



T-REX sea-trial Test Plan June 2007

## **June 2007 Sea Trial Test Plan**

By

Conor McGann

*([cmcgann@mbari.org](mailto:cmcgann@mbari.org))*



## Introduction

This document is presented in preparation for the coming sea trial scheduled for June 20<sup>th</sup>. We will have about 3 hours on site to work with the vehicle. The plan will be to find a region of operation reasonably close to shore with a safe operating radius of about 1 km at the surface and 50 meters to the bottom. The goal of the sea-trial is to verify baseline of functionality in place and fully integrated end-to-end.

### **Features of Interest**

- Vehicle Safety
- Straight Plan Execution
- Reactive Planning
- Deliberative Planning
- Distributed Planning

### **Measurements of Interest**

- All log data from the VCS
- Log for Playback on the AMC
- AMC Resource Usage (CPU & Memory)
- Search Efficiency - Step Count and Node Count for Synchronization and Deliberation for all reactors.

### **Behaviors**

In addition to the safety behaviors, the following behaviors are used:

- **Waypoint.** Attempts to achieve a position using a specified direction of approach. This behavior can be used on the surface (if traveling on the surface) or at depth.
- **Descend.** Drives the vehicle towards the bottom, to a specified depth. Terminates when a target depth or minAltitude are reached.
- **Ascend.** Drives the vehicle towards the surface. It terminates when the depth is less than a specified minDepth. We will use this to surface the vehicle.
- **GetGPS.** We will get the GPS whenever we reach the surface. It can only occur when at the surface. **It will be used with minHits = 30, and abortOnTimeout = true. It will have a max duration of 600 seconds.**
- **Waypoint\_Yoyo.** This behavior causes the vehicle to oscillate between two extremes of depth. Termination is based on a timeout or a completion of a specified number of cycles.
- **Setpoint.** This behavior sets heading, depth or pitch, and speed of the vehicle. This behavior is used to idle at the surface, to stop the prop, and to build speed for descent when at the surface.



## Units

- The unit of time will be a second. This is the finest granularity for all timing and also the units in which goals and constraints must be specified.
- All positional information will be provided in latitude and longitude. These are specified in degrees (rather than radians).
- All distances (horizontal and vertical) will be defined in meters.
- Speed will be in meters/per second

## Safety

The main strategy for safety is to fail over into an abort plan as soon as possible once a fault occurs, and to ensure the overall mission operates within a safety envelope for depth, altitude and time. This section reviews the design elements addressing vehicle safety.

## Startup Plan

The AMC ingests a base mission plan (*init.cfg*) in the standard AUV mission script format. We call this the *startup plan*. It contains the following:

- **Mission Timer.** This is a non-sequential behavior used to terminate the mission after the given duration has elapsed. It is defined in seconds. This allows the VCS to initiate termination of the mission. **The maximum mission duration we will permit is 30 minutes to insure we remain in our operating region. Most missions will be limited to 10.**
- **Depth Envelope.** This is a non-sequential behavior used to ensure the vehicle stays within a given depth range. We will have a **maximum depth of 50 meters and a minimum altitude of 10 meters.** The vehicle will be operating over a sandy bottom. In practice, missions will be commanded to no more than **30 meters depth.**
- **Setpoint.** This is the only sequential behavior in the base plan. It will be short (2 seconds). The completion of this behavior provides the **initialization signal** for the AMC.

## Watchdogs

Layered control has been modified to permit periods with no sequential behaviors on the stack. This is to allow some time for the AMC to generate behaviors reactively on termination of a behavior. This arises for example where the parameters for a waypoint are computed when the target depth is obtained. This may also arise if the plan breaks, triggering a recall, and requiring re-planning.

To enforce a maximum time for no behaviors on the stack, the AMC has been augmented with a watchdog timer that **starts when the command sequence timeline begins in an undefined state.** The timer is reset when the command sequence timeline switches to a



defined command. **If the timer reaches a timeout of 10 seconds, it sends an exit command to the VCS. This exit immediately jumps into the abort plan.**

To capture conditions where the AMC is not working correctly from the VCS perspective, we have implemented a **heartbeat using a ping command generated from the AMC**. The AMC sends a ping every second under normal operation. If it crashes or locks up, no ping is sent. The VCS has a separate watchdog that is reset on receipt of each ping and will accumulate otherwise. **If it reaches an elapsed time of 10 seconds without being cleared, it will jump into the abort plan.**

### ***AMC Mission Timer***

The AMC is configured with a mission duration upper bound. If the mission reaches this time, it terminates the mission. Termination will send an exit to the VCS, which will jump it into the abort plan. This will be set tightly on a case-by-case basis and will not exceed 30 minutes.

