ROSDextrousHandinterface

Generated by Doxygen 1.6.3

Mon Nov 8 12:48:46 2010

Contents

1 Main Page **1**

1.1 Introduction . . . 1

1.2 How to use this interface . . . 1

1.2.1 Virtual Hand . . . 1

1.2.2 Real Hand . . . 1

1.2.3 Generic use - Hand . . . 2

1.2.4 Generic use - Arm . . . 2

1.3 Details on how all this is organized . . . 3

1.4 Control Graphical Interface . . . 3

1.4.1 How to use the GUI. . . . 3

1.5 Simulation Using Gazebo . . . 4

1.5.1 Launching Gazebo and spawning your model . . . 4

1.5.2 Making your robot move . . . 4

1.5.3 Modifying the PID values . . . 4

1.6 Modifying the Models . . . 5

1.6.1 Modify the .urdf.xacro file . . . 5

1.6.2 Generate the .urdf file . . . 5

1.7 Code API . . . 6

1.8 Examples . . . 6

2 Todo List **7**

3 Namespace Index **9**

3.1 Namespace List . . . 9

4 Data Structure Index **11**

4.1 Class Hierarchy . . . 11

5 Data Structure Index **13**

5.1 Data Structures . . . 13

6 File Index **15**

6.1 File List 15

7 Namespace Documentation **17**

7.1 sr_hand::msg::_cartesian_data Namespace Reference 17

7.1.1 Detailed Description 17

7.2 sr_hand::msg::_cartesian_position Namespace Reference 18

7.2.1 Detailed Description 18

7.3 sr_hand::msg::_command Namespace Reference 19

7.3.1 Detailed Description 19

7.4 sr_hand::msg::_config Namespace Reference 20

7.4.1 Detailed Description 20

7.5 sr_hand::msg::_contrlr Namespace Reference 21

7.5.1 Detailed Description 21

7.6 sr_hand::msg::_joint Namespace Reference 22

7.6.1 Detailed Description 22

7.7 sr_hand::msg::_joints_data Namespace Reference 23

7.7.1 Detailed Description 23

7.8 sr_hand::msg::_reverseKinematics Namespace Reference 24

7.8.1 Detailed Description 24

7.9 sr_hand::msg::_sendupdate Namespace Reference 25

7.9.1 Detailed Description 25

8 Data Structure Documentation **27**

8.1 sr_hand::msg::_cartesian_data::cartesian_data Class Reference 27

8.1.1 Member Function Documentation 27

8.1.1.1 __init__ 27

8.1.1.2 deserialize 27

8.1.1.3 deserialize_numpy 28

8.1.1.4 serialize 28

8.1.1.5 serialize_numpy 28

8.2 sr_hand::cartesian_data_< ContainerAllocator > Struct Template Reference 29

8.3 sr_hand::msg::_cartesian_position::cartesian_position Class Reference 30

8.3.1 Member Function Documentation 30

8.3.1.1 __init__ 30

8.3.1.2 deserialize 30

8.3.1.3 deserialize_numpy 30

8.3.1.4 serialize . . . 31
8.3.1.5 serialize_numpy . . . 31
8.4 sr_hand::cartesian_position_< ContainerAllocator > Struct Template Reference . . . 32
8.5 sr_hand::msg::_command::command Class Reference . . . 33
8.5.1 Member Function Documentation . . . 33
8.5.1.1 __init__ . . . 33
8.5.1.2 deserialize . . . 33
8.5.1.3 deserialize_numpy . . . 33
8.5.1.4 serialize . . . 33
8.5.1.5 serialize_numpy . . . 34
8.6 sr_hand::command_< ContainerAllocator > Struct Template Reference . . . 35
8.7 sr_hand::msg::_config::config Class Reference . . . 36
8.7.1 Member Function Documentation . . . 36
8.7.1.1 __init__ . . . 36
8.7.1.2 deserialize . . . 36
8.7.1.3 deserialize_numpy . . . 36
8.7.1.4 serialize . . . 36
8.7.1.5 serialize_numpy . . . 37
8.8 sr_hand::config_< ContainerAllocator > Struct Template Reference . . . 38
8.9 sr_hand::msg::_contrlr::contrlr Class Reference . . . 40
8.9.1 Member Function Documentation . . . 40
8.9.1.1 __init__ . . . 40
8.9.1.2 deserialize . . . 40
8.9.1.3 deserialize_numpy . . . 40
8.9.1.4 serialize . . . 40
8.9.1.5 serialize_numpy . . . 41
8.10 sr_hand::contrlr_< ContainerAllocator > Struct Template Reference . . . 42
8.11 ros::message_traits::DataType< ::sr_hand::cartesian_data_< ContainerAllocator > > Struct Template Reference . . . 44
8.12 ros::message_traits::DataType< ::sr_hand::cartesian_position_< ContainerAllocator > > Struct Template Reference . . . 45
8.13 ros::message_traits::DataType< ::sr_hand::command_< ContainerAllocator > > Struct Template Reference . . . 46
8.14 ros::message_traits::DataType< ::sr_hand::config_< ContainerAllocator > > Struct Template Reference . . . 47
8.15 ros::message_traits::DataType< ::sr_hand::contrlr_< ContainerAllocator > > Struct Template Reference . . . 48

8.16 ros::message_traits::DataType< ::sr_hand::joint_< ContainerAllocator > > Struct Template Reference . . . 49
8.17 ros::message_traits::DataType< ::sr_hand::joints_data_< ContainerAllocator > > Struct Template Reference . . . 50
8.18 ros::message_traits::DataType< ::sr_hand::reverseKinematics_< ContainerAllocator > > Struct Template Reference . . . 51
8.19 ros::message_traits::DataType< ::sr_hand::sendupdate_< ContainerAllocator > > Struct Template Reference . . . 52
8.20 ros::message_traits::Definition< ::sr_hand::cartesian_data_< ContainerAllocator > > Struct Template Reference . . . 53
8.21 ros::message_traits::Definition< ::sr_hand::cartesian_position_< ContainerAllocator > > Struct Template Reference . . . 54
8.22 ros::message_traits::Definition< ::sr_hand::command_< ContainerAllocator > > Struct Template Reference . . . 55
8.23 ros::message_traits::Definition< ::sr_hand::config_< ContainerAllocator > > Struct Template Reference . . . 56
8.24 ros::message_traits::Definition< ::sr_hand::contrlr_< ContainerAllocator > > Struct Template Reference . . . 57
8.25 ros::message_traits::Definition< ::sr_hand::joint_< ContainerAllocator > > Struct Template Reference . . . 58
8.26 ros::message_traits::Definition< ::sr_hand::joints_data_< ContainerAllocator > > Struct Template Reference . . . 59
8.27 ros::message_traits::Definition< ::sr_hand::reverseKinematics_< ContainerAllocator > > Struct Template Reference . . . 60
8.28 ros::message_traits::Definition< ::sr_hand::sendupdate_< ContainerAllocator > > Struct Template Reference . . . 61
8.29 shadowrobot::DiagnosticData Struct Reference . . . 62
8.29.1 Detailed Description . . . 62
8.30 Grasp::Grasp Class Reference . . . 63
8.31 grasps_interpoler::GraspInterpoler Class Reference . . . 64
8.31.1 Member Function Documentation . . . 64
8.31.1.1 __init__ . . . 64
8.31.1.2 set_percentage_grasp1_to_grasp2 . . . 64
8.32 grasps_parser::GraspParser Class Reference . . . 65
8.32.1 Detailed Description . . . 65
8.32.2 Member Function Documentation . . . 65
8.32.2.1 parse_tree . . . 65
8.33 shadowhand_ros::Joint Class Reference . . . 66
8.34 sr_hand::msg::_joint::joint Class Reference . . . 67
8.34.1 Member Function Documentation . . . 67
8.34.1.1 __init__ . . . 67

8.34.1.2 deserialize . . . 67
8.34.1.3 deserialize_numpy . . . 67
8.34.1.4 serialize . . . 68
8.34.1.5 serialize_numpy . . . 68
8.35 sr_hand::joint_< ContainerAllocator > Struct Template Reference . . . 69
8.36 shadowrobot::JointControllerData Struct Reference . . . 71
8.36.1 Detailed Description . . . 71
8.37 shadowrobot::JointData Struct Reference . . . 72
8.37.1 Detailed Description . . . 72
8.38 sr_hand::msg::_joints_data::joints_data Class Reference . . . 73
8.38.1 Member Function Documentation . . . 73
8.38.1.1 __init__ . . . 73
8.38.1.2 deserialize . . . 73
8.38.1.3 deserialize_numpy . . . 73
8.38.1.4 serialize . . . 73
8.38.1.5 serialize_numpy . . . 74
8.39 sr_hand::joints_data_< ContainerAllocator > Struct Template Reference . . . 75
8.40 ros::message_traits::MD5Sum< ::sr_hand::cartesian_data_< ContainerAllocator > > Struct Template Reference . . . 76
8.41 ros::message_traits::MD5Sum< ::sr_hand::cartesian_position_< ContainerAllocator > > Struct Template Reference . . . 77
8.42 ros::message_traits::MD5Sum< ::sr_hand::command_< ContainerAllocator > > Struct Template Reference . . . 78
8.43 ros::message_traits::MD5Sum< ::sr_hand::config_< ContainerAllocator > > Struct Template Reference . . . 79
8.44 ros::message_traits::MD5Sum< ::sr_hand::contrlr_< ContainerAllocator > > Struct Template Reference . . . 80
8.45 ros::message_traits::MD5Sum< ::sr_hand::joint_< ContainerAllocator > > Struct Template Reference . . . 81
8.46 ros::message_traits::MD5Sum< ::sr_hand::joints_data_< ContainerAllocator > > Struct Template Reference . . . 82
8.47 ros::message_traits::MD5Sum< ::sr_hand::reverseKinematics_< ContainerAllocator > > Struct Template Reference . . . 83
8.48 ros::message_traits::MD5Sum< ::sr_hand::sendupdate_< ContainerAllocator > > Struct Template Reference . . . 84
8.49 shadowrobot::Parameters Struct Reference . . . 85
8.49.1 Detailed Description . . . 85
8.50 ros::message_operations::Printer< ::sr_hand::cartesian_data_< ContainerAllocator > > Struct Template Reference . . . 86

