

Contents

Preface.....	vii
Printed vs PDF Versions of the Book.....	ix
1. Scope of this Volume.....	1
2. Installing the ros-by-example Code.....	3
3. Task Execution using ROS.....	7
3.1 A Fake Battery Simulator.....	8
3.2 A Common Setup for Running the Examples.....	10
3.3 A Brief Review of ROS Actions.....	11
3.4 A Patrol Bot Example.....	12
3.5 The Patrol Bot using a Standard Script.....	13
3.6 Problems with the Script Approach.....	16
3.7 SMACH or Behavior Trees?.....	17
3.8 SMACH: Tasks as State Machines.....	17
3.8.1 SMACH review.....	18
3.8.2 Patrolling a square using SMACH.....	19
3.8.3 Testing SMACH navigation in the ArbotiX simulator.....	23
3.8.4 Accessing results from a SimpleActionState.....	26
3.8.5 SMACH Iterators.....	27
3.8.6 Executing commands on each transition.....	30
3.8.7 Interacting with ROS topics and services.....	31
3.8.8 Callbacks and Introspection.....	36
3.8.9 Concurrent tasks: Adding the battery check to the patrol routine.....	36
3.8.10 Comments on the battery checking Patrol Bot.....	44
3.8.11 Passing user data between states and state machines.....	44
3.8.12 Subtasks and hierarchical state machines.....	48
3.8.13 Adding the battery check to the house cleaning robot.....	54
3.8.14 Drawbacks of state machines.....	54
3.9 Behavior Trees.....	55
3.9.1 Behavior Trees versus Hierarchical State Machines.....	56
3.9.2 Key properties of behavior trees.....	57
3.9.3 Building a behavior tree.....	58
3.9.4 Selectors and sequences.....	60
3.9.5 Customizing behaviors using decorators (meta-behaviors).....	61
3.10 Programming with Behavior Trees and ROS.....	63
3.10.1 Installing the pi_trees library.....	63
3.10.2 Basic components of the pi_trees library.....	63
3.10.3 ROS-specific behavior tree classes.....	68
3.10.4 A Patrol Bot example using behavior trees.....	72
3.10.5 A housing cleaning robot using behavior trees.....	79
3.10.6 Parallel tasks.....	85
3.10.7 Adding and removing tasks.....	87

4. Creating a URDF Model for your Robot.....	89
4.1 Start with the Base and Wheels.....	90
4.1.1 The robot_state_publisher and joint_state_publisher nodes.....	91
4.1.2 The base URDF/Xacro file.....	92
4.1.3 Alternatives to using the /base_footprint frame.....	97
4.1.4 Adding the base to the robot model.....	97
4.1.5 Viewing the robot's transform tree.....	98
4.1.6 Using a mesh for the base.....	99
4.2 Simplifying Your Meshes.....	104
4.3 Adding a Torso.....	104
4.3.1 Modeling the torso.....	105
4.3.2 Attaching the torso to the base.....	106
4.3.3 Using a mesh for the torso.....	107
4.3.4 Adding the mesh torso to the mesh base.....	108
4.4 Measure, Calculate and Tweak.....	110
4.5 Adding a Camera.....	110
4.5.1 Placement of the camera.....	111
4.5.2 Modeling the camera.....	112
4.5.3 Adding the camera to the torso and base.....	114
4.5.4 Viewing the transform tree with torso and camera.....	115
4.5.5 Using a mesh for the camera.....	116
4.5.6 Using an Asus Xtion Pro instead of a Kinect.....	118
4.6 Adding a Laser Scanner (or other Sensors).....	119
4.6.1 Modeling the laser scanner.....	119
4.6.2 Attaching a laser scanner (or other sensor) to a mesh base.....	120
4.6.3 Configuring the laser node launch file.....	121
4.7 Adding a Pan and Tilt Head.....	122
4.7.1 Using an Asus Xtion Pro instead of a Kinect.....	124
4.7.2 Modeling the pan-and-tilt head.....	124
4.7.3 Figuring out rotation axes.....	127
4.7.4 A pan and tilt head using meshes on Pi Robot.....	128
4.7.5 Using an Asus Xtion Pro mesh instead of a Kinect on Pi Robot.....	129
4.8 Adding One or Two Arms.....	129
4.8.1 Placement of the arm(s).....	130
4.8.2 Modeling the arm.....	130
4.8.3 Adding a gripper frame for planning.....	133
4.8.4 Adding a second arm.....	134
4.8.5 Using meshes for the arm servos and brackets.....	136
4.9 Adding a Telescoping Torso to the Box Robot.....	138
4.10 Adding a Telescoping Torso to Pi Robot.....	139
4.11 A Tabletop One-Arm Pi Robot.....	140
4.12 Testing your Model with the ArbotiX Simulator.....	142
4.12.1 A fake Box Robot.....	142
4.12.2 A fake Pi Robot.....	145
4.13 Creating your own Robot Description Package.....	145
4.13.1 Using rosbUILD.....	145
4.13.2 Using catkin.....	146
4.13.3 Copying files from the rbx2_description package.....	147
4.13.4 Creating a test launch file.....	147