### ***Abort on Timeout***

Ascend, descend, and getGPS all have semantics of expected completion and that failure to complete is an indication of a mission termination event. This is not the case with setpoint, which has no measurable goal to accomplish. Nor is it the case in our utilization of waypoint and waypoint\_yoyo that can be allocated less time than is necessary to allow pre-emption of transects to accommodate check-in requirements. GetGPS is parameterized with abortOnTimeout = true in the model. We also check for post-conditions on ascend and descend thus detecting a plan failure. These issues are handled in the model.

### ***Stop Gaps***

It may occur that a command ends and the stack is empty for a period of as much as 10 seconds (i.e. the watchdog timeout). The control parameters retain the values of the last command. **It will be up to the model to correctly ensure the prop is stopped if that is the desirable thing to do.**

### ***Abort Plan***

Nominal completion of each mission will occur via execution of the abort plan. This will use a 0 speed setpoint with a duration of 90 minutes. It will then include a dropWeight.

## **System Operation**

- ***Machine Setup***
  - MVC IP\_ADDRESS – 134.89.32.39
- MVC Host Name – mvc-dmo1.shore.mbari.org
- State Publisher Socket Port: 8004
- AmcAgent Socket Port: 8002
- Trex2 IP\_ADDRESS – 134.89.32.39



## ***MVC Environment Variables***

- AUV
- AUV\_CONFIG\_DIR
- AUV\_LOG\_DIR
- See mvc-dmo1:~/auv\_autonomy/configSrc

## ***TREX Environment Variables***

- See TREN/amcConfig, TREN/devConfig
- ROOT\_DIR. Make sure this is set to the parent directory containing
- PLASMA\_HOME
- TREN\_HOME
- AMC\_HOME
- AUV\_HOME
- PATH
- LD\_LIBRARY\_PATH

## ***Configuration Files***

- amcAgent.cfg
- statePublisher.cfg
- abortPlan.cfg
- devices.cfg
- vcs.cfg

## ***Miscellaneous***

- Bootup:
  - The magnet should be in place.
  - Power on the power supply by pressing the red button (in the lab). Expect voltage of  $\leq 35$  V and current approx 2 Amps. There is an led indicator if remote mode is on. This can be an indicator of someone else using it so check with Doug, Duane or Hans.
  - If LED below current indicator is on, power down & call Doug.
  - Use a ping for mvc-dmo1 to wait for it to come up.
  - Telnet mvc-dmo1 (dorado1/dorado1)
  - qtalk -m /dev/ser13
  - !a <ENTER> // To turn on trex2



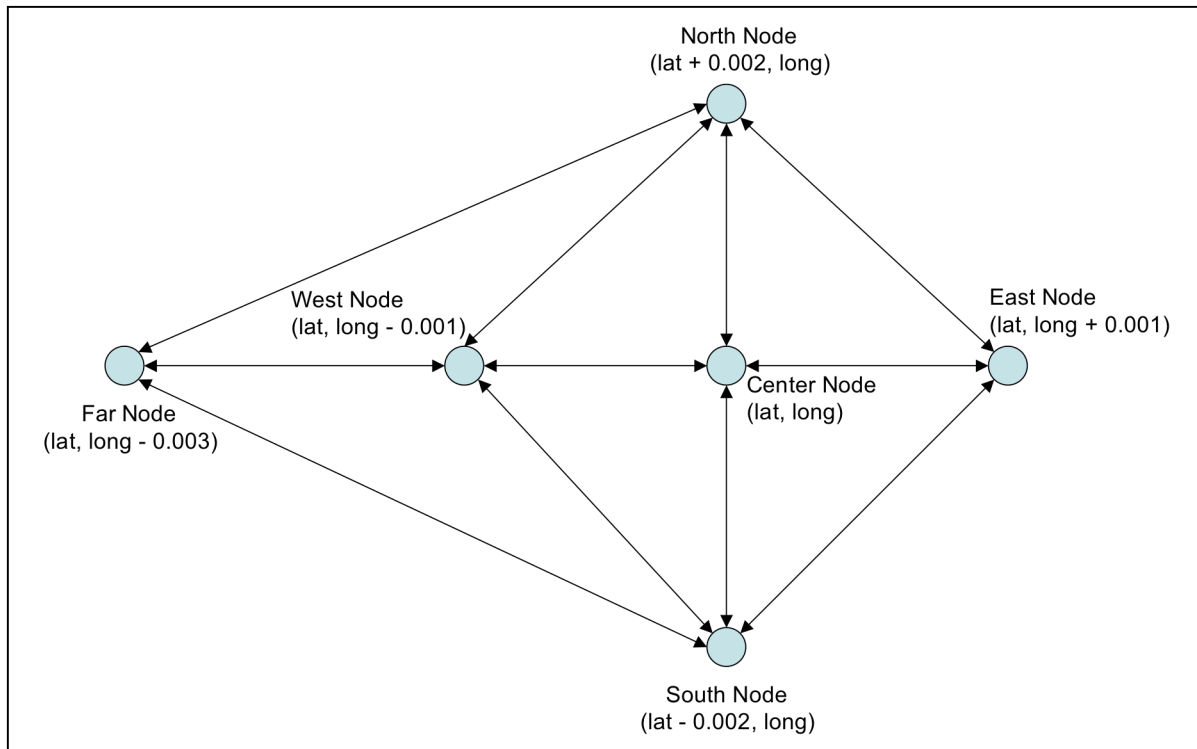
#### T-REX sea-trial Test Plan June 2007

