I_{yz}(q^2 - r^2) - I_{xz}(pq + \dot{r}) - Z_{\dot{w}}vw + Y_{\dot{v}}vw$$
$$- N_{\dot{r}}qr + M_{\dot{q}}qr + m[y_G(\dot{w} - uq + vp) - z_G(\dot{v} + ur - wp)]$$
$$= K_{\dot{p}}\dot{p} + K_{\dot{r}}\dot{r} + K_{pq}pq + K_{qr}qr + K_{\dot{v}}\dot{v} + K_p up + K_r ur + K_{vq}vq + K_{wp}wp + K_{wr}wr + K_v uv + K_{vw}vw$$
$$+ (y_G W - y_B B)\cos\phi\cos\theta - (z_G W - z_B B)\cos\theta\sin\phi$$
$$+ \left(\frac{Wz_G - Bz_B}{g}\right)\left[\dot{T}_{21}U_f + \dot{T}_{22}V_f + \dot{T}_{23}W_f + T_{21}\dot{U}_f + T_{22}\dot{V}_f + T_{23}\dot{W}_f\right]$$
$$- \left(\frac{Wy_G - By_B}{g}\right)\left[\dot{T}_{31}U_f + \dot{T}_{32}V_f + \dot{T}_{33}W_f + T_{31}\dot{U}_f + T_{32}\dot{V}_f + T_{33}\dot{W}_f\right]$$

Pitch Motion Equation:

$$I_y\dot{q} + (I_x - I_z)pr - I_{xy}(qr + \dot{p}) + I_{yz}(pq - \dot{r}) + I_{xz}(p^2 - r^2)$$
$$- m[x_G(\dot{w} - uq + vp) - z_G(\dot{u} - vr + wq)] + Z_{\dot{w}}uw - X_{\dot{u}}uw + N_{\dot{r}}pr - K_{\dot{p}}pr$$
$$= M_{\dot{q}}\dot{q} + M_{pp}p^2 + M_{pr}pr + M_{rr}r^2 + M_{\dot{w}}\dot{w} + M_q uq + M_{vp}vp + M_{vr}vr$$
$$+ M_{vv}v^2 + M_w uw + u|u|(M_{\delta sp}\delta_{sp} + M_{\delta bp}\delta_{bp}) + \frac{\rho}{2}\int_{x_{tail}}^{x_{nose}}\left[C_{dz}\,b(x)\left|(w - xq)\right|(w - xq)\mathrm{x}\right]dx$$
$$- \left(\frac{Wz_G - Bz_B}{g}\right)\left[\dot{T}_{11}U_f + \dot{T}_{12}V_f + \dot{T}_{13}W_f + T_{11}\dot{U}_f + T_{12}\dot{V}_f + T_{13}\dot{W}_f\right]$$
$$+ \left(\frac{Wx_G - Bx_B}{g}\right)\left[\dot{T}_{31}U_f + \dot{T}_{32}V_f + \dot{T}_{33}W_f + T_{31}\dot{U}_f + T_{32}\dot{V}_f + T_{33}\dot{W}_f\right]$$

Yaw Motion Equation:

$$I_z\dot{r} + (I_y - I_x)pq - I_{xy}(p^2 - q^2) - I_{yz}(pr + \dot{q}) + I_{xz}(qr - \dot{p})$$
$$+ m[x_G(\dot{v} + ur - wp) - y_G(\dot{u} - vr + wq)] - Y_{\dot{v}}uv + X_{\dot{u}}uv - M_{\dot{q}}pq + K_{\dot{p}}pq$$
$$= N_{\dot{p}}\dot{p} + N_{\dot{r}}\dot{r} + N_{pq}pq + N_{qr}qr + N_{\dot{v}}\dot{v} + N_p up + N_r ur + N_{vq}vq + N_{wp}wp + N_{wr}wr$$
$$+ N_v uv + N_{vw}vw + u|u|(N_{\delta sr}\delta_{sr} + N_{\delta br}\delta_{br}) - \frac{\rho}{2}\int_{x_{tail}}^{x_{nose}}\left[C_{dy}\,h(x)\left|(v + xr)\right|(v + xr)\mathrm{x}\right]$$
$$+ (x_G W - x_B B)\sin\phi\cos\theta + (y_G W - y_B B)\sin\theta + x_{blt}F_{blt} + x_{slt}F_{slt} - y_{ls}F_{ls} - y_{rs}F_{rs}$$
$$+ \left(\frac{Wy_G - By_B}{g}\right)\left[\dot{T}_{11}U_f + \dot{T}_{12}V_f + \dot{T}_{13}W_f + T_{11}\dot{U}_f + T_{12}\dot{V}_f + T_{13}\dot{W}_f\right]$$
$$- \left(\frac{Wx_G - Bx_B}{g}\right)\left[\dot{T}_{21}U_f + \dot{T}_{22}V_f + \dot{T}_{23}W_f + T_{21}\dot{U}_f + T_{22}\dot{V}_f + T_{23}\dot{W}_f\right]$$