I. Troch, F. Breitenecker, eds. ISBN 978-3-901608-35-3

DEVELOPMENT OF A MULTI-RATE SIMULATION MODEL OF AN UNMANNED UNDERWATER VEHICLE FOR REAL-TIME APPLICATIONS

J.J. Zenor[1], D.J. Murray-Smith[2], E.W. McGookin[2] , R.E. Crosbie[1]
[1]California State University, Chico, CA, U.S.A.; [2]University of Glasgow, Glasgow, Scotland, U.K.

Corresponding Author: D.J. Murray-Smith, Department of Electronics & Electrical Engineering, University of Glasgow, Glasgow G12 8QQ, Scotland, U.K., djms@elec.gla.ac.uk

Abstract. This paper describes work involving the further development and refinement of a mathematical model of an unmanned underwater vehicle (UUV), together with the development of an associated real-time multi-rate simulation that includes both high-speed power electronic sub-systems and slower components. The chosen vehicle uses a battery as its energy source and this feeds an a.c. motor drive through a d.c. to a.c. converter. The drive powers the vessel which is modelled as a six-degree of freedom vehicle with control surfaces. Tests carried out indicate that a careful choice of frame rates can increase the speed of solution by factors of several hundred over solution times when the shortest frame rate is used throughout. These multi-rate solutions were executed faster than real time on a typical laptop even when using 3-D graphical output for visualisation of vehicle motion. The conclusions of the paper are that the modelling and simulation of the UUV has provided a useful test-bed for ideas on multi-rate simulation and has demonstrated that multi-rate real-time simulation is feasible and useful for an application of this kind that includes very fast power electronic subsystems and relatively slow systems such as the vehicle and battery.

1 Introduction

A propulsion system for an unmanned underwater vehicle (UUV) involving an electrical drive system presents many interesting design challenges and involves electronic, electrical, mechanical, thermal and fluid dynamic sub-systems. As with any other complex system, analysis of the behaviour of a system of this kind can be greatly facilitated by the use of computer simulation techniques. However, in applications of this kind, simulation-based investigations of the complete system can be computationally very demanding and investigations involving multi-run optimisation studies or real-time simulation can present difficulties.

The conventional approach to simulation of a system of this kind involves the use of a single integration algorithm for all the sub-systems, using an integration step length small enough to produce results with acceptable accuracy for the subsystems with the greatest bandwidth and highest switching frequency. However, the step size chosen to provide adequate accuracy in the high bandwidth components may well be very much smaller than the minimum step length required for the slower sub-systems and the simulation may perform in a very inefficient and slow fashion. In most cases the approach chosen to overcome these problems involves the use of an error-controlled variable-step integration algorithm. Such algorithms make an estimate of the truncation error arising in each integration step and then automatically adjust the set length so that it is appropriate for the dynamics at all times. This is achieved as follows:

1) if the estimated error is larger than a pre-set tolerance the step is rejected and replaced by a shorter step, or
2) if the estimate is within the tolerance bounds the results of the step are accepted and the simulation continues using that step length, or
3) if the estimated error is much smaller than the pre-set error tolerance the results of the step are step is accepted and increase the step length in the next integration interval.

In addition, when a model includes switches that require the simulation of very fast changes, an additional feature often is included within the integration algorithm to detect switching events and establish the time (to within a user-defined tolerance) at which each event occurs. The integration step is then adjusted so that one step ends when the discontinuity occurs and the next step starts immediately after the event. The main benefit of this approach is increased simulation accuracy at the cost, inevitably, of additional computer time.

However, in the case of real-time simulation applications the use of variable step-length integration algorithms and discontinuity detection methods is not appropriate. One possible approach is to sub-divide the complete system model into sub-models having different dynamic ranges. Different time steps can then be used in different parts of the simulation model, giving a simulation which is capable of faster execution because the total number of calculations is significantly smaller. This approach is known as *multiple frame rate* or *multi-rate simulation* and can be applied with many different types of integration algorithm. Although this approach is

recognised as offering benefits in terms of reduced computational demands it also raises important issues in terms of the overall accuracy and stability of the simulation.