- !f <ENTER> // To turn off trex2
- ^a to exit qtalk.
- Ping trex2 to wait to come up. It can take a while. 134.89.32.39
- telnet trex2 (root/dorado1)
- remove the magnet before running a mission
- Shutdown
  - Insert magnet
  - Press red button on power supply. Make sure u press in all the way.
- Clearing shared memory
  - If the amcAgent terminates with resource temporarily unavailable messages, you need to nuke files for shared memory access
  - /dev/shmem may contain files of the for “\***Shmem**”
  - **rm /dev/shmem/\*Shmem\***
- Starting amcAgent
  - cd \$AUV/bin
  - ./amcAgent -v -sim // for simulation
  - ./amcAgent -v // for real missions
- Killing amcAgent
  - slay supervisor
  - ^c



## Region of Operation

The graph to be used for operation is shown below. Final co-ordinates of the central node are: N 36.8424, W 121.9044



## Test Cases

Nominal Termination Criteria:

- Correct execution of the plan. Arrived at expected position.
- No aborts.
- An exit sent from AMC to VCS
- VCS jumps into the abort plan.

It should look something like:

```
[VCS][VCS][400][1182204828.787091]Terminating due to watchdog timeout.
```

```
[VCS][VCS][400][1182204828.787140]Sending exit
```

```
[VCS][VCS][400][1182204828.787214]Exiting command dispatcher loop.
```

```
[VCS][VCS][400][1182204828.787313]Terminating Command Dispatcher
```

```
[VCS][VCS][400][1182204828.791884]Terminating State Listener.
```



## **Safety Checks**

### **checkin.0**

This test will execute a canned plan consisting of a call to **getgps** and then a call to **setpoint**. The mission time limit is 100 seconds. This will test that we can start the vehicle and run a canned plan.

Look for:

- Termination in less than 100 seconds
- Nominal Termination

### **checkin.0.timeout**

The purpose of this test is to verify that the mission timer will force an abort. To do this we employ a version of the base plan with a very short timeout window – 20 seconds. The AMC time limit will remain at 100 seconds.

**Setup: copy init.timer.cfg to init.cfg**

Look for:

- Termination initiated from the VCS after 20 seconds.
- Jumps into the abort plan

**Don't forget: copy init.base.cfg init.cfg**

### **checkin.0.terminate**

The purpose of this test is to ensure that if the AMC crashes, the VCS will jump into the abort plan within 10 seconds based on its own watchdog. Revert to standard base plan. Once again use checkin.0. After completion of the gps, kill the AMC.

Look for:

- VCS Jumps into the abort plan.

### **timing.1**

This test will run a **getgps** and a sequence of **setpoints**, all very short. It is a simple timing test to make sure the commanding and feedback is correct. The vehicle will remain at the surface. The AMC timeout will be 100 seconds.

### **go.base**

This test will run a **setpoint** and waypoint at the surface, followed by a **getgps**.

### **depth.abort**

This test will verify that the depth envelope behavior will force an abort if the vehicle goes too deep. We will use a canned plan to descend to a target depth of 20 meters but modify the base plan to have a max depth of 10 meters. We can set the AMC timeout to 100 seconds.



T-REX sea-trial Test Plan June 2007

Setup: copy init.depth.cfg to init.cfg

Look for :

- VCS "DepthEnvelope -- Measured depth 10.064657 over allowable depth 10.000000. Aborting!
- VCS : Abort request issued by behavior "depthEnvelope"
- AMC: Received a termination event from the VCS

**Don't forget: copy init.base.cfg init.cfg**

## **Simple Mobility**

This group of tests will execute traveling to a destination involving the main behaviors. All tests should terminate with the nominal termination sequence initiated by the AMC watchdog timing out after the last command has finished.

