Neural Network Progress at NSWCCD

David E. Hess, Naval Surface Warfare Center, Bethesda/USA, david.e.hess@navy.mil
William E. Faller, Applied Sim. Technologies, Cocoa Beach/USA, faller@appliedsimulation.com
Kurt Junghans, Naval Surface Warfare Center, Bethesda/USA, kurt.junghans@navy.mil
Joan L. Lewis, Naval Surface Warfare Center, Bethesda/USA, joan.l.lewis@navy.mil
Edward S. Ammeen, Naval Surface Warfare Center, Bethesda/USA, edward.ammeen@navy.mil

Abstract

We report on the current progress of three ongoing efforts at the Naval Surface Warfare Center, Carderock Division that have been funded by the U.S. Office of Naval Research. The first topic considers the use of a recursive neural network as a virtual sensor to provide real time analytic redundancy of sensor readings provided to the automatic control system of a submarine. Next, we discuss the use of feedforward neural networks to correct for differences between predictions produced by our standard submarine simulation and experimental maneuvering data. The final topic concerns the use of feedforward networks to model the six forces and moments acting on the hull of a 1/12th scale captive model of an air-cushioned vehicle.

1. Introduction

The application of neural networks to problems of naval interest remains an active area of research at the Naval Surface Warfare Center, Carderock Division (NSWCCD). The research is conducted within the Maneuvering and Control Division (MCD) and has been applied to the motion of submarines, Roddy et al. (2008b), Junghans et al. (2008), and Hess and Faller (2002); surface ships, Faller et al. (2008) and Hess et al. (2007); an air-cushioned vehicle, Hess et al. (2009); and to the predictions of forces and moments acting on propellers Roddy et al. (2008a) and Roddy et al. (2006). We will discuss the current progress of several ongoing efforts that have been funded by the U.S. Office of Naval Research (ONR). The first project employs a recursive neural network (RNN) as a virtual sensor to check sensor inputs to a submarine control system.

Advanced Automatic Control and Fault Detection systems are being developed for Navy submarines and surface ships. A critical concern is how the vehicle control system will respond to such factors as: the environment, damage that degrades vehicle performance, and unexpected sensor, actuator, or control surface failures. Such an occurrence may well lead to an automatic control system response that is either inadequate or inappropriate given the current state of the vehicle. To maintain mission effectiveness, changes in the vehicle dynamics, as well as component failures, must be rapidly detected and recovery actions must be promptly initiated within the automatic control loop. The current work makes use of a recursive neural network (RNN) as a virtual sensor to provide real-time analytic redundancy of the sensor readings provided to the automatic control system. Because sensor failures feed directly into the automatic control system, it is critical to check and verify the sensor readings prior to use in the automatic control loop since all subsequent commands are predicating on the assumption that the sensed information is correct. The virtual sensor system discussed here checks each measured sensor reading against real-time RNN simulation predictions of the expected sensor values, and a decision is made as to the validity of the measurement. The sensed values are either passed on as correct or flagged as being in error and an estimate of the true sensor reading values are provided to the automatic control system. Results are presented that demonstrate that typical sensor failure modes: sensor drift, sensor lock-up, sensor drop-out, sensor data spikes, and sensor noise, can be detected and corrected analytically using this approach with zero false positives. The next topic is called the Next Generation Tool.

This refers to a project designed to improve the Maneuvering and Control Simulation (MCSIM). MCSIM is the primary predictive tool in use by MCD for developing submarine hydrodynamic
products and responding to fleet needs. This necessitates a code that provides rapid solutions; hence, it is a potential flow code that has been augmented by adding some viscous effects and by tracking vorticity. While this code, in general, provides excellent results, there are instances where maneuvering predictions do not match well with submarine model or full scale behavior. To provide an improved tool under these circumstances, feedforward neural networks (FFNN) are used to estimate the force and moment residuals that, when added to the MCSIM output at each time step, correct for any differences between MCSIM and free-running model (FRM) or full scale submarine maneuvers. To test the feasibility of this approach, MCSIM predictions were prepared for maneuvers of ONR Body 1, which is an unclassified submarine FRM previously developed with ONR funding. After defining mutual FFNN and MCSIM requirements, FFNN solutions were developed for the residuals. Results are presented which sum the MCSIM predictions with the FFNN residuals and compare with ONR Body 1 experimental data. The results demonstrate that some of the maneuvers were corrected almost perfectly, while there were some others that displayed positive feedback in the errors. Clearly, further work is required, but the early conclusion is that feedforward neural networks models can accurately predict the moment residuals and that the solutions are valid across all maneuver types that were attempted (horizontal and vertical overshoots and turns). The approach appears to be feasible.