The task of modelling the power electronic and electrical drive system for an unmanned underwater vehicle (UUV) in real time, or in a time-scale faster than real-time, provides a useful basis for the investigation of general issues arising with multi-rate simulation techniques. Evaluation of the effects of the propeller, control surface and actuator sub-systems on the overall performance capabilities of the vehicle may well require detailed modelling of electrical motors and associated converters coupled to an accurate representation of the relevant actuators and a nonlinear model of the vessel itself. This combination of relatively fast events in the power electronic components and much slower dynamics in the vessel itself raises immediately many of the issues discussed above.

Although multi-rate simulation has been a familiar technique for many years and is featured in a number of commercial simulation software packages, this application is notable because of the combination of a real-time application and the need for very short frame times to handle part of the system. The non-real-time version of this simulation, which has been developed in preparation for the full real-time version, also features use of the Virtual Test Bed (VTB) [1] and its associated 3-D animated graphics output in a faster-than-real-time implementation on a conventional laptop.

2 The underwater vehicle model

The nonlinear equations of motion of an underwater vehicle is commonly represented either in the body-fixed or earth-fixed frames of reference, as described in standard texts on modelling of ocean vehicles (e.g., [2]). Using the notation adopted by Fossen [2] the general body-fixed vector representation involves the equations:

$$\boldsymbol{M}\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{v})\boldsymbol{v} + \boldsymbol{D}(\boldsymbol{v})\boldsymbol{v} + \boldsymbol{g}(\boldsymbol{\eta}) = \boldsymbol{\tau} \tag{1}$$

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{v} \tag{2}$$

In this representation the matrix $\boldsymbol{M}$ is the inertia matrix and includes added mass effects while the matrix $\boldsymbol{C}(\boldsymbol{v})$ is the matrix of Coriolis and centripetal terms and also includes added mass effects. The matrix $\boldsymbol{D}(\boldsymbol{v})$ is the damping matrix, the vector $\boldsymbol{g}(\boldsymbol{\eta})$ is the vector of gravitational forces and moments and $\boldsymbol{\tau}$ is the vector of external forces and moments. The matrix $\boldsymbol{J}(\boldsymbol{\eta})$ is the transformation matrix relating body-fixed and earth-fixed coordinate systems. In these equations the body-fixed frame has components of motion given by the six velocity components as defined by the vector $\boldsymbol{v}(t) = \left[u(t), v(t), w(t), p(t), q(t), r(t)\right]^T$ relative to a constant velocity coordinate frame moving with the ocean current velocity vector $\boldsymbol{u}_c$. The six components in the global reference frame are given by the vector $\boldsymbol{\eta}(t) = \left[x(t), y(t), z(t), \phi(t), \theta(t), \psi(t)\right]^T$. The angles $\phi(t)$, $\theta(t)$ and $\psi(t)$ are related through the Euler transformations to the body yaw, pitch and roll motions. Individual components of the vectors are defined in the list of symbols.

The inputs that generate the external forces and moments required to control the vessel arise from control surface deflections at the rudder $\left(\delta_r(t)\right)$, port bow plane $\left(\delta_{bp}(t)\right)$ and starboard bow plane $\left(\delta_{bs}(t)\right)$, the stern plane $\left(\delta_s(t)\right)$, together with inputs arising from propeller rotational rate $\left(n(t)\right)$ and buoyancy adjustment $\left(B(t)\right)$.

From the above equations it is possible to derive a set of six nonlinear equations for surge, sway, heave, roll pitch and yaw motion [2]. For the purposes of the investigation described in this paper the six degrees of freedom equations of motion of an existing underwater vehicle (the U.S. Naval Postgraduate School (NPS) AUV II) were used. The specific equations and parameters relating to the NPS AUV II vehicle upon which the work described in this paper is based are those given by Fossen [2], which are, in turn, based on information provided by Healey and Lienard in their paper [3].

The model of the UUV is converted into standard state space form by rearranging Equations (1) and (2) in the following manner:

$$\begin{bmatrix} \dot{\boldsymbol{\eta}} \\ \dot{\boldsymbol{v}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{v} \\ \boldsymbol{M}^{-1}\{-(\boldsymbol{C}(\boldsymbol{v}) - \boldsymbol{D}(\boldsymbol{v}))\boldsymbol{v} - \boldsymbol{g}(\boldsymbol{\eta}) + \boldsymbol{\tau}\} \end{bmatrix} \tag{3}$$

This is the standard state space form for a nonlinear system i.e.

$$\dot{\boldsymbol{x}} = \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{u}) \tag{4}$$

Where $\boldsymbol{x}$ is the state vector and $\boldsymbol{u}$ is the input vector.

