Open Simulation Platform - Workshop

Tutorial exercises

Introduction

In this tutorial, we will build up a simple model of a vessel and run a few simulations where we make it navigate along a given trajectory using its DP system. For the curious, the vessel in question is NTNU's research vessel *Gunnerus*; see Figure 1.

Figure 1. R/V Gunnerus, NTNU's research vessel.

The tutorial consists of four steps:

Exercise 1: Creating OSP model description files.
Exercise 2: Creating an OSP system structure file.
Exercise 3: Running a simulation.
Exercise 4: Running a simulation with a scenario.

The model we'll build is available in its entirety as an example on the OSP web site in a few variations, where it's called *Gunnerus-DP*. **We strongly recommend that you don't "cheat" by looking it up there, however, as that would ruin the fun and challenge of the exercise.** Partial solutions will be provided at every step during the course of the tutorial, so you will be able to continue with the next step even if you get stuck on one.

Good luck!

Prerequisites

The exercises can only be done on Windows, because some of the FMUs only support Windows.

You will need to download and install the following OSP software to complete the tutorial:

- OSP Validator v. 0.13.0 (download `osp-cli.jar`; see documentation)
- OSP Demo Application v. 0.9.0 (see documentation)
- cosim v. 0.7.0 (see documentation)

You can choose to run simulations using either the Demo Application or with *cosim*. However, the latter only produces CSV files, so in that case you'll need additional software to plot the results. Most plotting software supports CSV, including Excel.

To use the OSP Validator and *cosim*, you need to be comfortable with using the command line.

The model

Figure 2 shows a schematic of the complete system. You can and should use this diagram actively when solving the exercises, as it helps to visualise the relations and connections between the different components.

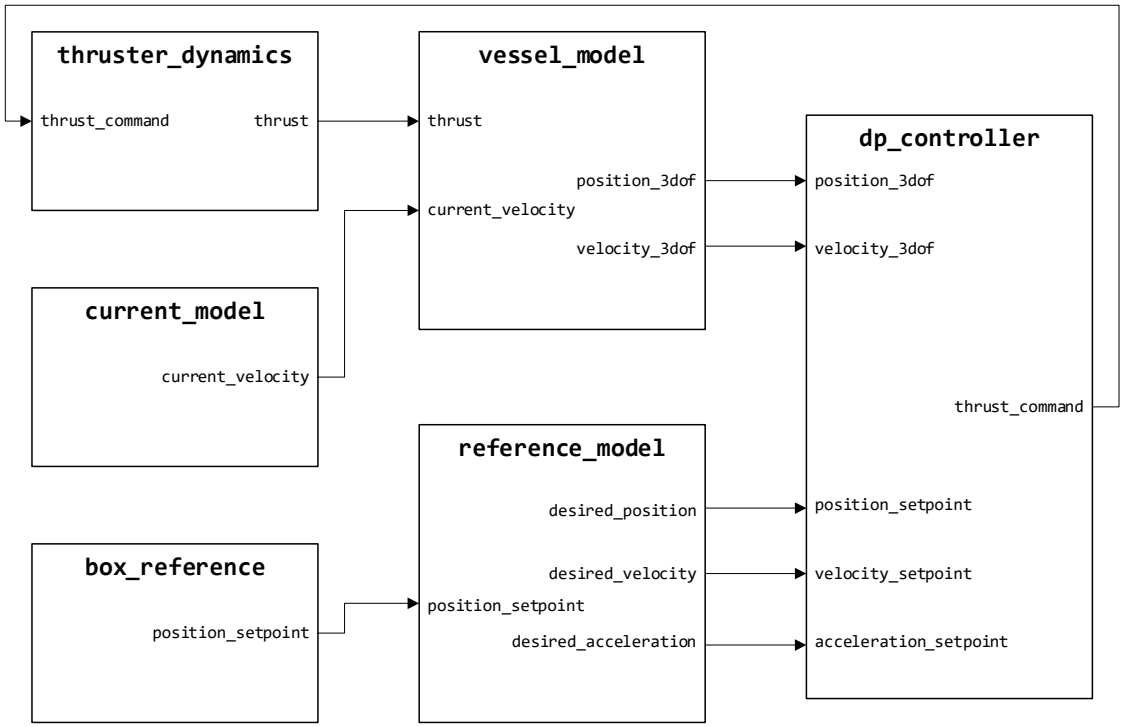

Figure 2. Coupling diagram for the Gunnerus-DP *vessel model.*

The model consists of 6 components:

- *VesselModel*, which represents the dynamics of the hull with 6 degrees of freedom.
- *ThrusterDynamics*, which represents the dynamics of the thruster system.
- *DPController*, which represents the dynamic positioning control system.
- *CurrentModel*, which generates surface current velocities.
- *BoxReference*, which generates position setpoints in a rectangular pattern.
- *ReferenceModel*, which provides smooth velocity and acceleration setpoint signals to the DP controller.

FMUs for all of these will be provided.

Exercise 1: Creating OSP model description files

Download and unzip the file called `osp-workshop-tutorial-step1.zip`. In it, you'll find FMUs for all components along with `OspModelDescription.xml` files for each of them.

One of the XML files is incomplete, however, and your task is to add the missing variable groups—at least the ones required to model the full *Gunnerus-DP* system.

Exercise

Complete the OSP model description for *VesselModel*. Use the OSP Validator tool to verify that your model description is correct.

