Task Descriptions

1. Purpose

The purpose of this document is to outline, at a high-level, the individual tasks that will constitute the 2019 Virtual RobotX (VRX) competition. The details of the individual tasks, including their implementation and scoring, are under active development. This document, along with the preliminary VRX Technical Guide, are being made available as a means for competitors to provide feedback early in the design of this new competition.

2. Goal and Approach

The VRX competition is a means of supporting engineering development for the Maritime RobotX Challenge. Participation in this competition should raise the level of vehicle performance at the Maritime RobotX Challenge. The tasks and scoring reflect this intent by emphasizing the foundational capabilities that lead to better autonomous performance. Testing autonomy algorithms in a relevant simulation environment is an efficient and cost-effective approach when compared to the challenges of testing system performance in a physical environment.

The simulation-based competition also rewards robust, repeatable performance by scoring each task over multiple trials where the environmental conditions (e.g., sea state, wind magnitude and direction, lighting, etc.) are varied between trials.

3. Preliminary Descriptions of Tasks

The task descriptions below are preliminary. They will be refined and modified incorporating feedback from the RobotX community. The inaugural VRX competition focuses on the capabilities of the WAM-V unmanned surface vehicle (USV), one subsystem of the larger RobotX autonomous maritime system (AMS).

For each task the scoring is based on rank ordering of task-specific metrics discussed below. The overall competition score is determined by the sum of the rank ordering for the individual tasks.

3.1. Level 1: Control and Perception Fundamentals

3.1.1. Task 1: Station-Keeping

Capability: System should be capable of 1) performing localization by fusing sensor data (e.g., Global Positioning System (GPS), inertial measurement unit (IMU), etc.) and 2) maintaining USV position and heading in the presence of environmental forcing (e.g., wind and waves).

Implementation: Simulation is initiated with USV near the goal pose (position and heading). The goal pose is supplied via the ROS API (See VRX Technical Guide for details) and will be published to the `/vrx/station_keeping/goal` ROS topic. Each simulation run continues until the maximum time is exceeded. During the run, a 2D pose error measuring the difference in position and heading between the true position of the USV and the goal pose is calculated continuously and published to the `/vrx/station_keeping/pose_error` ROS topic. The overall error for each run is calculated as the root-mean-square (RMS) of this instantaneous 2D pose error over the duration of the run. The overall error across all runs is the mean of the individual RMS error values. The final task score is determined by rank ordering the participants based on the total error.

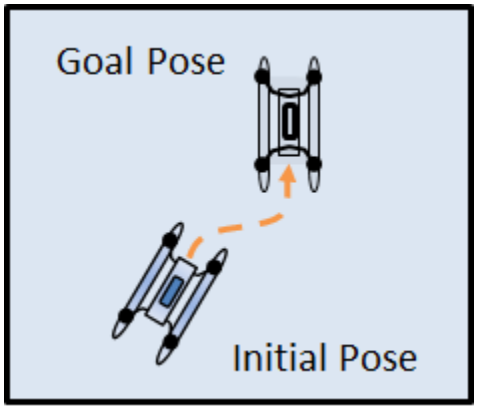

Figure 1: Station-Keeping poses

Table 1: Station Keeping API

Topic Name	Message Type	Description
`/vrx/station_keeping/goal`	`Geographic_msgs::GeoPoseStamped`	The goal pose, consisting of a position in spherical (WGS84) coordinates and a heading, given as a quaternion.
`/vrx/station_keeping/pose_error`	`std_msgs::Float64`	A 1 Hz sample of the current 2D pose error metric, which summarizes the current difference between USV and goal in terms of position and heading.
`/vrx/station_keeping/rms_error`	`std_msgs::Float64`	The total RMS pose error accumulated over the duration of the run so far.

3.1.2. Task 2: Wayfinding

Capability: System should be capable of command and control of USV to achieve a series of given goal states, specified as a series of locations/heading values.

Implementation: Simulation is initialized with USV in a random location within the operating area. The full series of goal states (waypoints) are provided via the ROS API; these are published to the `/vrx/wayfinding/waypoints` ROS topic when the task enters the "ready" state. The simulation continues until the the maximum time is exceeded. Teams are free to visit the goal states in any order. In each run of the simulation, a 2D pose error measuring the difference in position and heading between the true position of the USV and each of the waypoints is calculated continuously. The minimum error achieved so far for each waypoint is published to the `/vrx/wayfinding/min_errors` ROS topic. The overall state error is the mean of these minimum error values for each goal, published to the `/vrx/wayfinding/mean_error` ROS topic. The error across all runs is the average of the final mean errors achieved for each individual run. The task score is determined by rank ordering participants based on a weighted combination of overall state error and elapsed simulation time.