The final topic concerns the development of a six degree-of-freedom simulation of a free-running model of the U.S. Navy’s Landing Craft Air Cushion (LCAC) vehicle. This vehicle is a high-speed air-cushioned design with the mission to transport materiel from off-shore vessels and then onto the beach. The goal is to demonstrate a recursive neural network software platform for the simulation of LCAC maneuvering in calm water and in waves. The primary training data for the simulation will come from testing of a 1/6th scale free-running model which was acquired in March 2009 by the Naval Surface Warfare Center, Panama City Division in Florida. This simulation approach requires information on the input forces and moments acting on the vehicle. While previous experience modeling the motion of submarines and ships is of great utility for the overall design of the simulation, it provides no information on cushion dynamics, a key component of the vehicle motion. Captive model data from experiments with a 1/12th scale model of LCAC towed in calm water were available, and these data were used to train feedforward neural network models. The networks were used to model the six forces and moments acting on the hull of the vehicle as a function of available input data measured during the experiment. Each network is used to predict one force or moment component as its sole output. The intent is to use these models as input quantities to the larger RNN simulation effort in order to provide a well-posed problem. Some of these results will be presented and the current status of the overall effort will be reviewed.

2. Neural Network Virtual Sensors

An opportunity exists for significant improvements in such areas as the simulation, automatic control and fault monitoring of highly nonlinear platforms. Existing and future vehicles could realize significant increases in operational capability if the existing operating envelopes could be safely extended, and if failure modes could be detected and compensated for in real-time. Accordingly, techniques are being developed to augment and enhance the traditional automatic control loops for both submarines and surface ships. The advanced control systems being developed sense any unexpected sensor, actuator, or control surface failures and respond accordingly to any changes and/or damage that may degrade the vehicle performance. In order to maintain mission effectiveness, changes in the vehicle dynamics, as well as component failures must be rapidly detected, in real-time, and recovery actions must be promptly initiated within the automatic control loop. The overall system consists of combining three state-of-the-art techniques—Robust/Reconfigurable Control, Recursive Neural Networks (RNN), and Fault Detection and Isolation (FDI) algorithms—into a real-time Advanced Control & Monitoring (ACM) system that provides vehicle monitoring, fault protection, and automatic control of the vehicle from within the executive control loop. This section will concentrate on the virtual sensor (fault detection) component of the overall system.

The model-based component of the fault protection system models the vehicle using a recursive
neural network in order to provide a faster than real-time modeling and simulation capability for the vehicle reference model. RNNs are a revolutionary approach to nonlinear simulation that has been successfully utilized for the simulation of U.S. Navy submarines and surface ships. The choice of RNNs for the vehicle reference model is further dictated by the following requirements for the plant model: (1) the plant model of the vehicle must be a full 6-DOF simulation; (2) the reference model should be nonlinear, not linearized, in order to reflect the highly complex, nonlinear dynamics of the vehicle; (3) the reference model must computationally run faster than real-time; and (4) the reference model must be capable of providing predictions of the vehicle trajectory and attitude several seconds to minutes into the future. An additional advantage of the RNN approach is that the reference model can be developed off-line, directly from available model scale experimental data, and these model scale RNN simulations can then be used directly within the full-scale implementation of the ACM system. This eliminates requirements to collect full-scale data for development of the plant model since the necessary model scale data exists for all Navy submarines and surface ships. Alternatively, the faster than real-time RNN based reference model can be developed directly from CFD computations, if a suitable training set can be developed in the time frame available. The unique attributes of RNN plant models make them the logical choice for use in the development of a real-time system that provides vehicle monitoring, fault protection, and control.

The experimental data was obtained from a model scale, free-running submarine, which was funded by the U.S. Office of Naval Research (ONR) and is denoted ONR Body 1. The vehicle has a length of approximately 6m with an axisymmetric hull, and the motion of the vehicle is controlled by a single centerline mounted 3-bladed propeller, upper and lower rudders, and port and starboard sternplanes. All of the appendages are NACA sections, and the sail is rigidly mounted to the hull. The propeller used to drive the model is a modification of a commercially available 3-bladed right-handed motorboat propeller. While not representing any U.S. submarine, the ONR Body 1 model was designed to present computational model developers with similar types of maneuvers from a highly agile submarine in order to challenge the simulation and design codes. Hence, it is a suitable platform for use in verifying the performance of computational tools intended to predict submarine maneuvering characteristics.

The RNN for ONR Body 1 has 127 inputs. Each hidden layer contains 48 nodes, and each of these hidden unit nodes has a bias unit. The output layer consists of 6 nodes, and does not use bias units. The input vector consists of a series of terms that implement the various forces and moments that act on the vehicle. The RNN then predicts at each time step dimensionless forms of the six state variables: three linear velocity components, \( u \), \( v \), and \( w \), and three angular rate components, \( p \), \( q \) and \( r \). These six state variable predictions are then used to compute at each time step the remaining variables: the vehicle trajectory (\( x \), \( y \) and \( z \)), the attitude (pitch, roll and heading) and the accelerations.

