Abstract
This paper provides a brief User’s Guide for the MIT MOOS-IvP Undersea Autonomous Network Simulation environment.
## Contents

I Background and Introduction

1 Overview

  1.1 Purpose and Scope of this Document
  1.2 Sonar AUV MOOS Community

II MIT Distributed Network Simulation Environment

1 MOOS-IvP Simulator Architecture

2 Sonar-AUV Simulator

3 Running a Simulation Session

  3.1 Configuration Management
  3.2 Configuring a Virtual Experiment
    3.2.1 Cruise Configuration
    3.2.2 C2 Configuration
    3.2.3 Virtual Modem Network Configuration
    3.2.4 Acoustic Array Configuration
  3.3 Launching a Virtual Vehicle Mission
  3.4 Launching the Topside Command and Control

4 Sonar AUV Simulation Modules and Utilities

  4.1 uSimTargets
    4.1.1 Brief Overview
    4.1.2 Usage
    4.1.3 Parameters for the uSimTargets Configuration Block
    4.1.4 MOOS variables subscribed to by uSimTargets:
    4.1.5 MOOS variables published by uSimTargets:
  4.2 uSimTargetBearings
    4.2.1 Brief Overview
    4.2.2 Parameters for the uSimTargetBearings Configuration Block
    4.2.3 MOOS variables subscribed to by uSimTargetBearings:
    4.2.4 MOOS variables published by uSimTargetBearings:
  4.3 uSimTowedArray
    4.3.1 Brief Overview
    4.3.2 Parameters for the uSimTowedArray Configuration Block
    4.3.3 MOOS variables subscribed to by uSimTowedArray:
    4.3.4 MOOS variables published by uSimTowedArray:
  4.4 uSimPassiveSonar
    4.4.1 Brief Overview
    4.4.2 Parameters for the uSimPassiveSonar Configuration Block
4.4.3 MOOS variables subscribed to by uSimPassiveSonar: ........................................... 27
4.4.4 MOOS variables published by uSimPassiveSonar: .............................................. 27
4.4.5 uSimPassiveSonar Details ...................................................................................... 27
4.5 uSimGPS ......................................................................................................... 28
4.5.1 Brief Overview ......................................................................................................... 28
4.5.2 Parameters for the uSimGPS Configuration Block .................................................... 28
4.5.3 MOOS variables subscribed to by uSimGPS: ......................................................... 28
4.5.4 MOOS variables published by uSimGPS: ............................................................... 28
4.6 uSimCTD ........................................................................................................... 29
4.6.1 Brief Overview ......................................................................................................... 29
4.6.2 Parameters for the uSimCTD Configuration Block .................................................. 29
4.6.3 MOOS variables subscribed to by uSimCTD: ......................................................... 29
4.6.4 MOOS variables published by uSimCTD: ............................................................... 29
4.7 uSimBathy .......................................................................................................... 30
4.7.1 Brief Overview ......................................................................................................... 30
4.7.2 Parameters for the uSimBathy Configuration Block ................................................ 30
4.7.3 MOOS variables subscribed to by uSimBathy: ....................................................... 30
4.7.4 MOOS variables published by uSimBathy: ............................................................. 30
4.8 iMseasBathy ......................................................................................................... 31
4.8.1 Brief Overview ......................................................................................................... 31
4.8.2 Parameters for the iMseasBathy Configuration Block .............................................. 31
4.8.3 MOOS variables subscribed to by iMseasBathy: ...................................................... 31
4.8.4 MOOS variables published by iMseasBathy: ........................................................... 31
4.9 iMseas ..................................................................................................................... 32
4.9.1 Brief Overview ......................................................................................................... 32
4.9.2 Parameters for the iMseas Configuration Block ....................................................... 32
4.9.3 MOOS variables subscribed to by iMseas: .............................................................. 32
4.9.4 MOOS variables published by iMseas: ................................................................... 33
4.10 ctddisp.m ......................................................................................................... 34
4.10.1 Brief Overview ........................................................................................................ 34
4.10.2 Parameters for the ctddisp configuration Block ...................................................... 34
4.10.3 MOOS variables subscribed to by ctddisp.m: ......................................................... 34
4.10.4 MOOS variables published by ctddisp.m: .............................................................. 34
4.11 Arraysim.m ...................................................................................................... 35
4.11.1 Brief Overview ........................................................................................................ 35
4.11.2 Configuration Files ............................................................................................... 35
4.11.3 MOOS variables subscribed to: .............................................................................. 35
4.11.4 MOOS variables published: .................................................................................... 36
4.12 SealabMultiSim.m ................................................................................................. 37
4.12.1 Brief Overview ........................................................................................................ 37
4.12.2 Configuration MOOS-block ............................................................................... 37
4.12.3 MOOS variables subscribed to: .............................................................................. 38
4.12.4 MOOS variables published: .................................................................................... 38
Part I

Background and Introduction

1 Overview

1.1 Purpose and Scope of this Document

The purpose of this document is to provide a catalog style overview of modules used for creating a high-fidelity MOOS-IvP simulation environment for networks of autonomous underwater vehicles and surface craft, and to provide an introductory guide to operating the simulator. The scope of discussion includes, for each module, a brief description of the module function, authorship, source for download, rough measure of complexity, and module dependencies. Further, for use by developers of onboard processing modules, for example, the description includes a detailed listing of MOOS variables published by or subscribed to by each simulator module.

1.2 Sonar AUV MOOS Community

The MIT Laboratory for Autonomous Marine Sensing Systems (LAMSS) operates two Bluefin21 AUVs for research into adaptive and collaborative, autonomous acoustic sensing in the ocean. The vehicles, called Unicorn, and Caribou or Macrura depending on the configuration, can be carrying a variety of customized acoustic source/receiver payloads. In configurations for seabed object
detection, classification and localization the two vehicles each have a 16-element fixed hydrophone array mounted in the front section, and an active source for ionizing the seabed. For passive and multistatic acoustic research below 1 kHz, Unicorn is integrated with a 32-element towed hydrophone array.

In either case, the vehicles are operating with MOOS-IvP handling the higher-level intelligent autonomy, implemented on a computer stack in the payload section, integrated with the acoustic source control and the data acquisition system. The MOOS-IvP configuration is shown schematically in Fig. 1. The MOOS-IvP Autonomy System is connected to the main vehicle computer using the 'back-seat driver' paradigm, consistent with the proposed ASTM F41 standard for AUVControl. The MOOS interface to the front-seat driver is platform dependent, with several existing in the source tree, for example iHuxley for BF21 and iOEX for the Ocean Explorer. Other connections from the payload include the WHOI Micromodem, which forms the backbone of the undersea communication network. The modem communication is handled by the MOOS communication stack, consisting of a generic message coder-decoder pGeneralCodec, and the pAcommsHandler process handling message queueing to and from the modem driver module iMicroModem. The main purpose of the sonar payload is obviously to operate the acoustic sources and record and process data from the hydrophone array. The interface is provided by a dedicated data acquisition MOOS module iDAS (Digital Acquisition System), which interacts with the signal conditioning for each array element. iDAS subscribes for control parameters and commands from the MOOSDB, and publishes array element locations and a filename, every time a new snippet of data is available. The high-bandwidth nature of the raw acoustic data makes it inconvenient to publish these directly in the MOOSDB. Instead they are written to a file, which also serves as permanent recording, and only the file location and name is published, for other processes to access, such as the signal processing module, which for acoustic arrays will typically derive a bearing estimate for a source or a scatterer and publish the result in the MOOSDB for other processing such as tracking.

In addition to these three main connections, there is also a process iCTD controlling and receiving data from the CTD sensor which is standard for the vehicle. Also, some of the payloads have their own GPS antenna for accurate timing, and here the MOOS interfacing is handled by the iGPS process.
Part II
MIT Distributed Network Simulation Environment

1 MOOS-IvP Simulator Architecture

A cornerstone of the MIT multi-node undersea network simulator is the MOOS utility \texttt{iModemSim}, which simulates an undersea micromodem network used in the field as shown in Fig. 2 by transparently connecting all MOOS-communities on the local area network, or the same computer, running this process. The utility incorporates physically realistic message transmission uncertainty, intermittency and latency, as well as message collision, and therefore provides a high-fidelity ‘virtual ocean’ environment for realistic simulation of an undersea network. The network simulation architecture is shown in Fig. 2.

The topside command and control MOOS community \texttt{AUV\_Topside} is operated unchanged from that used in field deployments, except for the physical serial port to the WHOI micromodem gateway being replaced by a virtual serial port to the \texttt{iModemSim} process. Thus, the topside command and control GUIs and the situational and nodal status displays are transparently representing an actual operational environment.
Similarly, the MOOS Communities for an arbitrary number of underwater vehicles, or surface craft are running on one or more desktop or laptop computers - or on the actual vehicle payload - connected to the 'virtual ocean' by iModemSim. Each 'virtual vehicle' are commanded from the topside using CCL modem commands with realistic throughput statistics, and will transmit status reports via the virtual modems to the network when polled to do so. The architecture allows for all virtual nodes to share environmental and situational information. Thus, the environment allows for fully realistic simulation of adaptive and collaborative autonomy by the network assets.

In addition to the field-level display tools applied by the topside community, the simulator incorporates several graphical tools for real-time situational display of nodal information, including results of processed sensor data, such as the inboard processing of acoustic data collected by hydrophone arrays on sonar AUVs, as described in the following.