Table 2: Wayfinding API

Topic Name	Message Type	Description
`/vrx/wayfinding/waypoints`	`Geographic_msgs::GeoPath`	An array of waypoints, each consisting of a position given in spherical (WGS84) coordinates and a heading given as a quaternion.
`/vrx/wayfinding/min_errors`	`Std_msgs::Float64MultiArray`	An array containing the minimum 2D pose error so far achieved between the USV and each waypoint.
`/vrx/wayfinding/mean_error`	`std_msgs::Float64`	The mean of the minimum errors so far achieved for all waypoints.

3.1.3. Task 3: Landmark Localization and Characterization

Capability: Using perceptive sensors (cameras, LiDAR, etc.), system should be capable of identifying, characterizing and localizing given RobotX landmarks including buoys, totems, placards and docks. Perception should be robust with respect to vehicle motion (heave, pitch and roll) and environmental conditions (lighting, camera noise, etc.).

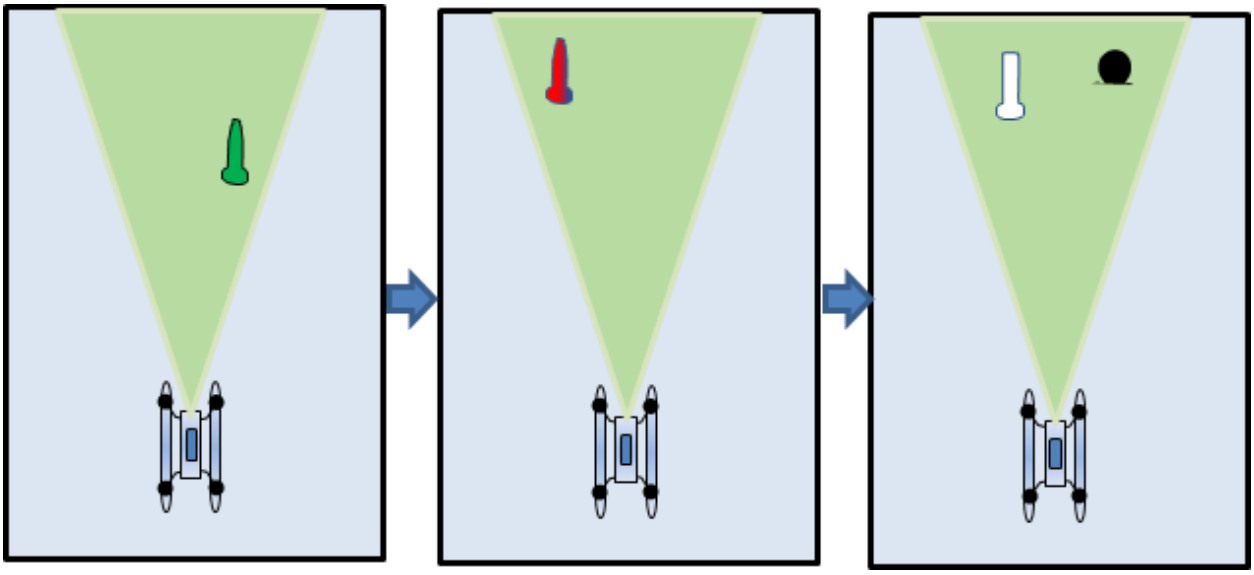

Figure 2: Three individual cases during the task as markers and objects are sequentially added to the field of view.

Implementation: Simulation is initiated with USV in a fixed location. The USV remains fixed for the duration of the tasks in the X (surge), Y (sway) and yaw degrees of freedom, but free to move in Z (heave), pitch and roll. During the task, a series of simulated RobotX landmarks will be generated within the field of view of the USV. Teams will locate and identify landmarks and communicate their results via the ROS API.

Two scores make up the overall task score. First, the identification score is determined by rank ordering teams based on the total number of correct object identifications. Second, the localization score is determined by rank ordering teams based on the total RMS position error between the estimated positions of the center-of-gravity of the objects and the true location of the simulated objects. The overall task score is the rank ordering of the combined identification score and localization score. In the event of a tie, the higher rank goes to the team with the smallest total RMS position error.

Table 3: Landmark localization and characterization API

Topic Name	Message Type	Description
`/vrx/perception/landmark`	`Geographic_msgs::GeoPoseStamped`	Teams report their estimated location (latitude and longitude) of a detected landmark. The identification of the landmark is reported using the message header `frame_id` string.

3.2. Level 2: Integrating Control and Perception

3.2.1. Task 4: Traverse Navigation Channel

Capability: System should be capable of creating and executing a motion plan to traverse a navigation channel specified by red and green markers. The solution should be robust with respect to environmental forcing and obstacles within the channel.