8.51 ros::message_operations::Printer< ::sr_hand::cartesian_position_< ContainerAllocator > > Struct Template Reference . . . 87
8.52 ros::message_operations::Printer< ::sr_hand::command_< ContainerAllocator > > Struct Template Reference . . . 88
8.53 ros::message_operations::Printer< ::sr_hand::config_< ContainerAllocator > > Struct Template Reference . . . 89
8.54 ros::message_operations::Printer< ::sr_hand::contrlr_< ContainerAllocator > > Struct Template Reference . . . 90
8.55 ros::message_operations::Printer< ::sr_hand::joint_< ContainerAllocator > > Struct Template Reference . . . 91
8.56 ros::message_operations::Printer< ::sr_hand::joints_data_< ContainerAllocator > > Struct Template Reference . . . 92
8.57 ros::message_operations::Printer< ::sr_hand::reverseKinematics_< ContainerAllocator > > Struct Template Reference . . . 93
8.58 ros::message_operations::Printer< ::sr_hand::sendupdate_< ContainerAllocator > > Struct Template Reference . . . 94
8.59 shadowrobot::RealShadowhand Class Reference . . . 95
8.59.1 Detailed Description . . . 95
8.59.2 Constructor & Destructor Documentation . . . 95
8.59.2.1 RealShadowhand . . . 95
8.59.3 Member Function Documentation . . . 95
8.59.3.1 getAllJointsData . . . 95
8.59.3.2 getConfig . . . 96
8.59.3.3 getContrl . . . 96
8.59.3.4 getDiagnostics . . . 96
8.59.3.5 getJointData . . . 96
8.59.3.6 initializeMap . . . 97
8.59.3.7 sendupdate . . . 97
8.59.3.8 setConfig . . . 97
8.59.3.9 setContrl . . . 98
8.60 sr_hand::msg::_reverseKinematics::reverseKinematics Class Reference . . . 99
8.60.1 Member Function Documentation . . . 99
8.60.1.1 __init__ . . . 99
8.60.1.2 deserialize . . . 99
8.60.1.3 deserialize_numpy . . . 99
8.60.1.4 serialize . . . 99
8.60.1.5 serialize_numpy . . . 100
8.61 sr_hand::reverseKinematics_< ContainerAllocator > Struct Template Reference . . . 101
8.62 sr_hand::msg::_sendupdate::sendupdate Class Reference . . . 102

8.62.1 Member Function Documentation . . . 102
8.62.1.1 __init__ . . . 102
8.62.1.2 deserialize . . . 102
8.62.1.3 deserialize_numpy . . . 102
8.62.1.4 serialize . . . 102
8.62.1.5 serialize_numpy . . . 103
8.63 sr_hand::sendupdate_< ContainerAllocator > Struct Template Reference . . . 104
8.64 ros::serialization::Serializer< ::sr_hand::cartesian_data_< ContainerAllocator > > Struct Template Reference . . . 105
8.65 ros::serialization::Serializer< ::sr_hand::cartesian_position_< ContainerAllocator > > Struct Template Reference . . . 106
8.66 ros::serialization::Serializer< ::sr_hand::command_< ContainerAllocator > > Struct Template Reference . . . 107
8.67 ros::serialization::Serializer< ::sr_hand::config_< ContainerAllocator > > Struct Template Reference . . . 108
8.68 ros::serialization::Serializer< ::sr_hand::contrlr_< ContainerAllocator > > Struct Template Reference . . . 109
8.69 ros::serialization::Serializer< ::sr_hand::joint_< ContainerAllocator > > Struct Template Reference . . . 110
8.70 ros::serialization::Serializer< ::sr_hand::joints_data_< ContainerAllocator > > Struct Template Reference . . . 111
8.71 ros::serialization::Serializer< ::sr_hand::reverseKinematics_< ContainerAllocator > > Struct Template Reference . . . 112
8.72 ros::serialization::Serializer< ::sr_hand::sendupdate_< ContainerAllocator > > Struct Template Reference . . . 113
8.73 shadowhand_ros::ShadowHand_ROS Class Reference . . . 114
8.73.1 Detailed Description . . . 115
8.73.2 Member Function Documentation . . . 115
8.73.2.1 __init__ . . . 115
8.73.2.2 callback . . . 115
8.73.2.3 callback_arm . . . 115
8.73.2.4 callVisualisationService . . . 115
8.73.2.5 check_hand_presence . . . 115
8.73.2.6 has_arm . . . 115
8.73.2.7 init_actual_joints . . . 115
8.73.2.8 read_all_current_arm_positions . . . 116
8.73.2.9 read_all_current_arm_targets . . . 116
8.73.2.10 read_all_current_positions . . . 116
8.73.2.11 read_all_current_targets . . . 116

8.73.2.12 record_step_to_file . . . 116
8.73.2.13 resend_targets . . . 116
8.73.2.14 sendupdate . . . 116
8.73.2.15 sendupdate_arm . . . 116
8.73.2.16 sendupdate_arm_from_dict . . . 117
8.73.2.17 sendupdate_from_dict . . . 117
8.73.2.18 set_sendupdate_topic . . . 117
8.73.2.19 set_shadowhand_data_topic . . . 117
8.73.2.20 valueof . . . 117
8.74 shadowrobot::SRArticulatedRobot Class Reference . . . 118
8.74.1 Detailed Description . . . 119
8.74.2 Member Typedef Documentation . . . 119
8.74.2.1 JointsMap . . . 119
8.74.2.2 ParametersMap . . . 119
8.74.3 Constructor & Destructor Documentation . . . 119
8.74.3.1 SRArticulatedRobot . . . 119
8.74.3.2 ~SRArticulatedRobot . . . 119
8.74.4 Member Function Documentation . . . 119
8.74.4.1 getAllJointsData . . . 119
8.74.4.2 getConfig . . . 119
8.74.4.3 getContrl . . . 120
8.74.4.4 getDiagnostics . . . 120
8.74.4.5 getJointData . . . 120
8.74.4.6 sendupdate . . . 121
8.74.4.7 setConfig . . . 121
8.74.4.8 setContrl . . . 121
8.75 shadowrobot::SRDiagnosticer Class Reference . . . 122
8.75.1 Constructor & Destructor Documentation . . . 122
8.75.1.1 SRDiagnosticer . . . 122
8.75.2 Member Function Documentation . . . 122
8.75.2.1 publish . . . 122
8.76 test::SRLibrary Class Reference . . . 123
8.76.1 Detailed Description . . . 123
8.77 shadowrobot::SRPublisher Class Reference . . . 124
8.77.1 Detailed Description . . . 124
8.77.2 Constructor & Destructor Documentation . . . 124

8.77.2.1 SRPublisher . . . 124
8.77.3 Member Function Documentation . . . 124
8.77.3.1 publish . . . 124
8.78 shadowrobot::SRSubscriber Class Reference . . . 125
8.78.1 Detailed Description . . . 125
8.78.2 Constructor & Destructor Documentation . . . 125
8.78.2.1 SRSubscriber . . . 125
8.78.2.2 SRSubscriber . . . 125
8.79 shadowrobot::SRTactileSensorPublisher Class Reference . . . 126
8.80 shadowrobot::VirtualArm Class Reference . . . 127
8.80.1 Constructor & Destructor Documentation . . . 127
8.80.1.1 VirtualArm . . . 127
8.80.2 Member Function Documentation . . . 127
8.80.2.1 getAllJointsData . . . 127
8.80.2.2 getConfig . . . 128
8.80.2.3 getContrl . . . 128
8.80.2.4 getDiagnostics . . . 128
8.80.2.5 getJointData . . . 128
8.80.2.6 initializeMap . . . 129
8.80.2.7 sendupdate . . . 129
8.80.2.8 setConfig . . . 129
8.80.2.9 setContrl . . . 129
8.80.2.10 toDegrees . . . 130
8.80.2.11 toRad . . . 130
8.81 shadowrobot::VirtualShadowhand Class Reference . . . 131
8.81.1 Detailed Description . . . 132
8.81.2 Member Typedef Documentation . . . 132
8.81.2.1 ControllersMap . . . 132
8.81.3 Constructor & Destructor Documentation . . . 132
8.81.3.1 VirtualShadowhand . . . 132
8.81.4 Member Function Documentation . . . 132
8.81.4.1 getAllJointsData . . . 132
8.81.4.2 getConfig . . . 132
8.81.4.3 getContrl . . . 132
8.81.4.4 getDiagnostics . . . 133
8.81.4.5 getJointData . . . 133

8.81.4.6 initializeMap . . . 133
8.81.4.7 sendupdate . . . 133
8.81.4.8 setConfig . . . 134
8.81.4.9 setContrl . . . 134
8.81.4.10 toDegrees . . . 134
8.81.4.11 toRad . . . 135