## 2 Sonar-AUV Simulator

![Diagram of MOOS-IvP Sonar-AUV Simulator](image)

Figure 3: MOOS-IvP Sonar-AUV Simulation Environment: The simulation environment is identical to the AUV payload MOOS environment, except for the 'outside world' interfaces, which are replaced by a set of simulation tools which use the same interface definitions as the actual vehicle hard- and software, making it transparent to the MOOS-IvP environment whether it is operating in the vehicle or a stand-alone computer.

The power of the MOOS-IvP simulation environment is that all processes can be operated unchanged, as long as the hardware and software handling the interface between actuators and sensors and the MOOSDB are replaced by dedicated simulation modules publishing and subscribing to the same MOOS variables as the on-board interfaces. Thus, a full simulation capability has been established developing a set of payload connection simulators, which can replace the actual hardware and software transparently to all other MOOS processes connected to the MOOSDB, as illustrated schematically in Fig. 3, where the blue-shaded inboard systems in Fig. 1 are replaced by their red-shaded simulator equivalents. Also, the simulation modules may be used in any combination with the real system components. Thus, the CTD simulator has been used on a vehicle during
scientific missions with a broken CTD unit, due to the fact that the BF21 front-seat driver requires CTD data for operating. Also, the target simulator has been applied in the field in cases of a malfunctioning acoustic array.

Figure 4: small_uVis.m: Real-time display of a passive acoustic sensing mission by an AUV towing a hydrophone array. The upper left frame shows the auv/array and target history for a sensing mission simulation. The two lower frames show zooms of the auv/array geometry in the horizontal and vertical, while the upper right shows the computed Beam-Time Record (BTR) achieved from the simulated time series, with the real-time stabilized bearing-track estimate indicated by the white curve.

In addition to the system simulator modules, the source tree incorporates several graphical tools for use with the simulator, in addition to the standard topside displays described elsewhere. Thus, the MATLAB script small_uVis.m generates a real-time display of auv/array dynamics, and real-time acoustic processing results, as shown in Fig. 4.

A key enabler of the acoustic sensing simulation framework is the multi-target simulator uSimTargets. Each new target is identified by an initial position, speed and heading, and its acoustic properties, which is passed to all simulated sensing nodes via the acoustic communication network or its simulator equivalent iModemSim, using a special message generated by the topside command module iCommander. Once a node receives a target message, uSimTargets will update its target list and pass on the new target situation to the acoustic simulators.

A list of the simulation modules used in the LAMSS simulation environment are given in Tables 1 and 2. A detailed description of the modules, their configuration, and their interaction with the MOOSDB is given in the User’s Guide [?].

3 Running a Simulation Session

The missions-lamss repository contains a set of scripts for building the MOOS-IvP configuration files and launching the backseat driver on all LAMSS vehicles, and the topside Command and Control (C2) infrastructure, allowing for executing virtual experiments with an arbitrary number
<table>
<thead>
<tr>
<th>#</th>
<th>Module Name</th>
<th>Module Description</th>
<th>Author</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>uSimTargets</td>
<td>Dynamically simulates an arbitrary number of targets for the multi-source acoustic simulators. <em>Libraries: mbutil, MOOS, MOOSGen, MOOSUtility</em></td>
<td>Schmidt</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>uSimTargetBearings</td>
<td>Low-fidelity, target bearing estimator replacing an onboard acoustic array processing module such as pBearingTrack. Highly efficient and useful for multi-aug collaborative mission simulations, and for onboard bearing simulation in case of a malfunctioning hydrophone array. <em>Libraries: mbutil, MOOS, MOOSGen, MOOSUtility</em></td>
<td>Eickstedt, Schmidt</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>uSimTowedArray</td>
<td>Towed array simulator used together with uMVSBluefin for coupled dynamics simulation. <em>Libraries: mbutil, MOOS, MOOSGen, MOOSUtility</em></td>
<td>Schmidt</td>
<td>4,803</td>
</tr>
<tr>
<td>4</td>
<td>uSimPassiveSonar</td>
<td>Medium- or High-fidelity acoustic simulator generating time-series received on array simulated by uSimTowedArray, incorporating ambient noise and signals from targets simulated by uSimTargets. In the high-fidelity mode it uses the embedded Bellhop raytracing code for incorporating multipaths and boundary interactions. In the medium-frequency mode target signals are simulated as cylindrical waves, incorporating bottom loss. <em>Libraries: mbutil, MOOS, MOOSGen, MOOSUtility</em></td>
<td>Schmidt</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>uSimGPS</td>
<td>Simulates received GPS information when vehicle is surfaced. <em>Libraries: mbutil, MOOS, MOOSGen, MOOSUtility</em></td>
<td>Schmidt</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>uSimCTD</td>
<td>Simulates CTD data by interpolating in HOPS-generated virtual ocean or real ocean database. Can be used in the field in the case of a malfunctioning CTD unit. <em>Libraries: mbutil, MOOS, MOOSGen, MOOSUtility</em></td>
<td>Lum</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>uSimBathy</td>
<td>Simulates bathymetry data by interpolating in bathymetry table. Used for simulating altimeter data. <em>Libraries: mbutil, MOOS, MOOSGen, MOOSUtility</em></td>
<td>Lum</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>uMVSBluefin</td>
<td>Dynamic simulator for Bluefin21 AUV with or without a towed array</td>
<td>Battle</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>iModemSim</td>
<td>Simulates a modem network. Communicates with iModemSim in all MOOS communities on local network. Handles propagation latency and transmission loss, and collisions. <em>Libraries: anrputil, MOOS, MOOSGen, MOOSUtility</em></td>
<td>Patrikalakis</td>
<td>1,474</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Total unique lines of code</strong></td>
<td></td>
<td>24,655</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Total aggregate lines of code</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1: MIT AUV simulation environment. C++ simulation Modules.
<table>
<thead>
<tr>
<th>#</th>
<th>Module Name</th>
<th>Module Description</th>
<th>Author</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ArraySim.m</td>
<td>Matlab version of array dynamics simulator <strong>uSimTowedArray</strong></td>
<td>Dumortier</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>SealabMultiSim.m</td>
<td>High-fidelity acoustic simulator generating timeseries received on array simulated by <strong>uSimTowedArray</strong>, incorporating ambient noise and signals from targets simulated by <strong>uSimTargets</strong>. Uses SEALAB synthetic sonar environment for environmental acoustic modeling.</td>
<td>Schmidt</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>small_uVis.m</td>
<td>Graphical display of AUV mission, including array dynamics, and processed acoustic data. Applied on each simulated platform. May also be used in conjunction with <strong>uPlayback</strong> for replay of actual at-sea missions.</td>
<td>Schmidt</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: MIT AUV simulation environment - MATLAB modules.

of acoustic and non-acoustic sensing nodes.

### 3.1 Configuration Management

The **missions-lamss** repository contains a complete set of MOOS ProcessConfig blocks for all relevant MOOS processes, each having extension .plug.

Similarly, there is a set of .plug files (bhv*.plug) for all IvP behaviors used in the LAMSS missions.

Key parameters (indicated by the bash-like symbols $(0)$ in the process and behavior ‘plugs’ are replaced by environmental variables, defined in a set of definition files with extension .def.

At mission launch the plugs and definitions are combined into a fresh set of complete .moos and .bhv configuration files by a dedicated pre-processor nsplug.

The repository applies to both virtual experiments and field deployments, and as such provides maximal consistency between the two. Also, it ensures complete consistency in the communication infrastructure among vehicles and topside command and control. This is achieved by a hierarchical repository structure:

**missions-lamss/global_plugs** : Plugs common among all nodes, e.g communication.

**missions-lamss/auv/auv_plugs** . Plugs that are common to all AUVs.

**missions-lamss/auv/macrura/macrura_plugs** . Plugs that are specific to the AUV Macrura.

**missions-lamss/cruise/current** Cruise-specific plugs for current cruise, e.g local UTM datum, viewers

**missions-lamss/topside/topside_plugs** : Topside specific plugs, e.g iCommander.

There is an additional level of granularity associated with the arrays and onboard processors on the vehicles. The configuration plugs for the acoustic arrays used for the AUV’s are defined in the
directory tree `/missions-lamss/auv/arrays/`. For example the configuration plugs for the MIT DURIP array are specified in the directory `/missions-lamss/auv/arrays/durip`, together with a definition file `array.def` containing array-specific parameters such as array spacing, frequency band, sampling rate, etc. The configuration includes a suite of generic towed hydrophone arrays with typical number of elements, `hla8` (8 elements), `hla16` (16 elements), and `hla32` (32 elements), the MIT DURIP array, and the two nose mounted arrays SINGLE and DUAL.

The selection of MOOS-IvP processes to be incorporated in the mission configuration is defined by `moos.meta`, also containing the mission launch using pAntler, and links to all the relevant configuration plugs.

The behavior file is configured in a similar manner using `bhv.meta`.