Implementation: Simulation is initiated with USV in a random location in the vicinity of the entrance gate (white-red buoys). Navigation course consists of series of gates; each gate is defined by a red and a green marker. Obstacles are included within the navigation channel. The simulation continues until the maximum time is exceeded or the team passes through the exit gate. For each simulation run, the gate score is determined as the sum of the gates that the USV crosses through. The direction of travel is specified based on red, right, return. Gates must be traversed in the appropriate direction. For each simulation run the team score is determined by rank ordering the gate score (number of gates successfully traversed) minus the number of obstacle collisions. Ties in the rank order are broken based on the elapsed time. The total task score is determined by the sum of the score from each individual run. In the event of a tie in the total task score, the tie is broken by the total elapsed time for all runs.

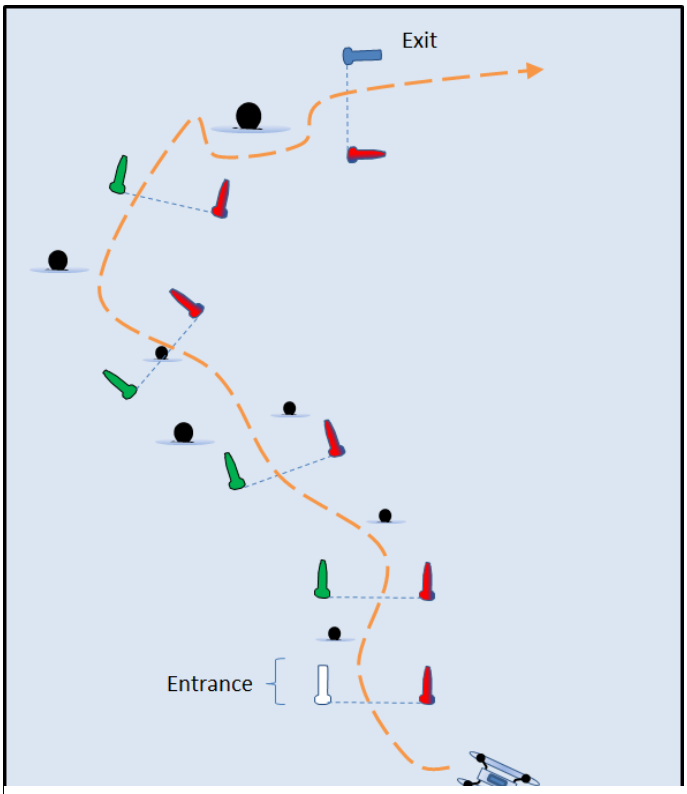

Figure 3: Navigation Channel

3.2.2. Task 5: Scan-the-Code and Dock

Capability: Given multiple docking bays (similar to the arrangement in the RobotX physical competition) the USV should be capable of 1) deciding on the appropriate bay for docking, 2) executing a safe and controlled docking maneuver and then 3) exiting the dock.

There are two versions of this task.

Implementation A: Simulation is initiated with USV in a random location in the vicinity of the dock. The correct docking bay is determined by the color and shape of the placard. The correct color and shape is provided via the ROS API. A maximum simulation time is specified. The USV must dock in the correct docking bay. Successful docking consists of having the USV fully enter the dock, stay within the dock area for a fixed amount of time (e.g., 10 s) and then exit the dock. The USV may only dock once per simulation run. The simulation ends when the USV has successfully docked and exited or the maximum simulation time is exceeded. Points are accumulated for successfully docking in any dock (e.g., 1 point) with additional points for docking in the correct dock, as specified by the placard color and shape (e.g., 1 additional point). The total task score is the sum of the points for all simulation runs. Any ties are broken based on the total elapsed time for all simulation runs.

Implementation B: This is identical to Implementation A except in this case the correct placard color and shape is not provided directly to the team via the ROS API. Instead, the team must read the color sequence from the scan-the-code buoy. The team reports the color sequence via the ROS API. Points are awarded for correctly reporting the color sequence. The color sequence uniquely determines the placard color and shape indicating the correct docking bay.

Table 4: Scan-the-code and dock API

Service Name	Message Type	Description
`/vrx/scan_dock/color_sequence`	`vrx_gazebo::ColorSequence`	The sequence of three colors perceived. Allowed values are "red", "green", "blue" and "yellow".
Topic Name	**Message Type**	**Description**
`/vrx/scan_dock/placard_symbol`	`std_msgs::String`	Contains the color and shape of a symbol with the format `<COLOR>_<SHAPE>`.

3.3. Task Names

During every task the status of the task is published as a custom ROS Task message over a ROS topic. The message Includes a unique task name.

Table 5: Task naming scheme

Task	Task Message Name
Station-Keeping	`station_keeping`
Wayfinding	`wayfinding`
Landmark Localization and Characterization	`Perception`
Traverse Navigation Channel	`navigation_course`
Scan-the-Code and Dock	`scan_dock`