NB: Timing constraints in each .cfg file for a test are based on how long the mission takes given an initial position. This may have to be changed if in fact we are further away from the destination. We tend to keep these bounds tight to limit the possibility of faults leading to a missing vehicle.

### **go.0**

This test is a predefined sequence of commands. No plan generation. It will timeout after 150 seconds in the worst case. Only plan validation on input. The plan will drive briefly at the surface, descend to 5 meters, drive to a waypoint, and then ascend to the surface. It uses **centerNorthing** and **centerEasting**.

```
behavior setpoint {
id=311;duration=6;heading=0.000000;speed=1.500000;verticalMode=pitch;pitch=0.000000; }

behavior descend { id=331;duration=59;horizontalMode=heading;horizontal=0.000000;pitch=-
30.000000;speed=1.500000;maxDepth=5.000000;minAltitude=10.000000; }

behavior waypoint {
id=355;duration=59;northing=4066281.115653;eastng=599908.200877;speed=1.500000;depth=5.000000;
}

behavior ascend { id=384;duration=59;pitch=30.000000;speed=1.500000;endDepth=0.000000; }
```

### **go.1**

This test will generate a decomposition of **Action.Go** to descend to a target depth of 5 meters. On termination, the commands to descend will be latched in so the vehicle will continue to descend at a pitch angle of -30 degrees and a speed of 1.5 meters for 10 seconds until the watchdog times out and sends an exit, which will jump into the abort plan. Default headings of 270 (due west) will be used. The commands generated should be:

```
behavior setpoint {
id=1139;duration=5;heading=270.000000;speed=1.500000;verticalMode=pitch;pitch=0.000000; }

descend { id=1094;duration=65;horizontalMode=heading;horizontal=270.000000;pitch=-
30.000000;speed=1.500000;maxDepth=5.000000;minAltitude=10.000000; }
```



Look for:

- Max depth in the log to be greater than 5 meters. It should not be more than 10 meters.
- Nominal termination.
- The vehicle should float to the surface.

## go.2

This test will acquire the centerNode. It will use a waypoint mode and a max depth of 5 meters. The heading generated will depend on where the vehicle is. The AMC timeout is 200 seconds. The commands generated should be approximately:

```
behavior setpoint {
id=1140;duration=5;heading=207.731830;speed=1.500000;verticalMode=pitch;pitch=0.000000; }

behavior descend { id=1095;duration=65;horizontalMode=heading;horizontal=207.731830;pitch=-
30.000000;speed=1.500000;maxDepth=5.000000;minAltitude=10.000000; }

behavior waypoint {
id=4807;duration=121;northing=4066281.115653;eastng=599908.200877;speed=1.500000;depth=5.00000
0; }
```

Look for:

- Arrival at the center node, at least crossing the line perpendicular to the track-line.
- A max depth of approximately 5 meters.

## go.3

This test will plan for 2 goals. The first will **dive** and the second will **ascend**. It can be run anywhere since no lat/long targets are used. Default headings will drive the vehicle west. The expected depth is 5 meters. It may go slightly deeper transitioning from descend to ascend. The AMC timeout is 100 seconds. The commands generated should be:

```
behavior setpoint {
id=1166;duration=5;heading=270.000000;speed=1.500000;verticalMode=pitch;pitch=0.000000; }

behavior descend { id=1121;duration=65;horizontalMode=heading;horizontal=270.000000;pitch=-
30.000000;speed=1.500000;maxDepth=5.000000;minAltitude=10.000000; }

behavior ascend { id=5071;duration=17;pitch=30.000000;speed=1.500000;endDepth=0.000000; }
```

## go.5

This test introduces a yoyo. It will make for the **west node** with drive mode set for yoyo and using a minDepth of 10 and a maxDepth of 20. No **checkin** requirements are imposed. A time limit of 250 seconds is used. This test should be initiated near the center node, or at least within a reachable distance from the **west node**. The commands generated should be:

```
behavior setpoint {
id=1145;duration=5;heading=235.449500;speed=1.500000;verticalMode=pitch;pitch=0.000000; }
```



## T-REX sea-trial Test Plan June 2007

```
behavior descend { id=1100;duration=72;horizontalMode=heading;horizontal=235.449500;pitch=-30.000000;speed=1.500000;maxDepth=10.000000;minAltitude=10.000000; }
```

```
behavior waypoint_yoyo {  
id=5774;duration=155;northing=4066280.065570;easting=599818.282261;speed=1.500000;minDepth=10.000000;maxDepth=20.000000;minPitch=-15.000000;maxPitch=15.000000;minAltitude=10.000000;maxCycles=100.000000; }
```

### **Checkin Tests**

This group of tests verify the checkin action which will get the vehicle to the surface, localize via gps, and then idle at the surface for a designated period of time.