### 3.2 Configuring a Virtual Experiment

#### 3.2.1 Cruise Configuration

The first step is to configure the repository for a specific geographical area or cruise. The repository has inherited several cruise definitions from other programs, including `glint08` and `swams09`. The configuration is performed by executing a script:

```
> cd ~/ssions-lamss
> ./cruise_onfig.sh glint08
```

This script will link the symbolic subdirectory cruise/current to cruise/glint08 containing all the GLINT08-specific plugs and definitions, such as operations box, local UTM datum etc. It also should contains definitions of obstacles to be avoided by the vehicles, such as the research vessel and moorings.

#### 3.2.2 C2 Configuration

The next step is to configure the command and report message set. For LAMSS there is a choice between the NaFCon message set used in PN07 or the new pAcommsHandler [1] message set. A script is provided which chooses the associated plugs and definitions. For either message set, the prototype is set up for the MIT C2 topside, and using pAcommsHandler, currently the only fully operational configurations. To choose the fixed NaFCon message set, append "nafcon" to the desired launch script (for topside and all the AUVs):

```
> ./simulation_launch.sh nafcon
```

The pAcommsHandler LAMSS message set is the default so no parameters are needed:

```
> ./simulation_launch.sh
```

For the LAMSS default pAcommsHandler message set there are only minor differences in the functionality of the autonomy system. Both message sets contain Deploy and Prosecute commands, and Status, Contact, and Track reports. The commands differ only slightly, mainly in the flexibility of defining the geometry of the survey patterns provided by the LAMSS message set, as opposed to the fixed configuration of NaFCON, where loiter patterns, racetracks etc. have to be defined in the configuration files, i.e. the plugs described above.

The more significant difference is in the Contact Report, where the LAMSS message support the multitarget tracking.
3.2.3 Virtual Modem Network Configuration

For network simulations, the user must configure a virtual serial port pair for each network node:

```bash
> sudo serial_loopback /tmp/ttyLOOPA1 /tmp/ttyLOOPA2 &
> sudo serial_loopback /tmp/ttyLOOPB1 /tmp/ttyLOOPB2 &
```

etc.

A script (`loopbacks_tmp`) is available for setting up 3 serial port pairs, configured for the vehicles Unicorn, Macrura, and the topside Command and Control. You can run this by hand, but `./simulation_launch.sh` will do it for you on the first launch, generating several virtual serial port pairs.

The association of a virtual port pair with a specific vehicle is set in the vehicle definition file, for the AUV Unicorn in `missions-lamss/auv/unicorn/all.def`, and for the topside in `missions-lamss/topside/all.def`.

For simulation of a single AUV and the topside, it may be advantageous to establish the modem connection through an actual modem pair or a modem emulator box. In this case the simulation is launched using the flag `hw_modem` for both the virtual vehicle and the topside. This mode is also used when a simulated virtual vehicle is ‘operating’ in a real underwater network using an over-the-side modem, a very cost-effective approach to testing multivehicle collaborative autonomy.

3.2.4 Acoustic Array Configuration

In each vehicle directory, a script is provided for associating a particular array with that vehicle. To select the DURIP array for the vehicle 'Unicorn', use the command sequence

```bash
> cd ~/missions-lamss/auv/unicorn
> ./array_config.sh durip
```

3.3 Launching a Virtual Vehicle Mission

Go into the directory for the vehicle, e.g.

```bash
> cd ~/missions-lamss/auv/unicorn
```

For launching a LAMSS passive sensing simulation with high-fidelity array dynamics and acoustic simulation and onboard passive signal processing:

```bash
> ./simulation_launch.sh passive
```

To alternatively launch a simulation with the low-fidelity multi-target tracking simulator

```bash
> ./simulation_launch.sh bearingsim
```

For launching a simulation of a SWAMSI multi-static active acoustic sensing mission:

```bash
> ./simulation_launch.sh active
```
When using a physical modem or modem emulator, the same mission is started using the command

```
> ./simulation_launch.sh active hw_modem
```

Finally, to launch the actual vehicle autonomy system for field missions (on the vehicle):

```
> ./runtime_launch.sh
```

The same parameters (active, passive, nafcon) available for simulation can be appended to `runtime_launch.sh` as well.

To launch the vehicle-side node display, automatically configured for the mission launched:

```
> matlab -nojvm -nosplash < misc/small_uVis.m
```

### 3.4 Launching the Topside Command and Control

To start the topside with the iCommander Command console [2] and the situational display:

```
> cd missions-lamss/topside
> ./simulation_launch.sh
```

To launch the Lat-Long to UTM converter tool, automatically configured for the current UTM grid and operations area:

```
> matlab
>> misc/geo_convert
```
4  Sonar AUV Simulation Modules and Utilities

4.1  uSimTargets

4.1.1  Brief Overview

This MOOS module simulates an arbitrary number of acoustic sources moving along straight-line paths. The sources are generated via the topside command console iCommander via a dedicate modem message broadcast to all nodes. uSimTargets will then keep a record of all sources within a critical distance from the vehicle, specified in the MOOS-block, and generate the MOOS variables needed by the acoustic timeseries simulators uSimPassiveSonar and SealabMultiSim.m.

4.1.2  Usage

The dynamic target simulator as well as the various-level fidelity acoustic simulators are always executed on each sensor node. However, the activation of the acoustic target and interferer sources may occur either locally in a simulated node MOOS community, or centrally, e.g. in the topside community. The latter is important for synchronizing sources simulated on multiple nodes for testing collaborative tracking processing and behaviors. The centralized target control is made possible by defining the interface between the topside C2 module iCommander and the target dynamics simulators using a format which is compatible with the generic messages CoDec in pAcommsHandler, such that the targets may be activated on all platforms in a synchronized manner via a modem transmission echoing the initial target status to all nodes. In addition to be used in entirely virtual experiments, this architecture allows for activating acoustic target simulators on actual, submerged nodes, e.g. in cases where equipment failure has made the sensing arrays unusable, thus allowing testing of collaborative tracking behaviors even in such cases.

The echoing of the target initialization to all network nodes is activated by including the XML file lamss_targetsim.xml in the configuration for pAcommsHandler, as described in Appendix ??.

Note that the MOOS variable TGT\_STATE\_OUT is assumed to contain the target initialization message on the local node, while the MOOS variable receiving the message on the receiving nodes is TGT\_STATE\_IN. Thus, it will be necessary to echo these variables from and into the standard MOOS variable T\_STATE published by PassiveTgtSim, using pEchoVar. In addition, the MOOS variable TARGET\_CONTROL is echoed to all other platforms to activate the source control through PassiveTgtSim. Thus, it is important to at least once broadcast a target initialization before issuing a Prosecute Command, which would otherwise activate the nodal source simulators.

4.1.3  Parameters for the uSimTargets Configuration Block

The following configuration parameters are defined for uSimTargets.

rangeCrit: Critical range beyond which target will be ignored.

Below is an example configuration block for the uSimTargets.

Listing 4.1 - An example uSimTargets configuration block.

```
1 LatOrigin = 42.3584
2 LongOrigin = -71.08745
3
```
4 // ------------ uSimTargets Configuration block ------------
5 ProcessConfig = uSimTargets
6 {
7  AppTick = 4
8  CommsTick = 4
9
10  rangeCrit = 5000 // maximum source range in meters
11 }

The `LatOrigin` and `LonOrigin` parameters on lines 1-2 are not specific to uSimTargets, but are specified at the global level in a MOOS file. They are needed to allow the conversion between local x-y coordinates and latitude-longitude coordinates. They are mandatory parameters and uSimTargets will refuse to launch if they are lacking - but they are mandatory for other common MOOS applications as well.

### 4.1.4 MOOS variables subscribed to by uSimTargets:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Published by</th>
</tr>
</thead>
<tbody>
<tr>
<td>TARGET_CONTROL</td>
<td>$</td>
<td>“ON”: Sources controlled via the <code>SIMULATE_TARGET</code> message, preample = T_</td>
<td>pAcommsHandler</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“OFF”: Sources controlled by Prosecute message, preample = TARGET_</td>
<td></td>
</tr>
<tr>
<td>T_STATE</td>
<td>$</td>
<td>Comma-separated list of source parameters: x,y,depth,heading,speed,freq,bw,spl, tn1_freq,tn1_dbl,...,tn5_freq,tn5_dbl,time,number</td>
<td>iCommander pAcommsHandler</td>
</tr>
<tr>
<td>TARGET_STATE</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The parameters listed in the MOOS variable `T_STATE`, posted by `pAcommsHandler` in response to a received `SIMULATED_TARGET` message are

- `tgt.x`: UTM x-coordinate of acoustic source starting point
- `tgt.y`: UTM y-coordinate of acoustic source starting point
- `tgt_depth`: Depth of acoustic source
- `tgt_hdg`: Heading of acoustic source in degrees
- `tgt_speed`: Speed of acoustic source in m/s.
- `tgt_freq`: Center frequency of broadband acoustic source
- `tgt_bw`: Bandwidth in Hz of broadband acoustic source
- `tgt_sp1`: Spectral level of broadband source
- `tn#_freq`: Frequency in Hz of tone number #. Message allows for up to 5 tones which will be superimposed to broadband source signature.
tn\#_dbl: Source level in dB of tone number \#. Message allows for up to 5 tones which will be superimposed to broadband source signature.

tgt\_utc: UTC time of source at starting location

tgt\_num: ID number assigned to acoustic source.