5. Controlling Dynamixel Servos: Take 2.....	149
5.1 Installing the ArbotiX Packages.....	149
5.2 Launching the ArbotiX Nodes.....	150
5.3 The ArbotiX Configuration File.....	154
5.4 Testing the ArbotiX Joint Controllers in Fake Mode.....	160
5.5 Testing the Arbotix Joint Controllers with Real Servos.....	162
5.6 Relaxing All Servos.....	165
5.7 Enabling or Disabling All Servos.....	168
6. Robot Diagnostics.....	169
6.1 The DiagnosticStatus Message.....	170
6.2 The Analyzer Configuration File.....	171
6.3 Monitoring Dynamixel Servo Temperatures.....	172
6.3.1 Monitoring the servos for a pan-and-tilt head.....	172
6.3.2 Viewing messages on the /diagnostics topic.....	175
6.3.3 Protecting servos by monitoring the /diagnostics topic.....	177
6.4 Monitoring a Laptop Battery.....	181
6.5 Creating your Own Diagnostics Messages.....	182
6.6 Monitoring Other Hardware States.....	188
7. Dynamic Reconfigure.....	191
7.1 Adding Dynamic Parameters to your own Nodes.....	192
7.1.1 Creating the .cfg file.....	192
7.1.2 Making the .cfg file executable.....	193
7.1.3 Configuring the CMakeLists.txt file.....	194
7.1.4 Building the package.....	194
7.2 Adding Dynamic Reconfigure Capability to the Battery Simulator Node.....	194
7.3 Adding Dynamic Reconfigure Client Support to a ROS Node.....	198
7.4 Dynamic Reconfigure from the Command Line.....	201
8. Multiplexing Topics with mux & yocs.....	203
8.1 Configuring Launch Files to Use mux Topics.....	204
8.2 Testing mux with the Fake TurtleBot.....	205
8.3 Switching Inputs using mux Services.....	206
8.4 A ROS Node to Prioritize mux Inputs.....	207
8.5 The YOCS Controller from Yujin Robot.....	210
8.5.1 Adding input sources.....	213
9. Head Tracking in 3D.....	215
9.1 Tracking a Fictional 3D Target.....	216
9.2 Tracking a Point on the Robot.....	217
9.3 The 3D Head Tracking Node.....	220
9.3.1 Real or fake head tracking.....	220
9.3.2 Projecting the target onto the camera plane.....	221
9.4 Head Tracking with Real Servos.....	224
9.4.1 Real servos and fake target.....	225

9.4.2 Real servos, real target.....	226
9.4.3 The nearest_cloud.py node and launch file.....	228
10. Detecting and Tracking AR Tags.....	233
10.1 Installing and Testing the ar_track_alvar Package.....	234
10.1.1 Creating your own AR Tags.....	234
10.1.2 Generating and printing the AR tags.....	236
10.1.3 Launching the camera driver and ar_track_alvar node.....	236
10.1.4 Testing marker detection.....	238
10.1.5 Understanding the /ar_pose_marker topic.....	238
10.1.6 Viewing the markers in RViz.....	240
10.2 Accessing AR Tag Poses in your Programs.....	240
10.2.1 The ar_tags_cog.py script.....	240
10.2.2 Tracking the tags with a pan-and-tilt head.....	244
10.3 Tracking Multiple Tags using Marker Bundles.....	245
10.4 Following an AR Tag with a Mobile Robot.....	245
10.4.1 Running the AR follower script on a TurtleBot.....	248
10.5 Exercise: Localization using AR Tags.....	249
11. Arm Navigation using MoveIt!.....	251
11.1 Do I Need a Real Robot with a Real Arm?.....	252
11.2 Degrees of Freedom.....	252
11.3 Joint Types.....	253
11.4 Joint Trajectories and the Joint Trajectory Action Controller.....	254
11.5 Forward and Inverse Arm Kinematics.....	257
11.6 Numerical versus Analytic Inverse Kinematics.....	258
11.7 The MoveIt! Architecture.....	258
11.8 Installing MoveIt!.....	260
11.9 Creating a Static URDF Model for your Robot.....	261
11.10 Running the MoveIt! Setup Assistant.....	262
11.10.1 Load the robot's URDF model.....	263
11.10.2 Generate the collision matrix.....	264
11.10.3 Add the base_odom virtual joint.....	264
11.10.4 Adding the right arm planning group.....	265
11.10.5 Adding the right gripper planning group.....	269
11.10.6 Defining robot poses.....	271
11.10.7 Defining end effectors.....	273
11.10.8 Defining passive joints.....	273
11.10.9 Generating the configuration files.....	273
11.11 Configuration Files Created by the MoveIt! Setup Assistant.....	275
11.11.1 The SRDF file (robot_name.srdf).....	275
11.11.2 The fake_controllers.yaml file.....	276
11.11.3 The joint_limits.yaml file.....	277
11.11.4 The kinematics.yaml file.....	278
11.12 The move_group Node and Launch File.....	280
11.13 Testing MoveIt! in Demo Mode.....	280
11.13.1 Exploring additional features of the Motion Planning plugin.....	284
11.13.2 Re-running the Setup Assistant at a later time.....	285