Fig.1: Implementation of the Virtual Sensor within the Advanced Control & Monitoring System.
The RNN based Virtual Sensor (VS) component of the ACM system is shown in Fig.1 in red. The critical issue is that any errors in the sensor readings can have an immediate and detrimental effect on the automatic control system. The real-time RNN predictions of the anticipated vehicle dynamics, and therefore the anticipated sensor readings, are compared directly to the measured values, a decision is made as to the accuracy of the sensor reading (good or bad), and corrections are made to the values returned to the automatic control system.

Within the VS implementation, it is necessary to periodically re-synchronize the RNN simulation with the submarine in order to permit extended, indefinite, periods of operation. The RNN is after all a simulation, albeit a fairly accurate one; nevertheless, all simulations drift over extended periods of time. The methodology for re-synchronizing the RNN based simulation is shown schematically in Fig.2 and discussed in more detail below. The general idea is to fix the integration constants, pitch, roll, heading and depth when these are known to be correct (during constant heading operation, for example), and to update the state variable inputs when possible. This combination permits the RNN based simulation to faithfully track correct sensor values indefinitely, and therefore be available for the comparisons that will detect a sensor failure should one occur.

The VS inputs/outputs are assumed to be at the control frequency update rate. Two files must be read by the VS when it is initialized, the weights and gains for the RNN, and these files must be accessible to the subroutine call. The control inputs to the VS are the ordered rudder angle, the ordered sternplane angle, and the measured RPM. The sensor inputs to the VS are the measured quantities: ship speed, pitch angle, heading angle, roll angle, and depth.

One output from the VS is the predicted value for ship speed. This value is assumed to always be correct! It will either be nearly identical to the measured ship value or it will be a best estimate of the true sensor reading if a sensor failure has been detected. In addition to the VS estimate of the speed, a flag for either a good or bad sensor value is returned. The flag is equal to 0 when the sensor reading is determined to be correct, or is equal to 1 if the sensor has been determined to be drifting. The flag is equal to 2 for sensor noise, or the value 3 for a sensor drop-out, or 4 for a sensor lock-up. Finally, the flag is equal to 5 for data spikes in the sensor readings. Additional outputs from the virtual sensor are the estimate of the pitch angle and the heading angle. Again, the VS values are assumed to be correct, and the error flags for these outputs are consistent with the values defined above for speed. These three virtual sensor readings are the critical inputs to the automatic control system for which software redundancy is required.

A general overview of the virtual sensor heuristics follows. On the first call to the VS all variables are initialized and all required files are read (weights and gains). The simulation assumes that the initial 4 seconds of data (at model scale: 25 Hz, 100 data points) are correct. This data is used to fill the initial
data arrays used for comparison against future sensor readings and to update the RNN VS predictions using the sensor data. After the first 4 seconds the decision is then made whether the boat is in a constant heading (CH) mode or depth keeping mode where speed, pitch, heading and depth would be expected to be relatively unchanged over time. Otherwise, the boat is assumed to be maneuvering. The heuristics for detecting sensor failures are different for CH and maneuvering. If the boat was determined to be in a constant heading mode then each of the sensors, speed, pitch and heading are tested for possible sensor failures using the following approach. If the sensor is operating within normal values the sensor reading is retained, the RNN VS integration constants and state variables are updated periodically and the flag returned indicates no sensor errors. The values of the VS sensor readings returned will match the measured sensor readings. If the sensor is not operating within normal limits then the sensor values are tested to determine if the sensor is showing drift, noise, a drop-out, a lock-up condition, or if there is a data spike in the sensor reading. If one of these conditions is detected then the returned flag will indicate the type of failure, and the returned VS sensor reading will be a best estimate of the true value of the sensor. If none of the sensor failure modes are satisfied then the default is to assume the sensor reading is correct. Again, the RNN VS will be updated periodically, the flag returned will indicate no sensor errors, and the value of the VS sensor reading returned will match the measured sensor reading. If the boat was determined to be in a maneuvering mode then each of the sensors, speed, pitch and heading are tested for possible sensor failures using the same approach with the exception that for normal sensor values only the integration constants are updated periodically.

Fig.3: Baseline horizontal overshoot maneuver, Left: controls, Right: maneuver variables
Experimental data: Black, Sensor data: Blue, Virtual Sensor: Red.

Once the RNN simulation has been coupled with the control system, known maneuvers can be used to
test the response of the virtual sensor. Fig.3 shows a baseline maneuver, a horizontal overshoot with a small rise on the sternplanes and constant rpm. The control time histories for rudder angle, sternplane angle and propeller RPM are shown on the left hand side of the figure. The right hand side of the figure shows the data for speed, roll, pitch, depth and heading. In all plots, the actual response of the ship (with no simulated failures) is shown in black, the blue line is the sensor data (including any failures that were specified), and the red line is the output of the virtual sensor. As shown in Fig.3, for the baseline case, all three curves should be superimposed. Since there are no sensor failures, the actual response of the ship should be identical to the sensor data, which should in turn be identical to the virtual sensor. This demonstrates the accuracy of the RNN for the baseline case.