Tips

- The missing information is marked with "TODO" comments in the XML files.
- Look at the other OSP model description files for inspiration.
- The FMU's variable names can be obtained in several ways:
    - Using the *Demo Application*, by entering the path to the directory that contains the FMUs under *Configuration* on the *Simulation setup* screen and selecting the model in the sidebar.
    - Using the *cosim* tool, in particular its `cosim inspect` subcommand.
    - Using a regular file archiver, by unzipping the FMU (which is just a regular ZIP file) and looking at the contained `ModelDescription.xml` file.
- Some models have 6 degrees of freedom (6DOF), but you are asked to create 3DOF variable groups containing a subset of the input/output variables. In such cases, the relevant variables are the x, y and yaw components (typically the 1st, 2nd and 6th components of a 6DOF group).
- The FMU has many variables. Focus on the inputs and outputs, and ignore the others.
- Look at the coupling diagram in Figure 2 to understand which groups you need.
- Use the OSP Validator to see whether your changes in VesselModel_OspModelDescription.xml are valid.
  To run the validator from inside the osp-workshop-tutorial-step1 folder, type:
  ```
  java -jar ../osp-cli.jar osp-model-description -file
  VesselModel_OspModelDescription.xml -fmu VesselModel.fmu
  ```
- The OSP Validator can only verify that your model descriptions are *valid*, but not that they are *complete*.

Exercise 2: Creating an OSP system structure file

Download and unzip the file called `osp-workshop-tutorial-step2.zip`. In it, you'll find FMUs for all components, complete `OspModelDescription.xml` files for each of them (which also provide a solution to Exercise 1), and an `OspSystemStructure.xml` file.

The OSP system structure file describes the connections between components as shown in the coupling diagram in Figure 2. However, it is incomplete, as one subsimulator and some connections are missing. Your task is to add the missing elements and validate the configuration.

For an extra challenge, copy the `OspSystemStructure.xml` file over to the workspace you used in Exercise 1, and use that as your starting point for this exercise. Then, you'll need to ensure that it works with the OSP model descriptions you created yourself.

Exercise

Complete the OSP system structure description. Use the OSP Validator tool to verify that it is valid.

Tips

- Look at the existing simulators and connections for inspiration.
- Look at the OSP model descriptions to see the names of variable groups.
- Look at the coupling diagram to see which connections you need.
- Use the OSP Validator to see whether your changes in OspSystemStructure.xml are valid. To run the validator from inside the osp-workshop-tutorial-step2 folder, type:
  ```
  java -jar ../osp-cli.jar osp-system-structure -file OspSystemStructure.xml
  ```
- The OSP Validator can only check that your configuration is *valid*, but not that it is *complete*.

Exercise 3: Running a simulation

Download and unzip the file called `osp-workshop-tutorial-step3.zip`. In it, you'll find FMUs for all components, complete `OspModelDescription.xml` files for each of them, and a complete `OspSystemStructure.xml` file (which also provides a solution to Exercise 2).

Your task is to run a simulation of the system and verify that the results make sense. For an extra challenge, continue using the files you created in exercises 1 and 2. This is the ultimate test that you've solved those exercises correctly.

Exercise

Run a simulation of the *Gunnerus-DP* system and plot the 2D position of the vessel. Verify that it sails in the following square pattern:

Time	Target position (North)	Target position (East)
0	0	0
500	40	0
1000	40	-40
2000	0	-40
2500	0	0

Tips

- The simulation runs very fast, so enable real-time simulation if you're using the Demo Application, since it plots results live. You may also want to set the target real-time factor to some value greater than 1, though, so you don't have to wait too long. 50 (simulated seconds per wall-clock second) is a good suggestion.
- Plot the position reference along with the actual position, so you can more easily see that the vessel follows the expected trajectory.
- You won't be able to see the entire 2500-second trajectory at once in the Demo Application, because it only shows results from a limited time window of maximum 20 minutes, i.e., 1200 seconds. (This is a known and reported issue.)
- To run the example with cosim.exe from inside the osp-workshop-tutorial-step3 folder, type:

```
cosim run OspSystemStructure.xml -b 0 -d 2500
```

Exercise 4: Running a simulation with a scenario

Download and unzip the file called `osp-workshop-tutorial-step4.zip`. In it, you'll find FMUs for all components, complete `OspModelDescription.xml` files for each of them, a complete `OspSystemStructure.xml` file, and a file called PlotConfig.json which contains the setup for a vessel position plot in the Demo Application (which also provides a solution to Exercise 3).

So far, we've used the *BoxReference* component to generate waypoints for the vessel. Now, we'll do the same with a scenario instead.

Exercise

Create a scenario file that causes the vessel to travel in a rectangular pattern between the following waypoints, in order:

Target position (North)	Target position (East)
0	0
0	100
50	100
50	0
0	0

Tips

- To make the Demo Application read your scenario file, write it in JSON format and place it in a subdirectory of the one that contains `OspSystemStructure.xml` named `scenarios`. The JSON scenario format is documented on the OSP software web site.
- Remove BoxReference entirely, since you don't need it anymore.
- Consult the documentation for how to actually *run* a scenario in the Demo Application.
- To run the example with cosim.exe from inside the osp-workshop-tutorial-step4 folder, type:

```
cosim run OspSystemStructure.xml -b 0 -d 2500 --scenario scenarios/box.json
 --scenario-start 0
```

Final solution

For a solution to exercise 4, download and unzip the file called `osp-workshop-tutorial-step5.zip`.