4.1.5 MOOS variables published by uSimTargets:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>TARGET_SIM_LIST</td>
<td>$</td>
<td>Comma-separated list of source numbers within critical distance</td>
<td>1,2,3,6</td>
</tr>
<tr>
<td>TGT#_SIM_STATE</td>
<td>$</td>
<td>Comma-separated list of source parameters for source number # #.on/off,utc.x.y.speed,heading,depth, spl,freq,bw,delay,tn1_freq,tn2_dbl,...</td>
<td>2,1,122...,3000,2100,...,0,8,50,120</td>
</tr>
</tbody>
</table>
4.2 uSimTargetBearings

4.2.1 Brief Overview

This MOOS module provides a bearing estimate with Gaussian noise for multiple targets simulated by uSimTargets. This is a low-fidelity bearing estimator which does not perform any processing of array data, but simply computes the current bearing to the targets and publishes the bearing state variable in the same format as the on-board acoustic bearing tracker modules. On the other hand, it is highly efficient and therefore well suited for developing adaptive and collaborative behaviors not sensitive to realistic environmental acoustic uncertainty.

4.2.2 Parameters for the uSimTargetBearings Configuration Block

The following configuration parameters are defined for uSimTargetBearings.

rap_acoustics: if set to true activates rap (Reliable Acoustic Path) acoustics, simulating the direct path propagation assuming the deep ocean linearly increasing sound speed with depth, leading to circular ray path. Used for simulating the vertical elevation angle of the acoustic arrival, and converting it to a range estimate.

svp_surface: Sound speed in m/s at sea surface.

svp_gradient: Sound speed in gradient in (m/s)/m. Typically 0.016 in the deep ocean.

max_rap_depth: Maximum depth in m of the turning point of the rap path. Typically close to the average depth.

detection_range: Detection range for target in meters. If target is outside the tracking state is set to TRACKING = NO_TRACK. When targets enter the detection range the state is changed to TRACKING = AMBIGUOUS, and when the left-right ambiguity is resolved, to TRACKING = TRACKING.

sigma: Standard deviation of bearing estimate. used to make the bearing estimate more realistic in regard to array processing uncertainty.

Below is an example configuration block for the uSimTargetBearings.

Listing 4.2 - An example uSimTargetBearings configuration block.

```
1 LatOrigin = 42.3584
2 LongOrigin = -71.08745
3 // ------------ uSimTargetBearings Configuration block----------
4 ProcessConfig = uSimTargetBearings
5 {
6   AppTick = 2
7   CommsTick = 5
8   rap_acoustics = true
9  svp_surface = 1480
10 svp_gradient = 0.016
11 max_rap_depth = 5500
12 detection_range = 30000 // Detection range in meters
13 sigma = 1.0 // Bearing standard deviation in degrees
14 }
```
The \texttt{LatOrigin} and \texttt{LonOrigin} parameters on lines 1-2 are not specific to \texttt{uSimTargetBearings}, but are specified at the global level in a MOOS file. They are needed to allow the conversion between local x-y coordinates and latitude-longitude coordinates. They are mandatory parameters and \texttt{uSimTargetBearings} will refuse to launch if they are lacking - but they are mandatory for other common MOOS applications as well.

### 4.2.3 MOOS variables subscribed to by \texttt{uSimTargetBearings}:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Published by</th>
</tr>
</thead>
<tbody>
<tr>
<td>VEHICLE_ID</td>
<td>D</td>
<td>Node number for ownship. Used for selecting dynamic parameters</td>
<td>pAcommsHandler</td>
</tr>
<tr>
<td>TRACKER_COMMAND</td>
<td>$</td>
<td>Tracking control: “ON” or “OFF”</td>
<td>pTrackMonitor</td>
</tr>
<tr>
<td>TARGET_SIM_LIST</td>
<td>$</td>
<td>Comma-separated list of source numbers within critical distance</td>
<td>uSimTargets</td>
</tr>
<tr>
<td>TGT_#_SIM_STATE</td>
<td>$</td>
<td>Comma-separated list of source parameters for source number # #.on/off,utc,x,y,speed,heading,depth,spl,freq,bw</td>
<td>uSimTargets</td>
</tr>
<tr>
<td>NAV_X</td>
<td>D</td>
<td>Ownship UTM x-coordinate</td>
<td>iHuxley</td>
</tr>
<tr>
<td>NAV_Y</td>
<td>D</td>
<td>Ownship UTM y-coordinate</td>
<td>iHuxley</td>
</tr>
<tr>
<td>NAV_HEADING</td>
<td>D</td>
<td>Ownship true heading in degrees</td>
<td>iHuxley</td>
</tr>
<tr>
<td>NAV_DEPTH</td>
<td>D</td>
<td>Depth of vehicle in meters (positive down).</td>
<td>iHuxley</td>
</tr>
<tr>
<td>NAV_SPEED</td>
<td>D</td>
<td>Ownship speed in m/s .</td>
<td>iHuxley</td>
</tr>
</tbody>
</table>

### 4.2.4 MOOS variables published by \texttt{uSimTargetBearings}:
<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Format</th>
</tr>
</thead>
</table>
| TRACKING      | $    | Tracking state.  
“NO_TRACK”: No detection yet  
“AMBIGUOUS”: Detection, but ambiguous bearing  
“TRACKING”: Un-ambiguous bearing tracking |        |
| “CLASSIFYING”: |      | Classifying sub-state |        |
| BEARING_TRACKS | $    | Comma-separated bearing state variables, published sequentially for each of the bearing tracks.  
See below for content |        |
| IN_RANGE      | $    | Reset flag for geo trackers | “FALSE” |
| CLOSE_RANGE   | $    | Reset flag for geo trackers | “FALSE” |

The principal output of uSimTargetBearings is the MOOS variable BEARING_TRACKS, which contain the bearing state variables for the current tracks, published sequentially for each active track.

The process pTrackMonitor subscribes to this variable and provides the onboard contact management, generates the Contact Reports, and manages the interface to the autonomy helm.
<table>
<thead>
<tr>
<th>State variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>track</td>
<td>Track number</td>
</tr>
<tr>
<td>priority</td>
<td>Track priority. Set to 10</td>
</tr>
<tr>
<td>template</td>
<td>not used</td>
</tr>
</tbody>
</table>
| state          | Tracking state: 0: Invalid Bearing  
1: Relative bearing  
2: Absolute bearing. |
| ips            | Port-starboard flag for relative bearings 0: ambiguous, relative baring  
1: Starboard relative bearing  
2: Port relative bearing |
| x              | Current UTM x-coordinate of array center |
| y              | Current UTM y-coordinate of array center |
| bearing        | Bearing to target in degrees. |
| sigma          | Bearing estimate uncertainty in degrees |
| rate           | Bearing rate in degrees per second |
| rate_sigma     | Bearing rate uncertainty in degrees per second |
| snr            | SNR of track in dB |
| time           | UTC time. |
| nfeat          | Not used |
4.3 uSimTowedArray

4.3.1 Brief Overview

This MOOS process models the dynamics of a multi-sectored towed array in response to the motion of the tow-point. In addition to computing and publishing in the MOOSDB the hydrophone positions, it also publishes the tension on the tow-point, a parameter which is then subscribed to by the model for the vehicle dynamics. This connectivity results in a high-fidelity modeling of the coupled platform/array system.

4.3.2 Parameters for the uSimTowedArray Configuration Block

The cable configuration and hydrophone separations are defined in the configuration (.moos) file. The following configuration parameters are defined for uSimTowedArray.

- **cond_check**: Flag for checking the condition number for Jacobian computed for the array dynamics updates. Should be set to 1 to avoid unstable solutions. If condition number exceeds $10^6$ the array will be re-initialized to steady-state solution with current AUV heading. This can happen in cases of sharp turns and depth changes. Remedied by increasing AppTick.

- **start_speed**: Minimum speed for which array dynamics model will be initiated.

- **create_dir_frame**: Flag identifying whether to write the frame number variables VSA_DIR and VSA_FRAME to the frame file. If set to zero this step will be done by the acoustic simulation module, which is the normal situation.

- **write_out_files**: Flag identifying whether to write the non-acoustic data to the file identified by VSA_DIR and VSA_FRAME. Under normal circumstances it should be set to 0, since the acoustic simulators perform this step.

- **filter_length**: Number of time samples used to achieve an average value of the towing speed and the cable tension, which provides a stable cable solution.

- **array_type**: Array type. 1: NUWC VSA array, 2: MIT DURIP array

- **number_sectors**: Number of array sections, each of which has homogeneous properties as defined in the SECTOR_# parameters.

- **sector_#**: Comma-separated list of 7 properties characterizing array sector number #.

  - Nsub Number of subdivisions applied in the numerical model for this sector.
  - Len length of sector in meters.
  - Wgt Weight of cable in N/m.
  - Dia Diameter of cable in m.
  - Td Tangential drag coefficient (along cable)
  - Nd Normal drag coefficient (perpendicular to cable)
E Elastic modulus for array section in N/m²

towbody: Flag indicating if towbody exists (1) or not (0).

towbody_parameters: Comma-separated list of 3 properties the tow body

  Wgt Weight of tow body in N
  Dcoef Drag coefficient for towbody
  Area Cross-sectional area of towbody.

array_section_number: cable section containing acoustic array

first_phone_offset: Offset in meters of the first hydrophone in array section.

phone_spacing_groups: Number of array element spacing groups. For group number # the number
of spacings is specified by the variable NUM_SPACINGS_GROUP #, and the spacing by SPACING_GROUP #

spacing_group #: Array element spacing for group number #.

num_spacings_group #: Number of hydrophone spacing intervals for group number #. Note that
the total number of spacings in the array must equal the total number of array elements minus
one.