Some important changes had to be introduced to the model as described by Healey and Lienard [3] in order to get the simulation model to behave in a credible fashion for the range of open-loop conditions and manoeuvres needed in the proposed real-time simulation application. Modifications were made to the equations representing the propeller in order to allow the model to function correctly as the propeller speed dropped to zero. This involved adopting an approach, first suggested by Fossen [2], in which the original representation of the thrust generated by the propellers was replaced by a quadratic quasi-steady thrust equation relating the propeller speed and the surge velocity of the vehicle. The modifications made to the model also ensured that an undesirable situation involving a divide-by-zero condition that could occur with the original representation with zero initial propeller speed was eliminated.

The main modification involved replacing the thrust equation within the existing model. First it was assumed that the UUV produces a single propulsive force, although it has two propellers. The resulting thrust produced by the propeller is calculated from the *Bilinear Thruster Model* [2] as:

$$T = T_{nn}|n|n + T_{nu}|n|u \tag{5}$$

The values for the T_{nn} and T_{nu} coefficients can be obtained from the following equations [2]:

$$T_{nn} = \rho D^4 \alpha_1 \tag{6}$$

$$T_{nu} = \rho D^3 \alpha_2 \tag{7}$$

$$\alpha_1 = 0.12 - 0.5\alpha_2 \tag{8}$$

Here D is the diameter of the propellers, which is given as 30cm. For this application the value for α_2 is chosen to be -0.16, which gives a value for α_1 of 0.019. These values provide the required surge velocity profile for the NPS AUV II. Combining Equations (5) to (8) gives the ideal total thrust produced by the propeller. However, the efficiency of the propeller is less than 100% and in this case the efficiency has been chosen to be 70%. Therefore, the surge force produced by the propulsion system is found to be:

$$X_{prop} = 0.7T \tag{9}$$

The other dynamic equations do not have propulsion components since the propeller only produces thrust along the longitudinal axis. Also, the rotational effects of the propellers in roll and yaw are assumed to be negligible since the propellers operate together in a counter-balancing manner.

As well as modelling the propeller in terms of thrust production, this model has an element that approximates the load on the motor shaft caused by the propeller rotating in the water. This approximation is based upon the propeller being considered as a rotating disc. By applying Newton's Second Law, the moments of the propeller can be considered thus:

$$\sum T_o = I_o \dot{n} \tag{10}$$

Here T_O represents the torques acting on the propeller, I_O is its moment of inertia and n is the rotation speed of the propeller. By considering the total number of moments acting on the propeller, we can derive the equation of motion for the load exerted on the shaft. Firstly there is the lateral drag moment caused by the rotation of the propeller. The drag force generated by the rotating propeller can be calculated using the following standard drag equation [4]:

$$F_{DRAG} = -\frac{1}{2}\rho C_D (n \times R_{DISC})^2 \tag{11}$$

Here R_{DISC} is the radius of the disc, C_D is the drag coefficient and ρ is the density of water. To calculate the drag moment, the drag force is multiplied by the moment arm of the propeller (i.e., its radius):

$$T_{DRAG} = F_{DRAG} R_{DISC} = -\frac{1}{2}\rho C_D n^2 {R_{DISC}}^3 \tag{12}$$

Note that the negative sign indicates that the moment is acting in opposition to the rotation of the propeller.

The second moment is the input torque applied by the shaft, T_{SHAFT}. This is the necessary torque needed to be applied to the propeller to make it overcome its drag and inertia components. Both this moment and the drag moment can be combined to give the equation of motion for the propeller:

$$T_{SHAFT} = I_o \dot{n} + \frac{1}{2} \rho C_D n^2 R_{DISC}{}^3 \tag{13}$$

It follows that since the input moment needs to overcome the drag and inertia of the propeller, that it is equal in amplitude to the load on the propeller but acting in the opposite direction. Therefore, the load moment on the propeller, T_{LOAD}, is calculated as:

$$T_{LOAD} = -T_{SHAFT} = -I_o \dot{n} - \frac{1}{2} \rho C_D n^2 R_{DISC}{}^3 \tag{14}$$

This gives the approximate load on the shaft. The only parameter that is not readily available in this expression is the drag coefficient, C_D , However it is known that a disc rotating about its longitudinal axis has a drag coefficient equal to 1.369×10^{-3} [4] and this value is used in the model. It should be noted that this representation does not take into account added mass and the dynamics of the shaft.