9 File Documentation **137**

9.1 link_joints.cpp File Reference . . . 137
9.1.1 Detailed Description . . . 137
9.1.2 Function Documentation . . . 138
9.1.2.1 callback . . . 138
9.1.2.2 main . . . 138
9.2 real_shadowhand.cpp File Reference . . . 139
9.2.1 Detailed Description . . . 139
9.3 real_shadowhand.h File Reference . . . 140
9.3.1 Detailed Description . . . 140
9.4 real_shadowhand_node.cpp File Reference . . . 141
9.4.1 Detailed Description . . . 141
9.4.2 Function Documentation . . . 141
9.4.2.1 main . . . 141
9.5 sr_diagnosticer.cpp File Reference . . . 143
9.5.1 Detailed Description . . . 143
9.6 sr_diagnosticer.h File Reference . . . 144
9.6.1 Detailed Description . . . 144
9.7 sr_publisher.cpp File Reference . . . 145
9.7.1 Detailed Description . . . 145
9.8 sr_publisher.h File Reference . . . 146
9.8.1 Detailed Description . . . 146
9.9 sr_subscriber.cpp File Reference . . . 147
9.9.1 Detailed Description . . . 147
9.10 sr_subscriber.h File Reference . . . 148
9.10.1 Detailed Description . . . 148
9.11 sr_tactile_sensor_pub.cpp File Reference . . . 149
9.11.1 Detailed Description . . . 149
9.12 sr_tactile_sensor_pub.h File Reference . . . 150
9.12.1 Detailed Description . . . 150

9.13 sr_tactile_sensor_pub_node.cpp File Reference 151
9.13.1 Detailed Description 151
9.13.2 Function Documentation 151
9.13.2.1 main 151
9.14 virtual_arm.cpp File Reference 152
9.14.1 Detailed Description 152
9.15 virtual_arm.h File Reference 153
9.15.1 Detailed Description 153
9.16 virtual_arm_node.cpp File Reference 154
9.16.1 Detailed Description 154
9.16.2 Function Documentation 154
9.16.2.1 main 154
9.17 virtual_shadowhand.cpp File Reference 156
9.17.1 Detailed Description 156
9.18 virtual_shadowhand.h File Reference 157
9.18.1 Detailed Description 157
9.19 virtual_shadowhand_node.cpp File Reference 158
9.19.1 Detailed Description 158
9.19.2 Function Documentation 158
9.19.2.1 main 158

10 Example Documentation **161**

10.1 /examples/shadowhand_publisher.py 161
10.2 /examples/shadowhand_subscriber.py 162
10.3 link_joints.cpp 163
10.4 link_joints.py 164

Chapter 1

Main Page

1.1 Introduction

This is a generic ROS interface used to access and experiment with Shadow Robot's hardware. It can be used either for accessing the real hardware or use a virtual hand to test your algorithm / utilities. It also contains an interface to Shadow Robot's muscle arm.

1.2 How to use this interface

If you didn't pull the branch in the stacks (.path/to/ros/stacks), don't forget to add the path to the shadow_robot stack to your `ROS_PACKAGE_PATH` for the relevant packages to be found by ROS. To do this, edit your .bashrc: after the line "source /path/to/ros/setup.sh", add :

```
export ROS_PACKAGE_PATH=${ROS_PACKAGE_PATH}:/path/to/shadow_robot
```

1.2.1 Virtual Hand

To run the virtual Dextrous Hand, just compile the code:

```
rosmake --rosdep-install sr_hand
```

Then start the ROS interface, if you want to have a motor hand:

```
roslaunch sr_hand srh_motor.launch
```

Otherwise, if it's a muscle hand:

```
roslaunch sr_hand srh_muscle.launch
```

1.2.2 Real Hand

If you have a real Shadow Dextrous Hand, you need to uncomment the relevent region in the CMakelist.txt

```
include_directories(/usr/realtime/include/)
LINK_LIBRARIES(pthread)
add_library(robot STATIC IMPORTED)
set_property(TARGET robot PROPERTY IMPORTED_LOCATION /usr/lib/robot.a)
```

```
rosbuild_add_executable(shadowhand_real src/hand/shadowhand.cpp src/hand/real_sha
      dowhand.cpp src/sr_publisher.cpp
src/sr_subscriber.cpp src/hand/real_shadowhand_node.cpp src/sr_diagnosticer.cpp)
rosbuild_link_boost(shadowhand_real thread)
target_link_libraries(shadowhand_real robot)
```

Then compile the code:

```
rosmake --rosdep-install sr_hand
```

You then need to modify slightly the srh_motor.launch file (or srh_muscle.launch), on line 5 replace virtual by real:

```
<node pkg="sr_hand" name="shadowhand" type="shadowhand_real" >
```

Then start the ROS interface:

```
roslaunch sr_hand [srh_motor.launch/srh_muscle.launch]
```

1.2.3 Generic use - Hand

Once you run either a virtual or real hand, you can use the different tools provided by ROS / Shadow without changing anything on either types of hand.

If you want to view the hand in **rviz**, just run

```
roslaunch sr_hand [rviz_muscle.launch/rviz_motor.launch]
```

You then need to add one `robot_model` to visualize the current positions of the hand. To do this, click on Add at the bottom left of rviz, then select **Robot Model**. You can also visualize the targets of the hand by using a second `robot_model` in **rviz**. Don't forget to specify the correct topics:

- To display the hand position: set **Robot Description** to **hand_description** and **TF Prefix** to **/srh/-position** (on the left hand side, in the parameters of your robot_model).
- To display the hand targets: set **Robot Description** to **hand_description** and **TF Prefix** to **/srh/target**.

You should see the model displayed in rviz:

If you need to monitor the diagnostics published by the hand, you can use the following command:

```
rosrun robot_monitor robot_monitor
```

.

1.2.4 Generic use - Arm

To start the arm with the hand attached, run:

```
roslaunch sr_hand [sr_arm_motor.launch/sr_arm_muscle.launch]
```

You can then visualize it in rviz:

```
roslaunch sr_hand [rviz_motor.launch/rviz_muscle.launch]
```

You then need to add one `robot_model` to view the current positions of the arm. You can also visualize the targets of the arm by using a second `robot_model` in **rviz**. Don't forget to specify the correct topics:

- To display the arm position: set **Robot Description** to **arm_description** and **TF Prefix** to **/sr_-arm/position** (on the left hand side, in the parameters of your robot_model).
- To display the arm targets: set Robot **Description** to **arm_description** and **TF Prefix** to **/sr_-arm/target**.

1.3 Details on how all this is organized

To visualize the current organization of the ros nodes, once you've started the shadowhand node by running one of the above commands, you can run the command

```
rxplot
```

As you can see on this diagram generated by rxplot, the shadowhand publishes to two different joint_-states topics: one for the targets, one for the position. The joint_states topic is a topic where the current angle, velocity and effort of each joint is published. Those joint_states topics are then transformed to position information by two different robot_state_publisher, using the file shadowhand.xml which contains the model of Shadow Dextrous Hand. The robot_state_publisher then publishes on the tf topic. This tf topic can then be visualized in rviz. The shadowhand node publishes to the diagnostics topic which is then aggregated by a diagnostic_aggregator node.

Additional data is also published on the /srh/shadowhand_data topic. If you want to look what's published on a topic, you can either look at a simple example (shadowhand_subscriber.py), or simply use the command:

```
rostopic echo /srh/shadowhand_data
```

To send commands to the shadowhand, you can send messages to the following topics.

- /srh/sendupdate to send new targets to the hand
- /srh/contrlr to set the controler parameters
- /srh/config to set the palm config

To send information to a topic, you can either use a c++ code or a python code. Please refer to the example: shadowhand_publisher.py.

For more information regarding those topics, please use the command `rostopic info`

1.4 Control Graphical Interface

sr_control_gui is a graphical application which integrates most of the tools needed by the end-user to control Shadow Robot's Hardware.

1.4.1 How to use the GUI.

To start the GUI, run:

```
rosrun sr_control_gui sr_control_gui
```

The GUI is organized in tabs :

- **Hand control** provides sliders to control the hand joint by joint, displaying actual positions and current targets set by the sliders. It also provides a grasp manager which allows to go from one saved grasp to another smoothly, moving all the fingers at the same time. It is also possible to come back to the former grasp. Finally, grasps can be saved into the grasps.xml file, using the saver. Beware, only the selected joints (ticked in the joint chooser) will be writen to the file.
- **Visualisation** provides various services to launch external useful programs, such as ROS utils. The main part, which is not implemented yet, will allow to choose which information should be displayed in Rviz.
- **Player** is used to choose a file containing a list of grasps to replay step by step. Two options are available : playing them by clicking on play for each grasps, or playing the whole file with a given time between each grasp. This is design to be used with the record button on the menu. The field must be filled with a filename to save the grasps.
- **Accessories** provides services to control some spare parts of the hand, such as the Shadow Arm, which can be controled like the hand, the Cyberglove (including a calibration service) or the Cybergrasp (not implemented yet). It is also possible to change the topics read by the program in order to change the hand controled.

1.5 Simulation Using Gazebo

The ROS interface comes with a model which can be used in gazebo. Gazebo is encapsulated in ROS. To compile the ROS version of gazebo, run:

```
rosmake --rosdep-install gazebo_worlds
```

1.5.1 Launching Gazebo and spawning your model

You then can start any model that you want using the different launchers in **launch/gazebo**. For instance, if you want to spawn the motor hand in gazebo:

```
roslaunch sr_hand gazebo_hand_motor.launch
```

1.5.2 Making your robot move

When Gazebo is running with the robot model, you can use motion commands to make the robot move. Assuming you have the hand running (see picture above), you can move FFJ3 (the first finger proximal joint) by running:

```
rostopic pub /ffproximal_controller/command std_msgs/Float64 0.6
```

In this example, ffproximal_controller can be any other hand controller, and 0.6 is the position of the joint, 0 being the lower limit and 1 the higher limit of the allowed motion angle. Back in Gazebo, the finger should be like this:

1.5.3 Modifying the PID values

The models that are spawned in Gazebo are moveable thanks to PID controllers, which values are easy to modify, in order to set the behaviour of the robot as you wish. If you want to modify the hand PID values, just modify the **model/hand_urdf/hand_controller.yaml** file. If you want to modify the arm PID values, modify the **model/arm_urdf/arm_controller.yaml** file In those files each controller looks like:

```
wrist_controller:
  type: robot_mechanism_controllers/JointPositionController
  joint: WRJ2
  pid:
    p: 0.0
    i: 0.0
    d: 0.0
    i_clamp: 1
```

Modifying the P, I and D values will affect the speed and accuracy of your robot motions. When those values are set, just save the file and run your launch file again to apply the modifications.