Below is an example configuration block for uSimTowedArray, here configured for the MIT DURIP
towed array.

Listing 4.3 - An example uSimTowedArray configuration block (MIT DURIP towed array).

```
0 //--------- uSimTowedArray Configuration block -----------------
1 ProcessConfig = uSimTowedArray
2 {
3   AppTick = 2
4   CommsTick = 5
5
6   cond_check = 1
7   start_speed = 0.5
8   create_dir_frame = 0
9   write_out_files = 0
10  filter_length = 40
11  array_type = 2
12  number_sectors = 3
13 // Nsub, Len  Wgt  dia  td  nd  nd  E
14 sector_1 = 20, 20.0, .0000, .0095, .001, 0.1, 1.6e9
15 sector_2 = 30, 36.0, .0000, .035, .0024, 0.5, 1.6e9
16 sector_3 = 20, 20.0, .0000, .0095, .001, 0.1, 1.6e9
17 tow_body = 0
18 // Wgt  Dcoef  Area
19 towbody_parameters = 0.0, 0.0, 0.0
20 array_section_number = 2
21 first_phone_offset = 3.0
22 phone_spacing_groups = 3
23 spacing_group_1 = 1.5
24 num_spacings_group_1 = 5
25 spacing_group_2 = 0.75
26 num_spacings_group_2 = 22
27 spacing_group_3 = 1.5
28 num_spacings_group_3 = 4
29 }
```
4.3.3 MOOS variables subscribed to by uSimTowedArray:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Published by</th>
</tr>
</thead>
<tbody>
<tr>
<td>VSA_DIR</td>
<td>$</td>
<td>Directory containing files with acoustic and non-acoustic array data</td>
<td>iVSA</td>
</tr>
<tr>
<td>VSA_FRAME</td>
<td>D</td>
<td>Frame number for acoustic and non-acoustic data files in VSA_DIR</td>
<td>iVSA</td>
</tr>
<tr>
<td>VEHICLE_ID</td>
<td>D</td>
<td>Node number for ownship. Used for selecting dynamic parameters</td>
<td>pAcommsHandler</td>
</tr>
<tr>
<td>TOW_POS_X</td>
<td>D</td>
<td>UTM x-coordinate of array cable tow point.</td>
<td>uMVS_Bluemin</td>
</tr>
<tr>
<td>TOW_POS_Y</td>
<td>D</td>
<td>UTM y-coordinate of array cable tow point.</td>
<td>uMVS_Bluemin</td>
</tr>
<tr>
<td>TOW_POS_Z</td>
<td>D</td>
<td>UTM z-coordinate of array cable tow point.</td>
<td>uMVS_Bluemin</td>
</tr>
<tr>
<td>TOW_VEL_X</td>
<td>D</td>
<td>x-velocity in m/s of array cable tow point.</td>
<td>uMVS_Bluemin</td>
</tr>
<tr>
<td>TOW_VEL_Y</td>
<td>D</td>
<td>y-velocity in m/s of array cable tow point.</td>
<td>uMVS_Bluemin</td>
</tr>
<tr>
<td>TOW_VEL_Z</td>
<td>D</td>
<td>z-velocity in m/s of array cable tow point.</td>
<td>uMVS_Bluemin</td>
</tr>
</tbody>
</table>

4.3.4 MOOS variables published by uSimTowedArray:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>CABLE_TENSION</td>
<td>D</td>
<td>Tension of tow cable in N at tow point</td>
<td></td>
</tr>
<tr>
<td>ARRAY_X</td>
<td>$</td>
<td>x-position of hydrophones in m relative to tow-point</td>
<td>17.23,18.33, ...</td>
</tr>
<tr>
<td>ARRAY_Y</td>
<td>$</td>
<td>y-position of hydrophones in m relative to tow-point</td>
<td>-6.13,-6.33, ...</td>
</tr>
<tr>
<td>ARRAY_Z</td>
<td>$</td>
<td>z-position of hydrophones in m relative to tow-point</td>
<td>0.01,0.02, ...</td>
</tr>
<tr>
<td>ARRAY_PITCH</td>
<td>$</td>
<td>pitch in degrees of array elements</td>
<td>0.01,0.00, ...</td>
</tr>
<tr>
<td>ARRAY_HEADING</td>
<td>$</td>
<td>true heading in degrees of array elements</td>
<td>301.01,300.99, ...</td>
</tr>
<tr>
<td>CABLE_SHAPE_X</td>
<td>$</td>
<td>x-position of cable elements in m relative to tow-point</td>
<td>0.43,0.84, ...</td>
</tr>
<tr>
<td>CABLE_SHAPE_Y</td>
<td>$</td>
<td>y-position of cable elements in m relative to tow-point</td>
<td>-0.10,-0.21, ...</td>
</tr>
<tr>
<td>CABLE_SHAPE_Z</td>
<td>$</td>
<td>z-position of cable elements in m relative to tow-point</td>
<td>0.00,0.00, ...</td>
</tr>
<tr>
<td>AEL_HEADING</td>
<td>D</td>
<td>average true heading in degrees of acoustic array section.</td>
<td></td>
</tr>
<tr>
<td>AEL_PITCH</td>
<td>D</td>
<td>average pitch in radians of acoustic array section.</td>
<td></td>
</tr>
</tbody>
</table>
4.4 uSimPassiveSonar

4.4.1 Brief Overview

This variable fidelity acoustic timeseries simulator uses a suite of embedded environmental acoustic models for simulating the acoustic propagation from the source to the array, including cylindrical spreading and bottom loss. It supports a range of acoustic arrays including hydrophones and directional motion sensors, using a simple generic output file format, which is easily interfaced to customized signal processing modules. The module supports three different levels of environmental acoustic fidelity:

1. Gaussian beam raytracing, supporting arbitrary range-independent sound speed profiles, reflection from the sea surface and a penetrable seabed. Uses the Bellhop legacy raytracing code to compute the ray arrival structure at a reference point and then uses a local WKB approximation to model the move-out of the arrivals along the array.

2. Rapid Acoustic Path (RAP) modeling. This mode uses an analytical model for the direct ray path connecting source and receiver in a stratified ocean with a linear sound speed profile in depth. No multipath is included in this mode.

3. Shallow Water Waveguide acoustics. This mode uses a medium fidelity cylindrical spreading model for shallow water environments. The field is represented by a single cylindrical wavefront exposed to a user-defined bottom loss.

4.4.2 Parameters for the uSimPassiveSonar Configuration Block

The following configuration parameters are defined for uSimPassiveSonar.

- **ray_acoustics**: Logical variable controlling the simulation fidelity. If set to true, the simulator will use the Bellhop raytracing code for computing the multipath array response. The number of boundary reflections included is controlled by the **max_number_bounces** configuration parameter. This is the highest level of environmental acoustic fidelity provided by uSimPassiveSonar. The environmental model is defined in the configuration for iBellhop.

- **rap_acoustics**: Logical variable controlling the simulation fidelity. If set to true, the simulator will assume the sound speed profile to be linear in depth, and use an analytical model for the ray arrival corresponding to the RAP-path (Reliable Acoustic Path), i.e. the direct arrival without any boundary reflections. This option includes the vertical directionality of the direct arrival.

- **max_number_bounces**: Only rays with the number of boundary reflections less than or equal to this parameter will be included. Default: 1. Only significant for **ray_acoustics = true**.

- **water_depth**: Local water depth (range-independent).

- **svp_surface**: Sound speed in water (m/s) at the sea surface and throughout water column for isovelocity waveguides

- **svp_gradient**: Sound speed depth gradient in water (1/s). Only significant for **rap_acoustics = true**.
bottom_loss: Estimate of average transmission loss due to bottom interaction. Specified in dB/km.

noise_level: Noise level in dB.

array_type: Array type, defining file format. Options are DURIP, VSA, SINGLE, or DOUBLE.

num_array_elements: Number of hydrophone elements in array.

sampling_frequency: Sampling frequency for synthesized time series in Hz.

num_samples: Number of samples per file.

hydrophone_gain: Conversion factor to dB rel. 1 Pa

base_dir_name: Path for directory to contain acoustic and non-acoustic data files.

acoustic_filename_prefix: Prefix for acoustic data file names

acoustic_filename_suffix: Extension for acoustic data file names

nonacoustic_filename_prefix: Prefix for non-acoustic (sensor geometries) data file names

nonacoustic_filename_suffix: Extension for non-acoustic (sensor geometries) data file names

Below is an example configuration block for the uSimPassiveSonar.

