**OCEAN21 Lightweight Nav to Custom Control Software ICD**

**Author:** Miles Liu

**Contact**: [mliu59@jhu.edu](mailto:mliu59@jhu.edu)

**This document should only be accessed by employees of Oceaneering International, Inc. or those individuals with a non-disclosure agreement with Oceaneering.**

**0.1 Relevant Abbreviations**

VTA – Vehicle Testing Asset (current project)

ROV – Remotely Operated Vehicle

TCP – Transmission Control Protocol

**1.0 System Interface Description.**

The interface is built to facilitate the retrieval of ROV vehicle state from Oceaneering's Lightweight Nav system to the OCEAN21 VTA surface vehicle control software.

This vehicle state information is used to define a moving target for the VTA's autopilot to follow.

Ultimately, the goal is to have the VTA proactively keep within a threshold distance of the ROV during its operation to prevent the wired connection to the VTA from ever coming in tension.

This system is built to function side-by-side with the mechanical based “tug steering” solution. Depending on the operational reliability of the active navigation system, the control software may opt to switch over to the **more reliable** but lower performance (comparatively higher amounts of tension) “tug steering” solution. Contact Miles Liu at [mliu59@jhu.edu](mailto:mliu59@jhu.edu) for information on the evaluation metrics and tug steering solution.

**1.1 Configurations.**

The OCEAN21 VTA vehicle control software will receive data packets through a TCP connection. So, the only hardware connection needed is an ethernet cable compatible with a Linux/Windows machine.

**1.2 Communications Link.**

The Lightweight Nav side will be the TCP host server, and the VTA control software will be the client connection.

TCP can send byte data to and from a client & server, but connection must be established in order for the data to be sent. On the lightweight nav side, there will need to be a TCP server that allows for client connections (VTA control software as a client).

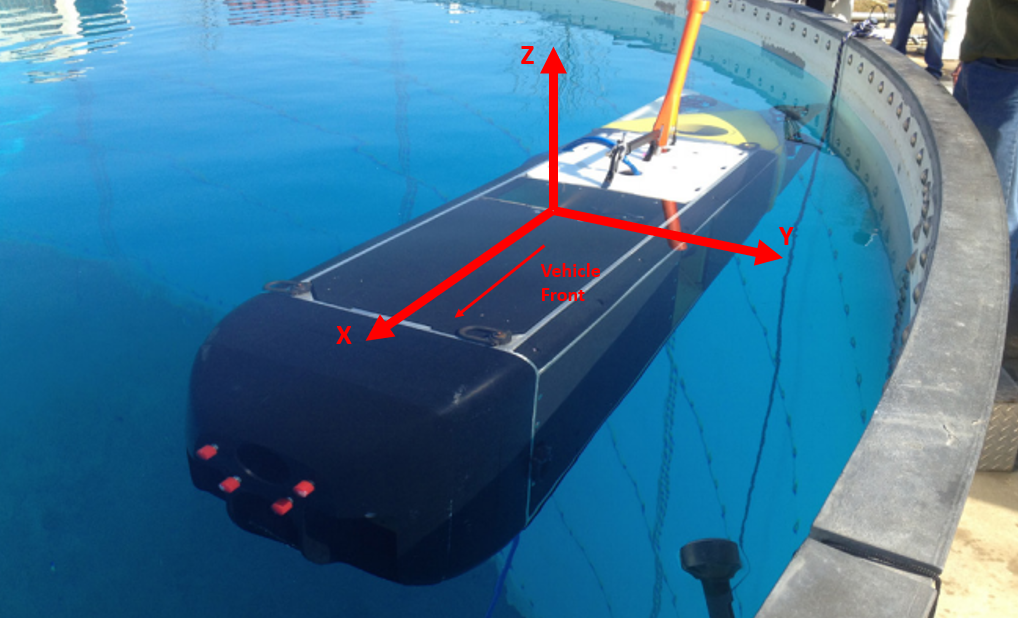
**2.0 System Basics.**

The data packets being sent should be a string converted to a byte packet with the UTF-8 encoding.