For more information on how to use gazebo in ROS, you should probably have a look at: http://www.ros.org/wiki/simulator_gazebo/Tutorials/Gazebo_ROS_API

1.6 Modifying the Models

The URDF Models used for the visualization (with Rviz) and the simulation (with Gazebo) are available in model/arm_urdf and model/hand_urdf. The arm and hand models are using xacro macros, which allow different robots to be linked in Gazebo (in order to simulate the behaviour of the arm with the hand at its extremity). If you want to modify something in the models, you have two things to do: change the informations in the right .urdf.xacro file, and then generate the .urdf file that will be used as a model.

1.6.1 Modify the .urdf.xacro file

The architecture of the robot is as follows (each robot has a motor version and a muscle version):

- An arm_and_hand is composed of a full_arm and a full_hand.
- An sr_arm is composed of a single full_arm
- A shadowhand is composed of a single full_hand
- A full_arm is composed of a shadowarm_base, a shadowarm_trunk, a shadowarm_upperarm, a shadowarm_lowerarm and a shadowarm_handsupport
- A full_hand is composed of a forearm, a wrist, a palm, a thumb, three fingers and a little_finger
- A finger is composed of a knuckle, a proximal, a middle and a distal, plus a metacarpal for the little_finger.
- A thumb is a little more complicated, with a hub and a base to add the missing motions.

All those parts are represented by "link" tags in the URDF files, and linked between them by "joint" tags. To modify data in a part of the hand, open the right .urdf.xacro file (for example to modify the size of the last part of each finger, you want to open the model/hand_urdf/finger/distal.urdf.xacro). Thus, the modification will be applied to all the identical parts of the hand. Then, just save the file.

1.6.2 Generate the .urdf file

the .urdf.xacro model has to be turned into a classic .urdf file to be used by ROS. To generate the good file, use the following command: (this example generates the motor version of the hand)

```
rosrun xacro xacro.py -o `rospack find sr_hand`/model/robots/shadowhand_motor.urdf `rospack find sr_hand`/
```

The .urdf.xacro robot model which is in the model/robots/xacro directory is turned into an urdf file in the model/robots directory (Warning: the urdf file in the model/robots directory has to exist before running the command, because the order is to overwrite what was in the file)

1.7 Code API

The classes directly interacting or simulating the Dextrous hand are the following:

- The generic parent class for the Dextrous Hands can be found at shadowhand::Shadowhand.
- The virtual Shadow Dextrous Hand can be found here shadowhand::VirtualShadowhand.
- To look at the real Dextrous Hand documentation refer to shadowhand::RealShadowhand.
- The virtual Shadow Arm can be found here: shadowhand::VirtualArm.

You can find the documentation regarding the different publishers and the subscriber here: shadowhand_publisher::ShadowhandPublisher, shadowhand_subscriber::ShadowhandSubscriber, shadowhand_diagnosticer::ShadowhandDiagnosticer .

The package also contains a python library to access the ROS interface through easy to use high level python commands. Please refer to shadowhand_ros.py. This library is used to develop the sr_control_gui interface.

1.8 Examples

There's a few examples in the examples directory in sr_hand. You can have a look at the example link_joints.cpp, or the same example written in python: link_joints.py. To run them, just run the following commands in different terminals:

```
roslaunch sr_hand srh_motor.launch
roslaunch sr_hand rviz_motor.launch
rosrun sr_control_gui __init__.py
```

and then

```
rosrun sr_hand link_joints
```

or

```
rosrun sr_hand link_joints.py
```

If you move the joint slider for FFJ3, then MFJ3 will move as well.

Chapter 2

Todo List

File real_shadowhand.h Make sure it works with the motor hand.

Global shadowrobot::SRArticulatedRobot::getConfig(std::string joint_name)=0 Not implemented yet

Global shadowrobot::SRArticulatedRobot::setConfig(std::vector< std::string > myConfig)=0 Not implemented yet

Global shadowrobot::VirtualArm::sendupdate(std::string joint_name, double target) This could be improved by implementing a control algorithm in this theoretic arm.

Global shadowrobot::VirtualShadowhand::sendupdate(std::string joint_name, double target) This could be improved by implementing a control algorithm in this theoretic hand.

Chapter 3

Namespace Index

3.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

sr_hand::msg::_cartesian_data 17
sr_hand::msg::_cartesian_position 18
sr_hand::msg::_command 19
sr_hand::msg::_config 20
sr_hand::msg::_contrlr 21
sr_hand::msg::_joint 22
sr_hand::msg::_joints_data 23
sr_hand::msg::_reverseKinematics 24
sr_hand::msg::_sendupdate 25

Chapter 4

Data Structure Index

4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

sr_hand::msg::_cartesian_data::cartesian_data . . . 27
sr_hand::cartesian_data_< ContainerAllocator > . . . 29
sr_hand::msg::_cartesian_position::cartesian_position . . . 30
sr_hand::cartesian_position_< ContainerAllocator > . . . 32
sr_hand::msg::_command::command . . . 33
sr_hand::command_< ContainerAllocator > . . . 35
sr_hand::msg::_config::config . . . 36
sr_hand::config_< ContainerAllocator > . . . 38
sr_hand::msg::_contrlr::contrlr . . . 40
sr_hand::contrlr_< ContainerAllocator > . . . 42
ros::message_traits::DataType< ::sr_hand::cartesian_data_< ContainerAllocator > > . . . 44
ros::message_traits::DataType< ::sr_hand::cartesian_position_< ContainerAllocator > > . . . 45
ros::message_traits::DataType< ::sr_hand::command_< ContainerAllocator > > . . . 46
ros::message_traits::DataType< ::sr_hand::config_< ContainerAllocator > > . . . 47
ros::message_traits::DataType< ::sr_hand::contrlr_< ContainerAllocator > > . . . 48
ros::message_traits::DataType< ::sr_hand::joint_< ContainerAllocator > > . . . 49
ros::message_traits::DataType< ::sr_hand::joints_data_< ContainerAllocator > > . . . 50
ros::message_traits::DataType< ::sr_hand::reverseKinematics_< ContainerAllocator > > . . . 51
ros::message_traits::DataType< ::sr_hand::sendupdate_< ContainerAllocator > > . . . 52
ros::message_traits::Definition< ::sr_hand::cartesian_data_< ContainerAllocator > > . . . 53
ros::message_traits::Definition< ::sr_hand::cartesian_position_< ContainerAllocator > > . . . 54
ros::message_traits::Definition< ::sr_hand::command_< ContainerAllocator > > . . . 55
ros::message_traits::Definition< ::sr_hand::config_< ContainerAllocator > > . . . 56
ros::message_traits::Definition< ::sr_hand::contrlr_< ContainerAllocator > > . . . 57
ros::message_traits::Definition< ::sr_hand::joint_< ContainerAllocator > > . . . 58
ros::message_traits::Definition< ::sr_hand::joints_data_< ContainerAllocator > > . . . 59
ros::message_traits::Definition< ::sr_hand::reverseKinematics_< ContainerAllocator > > . . . 60
ros::message_traits::Definition< ::sr_hand::sendupdate_< ContainerAllocator > > . . . 61
shadowrobot::DiagnosticData . . . 62
Grasp::Grasp . . . 63
grasps_interpoler::GraspInterpoler . . . 64
grasps_parser::GraspParser . . . 65
shadowhand_ros::Joint . . . 66

