Connection between ROS and Matlab Simulink and control of a simulated robot by Stateflow - EN

Goal of this tutorial is to show how to create a connection between Robot Operating System (ROS) and Matlab Simulink.

Requirements for Matlab

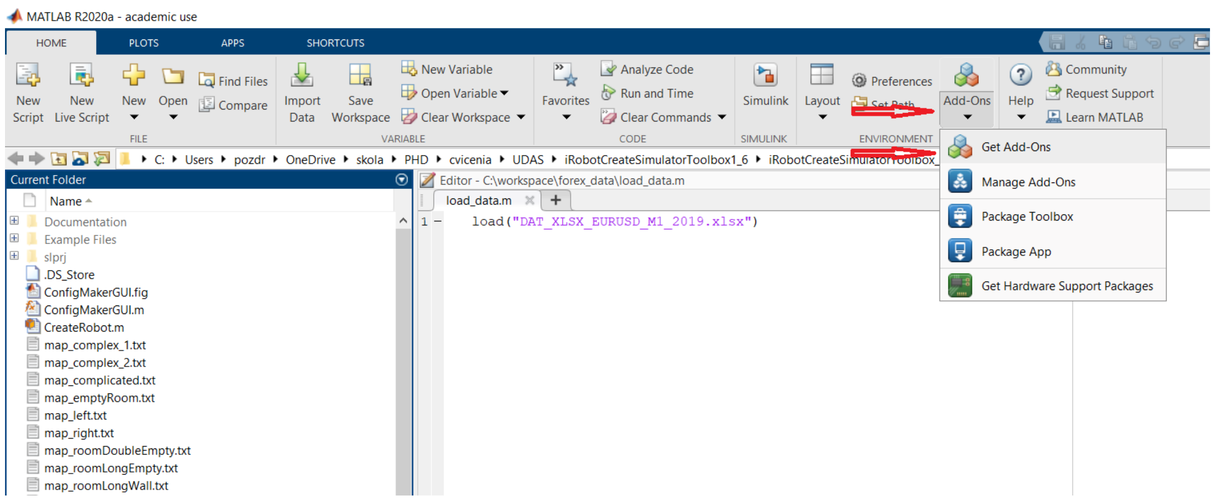
To be able to create a connection with ROS, the following is required:

- up-to-date version of Matlab
- Toolbox - Simulink
- Toolbox - Stateflow
- Toolbox - ROS
- Toolbox - Robotic System
- Add-on - Robotics System Toolbox Interface for ROS Custom Messages

Up-to-date version of Matlab and all necessary toolboxes it is possible to install via Matlab installation.

Custom ROS messages

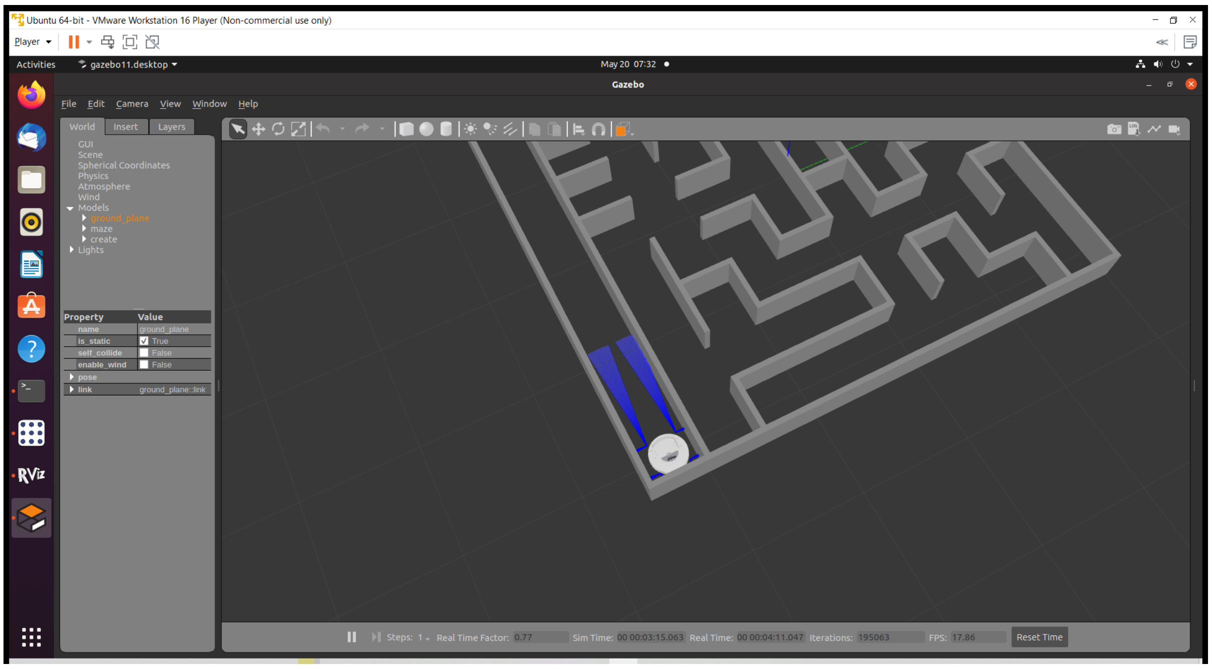
Package create_control uses custom ROS messages. So we could communicate with ROS package with custom ROS messages it is necessary to add Matlab App “Robotics System Toolbox Interface for ROS Custom Messages”. You can install Add-on by clicking in Matlab “Add-ons” and then “Get Add-ons”, and type in the name and find it. Follow the instructions and after that you should see more instructions in the Matlab console and follow those too. They should explain how to add custom ROS message into the Matlab. You will need the ROS package containing the custom message, in this case create_control and extract it into a known directory.



```
load("DAT_XLSX_EURUSD_M1_2019.xlsx")
```