11.14	Testing MoveIt! from the Command Line.....	286
11.15	Determining Joint Configurations and End Effector Poses.....	289
11.16	Using the ArbotiX Joint Trajectory Action Controllers.....	292
11.16.1	Testing the ArbotiX joint trajectory action controllers in simulation.....	292
11.16.2	Testing the ArbotiX joint trajectory controllers with real servos.....	300
11.17	Configuring MoveIt! Joint Controllers.....	301
11.17.1	Creating the controllers.yaml file.....	302
11.17.2	Creating the controller manager launch file.....	304
11.18	The MoveIt! API.....	305
11.19	Forward Kinematics: Planning in Joint Space.....	306
11.20	Inverse Kinematics: Planning in Cartesian Space.....	314
11.21	Pointing at or Reaching for a Visual Target.....	322
11.22	Setting Constraints on Planned Trajectories.....	324
11.22.1	Executing Cartesian Paths.....	324
11.22.2	Setting other path constraints.....	330
11.23	Adjusting Trajectory Speed.....	333
11.24	Adding Obstacles to the Planning Scene.....	337
11.25	Attaching Objects and Tools to the Robot.....	346
11.26	Pick and Place.....	348
11.27	Adding a Sensor Controller.....	360
11.28	Running MoveIt! on a Real Arm.....	363
11.28.1	Creating your own launch files and scripts.....	364
11.28.2	Running the robot's launch files.....	364
11.28.3	Forward kinematics on a real arm.....	365
11.28.4	Inverse kinematics on a real arm.....	366
11.28.5	Cartesian paths on a real arm.....	367
11.28.6	Pick-and-place on a real arm.....	367
11.28.7	Pointing at or reaching for a visual target.....	367
11.29	Creating a Custom Fast IK Plugin.....	368

12. Gazebo: Simulating Worlds and Robots.....375

12.1	Installing Gazebo.....	376
12.2	Hardware Graphics Acceleration.....	377
12.3	Installing the ROS Gazebo Packages.....	378
12.4	Installing the Kobuki ROS Packages.....	379
12.5	Installing the UBR-1 Files.....	379
12.6	Using the Gazebo GUI.....	379
12.7	Missing Model Bug in Gazebo 1.9.....	381
12.8	Testing the Kobuki Robot in Gazebo.....	383
12.8.1	Accessing simulated sensor data.....	385
12.8.2	Adding safety control to the Kobuki.....	389
12.8.3	Running the nav_square.py script from Volume 1.....	391
12.9	Loading Other Worlds and Objects.....	392
12.10	Testing the UBR-1 Robot in Gazebo.....	393
12.10.1	UBR-1 joint trajectories.....	394
12.10.2	The UBR-1 and MoveIt!.....	395
12.11	Real Pick-and-Place using the UBR-1 Perception Pipeline.....	397
12.11.1	Limitations of depth cameras.....	398

12.11.2 Running the demo.....	399
12.11.3 Understanding the real_pick_and_place.py script.....	404
12.12 Running Gazebo Headless + RViz.....	407
13. Rosbridge: Building a Web GUI for your Robot.....	411
13.1 Installing the rosbridge Packages.....	411
13.2 Installing the mjpeg_sever Package.....	412
13.3 Installing a Simple Web Server (mini-httpd).....	415
13.4 Starting mini-httpd, rosbridge and mjpeg_server.....	416
13.5 A Simple rosbridge HTML/Javascript GUI.....	417
13.6 Testing the GUI with a Fake TurtleBot.....	420
13.7 Testing the GUI with a Real Robot.....	420
13.8 Viewing the Web GUI on another Device on your Network.....	421
13.9 Using the Browser Debug Console.....	421
13.10 Understanding the Simple GUI.....	423
13.10.1 The HTML layout: simple_gui.html.....	423
13.10.2 The JavaScript code: simple_gui.js.....	428
13.11 A More Advanced GUI using jQuery, jqWidgets and KineticJS.....	438
13.12 Rosbridge Summary.....	443
Appendix: Plug and Play USB Devices for ROS: Creating udev Rules.....	445
13.13 Adding yourself to the dialout Group.....	445
13.14 Determining the Serial Number of a Device.....	446
13.15 UDEV Rules.....	447
13.16 Testing a UDEV Rule.....	448
13.17 Using a UDEV Device Name in a ROS Configuration File.....	448