sr_hand::msg::_joint::joint 67
sr_hand::joint_< ContainerAllocator > 69
shadowrobot::JointControllerData 71
shadowrobot::JointData 72
sr_hand::msg::_joints_data::joints_data 73
sr_hand::joints_data_< ContainerAllocator > 75
ros::message_traits::MD5Sum< ::sr_hand::cartesian_data_< ContainerAllocator > > 76
ros::message_traits::MD5Sum< ::sr_hand::cartesian_position_< ContainerAllocator > > 77
ros::message_traits::MD5Sum< ::sr_hand::command_< ContainerAllocator > > 78
ros::message_traits::MD5Sum< ::sr_hand::config_< ContainerAllocator > > 79
ros::message_traits::MD5Sum< ::sr_hand::contrlr_< ContainerAllocator > > 80
ros::message_traits::MD5Sum< ::sr_hand::joint_< ContainerAllocator > > 81
ros::message_traits::MD5Sum< ::sr_hand::joints_data_< ContainerAllocator > > 82
ros::message_traits::MD5Sum< ::sr_hand::reverseKinematics_< ContainerAllocator > > 83
ros::message_traits::MD5Sum< ::sr_hand::sendupdate_< ContainerAllocator > > 84
shadowrobot::Parameters 85
ros::message_operations::Printer< ::sr_hand::cartesian_data_< ContainerAllocator > > 86
ros::message_operations::Printer< ::sr_hand::cartesian_position_< ContainerAllocator > > 87
ros::message_operations::Printer< ::sr_hand::command_< ContainerAllocator > > 88
ros::message_operations::Printer< ::sr_hand::config_< ContainerAllocator > > 89
ros::message_operations::Printer< ::sr_hand::contrlr_< ContainerAllocator > > 90
ros::message_operations::Printer< ::sr_hand::joint_< ContainerAllocator > > 91
ros::message_operations::Printer< ::sr_hand::joints_data_< ContainerAllocator > > 92
ros::message_operations::Printer< ::sr_hand::reverseKinematics_< ContainerAllocator > > 93
ros::message_operations::Printer< ::sr_hand::sendupdate_< ContainerAllocator > > 94
sr_hand::msg::_reverseKinematics::reverseKinematics 99
sr_hand::reverseKinematics_< ContainerAllocator > 101
sr_hand::msg::_sendupdate::sendupdate 102
sr_hand::sendupdate_< ContainerAllocator > 104
ros::serialization::Serializer< ::sr_hand::cartesian_data_< ContainerAllocator > > 105
ros::serialization::Serializer< ::sr_hand::cartesian_position_< ContainerAllocator > > 106
ros::serialization::Serializer< ::sr_hand::command_< ContainerAllocator > > 107
ros::serialization::Serializer< ::sr_hand::config_< ContainerAllocator > > 108
ros::serialization::Serializer< ::sr_hand::contrlr_< ContainerAllocator > > 109
ros::serialization::Serializer< ::sr_hand::joint_< ContainerAllocator > > 110
ros::serialization::Serializer< ::sr_hand::joints_data_< ContainerAllocator > > 111
ros::serialization::Serializer< ::sr_hand::reverseKinematics_< ContainerAllocator > > 112
ros::serialization::Serializer< ::sr_hand::sendupdate_< ContainerAllocator > > 113
shadowhand_ros::ShadowHand_ROS 114
shadowrobot::SRArticulatedRobot 118
  shadowrobot::RealShadowhand 95
  shadowrobot::VirtualArm 127
  shadowrobot::VirtualShadowhand 131
shadowrobot::SRDiagnosticer 122
test::SRLibrary 123
shadowrobot::SRPublisher 124
shadowrobot::SRSubscriber 125
shadowrobot::SRTactileSensorPublisher 126

Chapter 5

Data Structure Index

5.1 Data Structures

Here are the data structures with brief descriptions:

sr_hand::msg::_cartesian_data::cartesian_data . . . 27
sr_hand::cartesian_data_< ContainerAllocator > . . . 29
sr_hand::msg::_cartesian_position::cartesian_position . . . 30
sr_hand::cartesian_position_< ContainerAllocator > . . . 32
sr_hand::msg::_command::command . . . 33
sr_hand::command_< ContainerAllocator > . . . 35
sr_hand::msg::_config::config . . . 36
sr_hand::config_< ContainerAllocator > . . . 38
sr_hand::msg::_contrlr::contrlr . . . 40
sr_hand::contrlr_< ContainerAllocator > . . . 42
ros::message_traits::DataType< ::sr_hand::cartesian_data_< ContainerAllocator > > . . . 44
ros::message_traits::DataType< ::sr_hand::cartesian_position_< ContainerAllocator > > . . . 45
ros::message_traits::DataType< ::sr_hand::command_< ContainerAllocator > > . . . 46
ros::message_traits::DataType< ::sr_hand::config_< ContainerAllocator > > . . . 47
ros::message_traits::DataType< ::sr_hand::contrlr_< ContainerAllocator > > . . . 48
ros::message_traits::DataType< ::sr_hand::joint_< ContainerAllocator > > . . . 49
ros::message_traits::DataType< ::sr_hand::joints_data_< ContainerAllocator > > . . . 50
ros::message_traits::DataType< ::sr_hand::reverseKinematics_< ContainerAllocator > > . . . 51
ros::message_traits::DataType< ::sr_hand::sendupdate_< ContainerAllocator > > . . . 52
ros::message_traits::Definition< ::sr_hand::cartesian_data_< ContainerAllocator > > . . . 53
ros::message_traits::Definition< ::sr_hand::cartesian_position_< ContainerAllocator > > . . . 54
ros::message_traits::Definition< ::sr_hand::command_< ContainerAllocator > > . . . 55
ros::message_traits::Definition< ::sr_hand::config_< ContainerAllocator > > . . . 56
ros::message_traits::Definition< ::sr_hand::contrlr_< ContainerAllocator > > . . . 57
ros::message_traits::Definition< ::sr_hand::joint_< ContainerAllocator > > . . . 58
ros::message_traits::Definition< ::sr_hand::joints_data_< ContainerAllocator > > . . . 59
ros::message_traits::Definition< ::sr_hand::reverseKinematics_< ContainerAllocator > > . . . 60
ros::message_traits::Definition< ::sr_hand::sendupdate_< ContainerAllocator > > . . . 61
shadowrobot::DiagnosticData . . . 62
Grasp::Grasp . . . 63
grasps_interpoler::GraspInterpoler . . . 64
grasps_parser::GraspParser . . . 65
shadowhand_ros::Joint . . . 66

sr_hand::msg::_joint::joint . . . 67
sr_hand::joint_< ContainerAllocator > . . . 69
shadowrobot::JointControllerData . . . 71
shadowrobot::JointData . . . 72
sr_hand::msg::_joints_data::joints_data . . . 73
sr_hand::joints_data_< ContainerAllocator > . . . 75
ros::message_traits::MD5Sum< ::sr_hand::cartesian_data_< ContainerAllocator > > . . . 76
ros::message_traits::MD5Sum< ::sr_hand::cartesian_position_< ContainerAllocator > > . . . 77
ros::message_traits::MD5Sum< ::sr_hand::command_< ContainerAllocator > > . . . 78
ros::message_traits::MD5Sum< ::sr_hand::config_< ContainerAllocator > > . . . 79
ros::message_traits::MD5Sum< ::sr_hand::contrlr_< ContainerAllocator > > . . . 80
ros::message_traits::MD5Sum< ::sr_hand::joint_< ContainerAllocator > > . . . 81
ros::message_traits::MD5Sum< ::sr_hand::joints_data_< ContainerAllocator > > . . . 82
ros::message_traits::MD5Sum< ::sr_hand::reverseKinematics_< ContainerAllocator > > . . . 83
ros::message_traits::MD5Sum< ::sr_hand::sendupdate_< ContainerAllocator > > . . . 84
shadowrobot::Parameters . . . 85
ros::message_operations::Printer< ::sr_hand::cartesian_data_< ContainerAllocator > > . . . 86
ros::message_operations::Printer< ::sr_hand::cartesian_position_< ContainerAllocator > > . . . 87
ros::message_operations::Printer< ::sr_hand::command_< ContainerAllocator > > . . . 88
ros::message_operations::Printer< ::sr_hand::config_< ContainerAllocator > > . . . 89
ros::message_operations::Printer< ::sr_hand::contrlr_< ContainerAllocator > > . . . 90
ros::message_operations::Printer< ::sr_hand::joint_< ContainerAllocator > > . . . 91
ros::message_operations::Printer< ::sr_hand::joints_data_< ContainerAllocator > > . . . 92
ros::message_operations::Printer< ::sr_hand::reverseKinematics_< ContainerAllocator > > . . . 93
ros::message_operations::Printer< ::sr_hand::sendupdate_< ContainerAllocator > > . . . 94
shadowrobot::RealShadowhand . . . 95
sr_hand::msg::_reverseKinematics::reverseKinematics . . . 99
sr_hand::reverseKinematics_< ContainerAllocator > . . . 101
sr_hand::msg::_sendupdate::sendupdate . . . 102
sr_hand::sendupdate_< ContainerAllocator > . . . 104
ros::serialization::Serializer< ::sr_hand::cartesian_data_< ContainerAllocator > > . . . 105
ros::serialization::Serializer< ::sr_hand::cartesian_position_< ContainerAllocator > > . . . 106
ros::serialization::Serializer< ::sr_hand::command_< ContainerAllocator > > . . . 107
ros::serialization::Serializer< ::sr_hand::config_< ContainerAllocator > > . . . 108
ros::serialization::Serializer< ::sr_hand::contrlr_< ContainerAllocator > > . . . 109
ros::serialization::Serializer< ::sr_hand::joint_< ContainerAllocator > > . . . 110
ros::serialization::Serializer< ::sr_hand::joints_data_< ContainerAllocator > > . . . 111
ros::serialization::Serializer< ::sr_hand::reverseKinematics_< ContainerAllocator > > . . . 112
ros::serialization::Serializer< ::sr_hand::sendupdate_< ContainerAllocator > > . . . 113
shadowhand_ros::ShadowHand_ROS . . . 114
shadowrobot::SRArticulatedRobot (This is a parent class for the different types of Shadowhand we can have. It makes it possible to swap from a virtual to a real hand while using the same utilities to interact with the hand. One (or more) ROS subscriber and publisher can then share the same instance of a Shadowhand object to update the information contained in this object) . . . 118
shadowrobot::SRDiagnosticer . . . 122
test::SRLibrary . . . 123
shadowrobot::SRPublisher . . . 124
shadowrobot::SRSubscriber . . . 125
shadowrobot::SRTactileSensorPublisher . . . 126
shadowrobot::VirtualArm . . . 127
shadowrobot::VirtualShadowhand (The Virtual Shadowhand can be used as a simulator. As both the real hand and the virtual hand are children from the shadowhand class, using a virtual or a real hand doesn't change anything in the way you call them in your programs) 131

