Developer Test Task

Nortal Summer University 2017

Autonomous Life Raft (Java)

1. The Story

Alfred is always building new technology for Batman. At the moment he is working on the next version of the utility belt, ultra strong grappling hook, a spacecraft and autonomous life raft. He is also organising the Bat-Library. Due to the huge amount of projects he has decided to outsource some of the work. As a test task for the autonomous life raft projects, he has offered you the opportunity to give it a try with implementing the algorithm for the engine control system.

When entering the hangar for briefing, you notice the vessel prototype. It is circular with four powerful thrusters attached to it. You are wondering why Alfred has attached jet engines on top of a life raft and is it safe at all, but he assures you, the design can be improved later. He also explains you, there will be a race amongst multiple control systems to identify the best performing implementation.

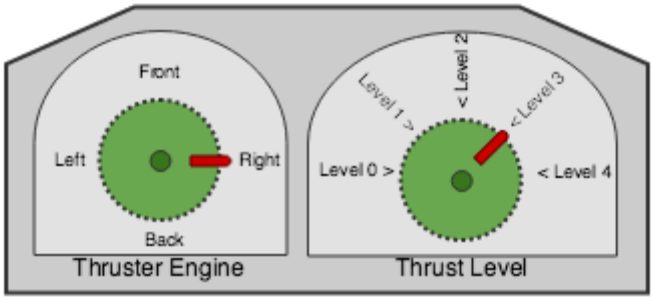
2. Assignment

Implement the control algorithm to collect all the targets on the race area without crashing.

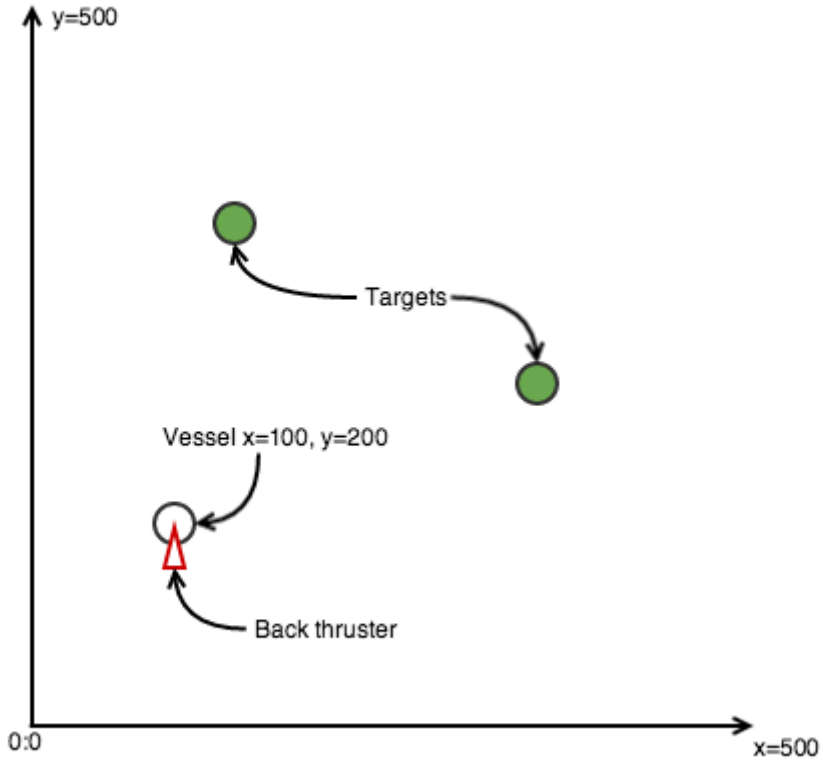
Every second your vessel receives a new set of parameters from the monitoring satellite system and needs to return a command for the engines. These parameters include a list of targets scattered around the race area. The goal for the vessel is to collect all of these targets in no particular order. The vessel also receives its current location and current speeds along X and Y axis.

Since you are working with a prototype, you can only enable one thruster at a time. For each thruster, you can control the level of thrust it provides between Level 0 and Level 4. For a thrust

Level X, a push force equivalent to X m/s² is generated. Once you enable a thruster, all other thrusters will turn off.



Since your vessel is on the water, you also need to consider drag forces applying to it. The drag forces applied are equivalent to 1.6 m/s² meaning that at least Level 2 thrust is required to start moving the vessel. Drag forces depend on the vessel movement speed.



The race is considered to be finished when one of the following events occurs

- all the targets have been collected
- the vessel crashes to race area boundaries
- the race lasts longer than 10 minutes or 1200 simulation cycles

Target is collected when your vessel coordinates are not further than 2m from the target coordinates.

3. Set Up

3.1. Gradle

Remember that gradle commands can be executed in three different ways:

1. When you have Gradle installed: *gradle <command>*
2. When working on Windows command line or PowerShell: *gradlew.bat <command>*
3. When working on MacOS, Linux or Windows 10 Shell: *./gradlew <command>*

In current guide all examples will use the first format that you can adapt if needed.

3.2. Tests

We have provided you a set of unit tests to simplify development of the controller code. We recommend to use your IDE to execute the tests that have been provided but you can do it from the command line as well.

- To execute single test just run:
  *gradle test --tests "*SimpleStraightTest.testStraightLineSingleTarget"*
  If you remove *--tests* option, then all tests will be run.
- You can also run the tests without simulation visualizations. Just add
  -Dsimulator.disable.gui=true to your command line.