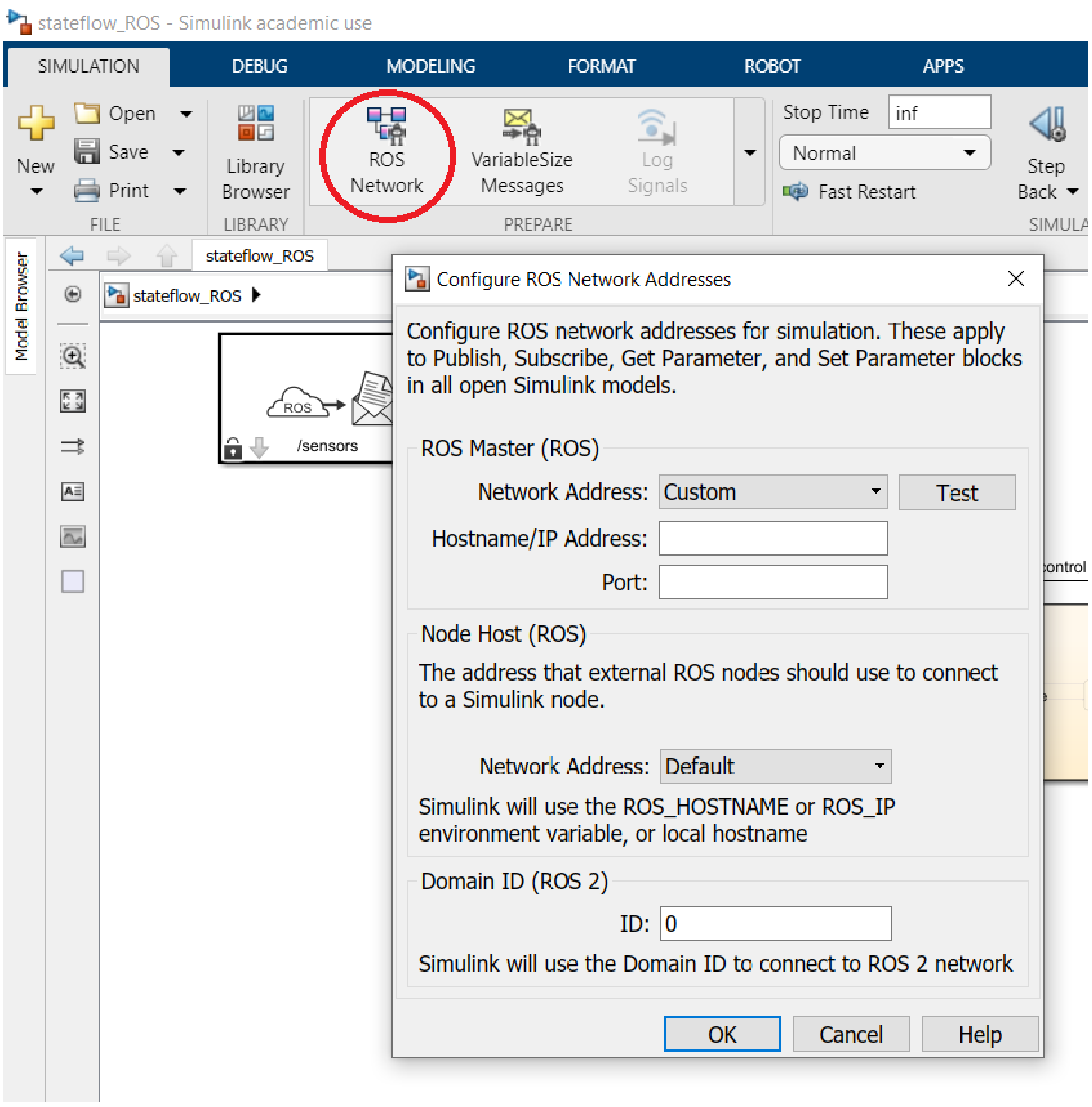
Launching of the simulated mobile robot

Before the next steps it is necessary to launch the simulated robot on a computer where we want the simulation to run according to the tutorial micromouse. Gazebo simulator with a mobile robot and a maze should be opened. Next we can continue with connecting the Simlulink to ROS_MASTER.



Configuration of ROS network

All ROS nodes need to be connected in the same network and thus each node must know IP address and port of ROS_MASTER node. By default nodes are configured to connect to local host, which is appropriate if Simulink and ROS_MASTER run on the same machine. But if they don’t we must configure it by opening “ROS Network” in Simulink “Simulation” tab. First change in the configuration “Network Address” to Custom. Then in the configuration we type in ROS_MASTER_URI into the IP and port where the ROS_MASTER is running. We can test the connection and then apply the changes. Now the communication should be established.



Configure ROS Network Addresses

Configure ROS network addresses for simulation. These apply to Publish, Subscribe, Get Parameter, and Set Parameter blocks in all open Simulink models.

ROS Master (ROS)

Network Address: Custom

Test

Hostname/IP Address:

Port:

Node Host (ROS)

The address that external ROS nodes should use to connect to a Simulink node.

Network Address: Default

Simulink will use the ROS_HOSTNAME or ROS_IP environment variable, or local hostname

Domain ID (ROS 2)