3 The power-electronic and motor drive system model

The vehicle model described in Section 2 was combined with a model of an electrical drive system involving a battery connected to a d.c. to a.c. inverter, consisting of a three-phase six switch network producing a variable-frequency a.c. waveform and an induction motor. The controlled switches in the inverter are operated by on-off commands from a pulse-width modulated controller switching at a frequency of 5kHz. The controller uses a form of proportional plus integral (PI) control for the timing of the on-off pulses supplied to the converter switches to maintain a desired current level in the motor. Switch timings are determined using a well-established approach involving comparison of sinusoidal and triangular waves. The relative amplitudes of these waveforms are adjusted by the feedback control system and switching occurs when the sine and triangular waves intersect. The a.c. output from the converter is filtered to remove high-frequency harmonics and is then supplied as input to the induction motor. The motor is connected directly to the propeller. This electrical sub-system has also been described in a number of previous publications (e.g., , [6], [8])

For full mission simulation capability the model must thus be capable of representing accurately the high speed power electronic subsystem, with phenomena involving time intervals of less than 10 μs, over periods of time of the order of seconds, minutes or longer. In some applications it may be important that the simulation model be capable of running in real time or faster than real time. From the brief description given above, the system is seen to divide naturally into sub-systems that may involve different ranges of frame time for simulation. The convertor is a fast sub-system compared with the dynamics of the vessel and the battery, which can undoubtedly both be classified as slow sub-systems. Between these extremes the controller and motor may be regarded as slow-medium and fast-medium sub-systems respectively. Figure 1 is a schematic diagram of the complete underwater vehicle system showing the interactions between these different sub-systems.

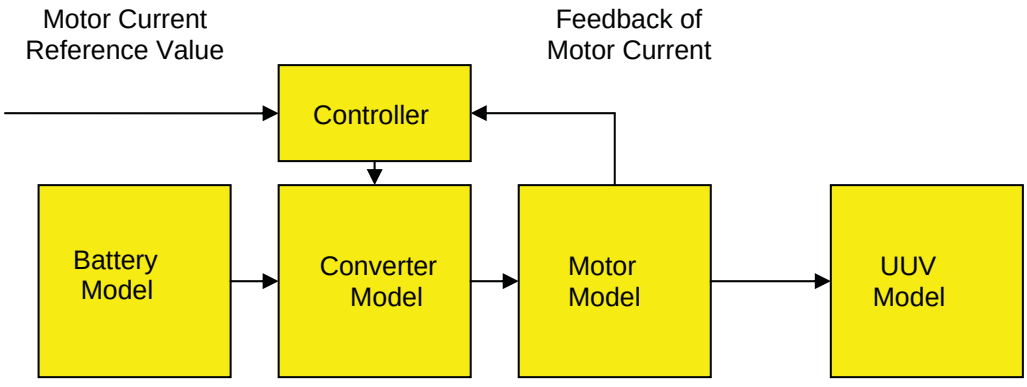

Figure 1: Conventional block diagram representation of UUV model with electrical drive system

4 Component connections in VTB

The VTB system allows component interconnection without the need for major changes in the coding of those components. This achieved using what are termed "natural" couplings in place of the conventional "signal" connections that are used in traditional continuous system simulation packages. A natural coupling is neither an input nor an output but is a form of connection in which values at each end of the connection must simply be consistent. In the case of an electrical circuit application, for example, the voltage at any connection point is the same for all connected components and the sum of all the currents at that point must be zero. This is achieved by providing as output the equation that relates the current to the voltage at the connection point, rather than simply outputting a specific value of current given the voltage at that point, as would be the case for a conventional signal connection. An external VTB solver then determines what the voltage would have to be to ensure that the sum of currents from all the connected component models is zero. The equation defining this has the form:

$$\boldsymbol{I} = \boldsymbol{G}\boldsymbol{V} - \boldsymbol{b} \tag{15}$$

Each model is called at time t to output the values of $\boldsymbol{G}$ and $\boldsymbol{b}$ for time $t + \Delta t$. The computation of the elements of $\boldsymbol{G}$ and $\boldsymbol{b}$ involves algebraic manipulation of the difference equations that result form the application of an implicit integration method to the differential equation of the model. Illustrative examples of this procedure may be found in the VTB model development manual [7] and in publications describing the development of the VTB software (e.g., [1]).