Listing 4.4 - An example uSimPassiveSonar configuration block.

```cpp
1 // ------------ uSimPassiveSonar Configuration block ----------
2 ProcessConfig = uSimPassiveSonar
3 {
4   AppTick = 12
5   CommsTick = 5
6 // Source Parameters
7 // Environmental Parameters
8   ray_acoustics = false
9   rap_acoustics = true
10  svp_surface = $(C_0)
11  svp_grad = $(C_GRAD)
12  max_rap_depth = $(MAX_RAP_DEPTH)
13  water_depth = $(WATER_DEPTH) // used for spherical spreading (m)
14  bottom_loss = 5.0 // Bottom loss in dB/km
15  noise_level = 60 // dB re 1 uPa
16 // Array and sampling parameters
17   array_type = DURIP // DURIP or VSA
18   num_array_elements = 32 // number of elements in the array
19   sampling_frequency = 4000 // Hz
20   num_samples = 8000 // number of samples per file
21   hydrophone_gain = 1000000000 // conversion from Pascal to float
22 // Output file location and name
23 // absolute path including initial and final forward slash.
24   base_dir_name = /home/henrik/DURIP/
25 // The output filename will be:
26 // [filename_prefix][frame_number][filename_suffix]
27   acoustic_filename_prefix = ACO
28   acoustic_filename_suffix = .DAT
29   nonacoustic_filename_prefix = NAS
30   nonacoustic_filename_suffix = .DAT
31 }
```
4.4.3 MOOS variables subscribed to by uSimPassiveSonar:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Published by</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOW_POS_X</td>
<td>D</td>
<td>UTM x-coordinate of array cable tow point.</td>
<td>uMVS_Bluefin</td>
</tr>
<tr>
<td>TOW_POS_Y</td>
<td>D</td>
<td>UTM y-coordinate of array cable tow point.</td>
<td>uMVS_Bluefin</td>
</tr>
<tr>
<td>TOW_POS_Z</td>
<td>D</td>
<td>UTM z-coordinate of array cable tow point.</td>
<td>uMVS_Bluefin</td>
</tr>
<tr>
<td>ARRAY_X</td>
<td>$</td>
<td>x-position of hydrophones in m relative to tow-point</td>
<td>uSimTowedArray</td>
</tr>
<tr>
<td>ARRAY_Y</td>
<td>$</td>
<td>y-position of hydrophones in m relative to tow-point</td>
<td>uSimTowedArray</td>
</tr>
<tr>
<td>ARRAY_Z</td>
<td>$</td>
<td>z-position of hydrophones in m relative to tow-point</td>
<td>uSimTowedArray</td>
</tr>
<tr>
<td>ARRAY_PITCH</td>
<td>$</td>
<td>pitch in radians of array elements</td>
<td>uSimTowedArray</td>
</tr>
<tr>
<td>ARRAY_HEADING</td>
<td>$</td>
<td>true heading in degrees of array elements</td>
<td>uSimTowedArray</td>
</tr>
<tr>
<td>TARGET_SIM_LIST</td>
<td>$</td>
<td>Comma-separated list of source numbers within critical distance</td>
<td>uSimTargets</td>
</tr>
<tr>
<td>TGT_##_SIM_STATE</td>
<td>$</td>
<td>Comma-separated list of source parameters for source number #</td>
<td>uSimTargets</td>
</tr>
<tr>
<td></td>
<td></td>
<td>#.on/off,utc,x,y,speed,heading,depth,spl,freq,bw</td>
<td></td>
</tr>
<tr>
<td>TGT_##_SIM_ARR_FILE</td>
<td>$</td>
<td>Path to file containing ray arrival time, angle and pressure level</td>
<td>iBellhop</td>
</tr>
</tbody>
</table>

4.4.4 MOOS variables published by uSimPassiveSonar:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>BELLHOP_REQUEST</td>
<td>$</td>
<td>Contact and simulation type information passed to i Bellhop</td>
<td>contact=TGT_##_SIM #output=arrival_times</td>
</tr>
<tr>
<td>TARGET_XPOS</td>
<td>D</td>
<td>UTM x position for current target for plotting by small_uVis.m</td>
<td></td>
</tr>
<tr>
<td>TARGET_YPOS</td>
<td>D</td>
<td>UTM x position for current target for plotting by small_uVis.m</td>
<td></td>
</tr>
</tbody>
</table>

4.4.5 uSimPassiveSonar Details

uSimPassiveSonar models acoustic sources that have a flat spectrum in addition to sinusoidal tones. In the default shallow water mode, uSimPassiveSonar takes into account spreading loss (spherical to the waveguide depth, cylindrical thereafter), as well as bottom loss. The waves impinging upon the hydrophones are cylindrical waves originating from the sources’ respective locations. Because the modeled waves are cylindrical, the depth of each hydrophone is irrelevant. In the higher fidelity modes, it uses raytracing to account for vertical refraction. In the ‘rap’ mode only the direct ray is included, while in the ‘ray’ mode, a full raytracing solution is applied, including all multiples within the ray fan specified in the environment file used by iBellhop.
4.5 uSimGPS

4.5.1 Brief Overview

This module is used for simulating GPS fixes during surfacing. Used e.g. in connection with BHV_PeriodicSurface behavior.

4.5.2 Parameters for the uSimGPS Configuration Block

The following configuration parameters are defined for uSimTargets.

\textbf{gps\_depth}: Depth above which GPS is assumed to be active.

\textbf{gps\_interval}: Time between GPS fixes in seconds

\textbf{min\_gps\_time}: Minimum time spent on surface for GPS fixes in seconds

Below is an example configuration block for the uSimTargets.

\textit{Listing 4.5 - An example uSimTargets configuration block.}

\begin{verbatim}
1 ProcessConfig = uSimGPS
2 {
3   AppTick = 1
4   CommsTick = 1
5   gps_depth = 0.5 // Depth at which GPS becomes active
6   gps_interval = 2 // Interval between gps fixes in seconds
7   min_gps_time = 10 // Minimum surface time for achieving GPS
8 }
\end{verbatim}

4.5.3 MOOS variables subscribed to by uSimGPS:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Published by</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAV_X</td>
<td>D</td>
<td>UTM x-coordinate of vehicle.</td>
<td>iHuxley</td>
</tr>
<tr>
<td>NAV_Y</td>
<td>D</td>
<td>UTM y-coordinate of vehicle.</td>
<td>iHuxley</td>
</tr>
<tr>
<td>NAV_DEPTH</td>
<td>D</td>
<td>Vehicle depth in meter.</td>
<td>iHuxley</td>
</tr>
</tbody>
</table>

4.5.4 MOOS variables published by uSimGPS:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>SECS_TO_DIVE</td>
<td>D</td>
<td>Time in seconds remaining before diving</td>
<td></td>
</tr>
</tbody>
</table>
| GPS\_UPDATE\_RECEIVED$ |     | GPS 'measurement'            | Timestamp=122..,
                              |                          | Latitude=42.12345,     |
                              |                          | Longitude=10.98765      |
4.6 uSimCTD

4.6.1 Brief Overview

This module is used for simulating CTD data using HOPS-generated virtual ocean or real ocean database.

4.6.2 Parameters for the uSimCTD Configuration Block

The following configuration parameters are defined for uSimCTD.

\texttt{ Mods\_Bin\_File: Name of binary file containing CTD database.}

Below is an example configuration block for the uSimCTD.

\textit{Listing 4.6 - An example uSimCTD configuration block.}

1 ProcessConfig = uSimCTD
2 {
3   AppTick = 5
4   CommsTick = 5
5   Mods\_Bin\_File = ../../../data/may07_ctdsim2.bin
6 }

4.6.3 MOOS variables subscribed to by uSimCTD:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Published by</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAV_LONG</td>
<td>D</td>
<td>Longitude of vehicle in degrees.minutes.</td>
<td>iHuxley</td>
</tr>
<tr>
<td>NAV_LAT</td>
<td>D</td>
<td>Latitude of vehicle in degrees.minutes.</td>
<td>iHuxley</td>
</tr>
<tr>
<td>NAV_DEPTH</td>
<td>D</td>
<td>Depth of vehicle in meter.</td>
<td>iHuxley</td>
</tr>
</tbody>
</table>

4.6.4 MOOS variables published by uSimCTD:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTD1</td>
<td>$</td>
<td>Message concatenating published and subscribed fields</td>
<td></td>
</tr>
<tr>
<td>CTD_SOUND_VELOCITY</td>
<td>D</td>
<td>Sound velocity in m/s given longitude, latitude and depth of vehicle</td>
<td></td>
</tr>
<tr>
<td>CTD_TEMPERATURE</td>
<td>D</td>
<td>Temperature in degree Celsius given longitude, latitude and depth of vehicle</td>
<td></td>
</tr>
<tr>
<td>CTD_SALINITY</td>
<td>D</td>
<td>Salinity in PSU given longitude, latitude and depth of vehicle</td>
<td></td>
</tr>
</tbody>
</table>
4.7 uSimBathy

4.7.1 Brief Overview

This module is used for simulating bathymetry data using bathymetry table.

4.7.2 Parameters for the uSimBathy Configuration Block

The following configuration parameters are defined for uSimBathy.

Bathy Bin File: Name of binary file containing bathymetry table.

water depth: Default water depth in meters, used outside area covered by bathymetry file.

Below is an example configuration block for the uSimBathy.