For this specific application, the ROV vehicle state packets will be sent at a frequency of **1 Hz**, functioning as a heartbeat for the vehicle.

Vehicle state should always be passed at the above frequency. A communications error will be flagged on the control software side if no packets arrive in **10 seconds**, and the VTA will revert to the mechanical tug steering solution.

**2.1 Vehicle Axis Convention**

****

X axis points to vehicle front, Y axis points to vehicle left, Z axis points vertically upwards.

Roll represents rotation about the X axis, Pitch represents rotation about the Y axis, and Yaw represents rotation about the Z axis.

The system will follow right-hand-rule convention for rotations. When facing the same direction as a positive axis, CW rotation is positive.

 [Image Source](https://bobcad.com/tech-tuesday-understanding-the-right-hand-rule-and-the-cartesian-coordinate-system/)

**3.0 Interface – Packet Descriptions.**

The packet being sent from the server to client contains the ROV vehicle state data, sufficient to help the VTA control software determine the last known location for the ROV.

ROV speed data should also be included for the control software to be able to predict the ROV's navigational direction and determine if the ROV vehicle state data is accurate. This will also be helpful for the development of an ROV watchdog.

**3.1 ROV Vehicle State Data Packet**

General Format:

<0>,<1>,<2>,<3>,<4>,<5>,<6>,<7>,<8>,<9>,<10>,<11>\*

General note: positive values **do not** have the ‘+’ symbol before the numeric representation, negative values will require ‘-’.

Message fields

Field Meaning

0 Message Status: A single character field that is used to represent message state. ‘0’ = Vehicle state is good and can be used for navigation. Any other character suggests that the data is faulty/not reliable, and should be discarded. This field can be used in the future to denote other message flags.

1 Message Type: D = incoming ROV vehicle state data, K = kill, any other value means invalid packet.

2 ROV Latitude Info: Latitude in degrees decimal (range: -90.000000 to 90.000000), accurate to 6 decimal places.

3 ROV Longitude Info: Longitude in degrees decimal (range: -180.000000 to 180.000000), accurate to 6 decimal places.

4 ROV Depth Info: magnitude of ROV depth in meters relative to the surface, accurate to 1 decimal place

5 ROV Roll: Roll (rotation about X) angle in degrees, accurate to 1 decimal place (range: -180.0 to 180.0)

6 ROV Pitch: Pitch (rotation about Y) angle in degrees, accurate to 1 decimal place (range: -180.0 to 180.0)

7 ROV Yaw: Yaw (rotation about Z) angle in degrees, accurate to 1 decimal place. (range: -180.0 to 180.0) Vehicle yaw is equivalent to true heading. At 0 degrees yaw, the vehicle points to true north.

8 ROV X velocity: X velocity in m/s, accurate to 1 decimal place (Ex. ‘-0.2’, ‘6.2’)

9 ROV Y velocity: Y velocity in m/s, accurate to 1 decimal place (Ex. ‘-0.2’, ‘6.2’)

10 ROV Z velocity: Z velocity in m/s, accurate to 1 decimal place. (Ex. ‘-0.2’, ‘6.2’) When vehicle is upright, positive Z represents decrease in depth (ascent), and negative Z represents increase in depth (descent).

11 String End Character: ‘\*’, always present

Each field is delimited by a comma ‘,’.

For fields 2 to 10, if the specific field data is unavailable / has an ERROR, replace with “NAN”.

Examples of the data message string:

0,D,39.331595,-76.616543,2.2,5.2,3.2,45.1,2.1,-0.2,-0.3\*

0,D,39.332142,-76.617961,1.0,3.2,NAN,-60.3,0.1,NAN,-1.2\*

If the ROV system is shutdown, or if the server connection is to be disconnected, a kill message would be sent (all fields except 0, 1, and 11 would be empty):

0,K,,,,,,,,,\*