The left hand side of Fig.4 shows the same maneuver but with a simulated failure in the speed sensor. In this case, the failure introduced was drift in the sensor reading for speed. Based on historical data, this is a realistic failure scenario for a submarine. The baseline response of the ship, in the absence of any failures, is repeated in black. The speed sensor reading including the sensor failure is shown in blue, and the virtual sensor detection and estimation of the true speed is shown in red. After a small period of time is allowed to elapse to avoid reporting false errors, the virtual sensor takes over and computes its own speed estimate, based on the current control and state values, and provides it to the automatic control system. Without the virtual sensor, the automatic control system would have received the information directly from the sensor reading, the blue line in the speed plot. Clearly, with gain scheduling based on speed this would be a very undesirable occurrence. With the virtual sensor system in place, the sensor is flagged as being in error and the speed estimate provided by the RNN allows normal operations to be maintained while the error in the speed sensor is corrected. As can be
seen, the change in the velocity time history, due to the small period of drift, causes the predicted maneuver (in red) to be different from the original baseline case with no error, but the resulting maneuver is benign because the error is captured very quickly.

The right hand side of Fig.4 shows the same maneuver again but with a drop-out failure in the speed sensor. This is also a realistic failure scenario for a submarine. The baseline response of the ship, assuming no failures, is shown in black. The speed sensor reading including the sensor failure is shown in blue, and the virtual sensor detection and estimation of the true speed is shown in red. In this case, the failure is so dramatic that the virtual sensor recognizes it nearly instantaneously and responds with its own prediction of speed. Without the virtual sensor, the automatic control system would have received the information directly from the sensor reading, the blue line in the speed plot. Clearly, with gain scheduling based on speed this type of failure could result in catastrophic consequences. With the virtual sensor system in place, the sensor is flagged as being in error and the speed estimate shown in red is provided to the automatic control system. This estimate allows normal operations to be maintained while the error in the speed sensor is corrected.

The virtual sensor approach makes it possible to provide analytic redundancy of the ship sensor readings prior to utilization by the automatic control system. Consistent results were obtained across hundreds of maneuvers tested for both ONR Body1 and for a full scale vehicle. The virtual sensor system, in all cases, correctly classified the failure type, flagged the error and provided a reasonable estimate of the true sensor reading. Critically, zero false positives were returned for any of the cases tested. Since sensor failures will feed directly into the automatic control system, it is critical to check and verify the sensor readings prior to utilization in the automatic control loop since all subsequent commands are predicated on the assumption that the sensed information is correct. Results show that the typical sensor failure modes: drift, lock-up, drop-out, noise, and data spikes can be detected and corrected analytically using this approach. The work described here provides a key component of the overall ACM system under development. These real-time systems can, mean the difference between safe, continued operation and potentially catastrophic failures. Next, we describe a means for improvement in simulation results by coupling a feedforward neural network (FFNN) to a submarine maneuvering simulation; the combined system is called the Next Generation Tool.

3. Next Generation Tool

The Maneuvering and Control Division has long recognized the need to incorporate emerging capabilities into the submarine maneuvering prediction process as they become available. Therefore, MCD embarked upon a task with funding from ONR to begin the development of a Next Generation Tool (NGT) which would represent an amalgam of simulation strategies to: model new problems such as the two-body problem; to address alternative geometries such as non-axisymmetric vehicles; to elucidate difficult physics; and to further reduce simulation time for problems requiring many simulations. The approach relies on the development of a process architecture that can exploit existing, validated hydrodynamic, maneuvering, and control tools to enable the proper tools to be applied to the appropriate problem.

MCD requires an accurate submarine motion prediction code that can run faster than real time in order to support current and future Navy needs. The primary simulation tool has been the Maneuvering and Control Simulation (MCSIM), and it has served as a key link within the Division’s overall process for combining analytic methods (hydrodynamic loads predictions), experimental data (captive model, FRM and full scale), and simulation techniques (MCSIM, coefficient-based and recursive neural networks) to create a mathematical model of submarine maneuvering behavior to respond to fleet needs.

The faster than real-time MCSIM uses potential flow computations that are augmented by approximations of viscous effects in the streamwise and crossflow directions. In the streamwise direction viscous effects are well approximated by potential flow theory if the body is augmented by the displacement thickness of the boundary layer. The separated crossflow vortex wake is
approximated by thin vortex singularities in the form of point vortices. The creation and movement of these point vortex singularities within the potential flow framework comprises the evolution of the wake. Appendage force computations employ lifting line theory, and unsteady effects are captured by tracking the time history of the strength and propagation of the vorticity.

This code normally provides excellent results; however, there are instances where maneuvering predictions do not match well with submarine model or full scale behavior. Therefore, within the NGT program, feedforward neural networks (FFNN) were considered as a means to provide a correction to the simulation data such that the sum of the original prediction and the correction yields the measured vehicle behavior. Specifically, FFNNs are used to estimate the force and moment residuals that, when added to the MCSIM output at each time step, correct for any differences between MCSIM and free-running model (FRM) or full scale submarine maneuvers. A schematic diagram of the process is shown in Fig.5.