#### **checkin.1**

The vehicle will cycle through a sequence of active checkin windows for the duration of the mission. The separation between checkin's is 1 second. The duration of each is 40 seconds. This test can be run anywhere. The commands generated will be something like:

```
|behavior getgps { id=1821;duration=599;abortOnTimeout=True;minHits=30; }  
|behavior setpoint {  
id=5954;duration=9;heading=0.000000;speed=0.000000;verticalMode=pitch;pitch=10.000000; }  
behavior getgps { id=7640;duration=599;abortOnTimeout=True;minHits=30; }  
behavior setpoint {  
id=11773;duration=9;heading=0.000000;speed=0.000000;verticalMode=pitch;pitch=10.000000; }  
behavior getgps { id=13459;duration=599;abortOnTimeout=True;minHits=30; }  
behavior setpoint {  
id=17592;duration=9;heading=0.000000;speed=0.000000;verticalMode=pitch;pitch=10.000000; }  
behavior getgps { id=19278;duration=599;abortOnTimeout=True;minHits=30; }  
behavior setpoint {  
id=23411;duration=9;heading=0.000000;speed=0.000000;verticalMode=pitch;pitch=10.000000; }  
behavior getgps { id=25043;duration=599;abortOnTimeout=True;minHits=30; }
```

#### **checkin.2**

This test will introduce pre-emption of a transect in order to meet checkin requirements. A MAX\_SEPARATION of 60 seconds and a CHECKIN\_DURATION of 45 seconds will be used. The vehicle will be tasked to make the **center node**. Thus it matters where we run it from to reach it in time. The commands generated should be something like:

```
behavior setpoint {  
id=1249;duration=5;heading=207.731830;speed=1.500000;verticalMode=pitch;pitch=0.000000; }  
behavior descend { id=1204;duration=52;horizontalMode=heading;horizontal=207.731830;pitch=-30.000000;speed=1.500000;maxDepth=5.000000;minAltitude=10.000000; }  
behavior waypoint {  
id=4987;duration=28;northing=4066281.115653;easting=599908.200877;speed=1.500000;depth=5.000000;  
; }  
behavior ascend { id=8799;duration=14;pitch=30.000000;speed=1.500000;endDepth=0.000000; }  
behavior getgps { id=10577;duration=599;abortOnTimeout=True;minHits=30; }
```



## T-REX sea-trial Test Plan June 2007

```
behavior setpoint {
id=14563;duration=14;heading=0.000000;speed=0.000000;verticalMode=pitch;pitch=10.000000; }

behavior setpoint {
id=16789;duration=5;heading=186.941852;speed=1.500000;verticalMode=pitch;pitch=0.000000; }

behavior descend { id=16744;duration=53;horizontalMode=heading;horizontal=186.941852;pitch=-
30.000000;speed=1.500000;maxDepth=5.000000;minAltitude=10.000000; }

behavior waypoint {
id=20094;duration=32;northing=4066281.115653;eastings=599908.200877;speed=1.500000;depth=5.00000
0; }
```

### **checkin.3**

This will send the vehicle 500 meters north and 500 meters west of the center node. It will employ a **waypoint\_yoyo** drive mode this time, and should have a max separation of 150 seconds with 45 seconds checkin duration. An AMC timeout of 1000 seconds is used. It should switch into yoyo mode at the minDepth of 10 meters.

### ***Path Navigation***

This next set of tests involve commanding the vehicle at a higher level of abstraction – i.e. a path. They also introduce multi-link missions and employ the skipper in planning, thus testing distributed planning among 2 deliberative reactors.

#### **path.0**

Go to the **center node** and check in when done. Will travel using a waypoint and a depth of 10 meters. The checkin window will be 50 seconds. No separation limits are imposed. AMC timeout set to 200 seconds.

#### **path.1**

This test interleaves path level planning and checkin window management to pre-empt go actions. The mission is directed to the **north node** with a max separation of 100 seconds and a checkin duration of 40 seconds. AMC timeout is 400 seconds.

#### **path.2**

This test will head to the **south node** with a max separation of 100 seconds and a checkin duration of 40 seconds. AMC timeout is 800 seconds.. It utilizes the **skipper**, so it is primarily a test of distributed planning.