ID: 0

Simulink will use the Domain ID to connect to ROS 2 network

OK Cancel Help

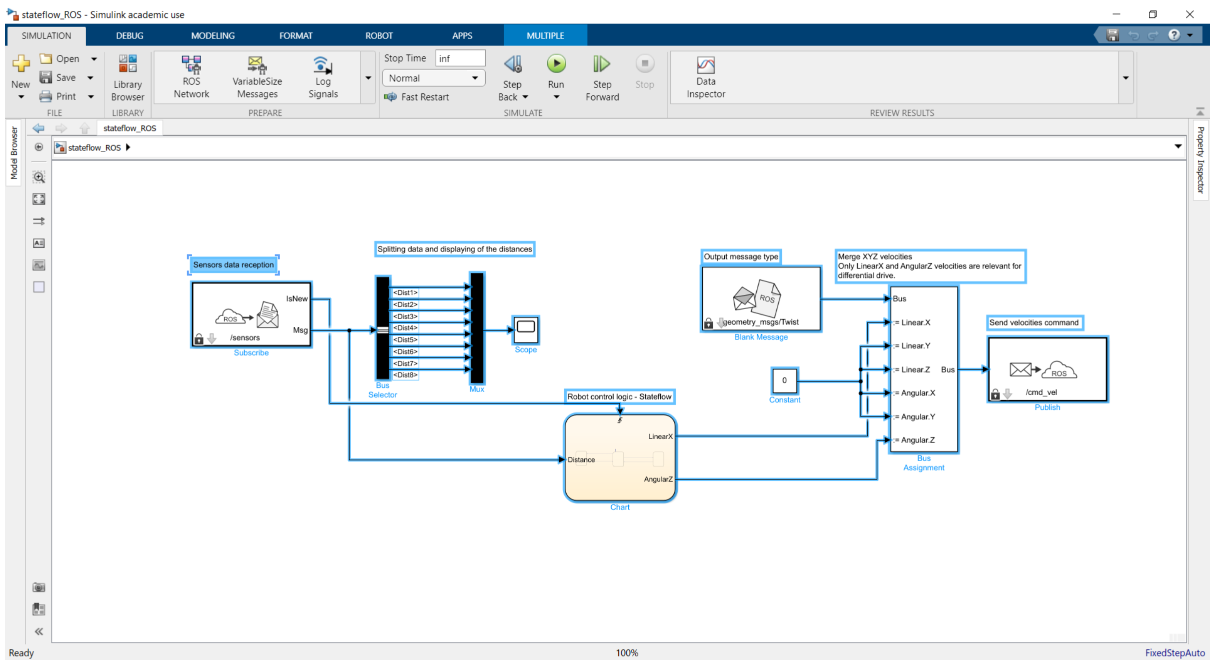
Publishing and subscribing ROS messages

If everything is correctly configured now it is possible to create the communication interface between Simulink and ROS. In a new created simulink schematic add the following blocks:

- Subscribe
- Bus selector
- Stateflow chart
- Blank message
- Bus assignment
- Publish

If we want to receive messages from ROM, we use a block “subscribe”. For the block it is necessary to configure “Topic”, choose from the list the intended one, in our case “/sensors”. Automatically will be created a bus of outputs because the message of type “sensors”

contains multiple values. With block “Bus Selector” we can split the bus and display individual signals.