![Fig.5: Maneuvering and Control Simulation with FFNN Predicted Residuals](image)

The figure shows that at a given time step MCSIM makes a prediction of the forces and moments that are acting on the vehicle one time step into the future using control information and vehicle state information from the previous time step. Dividing by mass or moment of inertia terms allows the accelerations to be determined. Integrating provides the velocities, trajectory and attitude, and the vehicle is advanced to the next time step in the maneuver. The cycle then repeats and this is the loop shown in the top of the figure.

Study of MCSIM predictions has indicated that the dominant, required corrections are in roll, pitch and yaw moments, $\Delta K$, $\Delta M$, and $\Delta N$. Therefore, the box labeled NNs shown in Fig.5 consists of three feedforward neural networks. Each network uses a variety of inputs and computes one output:

$$\Delta p = p_{\text{RM}} - p_{\text{MCSIM}}$$

$$\Delta q = q_{\text{RM}} - q_{\text{MCSIM}}$$

$$\Delta r = r_{\text{RM}} - r_{\text{MCSIM}},$$

as shown in Fig.6. These angular accelerations in roll, pitch and yaw are related to the moment residuals by moments of inertia.

A set of maneuvers of the free-running submarine model ONR Body 1 were selected, and the simulation was exercised to create predictions for the identical maneuvers. Time histories of the residuals given in Eq.1 were computed and grouped with the various inputs. This data formed the training data for the networks. The networks were then trained offline; that is, they were removed from the loop shown in Fig.5. A large set of inputs, those shown in Fig.6 as well as various products of these, were selected to provide a rich input data set and more than enough information to pose the problem well to the networks.

The solutions converged quickly, and they were very accurate. Error measures such as correlation coefficients and average angle measures, Roddy et al. (2006), comparing the FFNN residual predictions to the target values in the training set were consistently 0.85 to 0.95 showing excellent predictions. Training was ceased, and the three neural networks were now able to recover the angular acceleration residuals over the entire set of ONR Body 1 maneuvers.
Despite the accurate residual predictions when trained offline, difficulties were encountered when the FFNNs were inserted into the prediction loop of Fig.5. For half of the set of maneuvers, the residuals were correctly predicted, and summing the residuals with the MCSIM predictions recovered the measured maneuvering data extremely well. For these maneuvers, the system was working as planned. For the other half of the maneuvers in the set, a positive feedback problem was occurring. To learn more about the nature of this problem, time series of the inputs to, and the outputs from, the networks for maneuvers that were poorly predicted were studied.

When the networks are inserted into the prediction loop, the input quantities are provided by MCSIM. The FFNNs produce the residuals corresponding to that set of inputs. However, the residuals are not perfectly correct. The residuals are added to produce the total moment predictions at that time step and these are imperfect. Integrating provides the velocities, trajectory and attitude, and the vehicle is advanced to the next time step in the maneuver, but there are small errors in all of these quantities. MCSIM uses this imperfect information to provide the next set of predictions. These predictions contain errors from two sources: the error propagated into the inputs to MCSIM, and the fact that MCSIM is itself imperfect and provides some prediction error. This error then reenters the networks.

There are a large number of inputs to the FFNNs, and this is by design. Some of these inputs are very important for producing the FFNN output, and others less so. Some of these inputs were exposed to a large dynamic range over all of the maneuvers during training, whereas others experienced a smaller dynamic range. So, when error enters these inputs, two problems are occurring. First, certain inputs may simply be driven outside of the dynamic range that they experienced during the training process. If these inputs are critical to the solution, then they are very sensitive; if they are of lesser importance to the solution, then driving them a bit out of range is not such a problem. They are less sensitive. The second, more subtle, problem is that when the inputs are combined with error, then the input vector may lie outside of the parameter space encountered during training. This is extrapolation error that is produced at the output. This error may be small or large depending upon the slope of the solution surface in the neighborhood of this input vector.

The point is that error is introduced into the FFNNs, which in turn, produces errors at the outputs. If the error grows during successive iterations, then the positive feedback problem leads to a poor prediction. An example of time series for a few of the inputs to one of the networks during a poorly predicted maneuver is shown in Fig.7. These inputs after a short time become quite different from those encountered during training, whereas other inputs (not shown) display benign behavior in the presence of error.

The solution to the positive feedback problem requires that inputs that contribute to the amplification of errors (like the two inputs shown in Fig.7) be removed (lesioned). The procedure is conservative by design. The inputs were studied over the entire set of maneuvers, and a few of the worst-case inputs
responsible for amplifying error were removed for each of the FFNNs. Then, the networks were removed from the prediction loop and retrained with the reduced set of inputs. The reduced input set remained quite large and continued to provide ample information for the FFNNs to converge to accurate solutions. The modified networks were then reinserted into the prediction loop shown in Fig. 5 and the performance of the system was reevaluated. The number of maneuvers that displayed positive feedback errors was reduced from 50% to less than 5%.