Listing 4.7 - An example uSimBathy configuration block.

```plaintext
ProcessConfig = uSimBathy
{
  AppTick = 5
  CommsTick = 5
  Bathy_Bin_File = /home/raylum/project-plusnet/data/monterey.xyz.bin
  water_depth = 200
}
```

4.7.3 MOOS variables subscribed to by uSimBathy:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Published by</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAV_X</td>
<td>D</td>
<td>x position of vehicle in meter.</td>
<td>iHuxley</td>
</tr>
<tr>
<td>NAV_Y</td>
<td>D</td>
<td>y position of vehicle in meter.</td>
<td>iHuxley</td>
</tr>
<tr>
<td>NAV_Z</td>
<td>D</td>
<td>z position of vehicle in meter.</td>
<td>iHuxley</td>
</tr>
</tbody>
</table>

4.7.4 MOOS variables published by uSimBathy:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>BATHY</td>
<td>$</td>
<td>Message concatenating published and subscribed fields</td>
<td></td>
</tr>
<tr>
<td>BATHY_Z</td>
<td>D</td>
<td>Depth of water column given x and y positions of vehicle</td>
<td></td>
</tr>
</tbody>
</table>
4.8 iMseasBathy

4.8.1 Brief Overview
This module is used for simulating bathymetry data using an MSEAS ocean model. It applies linear interpolation based on a lat/lon grid of bathymetry points stored in netCDF format.

4.8.2 Parameters for the iMseasBathy Configuration Block
The following configuration parameters are defined for iMseasBathy.

\textbf{nc} \_\text{ts} \_\text{file} \_\text{loc}: Name of MSEAS ts ("temperature-salinity") netCDF file containing bathymetry data, as well as other environmental data specifying the model ocean.

\textbf{publish} \_\text{interval}: Number of seconds between publishes of MSEAS bathymetry data.

Below is an example configuration block for iMseasBathy.

\textit{Listing 4.8 - An example iMseasBathy configuration block.}

```
1 ProcessConfig = iMseasBathy
2 {
3   // available to all moos processes.
4   AppTick = 4
5   CommsTick = 4
6
7   // all case insensitive
8   nc_ts_file_loc = /home/spetillo/Desktop/mseas_netcdf_files/pe_out_ts.nc
9
10  // number of seconds between publishes of mseas bathy data
11  publish_interval = 5;
12 }
```

4.8.3 MOOS variables subscribed to by iMseasBathy:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Published by</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAV_LAT</td>
<td>D</td>
<td>Latitude of vehicle in decimal degrees North.</td>
<td>iHuxley</td>
</tr>
<tr>
<td>NAV_LONG</td>
<td>D</td>
<td>Longitude of vehicle in decimal degrees East.</td>
<td>iHuxley</td>
</tr>
</tbody>
</table>

4.8.4 MOOS variables published by iMseasBathy:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>BATHY_Z</td>
<td>D</td>
<td>Depth (negative down) in meters of water column given latitude and longitude of vehicle. Defaults to -100m bathymetry if AUV is over 5km from the closest lat/lon grid point.</td>
<td></td>
</tr>
</tbody>
</table>
4.9 iMseas

4.9.1 Brief Overview

This module is used to read temperature, salinity, sound speed, and zonal and meridional current data from MSEAS spatiotemporal simulated ocean models. It applies a C wrapper to the read-hopspe.m script provided by the MSEAS group at MIT. MSEAS ocean model data are stored in netCDF format.

4.9.2 Parameters for the iMseas Configuration Block

The following configuration parameters are defined for iMseas.

verbosity: Verbosity of the output to the screen (verbose, terse, or quiet).

c_tsf_file_loc: Name of MSEAS ts (“temperature-salinity”) netCDF file containing hydrographic data specifying the model ocean.

c_file_loc: Name of MSEAS v (“velocity”) netCDF file containing water velocity data specifying the model ocean. We prefer to use the rotated local grid (parallel to the coastline) specified by “vrot” in the file name.

publish_interval: Number of seconds between publishes of MSEAS environmental data.

Below is an example configuration block for iMseas.

Listing 4.9 - An example iMseas configuration block.

```
1 ProcessConfig = iMseas
2 {
3   // available to all moos processes.
4   AppTick = 4
5   CommsTick = 4
6   // available to all moos processes
7   verbosity = verbose
8   // all case insensitive
9   nc_ts_file_loc = /Users/spetillo/Desktop/mseas_netcdf_files/pe_out_ts.nc
10  nc_v_file_loc = /Users/spetillo/Desktop/mseas_netcdf_files/pe_out_vrot.nc
11  publish_interval = 5;
12 }
```

4.9.3 MOOS variables subscribed to by iMseas:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Published by</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAV_LAT</td>
<td>D</td>
<td>Latitude of vehicle in decimal degrees North.</td>
<td>iHuxley</td>
</tr>
<tr>
<td>NAV_LONG</td>
<td>D</td>
<td>Longitude of vehicle in decimal degrees East.</td>
<td>iHuxley</td>
</tr>
<tr>
<td>NAV_DEPTH</td>
<td>D</td>
<td>Depth of vehicle in meters (positive down).</td>
<td>iHuxley</td>
</tr>
</tbody>
</table>
4.9.4 MOOS variables published by iMseas:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSEAS_TEMPERATURE</td>
<td>D</td>
<td>Temperature in degrees C at vehicle location in space and time.</td>
<td></td>
</tr>
<tr>
<td>MSEAS_SALINITY</td>
<td>D</td>
<td>Salinity at vehicle location in space and time.</td>
<td></td>
</tr>
<tr>
<td>MSEAS_VELOCITY_U</td>
<td>D</td>
<td>Water velocity in the zonal direction in m/s at vehicle location in space and time.</td>
<td></td>
</tr>
<tr>
<td>MSEAS_VELOCITY_V</td>
<td>D</td>
<td>Water velocity in the meridional direction in m/s at vehicle location in space and time.</td>
<td></td>
</tr>
<tr>
<td>MSEAS_SOUNDSPEED</td>
<td>D</td>
<td>Mackenzie sound speed in m/s at vehicle location in space and time.</td>
<td></td>
</tr>
</tbody>
</table>
4.10  ctd\_disp.m

4.10.1  Brief Overview

This MATLAB script is run during an actual or simulated AUV mission to graphically display CTD data in real-time, as shown in Fig. ??.

4.10.2  Parameters for the ctd\_disp configuration Block

The following configuration parameters are defined for ctd\_disp.

SUBSCRIBE: MOOS variable subscribed to by ctd\_disp.m.

Below is an example configuration block for ctd\_disp.

Listing 4.10 - An example ctd\_disp configuration block.

```
1 //+++++++++++++++++++++++++++++++
2 // global variables anyone can use them
3
4 ServerHost = localhost
5 //ServerHost = 192.168.2.113
6 ServerPort = 9123
7 Community = topside
8
9 //+++++++++++++++++++++++++++++++
10 ProcessConfig = ctd\_disp
11 {
12   AppTick = 10
13   CommsTick = 10
14   Port = COM6
15   BaudRate = 4800
16   Verbose = true
17   Streaming = false
18   MOOSComms = true
19   SerialComms = false
20
21 SUBSCRIBE = IN\_CTD\_254B\_DEC @ 0
22 SUBSCRIBE = IN\_CTD\_62B\_DEC @ 0
23 SUBSCRIBE = IN\_CTD\_30B\_DEC @ 0
24 }
```

4.10.3  MOOS variables subscribed to by ctd\_disp.m:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Published by</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN_CTD_254B_DEC</td>
<td>D</td>
<td>Decoded string of CTD data.</td>
<td>pCTDCodec</td>
</tr>
<tr>
<td>IN_CTD_62B_DEC</td>
<td>D</td>
<td>Decoded string of CTD data.</td>
<td>pCTDCodec</td>
</tr>
<tr>
<td>IN_CTD_30B_DEC</td>
<td>D</td>
<td>Decoded string of CTD data.</td>
<td>pCTDCodec</td>
</tr>
</tbody>
</table>

4.10.4  MOOS variables published by ctd\_disp.m:

None.
4.11 Arraysim.m

4.11.1 Brief Overview

This is the MATLAB version of the array dynamics simulator. The functionality of the two are identical, and they use identical MOOSDB interface definitions.

4.11.2 Configuration Files

The array information is defined in the configuration file config.txt: For description of the significance of each parameter, see the description for the equivalents for uSimTowedArray.

Listing 4.11 - An example uSimTowedArray configuration block (MIT DURIP towed array).

```
1 moos_file    Array.moos
2 moos_name    ArraySim
3 cond_check  1
4 start_speed  0.5
5 create_dir_frame  0
6 write_out_files  0
7 filter_length  40
```

and a cable definition file, which for the MIT DURIP array is cable_durip.dat. Again the significance of the parameters is identical to the uSimTowedArray equivalents.

```
0 Parameters for DURIP array with 3 sections (tow cable+acoustic section+drogue)
1 3
2 20 20.0   .0000   .0095   .001   0.2   1.6e9
3 30 36.0   .0000   .035    .0024  0.3   1.6e9
4 20 20.0   .0000   .0095   .001   0.2   1.6e9
5 0.0 0.0    0.0
```

The remaining parameters, such as those defining hydrophone spacing etc. are specified internally, based on the MOOS variable VEHICLE_ID, which for DURIP should be 3 (Unicorn). This in contrast to the C++ version uSimTowedArray which is entirely generic, with all variables specified in the configuration file.