A further important feature of the VTB is the provision for "layered models" which allows several versions of a simulation component to be packaged within a single VTB component. Separate icons may be used for each version, with each having different connections.

5 Multi-rate simulation

5.1 Fundamentals of multi-rate simulation techniques

The reason for using multi-rate simulation methods is the fact that this approach can reduce significantly the time for execution of a simulation. This approach was used extensively in the early days of simulation using digital computers but has become less widely used as computers have become more powerful. However the complexity of some present day engineering applications of simulation, such as those arising in the design and development of electric ship systems, makes these techniques potentially attractive, especially in the context of real-time applications or multiple-run simulation studies for design optimisation.

An integration step length of h seconds gives a frame rate of $f = \dfrac{1}{h}$ frames per second. In multi-rate simulation it is assumed that integration step lengths $h_1, h_2, h_3, \ldots\ldots$ (where h_1 is the smallest) can be chosen so that the ratio of successive step length values

$$r_{ij} = \frac{h_i}{h_j} \tag{16}$$

is an integer.

Figure 2 illustrates the simplest case of multi-rate simulation with two frame rates corresponding to integration step lengths h_1 and h_2 where h_2 is related to h_1 through an integer N according to the equation:

$$h_2 = Nh_1 \tag{17}$$

Thus Segment 1 of the model, as illustrated in Figure 2, produces results at a rate of

$$R_1 = \frac{1}{h_1} \tag{18}$$

while Segment 2 produces results at the slower rate of

$$R_2 = \frac{1}{h_2} = NR_1 \tag{19}$$

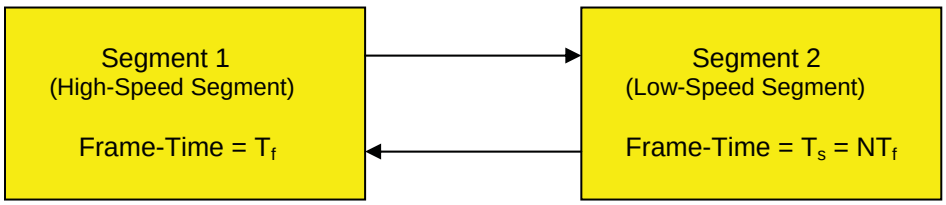

Figure 2: Block diagram illustrating a two-rate simulation with the simulation partitioned into two parts which use different frame times. The slow frame time is an integer multiple of the fast frame time. Communication takes place at the slower frame rate.

There are several methods that can be used for communicating results between segments. In the case of the two-segment case considered here, the simplest of these involves use of a zero-order hold for transferring data from Segment 2 to Segment 1. Thus Segment 1 uses the same last value received from Segment 2 for *N* steps until the value from Segment 2 is updated. Values of the output from Segment 1 for communication to Segment 2 have to be averaged in some way over the last *N* steps of Segment 1 before transfer. Simple filtering can often provide a natural and convenient form of averaging. An alternative, but more complex, approach to inter-segment communication involves use of a first-order hold involving linear extrapolation to estimate input values for each of the fast steps. An approach which is very similar, but which avoids the use of the first-order hold, can be applied for cases where the value of the derivative of a variable is available as well as the variable itself. The basis of the method is to calculate both quantities within the slow segment and pass them to the fast segment to allow estimation of the intermediate values using the derivative information. A further possibility is to stagger the timing of the segments by delaying the slow segment by a time $h_2/2$ so that the slow segment passes values to the faster segment at the mid-point of its cycle of *N* steps. If there is a significant difference in the step sizes anti-aliasing filters may be required.

5.2 The UUV application

The model of the UUV has been partitioned into sub-systems, as discussed in Section 3 and shown in Figure 1. There is a fast component consisting of the model of the d.c. to a.c. converter which requires step sizes as small as 3 μs, a slow-medium speed component involving the feedback controller, a fast-medium speed component which is electric motor model and a slow component representing the battery, the vehicle and its control surfaces and also involving the interface graphics. The simulation frame rates used in these four areas correspond to integration periods of 2μs for the converter, 100μs for the motor, 800μs for the feedback controller and 100ms for the battery, vessel and graphical output.

The simulation has been implemented using the Virtual Test Bed (VTB) together with the VXE graphics software which can be used to display graphical and 3-D animations of model behaviour. Both of these software tools were developed at the University of South Carolina [1],[7]. The main benefit of this approach is that the VTB provides a flexible simulation environment which allows sub-system models which have been developed using different simulation tools to be combined. The VTB also incorporates a library which contains a large range of mechanical and electrical components.