Fig. 7: Selected FFNN inputs for a poorly predicted maneuver
Blue: Input provided during offline training, Purple: Input provided within prediction loop

An example of improved predictions resulting from combined MCSIM and FFNN tools is shown in Fig. 8. This is one selected case from the set of ONR Body 1 maneuvers and represents a hard turn to starboard with rise angle on the sternplanes. The bold black line indicates the measured data from the free-running model for this maneuver. The red curve represents the initial MCSIM prediction for this difficult combined planes maneuver. This error in the simulation prediction provided the motivation
for this work. The green curve denotes an early prediction by the combined MCSIM + FFNN tool and shows how the positive feedback problem led to predictions that were worse than before for certain variables such as pitch and depth. The lesioning process described above was then performed, and the purple curve indicates how the combined MCSIM + FFNN tool predicted the maneuver. This solution quality is typical of all but the few remaining runs with the feedback issue and is the kind of improved simulation predictions that were envisioned at the outset of this task.

Work is continuing towards the realizable goal of zero maneuvers with positive feedback. Summarizing, the iterative process that produces new networks with modified inputs is not taking place in order to produce better output residual solutions. In fact, each of the versions produces excellent results with the ample and rich set of inputs that remain after lesioning. Instead, what is occurring is a systematic and orderly study of how to tailor the FFNN solutions to the particular kind of errors that are generated by the combined MCSIM + FFNN system such that the positive feedback problem is eliminated over the entire set of free-running model maneuvers.

This effort has demonstrated that the feedforward neural networks can accurately predict the moment residuals, and that the Next Generation Tool can provide accurate predictions over a varied array of difficult free-running submarine model maneuvers such as horizontal overshoots, vertical overshoots, turns and constant heading maneuvers. The task has shown that using FFNNs to provide a correction to an existing simulation is feasible. An additional goal of this effort is to study the final network solutions for the residuals in an attempt to shed light on any missing physics that may not be correctly captured by the simulation. We turn now to the final topic, which concerns the use of feedforward networks to model the six forces and moments acting on the hull of a 1/12th scale captive model of an air-cushioned vehicle.

4. Force and Moment Predictions for LCAC

The U.S. Navy’s Landing Craft Air Cushion (LCAC) vehicle, currently in operation in the fleet, is a high-speed air-cushioned vehicle which operates from off-shore well deck ships to transport materiel onto the beach. A picture of the vehicle is provided on the left of Fig.9. Work supporting research initiatives for LCAC in the areas of Advanced Lift Fan and Advanced Skirt Development is currently being conducted with funding from the Office of Naval Research. An additional need funded by this program is a six degree of freedom maneuvering simulation for LCAC. Experimental data to support the simulation development are available from captive model testing of a 1/12th scale model of the vehicle acquired in the Fall of 2007 at NSWCCD, and from testing of a 1/6th scale free-running model recently completed in March 2009.

The primary goal is to develop a fully nonlinear, time domain simulation of LCAC to learn about the dynamics of the vehicle in calm water and in irregular waves. This program leverages the RNN development work that has been done previously for surface ships at model scale, Faller et al. (2008) and Hess et al. (2007). The LCAC captive model and proposed free-running model testing provides a unique opportunity to develop a nonlinear simulation of the dynamics of the vehicle in waves that can be utilized to study vehicle performance and stability. Furthermore, methods are available to mine the nonlinear RNN solution to extract details about physics and to support controller development.

The free-running model test will serve as the primary source of training data for the simulation. However, to pose the problem well to the recursive neural network, accurate models of the six forces and moments that act on the vehicle are required, and the captive model data was used for this purpose. Specifically, feedforward neural network models were constructed to recall the experimental data, and further, to provide reasonable predictions for some experimental conditions not performed in the tow tank. Because different skirt designs and ballast conditions were tested, the models can further be interrogated to discover how varying skirt designs and ballast conditions affect heave, pitch and roll. The feedforward networks trained by the captive model data of the six forces and moments in both dimensional and dimensionless forms acting on LCAC under calm water conditions are the subject of this section.
The 1/12th scale captive model experiment was carried out with the model free to heave, pitch and roll. The purpose of the experiment was to acquire data characterizing the response of LCAC over a series of speeds in calm water, various regular wave conditions and irregular waves representing sea states 3, 4 and 5. The test measured 6-DOF forces and moments with 4 different skirt designs and 3 distinct ballast conditions. A photograph of the 7.3ft long by 3.9ft wide vehicle mounted to Carriage III and operating in the High Speed Basin is given on the right of Fig.9. Some of the measurements that were acquired include: Speed (tow carriage), Fan speed, Wave height (front of model, CG, other locations), Forces and moments from a 6-DOF dynamometer, Displacements (Heave, Pitch and Roll), Vertical accelerations (bow, CG and stern), Cushion pressures, Bag pressures and Fan pressures.