4.11.3 MOOS variables subscribed to:

. 
<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Published by</th>
</tr>
</thead>
<tbody>
<tr>
<td>VSA_DIR</td>
<td>$</td>
<td>Directory containing files with acoustic and non-acoustic array data</td>
<td>iVSA</td>
</tr>
<tr>
<td>VSA_FRAME</td>
<td>D</td>
<td>Frame number for acoustic and non-acoustic data files in VSA_DIR</td>
<td>iVSA</td>
</tr>
<tr>
<td>VEHICLE_ID</td>
<td>D</td>
<td>Node number for ownship. Used for selecting dynamic parameters</td>
<td>pAcommsHandler</td>
</tr>
<tr>
<td>TOW_POS_X</td>
<td>D</td>
<td>UTM x-coordinate of array cable tow point.</td>
<td>uMVS, Bluefin</td>
</tr>
<tr>
<td>TOW_POS_Y</td>
<td>D</td>
<td>UTM y-coordinate of array cable tow point.</td>
<td>uMVS, Bluefin</td>
</tr>
<tr>
<td>TOW_POS_Z</td>
<td>D</td>
<td>UTM z-coordinate of array cable tow point.</td>
<td>uMVS, Bluefin</td>
</tr>
<tr>
<td>TOW_VEL_X</td>
<td>D</td>
<td>x-velocity in m/s of array cable tow point.</td>
<td>uMVS, Bluefin</td>
</tr>
<tr>
<td>TOW_VEL_Y</td>
<td>D</td>
<td>y-velocity in m/s of array cable tow point.</td>
<td>uMVS, Bluefin</td>
</tr>
<tr>
<td>TOW_VEL_Z</td>
<td>D</td>
<td>z-velocity in m/s of array cable tow point</td>
<td>uMVS, Bluefin</td>
</tr>
</tbody>
</table>

### 4.11.4 MOOS variables published:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>CABLE_TENSION</td>
<td>D</td>
<td>Tension of tow cable in N at tow point</td>
<td></td>
</tr>
<tr>
<td>ARRAY_X</td>
<td>$</td>
<td>x-position of hydrophones in m relative to tow-point</td>
<td>17.23,18.33, ...</td>
</tr>
<tr>
<td>ARRAY_Y</td>
<td>$</td>
<td>y-position of hydrophones in m relative to tow-point</td>
<td>-6.13,-6.33, ...</td>
</tr>
<tr>
<td>ARRAY_Z</td>
<td>$</td>
<td>z-position of hydrophones in m relative to tow-point</td>
<td>0.01,0.02, ...</td>
</tr>
<tr>
<td>ARRAY_PITCH</td>
<td>$</td>
<td>pitch in radians of array elements</td>
<td>0.01,0.00, ...</td>
</tr>
<tr>
<td>ARRAY_HEADING</td>
<td>$</td>
<td>true heading in degrees of array elements</td>
<td>301.01,300.99, ...</td>
</tr>
<tr>
<td>CABLE SHAPE_X</td>
<td>$</td>
<td>x-position of cable elements in m relative to tow-point</td>
<td>0.43,0.84, ...</td>
</tr>
<tr>
<td>CABLE SHAPE_Y</td>
<td>$</td>
<td>y-position of cable elements in m relative to tow-point</td>
<td>-0.10,-0.21, ...</td>
</tr>
<tr>
<td>CABLE SHAPE_Z</td>
<td>$</td>
<td>z-position of cable elements in m relative to tow-point</td>
<td>0.00,0.00, ...</td>
</tr>
<tr>
<td>'auv'AEL_HEADING</td>
<td>D</td>
<td>average true heading in degrees of acoustic array section. The preample 'auv' is defined by VEHICLE_ID, e.g UNICORN.</td>
<td></td>
</tr>
<tr>
<td>'auv'AEL_PITCH</td>
<td>D</td>
<td>average pitch in radians of acoustic array section</td>
<td></td>
</tr>
</tbody>
</table>
4.12 SealabMultiSim.m

4.12.1 Brief Overview

This is the MATLAB-based interface between the MOOSDB and the Sealab sonar simulation environment. Sealab is a commercial product (VASA Associates Inc.), which uses state-of-the-art legacy propagation models and a native 3-D coupled mode model to produce high-fidelity element-level timeseries simulation for arbitrary 3D arrays and source configurations in a complex ocean model. It incorporates ocean wave-guide acoustic effects including mode coupling, Doppler spread etc. It is sufficiently efficient for producing high-fidelity simulated data in real time on a single platform.

4.12.2 Configuration MOOS-block

//.moos for SealabMultiTarget
ProcessConfig = SealabMultiTarget
{
    AppTick = 10
    CommsTick = 10
    Port = COM6
    BaudRate = 4800
    Verbose = true
    Streaming = false
    MOOSComms = true // Publish output to MOOSDB
    SerialComms = false // Send output over serial line.
    SERIAL_TIMEOUT = 10.0
    SUBSCRIBE = VEHICLE_ID @ 0
    SUBSCRIBE = DB_TIME @ 0
    SUBSCRIBE = NAV_SPEED @ 0
    SUBSCRIBE = TOW_POS_X @ 0
    SUBSCRIBE = TOW_POS_Y @ 0
    SUBSCRIBE = TOW_POS_Z @ 0
    SUBSCRIBE = TOW_VEL_X @ 0
    SUBSCRIBE = TOW_VEL_Y @ 0
    SUBSCRIBE = TOW_VEL_Z @ 0
    SUBSCRIBE = ARRAY_X @ 0
    SUBSCRIBE = ARRAY_Y @ 0
    SUBSCRIBE = ARRAY_Z @ 0
    SUBSCRIBE = ARRAY_PITCH @ 0
    SUBSCRIBE = ARRAY_HEADING @ 0
    SUBSCRIBE = TARGET_SIM_LIST @ 0
    SUBSCRIBE = VSA_DIR @ 0
    SUBSCRIBE = VSA_FRAME @ 0
}
4.12.3 MOOS variables subscribed to:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Published by</th>
</tr>
</thead>
<tbody>
<tr>
<td>VEHICLE_ID</td>
<td>D</td>
<td>Node number for ownship. Used for selecting dynamic parameters</td>
<td>pAcommsHandler</td>
</tr>
<tr>
<td>TOW_POS_X</td>
<td>D</td>
<td>UTM x-coordinate of array cable tow point.</td>
<td>uMVS_Bluefin</td>
</tr>
<tr>
<td>TOW_POS_Y</td>
<td>D</td>
<td>UTM y-coordinate of array cable tow point.</td>
<td>uMVS_Bluefin</td>
</tr>
<tr>
<td>TOW_POS_Z</td>
<td>D</td>
<td>UTM z-coordinate of array cable tow point.</td>
<td>uMVS_Bluefin</td>
</tr>
<tr>
<td>TOW_VEL_X</td>
<td>D</td>
<td>x-velocity of array cable tow point. For doppler.</td>
<td>uMVS_Bluefin</td>
</tr>
<tr>
<td>TOW_VEL_Y</td>
<td>D</td>
<td>y-velocity of array cable tow point. For doppler.</td>
<td>uMVS_Bluefin</td>
</tr>
<tr>
<td>TOW_VEL_Z</td>
<td>D</td>
<td>z-velocity of array cable tow point. For doppler.</td>
<td>uMVS_Bluefin</td>
</tr>
<tr>
<td>ARRAY_X</td>
<td>$</td>
<td>x-position of hydrophones in m relative to tow-point</td>
<td>uSimTowedArray</td>
</tr>
<tr>
<td>ARRAY_Y</td>
<td>$</td>
<td>y-position of hydrophones in m relative to tow-point</td>
<td>uSimTowedArray</td>
</tr>
<tr>
<td>ARRAY_Z</td>
<td>$</td>
<td>z-position of hydrophones in m relative to tow-point</td>
<td>uSimTowedArray</td>
</tr>
<tr>
<td>ARRAY_PITCH</td>
<td>$</td>
<td>pitch in radians of array elements</td>
<td>uSimTowedArray</td>
</tr>
<tr>
<td>ARRAY_HEADING</td>
<td>$</td>
<td>true heading in degrees of array elements</td>
<td>uSimTowedArray</td>
</tr>
<tr>
<td>VSA_DIR</td>
<td>$</td>
<td>Directory containing files with acoustic and non-acoustic array data</td>
<td>iVSA</td>
</tr>
<tr>
<td>VSA_FRAME</td>
<td>D</td>
<td>Frame number for acoustic and non-acoustic data files in VSA_DIR</td>
<td>iVSA</td>
</tr>
<tr>
<td>TARGET_SIM_LIST</td>
<td>$</td>
<td>Comma-separated list of source numbers within critical distance</td>
<td>uSimTargets</td>
</tr>
<tr>
<td>TGT_#SIM_STATE</td>
<td>$</td>
<td>Comma-separated list of source parameters for source number # #, on/off, utc, x, y, speed, heading, depth, spl, freq, bw</td>
<td>uSimTargets</td>
</tr>
</tbody>
</table>

4.12.4 MOOS variables published:

<table>
<thead>
<tr>
<th>MOOS variable</th>
<th>Type</th>
<th>Description</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>TARGET_XPOS</td>
<td>D</td>
<td>UTM x position for current target for plotting by small_uVis.m</td>
<td></td>
</tr>
<tr>
<td>TARGET_YPOS</td>
<td>D</td>
<td>UTM x position for current target for plotting by small_uVis.m</td>
<td></td>
</tr>
</tbody>
</table>
References