Chapter 6

File Index

6.1 File List

Here is a list of all documented files with brief descriptions:

cartesian_data.h . . . . . **??**
cartesian_position.h . . . . . **??**
command.h . . . . . **??**
config.h . . . . . **??**
contrlr.h . . . . . **??**
joint.h . . . . . **??**
joints_data.h . . . . . **??**
link_joints.cpp (This is an example to show how to get data from the hand, read the position for a specific joint and send this as the target to another joint) 137
real_shadowhand.cpp 139
real_shadowhand.h (The real shadowhand is the ROS interface to Shadow Robot robotic hand) . 140
real_shadowhand_node.cpp (Contains the main for the real Shadow Dextrous Hand. We start the publishers / subscribers in this node. They all share the same RealShadowhand object, this way the subscriber can update the hand properties, while the publishers publish up to date data. The diagnostics and the other publisher are started in two different threads, to allow them to be published at different frequencies) 141
reverseKinematics.h . . . . . **??**
sendupdate.h . . . . . **??**
sr_articulated_robot.h . . . . . **??**
sr_diagnosticer.cpp (The goal of this ROS publisher is to publish relevant data concerning the hand at a regular time interval. Those data are (not exhaustive): positions, targets, temperatures, currents, forces, error flags, ..) 143
sr_diagnosticer.h (The Diagnosticer is a ROS publisher which publishes diagnostic data regarding the Dextrous Hand or the Shadow Robot Arm, on the \/diagnostics topic. The diagnostics can be viewed using the robot_monitor package) 144
sr_publisher.cpp (The goal of this ROS publisher is to publish relevant data concerning the hand at a regular time interval. Those data are (not exhaustive): positions, targets, temperatures, currents, forces, error flags, ..) 145
sr_publisher.h (This class reads and publishes data concerning the shadowhand / shadowarm. To publish those data, just call the publish() function) 146
sr_subscriber.cpp (The role of this ROS subscriber is to receive commands messages (sent by a publisher) and pass them to the hand after translating them to the correct format) . . . 147

sr_subscriber.h (This ROS subscriber is used to issue commands to the hand / arm, from sending a set of targets, to changing the controller parameters) 148
sr_tactile_sensor_pub.cpp (The goal of this ROS publisher is to publish relevant data concerning the hand at a regular time interval. Those data are (not exhaustive): positions, targets, temperatures, currents, forces, error flags, ..) 149
sr_tactile_sensor_pub.h (This class reads and publishes data concerning the shadowhand. To publish those data, just call the publish() function) 150
sr_tactile_sensor_pub_node.cpp 151
virtual_arm.cpp (The virtual arm can be used as a simulator. It modelizes the Shadow Robot muscle arm) 152
virtual_arm.h (The virtual arm can be used as a simulator. It modelizes the Shadow Robot muscle arm) 153
virtual_arm_node.cpp (Contains the main for the virtual arm. We start the publishers / subscribers in this node. They all share the same VirtualArm object, this way the subscriber can update the arm properties, while the publishers publish up to date data. The diagnostics and the other publisher are started in two different threads, to allow them to be published at different frequencies) 154
virtual_shadowhand.cpp 156
virtual_shadowhand.h (The Virtual Shadowhand can be used as a simulator. As both the real hand and the virtual hand are children from the shadowhand class, using a virtual or a real hand doesn't change anything in the way you call them in your programs) 157
virtual_shadowhand_node.cpp (Contains the main for the virtual hand. We start the publishers / subscribers in this node. They all share the same VirtualHand object, this way the subscriber can update the hand properties, while the publishers publish up to date data. The diagnostics and the other publisher are started in two different threads, to allow them to be published at different frequencies) 158

Chapter 7

Namespace Documentation

7.1 sr_hand::msg::_cartesian_data Namespace Reference

Data Structures

- class cartesian_data

Variables

- **_struct_I** = roslib.message.struct_I
- tuple **_struct_b** = struct.Struct("<b")
- tuple **_struct_6f** = struct.Struct("<6f")

7.1.1 Detailed Description

```
autogenerated by genmsg_py from cartesian_data.msg. Do not edit.
```

7.2 sr_hand::msg::_cartesian_position Namespace Reference

Data Structures

- class cartesian_position

Variables

- **_struct_I** = roslib.message.struct_I
- tuple **_struct_6f** = struct.Struct("<6f")

7.2.1 Detailed Description

autogenerated by genmsg_py from cartesian_position.msg. Do not edit.

7.3 sr_hand::msg::_command Namespace Reference

Data Structures

- class command

Variables

- **_struct_I** = roslib.message.struct_I
- tuple **_struct_B** = struct.Struct("<B")
- tuple **_struct_5f** = struct.Struct("<5f")
- tuple **_struct_2b** = struct.Struct("<2b")

7.3.1 Detailed Description

```
autogenerated by genmsg_py from command.msg. Do not edit.
```

7.4 sr_hand::msg::_config Namespace Reference

Data Structures

- class config

Variables

- **_struct_I** = roslib.message.struct_I
- tuple **_struct_B** = struct.Struct("<B")

7.4.1 Detailed Description

autogenerated by genmsg_py from config.msg. Do not edit.

7.5 sr_hand::msg::_contrlr Namespace Reference

Data Structures

- class contrlr

Variables

- **_struct_I** = roslib.message.struct_I
- tuple **_struct_B** = struct.Struct("<B")

7.5.1 Detailed Description

```
autogenerated by genmsg_py from contrlr.msg. Do not edit.
```

7.6 sr_hand::msg::_joint Namespace Reference

Data Structures

- class joint

Variables

- **_struct_I** = roslib.message.struct_I
- tuple **_struct_5f** = struct.Struct("<5f")

7.6.1 Detailed Description

```
autogenerated by genmsg_py from joint.msg. Do not edit.
```

7.7 sr_hand::msg::_joints_data Namespace Reference

Data Structures

- class joints_data

Variables

- **_struct_I** = roslib.message.struct_I
- tuple **_struct_5f** = struct.Struct("<5f")
- tuple **_struct_b** = struct.Struct("<b")

7.7.1 Detailed Description

```
autogenerated by genmsg_py from joints_data.msg. Do not edit.
```

7.8 sr_hand::msg::_reverseKinematics Namespace Reference

Data Structures

- class reverseKinematics

Variables

- **_struct_I** = roslib.message.struct_I

7.8.1 Detailed Description

```
autogenerated by genmsg_py from reverseKinematics.msg. Do not edit.
```

7.9 sr_hand::msg::_sendupdate Namespace Reference

Data Structures

- class sendupdate

Variables

- **_struct_I** = roslib.message.struct_I
- tuple **_struct_5f** = struct.Struct("<5f")
- tuple **_struct_b** = struct.Struct("<b")

7.9.1 Detailed Description

```
autogenerated by genmsg_py from sendupdate.msg. Do not edit.
```

Chapter 8

Data Structure Documentation

8.1 sr_hand::msg::_cartesian_data::cartesian_data Class Reference

Public Member Functions

- def __init__
- def serialize
- def deserialize
- def serialize_numpy
- def deserialize_numpy

Data Fields

- **cartesian_positions_length**
- **cartesian_positions**

8.1.1 Member Function Documentation

8.1.1.1 def sr_hand::msg::_cartesian_data::cartesian_data::__init__ (*self*, *args*, *kwds*)

```
Constructor. Any message fields that are implicitly/explicitly
set to None will be assigned a default value. The recommend
use is keyword arguments as this is more robust to future message
changes.  You cannot mix in-order arguments and keyword arguments.

The available fields are:
   cartesian_positions_length,cartesian_positions

@param args: complete set of field values, in .msg order
@param kwds: use keyword arguments corresponding to message field names
to set specific fields.
```

8.1.1.2 def sr_hand::msg::_cartesian_data::cartesian_data::deserialize (*self*, *str*)

```
unpack serialized message in str into this message instance
@param str: byte array of serialized message
@type  str: str
```

8.1.1.3 def sr_hand::msg::_cartesian_data::cartesian_data::deserialize_numpy (*self*, *str*, *numpy*)

```
unpack serialized message in str into this message instance using numpy for array types
@param str: byte array of serialized message
@type  str: str
@param numpy: numpy python module
@type  numpy: module
```

8.1.1.4 def sr_hand::msg::_cartesian_data::cartesian_data::serialize (*self*, *buff*)

```
serialize message into buffer
@param buff: buffer
@type  buff: StringIO
```

8.1.1.5 def sr_hand::msg::_cartesian_data::cartesian_data::serialize_numpy (*self*, *buff*, *numpy*)

```
serialize message with numpy array types into buffer
@param buff: buffer
@type  buff: StringIO
@param numpy: numpy python module
@type  numpy module
```

The documentation for this class was generated from the following file:

- _cartesian_data.py

8.2 sr_hand::cartesian_data_< ContainerAllocator > Struct Template Reference

Public Types

- typedef cartesian_data_< ContainerAllocator > **Type**
- typedef int8_t **_cartesian_positions_length_type**
- typedef std::vector< ::sr_hand::cartesian_position_< ContainerAllocator >, typename ContainerAllocator::template rebind< ::sr_hand::cartesian_position_< ContainerAllocator > >::other > **_cartesian_positions_type**
- typedef boost::shared_ptr< ::sr_hand::cartesian_data_< ContainerAllocator > > **Ptr**
- typedef boost::shared_ptr< ::sr_hand::cartesian_data_< ContainerAllocator > const > **ConstPtr**