A subset of the described captive model data was used to train two sets of six FFNN models, with each model having one force or moment component as the sole output as shown in Fig.10. The first set of models is in dimensional form and is used to predict force components $X$, $Y$, $Z$ and moment components $K$, $M$, $N$. The inputs are chosen from: $L_{cg}$, the length to the center of gravity of the vehicle; $W$, weight; $z$, vertical (heave) displacement; $U$, speed; and $\phi, \theta, \psi$, roll, pitch, and yaw.

The second set of models predicts dimensionless force components $X', Y', Z'$, given by $(= X/0.5 \rho U^2 L_{cg})$, and dimensionless moments $K', M', N'$, defined as $(= K/0.5 \rho U^2 L_{cg})$. The inputs for these models are chosen from: $Re$, a Reynolds number $(=UL_{cg}/\nu)$; $Fr$, a Froude number $(=U^2/g L_{cg})$; $W'$, dimensionless weight $(= W/0.5 \rho U^2 L_{cg})$; $z'$, dimensionless vertical displacement $(= z/ L_{cg})$; and $\phi, \theta, \psi$, roll, pitch, and yaw.

Fig.9: Left: Landing Craft, Air Cushion (LCAC), Overview sketch, Right: 1/12th-scale LCAC model mounted to carriage 3 at NSWCCD

Fig.10: Feedforward neural network models of LCAC forces and moments
The numbers given on the left side of Fig. 10 denote the number of inputs, the number of nodes in each hidden (internal) layer and the number of outputs, respectively, for the set of dimensional networks. A similar list exists for the dimensionless networks. Initially, each network in each set is trained using all seven inputs. After training, each network is then put through a process, known as lesion analysis, in which each input is sequentially removed (lesioned), one at a time, in order to observe the degradation in the output. Dramatic changes in the output indicate the relative importance of a particular input, whereas minor changes indicate relative insensitivity. Selected inputs may then be removed and the network retrained with the smaller input set. The numbers given on the left side of Fig. 10 include the final number of inputs that were retained after the lesion analysis.

The training data taken from the captive model experiment consisted of 180 ordered n-tuples (inputs + output) of captive model data for the LCAC model fitted with the currently used baseline skirt. The data spanned a variety of CG locations, ballast conditions, heave displacements, speeds and attitude conditions. This data was split into two sets: 120 conditions used to train the networks (training data) and 60 conditions (validation data) set aside to measure how well the trained network can predict conditions similar to the training data, but different. The AAM and R (see below) were computed using only the validation data as this is the most rigorous representation of the results.

The functional form of the model, the architecture of the network and the number of epochs required for training for each of the six networks using a dimensional representation are given in Table I. Also in the table are the results of two error measures used to quantify the comparisons between model predictions and the measured data: an average angle measure (AAM) and a correlation coefficient (R). Note that the AAM was developed at NSWCCD and is defined in Roddy et al. (2006). Both error measures vary between 0 and 1, with 1 representing a perfect solution and 0 representing no agreement.

<table>
<thead>
<tr>
<th></th>
<th>Functional Form</th>
<th>Architecture</th>
<th># Epochs</th>
<th>AAM</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>( X = X(L_{cg}, W, z, U) )</td>
<td>4-16-16-1</td>
<td>109500</td>
<td>0.96</td>
<td>0.95</td>
<td></td>
</tr>
<tr>
<td>( Y = Y(L_{cg}, W, z, U, \varphi, \theta, \psi) )</td>
<td>7-11-11-1</td>
<td>48400</td>
<td>0.85</td>
<td>0.84</td>
<td></td>
</tr>
<tr>
<td>( Z = Z(L_{cg}, W, z, U, \psi) )</td>
<td>5-3-3-1</td>
<td>4800</td>
<td>0.73</td>
<td>0.69</td>
<td></td>
</tr>
<tr>
<td>( K = K(L_{cg}, W, z, U, \varphi, \theta, \psi) )</td>
<td>7-4-4-1</td>
<td>106100</td>
<td>0.86</td>
<td>0.87</td>
<td></td>
</tr>
<tr>
<td>( M = M(L_{cg}, W, z, U, \theta) )</td>
<td>5-36-36-1</td>
<td>27800</td>
<td>0.94</td>
<td>0.95</td>
<td></td>
</tr>
<tr>
<td>( N = N(L_{cg}, W, z, U, \varphi, \theta, \psi) )</td>
<td>7-34-34-1</td>
<td>27800</td>
<td>0.67</td>
<td>0.72</td>
<td></td>
</tr>
</tbody>
</table>

The results for drag force and pitch moment are excellent; side force and roll moment are very good; and heave force and yaw moment are satisfactory. The latter two quantities are small in magnitude, and the experimental data show considerable scatter. Also possible is the fact that additional independent variables, not measured, might be required to fully characterize the functional dependence of these two variables. Nevertheless, the results clearly show that acceptable models for all six forces and moments were obtained.