Three different programming approaches have been used in the development of the sub-system models because models had been developed by different groups in different locations. The converter, controller and motor sub-models have been programmed using C++ code whereas the model of the vessel is written in Matlab and the d.c. power source is represented using native VTB battery models. Modules exist in the VTB environment to allow interfacing to many other simulation environments but VTB natural couplings, as described in Section 4, are not directly compatible with traditional signal connections.

Although the converter model is coded using C++, it is implemented as a native VTB simulation model. In this sub-model the d.c. input connection and the a.c. output connections are natural couplings while the connections to the controller are signal couplings. The converter simulation involves Euler integration for the input d.c. filter capacitor and trapezoidal integration for the filter components at the output. The sine and triangular waveforms of the controller are generated by table lookup with the table entries being scanned at a rate determined by the desired frequency. Linear interpolation is employed with sufficient table entries to ensure a maximum error of one bit. The “layered model” feature of the VTB means that a stand-alone model of the converter can exist

alongside a combination system of converter, controller and motor with the icon and executed code being changed by an input parameter. This type of facility has been found to be very useful, especially for fault-finding during the development of the model.

The six degree of freedom simulation model of the vessel was implemented originally in Matlab and involves the use of a fourth-order Runge-Kutta integration algorithm. Fin deflections are applied through the user interface and the propeller shaft RPM is a signal input. The Matlab simulation was translated to C++ and implemented as a native VTB model although the original Matlab model may still be used through the VTB/Matlab interface.

The motor simulation was also implemented in C++ as a native VTB simulation using trapezoidal integration. The a.c. input involves a natural connection but the output connections for the motor model, in terms of the torque and shaft RPM variables, had to be compatible with the model of the vessel. Thus the original VTB model equations for the motor model were re-implemented to have signal connections for the mechanical variables.

Two implementations of the multi-rate simulation have been under development. One of these approaches is based upon a distributed multi-rate solver that forms an integral part of the VTB environment. In this approach the system is partitioned in such a way that each partition contains models that have to be run at the same time step. Connections between the partitions terminate in special VTB "ports". Each partition has to be distributed to a separately running copy of the VTB, connected using an Ethernet network.

Since the VTB distributed solver was not available when the work started, a less-flexible but simpler, special-purpose, multi-rate solver was developed for initial development and testing of the multi-rate approach. This second "Chico" solver approach involves bundling together the models to be run at different rates within a single VTB "super-model". The VTB then runs at the rate of the slowest model and other models run with a step size that is an integral divisor of the VTB step size. At each VTB time step the complete bundle of models is called by the VTB and an internal scheduler calls the internal models at the appropriate rate, returning values to the VTB when the elapsed time corresponds to the start of the next VTB step. Natural couplings are handled by a special-purpose internal solver.

Apart from its immediate availability an advantage of the "Chico" solver is a slightly lower computational overhead than that of the general-purpose VTB distributed solver. The main disadvantage is the fact that results could only be displayed at the user interface at the VTB time step.

Other work carried out to support the development of the multi-rate simulation has involved the use of an algebraic manipulation program based on Mathematica. The program takes the differential equations for the model, a list of program names and the corresponding mathematical symbols and a definition of the implicit integation algorithm to be used (e.g., the trapezoidal rule) and computes the difference equations, the equations for $\boldsymbol{G}$ and $\boldsymbol{b}$ and outputs the C++ code for insertion into a VTB program for communicating the values of $\boldsymbol{G}$ and $\boldsymbol{b}$ at each time step. Mathematica could be copied directly into the VTB simulation to implement the changes in a very simple and reliable way.

5.3 Results and verification of the multi-rate simulation

The models were not constrained to real-time execution and faster than real-time execution has been achieved on a typical laptop computer even when using the 3-D graphical output capability provided with the VXE. Work was also carried out to develop non-real-time simulations of the system to allow detailed investigation of the effectiveness of the multi-rate approach, consider detailed issues of frame rates for different sub-systems and prepare the ground for real-time implementation.