Public Member Functions

- **cartesian_data_** (const ContainerAllocator &_alloc)
- ROSCPP_DEPRECATED uint32_t **get_cartesian_positions_size** () const
- ROSCPP_DEPRECATED void **set_cartesian_positions_size** (uint32_t size)
- ROSCPP_DEPRECATED void **get_cartesian_positions_vec** (std::vector< ::sr_hand::cartesian_position_< ContainerAllocator >, typename ContainerAllocator::template rebind< ::sr_hand::cartesian_position_< ContainerAllocator > >::other > &vec) const
- ROSCPP_DEPRECATED void **set_cartesian_positions_vec** (const std::vector< ::sr_hand::cartesian_position_< ContainerAllocator >, typename ContainerAllocator::template rebind< ::sr_hand::cartesian_position_< ContainerAllocator > >::other > &vec)
- ROSCPP_DEPRECATED const std::string **__getDataType** () const
- ROSCPP_DEPRECATED const std::string **__getMD5Sum** () const
- ROSCPP_DEPRECATED const std::string **__getMessageDefinition** () const
- virtual ROSCPP_DEPRECATED uint8_t * **serialize** (uint8_t *write_ptr, uint32_t seq) const
- virtual ROSCPP_DEPRECATED uint8_t * **deserialize** (uint8_t *read_ptr)
- virtual ROSCPP_DEPRECATED uint32_t **serializationLength** () const

Static Public Member Functions

- static ROSCPP_DEPRECATED const std::string **__s_getDataType** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMD5Sum** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMessageDefinition** ()

Data Fields

- int8_t **cartesian_positions_length**
- std::vector< ::sr_hand::cartesian_position_< ContainerAllocator >, typename ContainerAllocator::template rebind< ::sr_hand::cartesian_position_< ContainerAllocator > >::other > **cartesian_positions**

template<class ContainerAllocator> struct sr_hand::cartesian_data_< ContainerAllocator >

The documentation for this struct was generated from the following file:

- cartesian_data.h

8.3 sr_hand::msg::_cartesian_position::cartesian_position Class Reference

Public Member Functions

- def __init__
- def serialize
- def deserialize
- def serialize_numpy
- def deserialize_numpy

Data Fields

- **tip_name**
- **tip_pos_x**
- **tip_pos_y**
- **tip_pos_z**
- **tip_orientation_rho**
- **tip_orientation_theta**
- **tip_orientation_sigma**

8.3.1 Member Function Documentation

8.3.1.1 def sr_hand::msg::_cartesian_position::cartesian_position::__init__ (*self*, *args*, *kwds*)

```
Constructor. Any message fields that are implicitly/explicitly
set to None will be assigned a default value. The recommend
use is keyword arguments as this is more robust to future message
changes.  You cannot mix in-order arguments and keyword arguments.

The available fields are:
   tip_name,tip_pos_x,tip_pos_y,tip_pos_z,tip_orientation_rho,tip_orientation_theta,tip_orientation_sigma

@param args: complete set of field values, in .msg order
@param kwds: use keyword arguments corresponding to message field names
to set specific fields.
```

8.3.1.2 def sr_hand::msg::_cartesian_position::cartesian_position::deserialize (*self*, *str*)

```
unpack serialized message in str into this message instance
@param str: byte array of serialized message
@type  str: str
```

8.3.1.3 def sr_hand::msg::_cartesian_position::cartesian_position::deserialize_numpy (*self*, *str*, *numpy*)

```
unpack serialized message in str into this message instance using numpy for array types
@param str: byte array of serialized message
@type  str: str
@param numpy: numpy python module
@type  numpy: module
```

8.3.1.4 def sr_hand::msg::_cartesian_position::cartesian_position::serialize (*self*, *buff*)

```
serialize message into buffer
@param buff: buffer
@type  buff: StringIO
```

8.3.1.5 def sr_hand::msg::_cartesian_position::cartesian_position::serialize_numpy (*self*, *buff*, *numpy*)

```
serialize message with numpy array types into buffer
@param buff: buffer
@type  buff: StringIO
@param numpy: numpy python module
@type  numpy module
```

The documentation for this class was generated from the following file:

- _cartesian_position.py

8.4 sr_hand::cartesian_position_< ContainerAllocator > Struct Template Reference

Public Types

- typedef cartesian_position_< ContainerAllocator > **Type**
- typedef std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > **_tip_name_type**
- typedef float **_tip_pos_x_type**
- typedef float **_tip_pos_y_type**
- typedef float **_tip_pos_z_type**
- typedef float **_tip_orientation_rho_type**
- typedef float **_tip_orientation_theta_type**
- typedef float **_tip_orientation_sigma_type**
- typedef boost::shared_ptr< ::sr_hand::cartesian_position_< ContainerAllocator > > **Ptr**
- typedef boost::shared_ptr< ::sr_hand::cartesian_position_< ContainerAllocator > const > **ConstPtr**

Public Member Functions

- **cartesian_position_** (const ContainerAllocator &_alloc)
- ROSCPP_DEPRECATED const std::string **__getDataType** () const
- ROSCPP_DEPRECATED const std::string **__getMD5Sum** () const
- ROSCPP_DEPRECATED const std::string **__getMessageDefinition** () const
- virtual ROSCPP_DEPRECATED uint8_t * **serialize** (uint8_t *write_ptr, uint32_t seq) const
- virtual ROSCPP_DEPRECATED uint8_t * **deserialize** (uint8_t *read_ptr)
- virtual ROSCPP_DEPRECATED uint32_t **serializationLength** () const

Static Public Member Functions

- static ROSCPP_DEPRECATED const std::string **__s_getDataType** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMD5Sum** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMessageDefinition** ()

Data Fields

- std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > **tip_name**
- float **tip_pos_x**
- float **tip_pos_y**
- float **tip_pos_z**
- float **tip_orientation_rho**
- float **tip_orientation_theta**
- float **tip_orientation_sigma**

template<class ContainerAllocator> struct sr_hand::cartesian_position_< ContainerAllocator >

The documentation for this struct was generated from the following file:

- cartesian_position.h

8.5 sr_hand::msg::_command::command Class Reference

Public Member Functions

- def __init__
- def serialize
- def deserialize
- def serialize_numpy
- def deserialize_numpy

Data Fields

- **command_type**
- **sendupdate_command**
- **contrlr_command**

8.5.1 Member Function Documentation

8.5.1.1 def sr_hand::msg::_command::command::__init__ (*self*, *args*, *kwds*)

```
Constructor. Any message fields that are implicitly/explicitly
set to None will be assigned a default value. The recommend
use is keyword arguments as this is more robust to future message
changes.  You cannot mix in-order arguments and keyword arguments.

The available fields are:
   command_type,sendupdate_command,contrlr_command

@param args: complete set of field values, in .msg order
@param kwds: use keyword arguments corresponding to message field names
to set specific fields.
```

8.5.1.2 def sr_hand::msg::_command::command::deserialize (*self*, *str*)

```
unpack serialized message in str into this message instance
@param str: byte array of serialized message
@type  str: str
```

8.5.1.3 def sr_hand::msg::_command::command::deserialize_numpy (*self*, *str*, *numpy*)

```
unpack serialized message in str into this message instance using numpy for array types
@param str: byte array of serialized message
@type  str: str
@param numpy: numpy python module
@type  numpy: module
```

8.5.1.4 def sr_hand::msg::_command::command::serialize (*self*, *buff*)

```
serialize message into buffer
@param buff: buffer
@type  buff: StringIO
```

8.5.1.5 def sr_hand::msg::_command::command::serialize_numpy (*self*, *buff*, *numpy*)

```
serialize message with numpy array types into buffer
@param buff: buffer
@type  buff: StringIO
@param numpy: numpy python module
@type  numpy module
```

The documentation for this class was generated from the following file:

- _command.py

8.6 sr_hand::command_< ContainerAllocator > Struct Template Reference

Public Types

- typedef command_< ContainerAllocator > **Type**
- typedef int8_t **_command_type_type**
- typedef ::sr_hand::sendupdate_< ContainerAllocator > **_sendupdate_command_type**
- typedef ::sr_hand::contrlr_< ContainerAllocator > **_contrlr_command_type**
- typedef boost::shared_ptr< ::sr_hand::command_< ContainerAllocator > > **Ptr**
- typedef boost::shared_ptr< ::sr_hand::command_< ContainerAllocator > const > **ConstPtr**

Public Member Functions

- **command_** (const ContainerAllocator &_alloc)
- ROSCPP_DEPRECATED const std::string **__getDataType** () const
- ROSCPP_DEPRECATED const std::string **__getMD5Sum** () const
- ROSCPP_DEPRECATED const std::string **__getMessageDefinition** () const
- virtual ROSCPP_DEPRECATED uint8_t * **serialize** (uint8_t *write_ptr, uint32_t seq) const
- virtual ROSCPP_DEPRECATED uint8_t * **deserialize** (uint8_t *read_ptr)
- virtual ROSCPP_DEPRECATED uint32_t **serializationLength** () const

Static Public Member Functions

- static ROSCPP_DEPRECATED const std::string **__s_getDataType** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMD5Sum** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMessageDefinition** ()

Data Fields

- int8_t **command_type**
- ::sr_hand::sendupdate_< ContainerAllocator > **sendupdate_command**
- ::sr_hand::contrlr_< ContainerAllocator > **contrlr_command**

template<class ContainerAllocator> struct sr_hand::command_< ContainerAllocator >

The documentation for this struct was generated from the following file:

- command.h

8.7 sr_hand::msg::_config::config Class Reference

Public Member Functions

- def __init__
- def serialize
- def deserialize
- def serialize_numpy
- def deserialize_numpy

Data Fields

- **node_name**
- **list_of_parameters**
- **length_of_list**

8.7.1 Member Function Documentation

8.7.1.1 def sr_hand::msg::_config::config::__init__ (*self*, *args*, *kwds*)

```
Constructor. Any message fields that are implicitly/explicitly
set to None will be assigned a default value. The recommend
use is keyword arguments as this is more robust to future message
changes.  You cannot mix in-order arguments and keyword arguments.

The available fields are:
   node_name,list_of_parameters,length_of_list

@param args: complete set of field values, in .msg order
@param kwds: use keyword arguments corresponding to message field names
to set specific fields.
```