Fig. 11 shows results for the drag models. At top right, the measured drag force is plotted using blue diamond symbols for each of the 180 n-tuples (cases). The resulting neural network predictions are over plotted using red squares. The data are ordered such that cases 1-120 are the training data, and cases 121-180 are the validation data points. This type of plot gives a quick overview of the level of agreement between the predicted and measured dependent variables. Of particular note here is the observation that the validation cases are predicted to the same level of accuracy as the training cases. This indicates a well-trained network with the drag well represented by the four chosen independent variables. This is reinforced by the plot shown at bottom right in Fig. 11. Here 180 ordered pairs are plotted with measured drag as the abscissa and predicted drag as the ordinate. If the level of agreement is good, then the data will cluster about a line oriented at 45°; this line is included on the graph for reference. The data are indeed grouped about the line representing good agreement.
The two plots on the left represent slices through the multi-dimensional space of the function. Drag as a function of speed and heave are shown. These plots reinforce the fact that the neural network models are predicting accurately in multiple dimensions simultaneously. Note that the plot of drag vs. heave appears to indicate a multi-valued dependence; however, only the heave independent variable is constant at a particular point on the x-axis while the remaining independent variables assume differing values.

Fig.11: Dimensional Drag Force (Positive to bow), AAM = 0.96

The Buckingham Pi theorem was used to devise the set of governing dimensionless variables corresponding to dimensionless forms of the six dependent variables. The purpose here was to examine the dimensionless forces and moments and develop simple models to relate them to dimensionless parameters with familiar physical interpretation, such as Re and Fr numbers. The quantitative error measures are given in Table II, and graphical results are included in Figs.12 for the dimensionless drag force. Again, we see that the poorest performing models are of the heave force and the yaw moment. Overall, the dimensionless solutions are clearly adequate models of the very complicated behavior of the vehicle.

Table II: Architecture, training and error results for the dimensionless networks

<table>
<thead>
<tr>
<th>Functional Form</th>
<th>Architecture</th>
<th># Epochs</th>
<th>AAM</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X' = X'(Re, Fr, W', z')$</td>
<td>4-7-7-1</td>
<td>66900</td>
<td>0.86</td>
<td>0.90</td>
</tr>
<tr>
<td>$Y' = Y'(Re, Fr, W', z', \phi, \theta, \psi)$</td>
<td>7-39-39-1</td>
<td>64100</td>
<td>0.68</td>
<td>0.84</td>
</tr>
<tr>
<td>$Z' = Z'(Re, Fr, W')$</td>
<td>3-16-16-1</td>
<td>2900</td>
<td>0.60</td>
<td>0.75</td>
</tr>
<tr>
<td>$K' = K'(Re, Fr, W', z', \phi, \theta, \psi)$</td>
<td>7-26-26-1</td>
<td>4100</td>
<td>0.71</td>
<td>0.86</td>
</tr>
<tr>
<td>$M' = M'(Re, Fr, W', z', \theta)$</td>
<td>5-4-4-1</td>
<td>16600</td>
<td>0.78</td>
<td>0.84</td>
</tr>
<tr>
<td>$N' = N'(Re, Fr, z', \phi, \theta)$</td>
<td>5-2-2-1</td>
<td>4200</td>
<td>0.42</td>
<td>0.31</td>
</tr>
</tbody>
</table>

Summarizing, the trained neural network models developed here can now be used to effectively recall the numerous data obtained from the experiment in a very simple fashion. Moreover, they can be used to interpolate through the n-dimensional space to recover predictions for conditions that were not
experimentally measured. Used judiciously in conjunction with a series of experimental or computational investigations, neural network models can be used to reduce time requirements and save money while also producing excellent models of physical behavior.

Finally, the overall goal of the program is to develop a six degree-of-freedom time domain simulation of LCAC to learn about the dynamics of the vehicle in calm water and in irregular waves. This approach employs a recursive neural network, which takes forces and moments acting on the vehicle as inputs and which computes three linear and three angular velocities as outputs. The outputs can be differentiated to recover accelerations and integrated to recover trajectory and attitude. Hence the model will produce time series predictions of the maneuvering behavior of the vehicle. The hull force and moment models described in this section resulting from captive model experiments will be used in conjunction with the free-running model data to advance that simulation effort.

![Graphs showing dimensionless drag force](image)

Fig. 12: Dimensionless Drag Force, AAM = 0.86

Acknowledgments

This work is supported by the U.S. Office of Naval Research through Code 331. The program officer for the Virtual Sensor and Next Generation Tool work is Dr. Ronald Joslin, and the program officer for the LCAC work is Dr. Ki-Han Kim. Dr. Patrick Purtell, Code 331 has also provided funding for neural network research. The automated feedforward neural network development code, Intelligent Calculation of Equations (ICE), was used to create the neural networks used for the LCAC work. This code was developed by Applied Simulation Technologies and is available for free by contacting the company via faller@appliedsimulation.com.

References