Next it is possible to send messages by “publish” block. Firstly we need to configure block “Blank Message” to an appropriate type, in our case the “geometry_msgs/Twist” for sending required velocities to the mobile robot. Connect output of the “Blank Message” to the “Bus” of the “Bus Assignment” block. With the “Bus Assignment” block we can assign inputs to individual values of the message. The resulting bus contains the “Twist” message with velocities and can be connected to the “publish” block.
Now receiving and sending messages is complete.



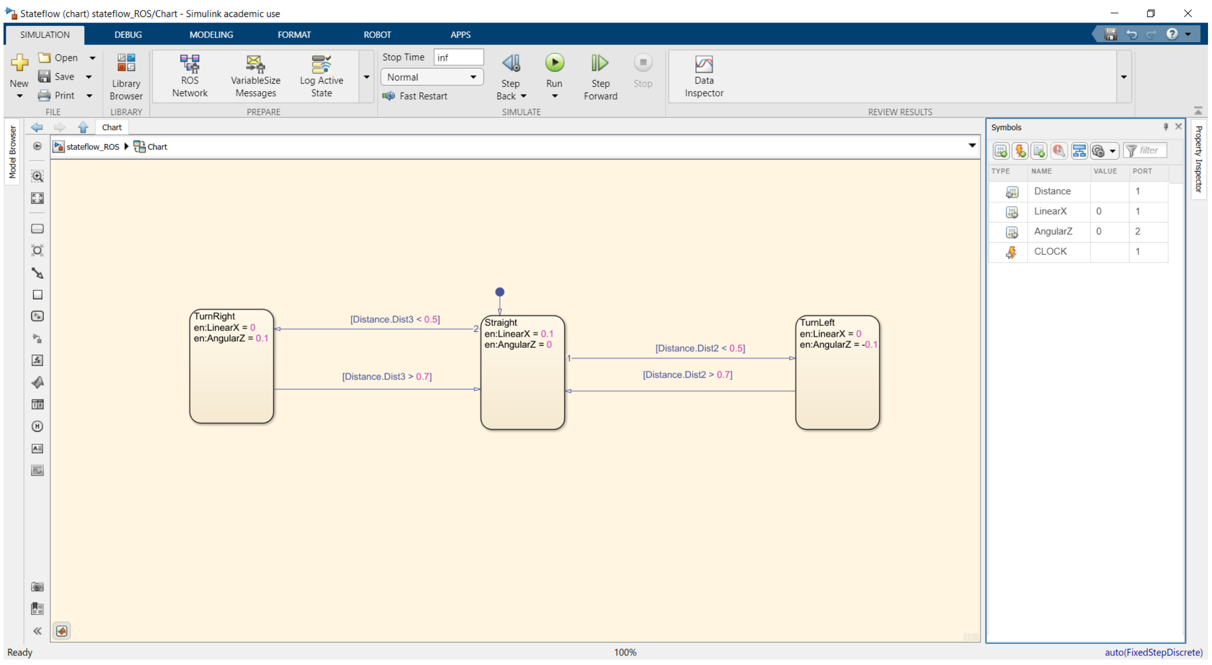
After launching of the Simulink simulation, ROS messages will be received and command messages will be sent back to ROS with required velocities for the robot.