The European Simulation Language (ESL) [9] was used to make comparisons of results obtained using multi-rate simulations with conventional simulation results. ESL, which was developed at the University of Salford for the European Space Agency, incorporates a parallel segment feature that supports multi-rate simulations. One of the important features of ESL is that it can automatically locate switching points, accurately integrate up to the discontinuity and then continue form that exact point. This discontinuity detector allows investigation of the effects of taking fixed time steps in a fast multi-rate simulation and the switching errors introduced for different step lengths. ESL has recently been integrated into the VTB system [10].

6. Discussion and conclusions

The UUV model of Healey and Lienard [3] proved to be a useful starting point in the development of the multi-rate simulation involving the combined model of the vehicle and the electrical drive system. Modifications required in the model of the vehicle were due to problems observed in the behaviour of the model at low values of forward speed and when starting from rest. These changes in the model, which were successfully implemented in the multi-rate simulation studies, are almost certainly due to the fact that the published model equations were used originally for research involving the development of nonlinear control systems for this underwater vehicle for manoeuvres involving some specific ranges of forward speed. The observation of some anomalous behaviour

in applying the same model for open-loop and manual control investigations over a wider range of speed is perhaps not entirely surprising and emphasises the difficulties likely to be encountered when a model developed for one type application is used for an entirely different type of investigation.

The conclusions of the paper are that the modelling and simulation of the UUV has provided a useful test-bed for ideas on multi-rate simulation and has demonstrated that multi-rate real-time simulation is feasible for an application of this kind that includes very fast power electronic subsystems and relatively slow systems such as the vehicle and the battery.. The VTB software has been a unifying tool in this development work and recent work on the development of an interface between ESL and the VTB makes ESL very attractive as a development tool in work of this kind.

Acknowledgements. The authors acknowledge the financial support from the U.S. Office of Naval Research through Awards N00014-01-1-0394, N00014-04-1-0373, N00014-05-1-0534 and N00014-08-1-0687.. They also recognise valuable contributions by Professor Roger Dougal and Antonello Monti and others in the VTB team at the University of South Carolina, and by Dr. John Pearce of ISIM International Simulation Ltd. Contributions made to the development of the simulation by a number of Chico students, including Sourabh Bhalerao and Robert Powelson, are also acknowledged.

References

[1] Dougal, R. A. “Design Tools for Electric Ship Systems”, In *Proceedings 2005 IEEE Electric Ship Technologies Symposium, Philadelphia, PA, July 2005*, IEEE, pp 8-11, 2005.

[2] Fossen, T. I., *Guidance and Control of Ocean Vehicles*, John Wiley & Sons Ltd, Chichester, 1994.

[3] Healey, A. J. and Lienard, D., "Multivariable Sliding Mode Control for Autonomous Diving and Steering of Unmanned Underwater Vehicles", *IEEE Journal of Oceanic Engineering*, Vol. 18, No. 3, pp 327-339, 1993.

[4] Hoerner, S.F., *Fluid Dynamic Drag*, Hoerner Publications, New York, 1968.

[5] Word, D., Zenor, J.J., Bednar, R., Crosbie, R.E. and Hingorani, N.G. “Multi-Rate Real-Time Simulation Techniques”. In *Proceedings of 2007 Summer Simulation Multiconference, San Diego, CA, July 2007*, SCS San Diego, USA, 2007.

[6] Zenor, J.J., Bednar, R., Sourabh Bhalerao, “Multi-Party, Multi-Rate Simulation of an Unmanned Underwater Vehicle”. In *Proceedings of 2007 Summer Simulation Multiconference, Edinburgh, UK, June 2008*, SCS San Diego, USA, 2008.

[7] The University of South Carolina, *The Virtual Test Bed* [Internet]:<http://vtb.engr.sc.edu/> [Accessed 23 December 2008]

[8] Word, D,, R. Bednar, J. J. Zenor, and N. G. Hingorani, “High-Speed Real-Time Simulation for Power Electronic Systems”*, Simulation, Vol.* 84., pp 441-456, 2008.

[9] Zenor, J.J., Pearce, J.G. and Bednar, R., “Testing of Multi-Rate Simulations Using the ESL Simulation language”, in *Proceedings of 2007 Summer Simulation Multiconference, San Diego, CA, July 2007*, SCS San Diego, USA, 2007.

[10] Pearce, J.G., “Interfacing the ESL Simulation Language to the Virtual Test Bed”, in *Proceedings of 2007 Western Simulation Multiconference, San Diego, CA, January 2007*, SCS San Diego, USA, 2007.