For detailed explanation see mathworks tutorial:
https://www.mathworks.com/help/ros/ug/get-started-with-ros-in-simulink.html

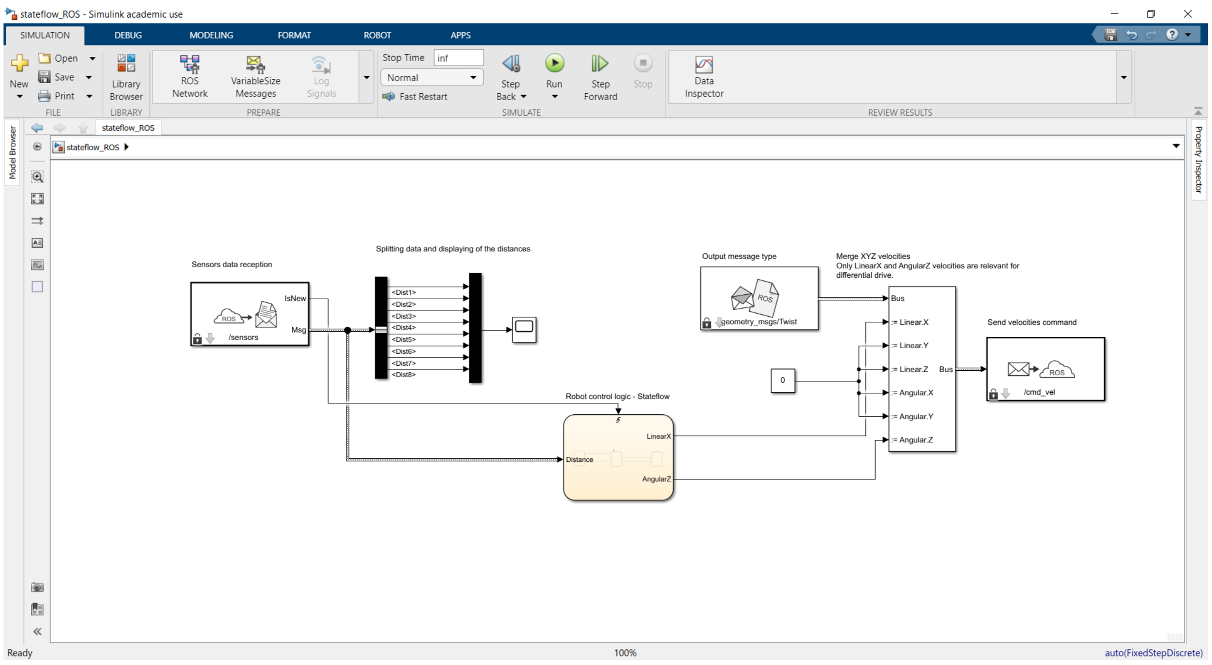
Creation of stateflow logic

After creation and testing of communication between ROS and Matlab we can proceed with creation of control logic for the mobile robot.

Create a vector of inputs from sensors, triggering events and output for linear velocity and output for rotation velocity. We will create three states, while the initial state is “go forward” and the second and third states are to turn left and right. Simple conditions reacting to distances measured by simulated range finders change the logic to the appropriate state.



TYPE	NAME	VALUE	PORT
	Distance		1
	LinearX	0	1
	AngularZ	0	2
	CLOCK		1


stateflow_ROS - Simulink academic use
SIMULATION
DEBUG
MODELING
FORMAT
ROBOT
APPS
Stop Time inf
Normal
Fast Restart
Sensors data reception
IsNew
Msg
/sensors
Splitting data and displaying of the distances
<Dist1>
<Dist2>
<Dist3>
<Dist4>
<Dist5>
<Dist6>
<Dist7>
<Dist8>
Output message type
geometry_msgs/Twist
Merge XYZ velocities
Only LinearX and AngularZ velocities are relevant for differential drive.
Bus
:= Linear.X
:= Linear.Y
:= Linear.Z
:= Angular.X
:= Angular.Y
:= Angular.Z
0
Send velocities command
/cmd_vel
Robot control logic - Stateflow
Distance
LinearX
AngularZ
Ready
100%
auto(FixedStepDiscrete)