8.7.1.2 def sr_hand::msg::_config::config::deserialize (*self*, *str*)

```
unpack serialized message in str into this message instance
@param str: byte array of serialized message
@type  str: str
```

8.7.1.3 def sr_hand::msg::_config::config::deserialize_numpy (*self*, *str*, *numpy*)

```
unpack serialized message in str into this message instance using numpy for array types
@param str: byte array of serialized message
@type  str: str
@param numpy: numpy python module
@type  numpy: module
```

8.7.1.4 def sr_hand::msg::_config::config::serialize (*self*, *buff*)

```
serialize message into buffer
@param buff: buffer
@type  buff: StringIO
```

8.7.1.5 def sr_hand::msg::_config::config::serialize_numpy (*self*, *buff*, *numpy*)

```
serialize message with numpy array types into buffer
@param buff: buffer
@type  buff: StringIO
@param numpy: numpy python module
@type  numpy module
```

The documentation for this class was generated from the following file:

- _config.py

8.8 sr_hand::config_< ContainerAllocator > Struct Template Reference

Public Types

- typedef config_< ContainerAllocator > **Type**
- typedef std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > **_node_name_type**
- typedef std::vector< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other >, typename ContainerAllocator::template rebind< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > >::other > **_list_of_parameters_type**
- typedef uint8_t **_length_of_list_type**
- typedef boost::shared_ptr< ::sr_hand::config_< ContainerAllocator > > **Ptr**
- typedef boost::shared_ptr< ::sr_hand::config_< ContainerAllocator > const > **ConstPtr**

Public Member Functions

- **config_** (const ContainerAllocator &_alloc)
- ROSCPP_DEPRECATED uint32_t **get_list_of_parameters_size** () const
- ROSCPP_DEPRECATED void **set_list_of_parameters_size** (uint32_t size)
- ROSCPP_DEPRECATED void **get_list_of_parameters_vec** (std::vector< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other >, typename ContainerAllocator::template rebind< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > >::other > &vec) const
- ROSCPP_DEPRECATED void **set_list_of_parameters_vec** (const std::vector< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other >, typename ContainerAllocator::template rebind< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > >::other > &vec)
- ROSCPP_DEPRECATED const std::string **__getDataType** () const
- ROSCPP_DEPRECATED const std::string **__getMD5Sum** () const
- ROSCPP_DEPRECATED const std::string **__getMessageDefinition** () const
- virtual ROSCPP_DEPRECATED uint8_t * **serialize** (uint8_t *write_ptr, uint32_t seq) const
- virtual ROSCPP_DEPRECATED uint8_t * **deserialize** (uint8_t *read_ptr)
- virtual ROSCPP_DEPRECATED uint32_t **serializationLength** () const

Static Public Member Functions

- static ROSCPP_DEPRECATED const std::string **__s_getDataType** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMD5Sum** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMessageDefinition** ()

Data Fields

- std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > **node_name**
- std::vector< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other >, typename ContainerAllocator::template rebind< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > >::other > **list_of_parameters**
- uint8_t **length_of_list**

template<class ContainerAllocator> struct sr_hand::config_< ContainerAllocator >

The documentation for this struct was generated from the following file:

- config.h

8.9 sr_hand::msg::_contrlr::contrlr Class Reference

Public Member Functions

- def __init__
- def serialize
- def deserialize
- def serialize_numpy
- def deserialize_numpy

Data Fields

- **contrlr_name**
- **list_of_parameters**
- **length_of_list**

8.9.1 Member Function Documentation

8.9.1.1 def sr_hand::msg::_contrlr::contrlr::__init__ (*self*, *args*, *kwds*)

```
Constructor. Any message fields that are implicitly/explicitly
set to None will be assigned a default value. The recommend
use is keyword arguments as this is more robust to future message
changes.  You cannot mix in-order arguments and keyword arguments.

The available fields are:
   contrlr_name,list_of_parameters,length_of_list

@param args: complete set of field values, in .msg order
@param kwds: use keyword arguments corresponding to message field names
to set specific fields.
```

8.9.1.2 def sr_hand::msg::_contrlr::contrlr::deserialize (*self*, *str*)

```
unpack serialized message in str into this message instance
@param str: byte array of serialized message
@type  str: str
```

8.9.1.3 def sr_hand::msg::_contrlr::contrlr::deserialize_numpy (*self*, *str*, *numpy*)

```
unpack serialized message in str into this message instance using numpy for array types
@param str: byte array of serialized message
@type  str: str
@param numpy: numpy python module
@type  numpy: module
```

8.9.1.4 def sr_hand::msg::_contrlr::contrlr::serialize (*self*, *buff*)

```
serialize message into buffer
@param buff: buffer
@type  buff: StringIO
```

8.9.1.5 def sr_hand::msg::_contrlr::contrlr::serialize_numpy (*self*, *buff*, *numpy*)

```
serialize message with numpy array types into buffer
@param buff: buffer
@type  buff: StringIO
@param numpy: numpy python module
@type  numpy module
```

The documentation for this class was generated from the following file:

- _contrlr.py

8.10 sr_hand::contrlr_< ContainerAllocator > Struct Template Reference

Public Types

- typedef contrlr_< ContainerAllocator > **Type**
- typedef std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > **_contrlr_name_type**
- typedef std::vector< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other >, typename ContainerAllocator::template rebind< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > >::other > **_list_of_parameters_type**
- typedef uint8_t **_length_of_list_type**
- typedef boost::shared_ptr< ::sr_hand::contrlr_< ContainerAllocator > > **Ptr**
- typedef boost::shared_ptr< ::sr_hand::contrlr_< ContainerAllocator > const > **ConstPtr**

Public Member Functions

- **contrlr_** (const ContainerAllocator &_alloc)
- ROSCPP_DEPRECATED uint32_t **get_list_of_parameters_size** () const
- ROSCPP_DEPRECATED void **set_list_of_parameters_size** (uint32_t size)
- ROSCPP_DEPRECATED void **get_list_of_parameters_vec** (std::vector< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other >, typename ContainerAllocator::template rebind< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > >::other > &vec) const
- ROSCPP_DEPRECATED void **set_list_of_parameters_vec** (const std::vector< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other >, typename ContainerAllocator::template rebind< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > >::other > &vec)
- ROSCPP_DEPRECATED const std::string **__getDataType** () const
- ROSCPP_DEPRECATED const std::string **__getMD5Sum** () const
- ROSCPP_DEPRECATED const std::string **__getMessageDefinition** () const
- virtual ROSCPP_DEPRECATED uint8_t * **serialize** (uint8_t *write_ptr, uint32_t seq) const
- virtual ROSCPP_DEPRECATED uint8_t * **deserialize** (uint8_t *read_ptr)
- virtual ROSCPP_DEPRECATED uint32_t **serializationLength** () const

Static Public Member Functions

- static ROSCPP_DEPRECATED const std::string **__s_getDataType** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMD5Sum** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMessageDefinition** ()

Data Fields

- std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > **contrlr_name**
- std::vector< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other >, typename ContainerAllocator::template rebind< std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > >::other > **list_of_parameters**
- uint8_t **length_of_list**

template<class ContainerAllocator> struct sr_hand::contrlr_< ContainerAllocator >

The documentation for this struct was generated from the following file:

- contrlr.h

8.11 ros::message_traits::DataType< ::sr_hand::cartesian_data_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::cartesian_data_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::DataType< ::sr_hand::cartesian_data_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- cartesian_data.h

8.12 ros::message_traits::DataType< ::sr_hand::cartesian_position_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::cartesian_position_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::DataType< ::sr_hand::cartesian_position_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- cartesian_position.h

8.13 ros::message_traits::DataType< ::sr_hand::command_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::command_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::DataType< ::sr_hand::command_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- command.h

8.14 ros::message_traits::DataType< ::sr_hand::config_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::config_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::DataType< ::sr_hand::config_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- config.h

8.15 ros::message_traits::DataType< ::sr_hand::contrlr_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::contrlr_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::DataType< ::sr_hand::contrlr_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- contrlr.h

8.16 ros::message_traits::DataType< ::sr_hand::joint_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::joint_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::DataType< ::sr_hand::joint_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- joint.h

8.17 ros::message_traits::DataType< ::sr_hand::joints_data_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::joints_data_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::DataType< ::sr_hand::joints_data_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- joints_data.h

8.18 ros::message_traits::DataType< ::sr_hand::reverseKinematics_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::reverseKinematics_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::DataType< ::sr_hand::reverseKinematics_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- reverseKinematics.h

8.19 ros::message_traits::DataType< ::sr_hand::sendupdate_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::sendupdate_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::DataType< ::sr_hand::sendupdate_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- sendupdate.h

8.20 ros::message_traits::Definition< ::sr_hand::cartesian_data_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::cartesian_data_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::Definition< ::sr_hand::cartesian_data_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- cartesian_data.h

8.21 ros::message_traits::Definition< ::sr_hand::cartesian_position_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::cartesian_position_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::Definition< ::sr_hand::cartesian_position_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- cartesian_position.h

8.22 ros::message_traits::Definition< ::sr_hand::command_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::command_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::Definition< ::sr_hand::command_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- command.h

8.23 ros::message_traits::Definition< ::sr_hand::config_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::config_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::Definition< ::sr_hand::config_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- config.h

8.24 ros::message_traits::Definition< ::sr_hand::contrlr_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::contrlr_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::Definition< ::sr_hand::contrlr_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- contrlr.h

8.25 ros::message_traits::Definition< ::sr_hand::joint_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::joint_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::Definition< ::sr_hand::joint_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- joint.h

8.26 ros::message_traits::Definition< ::sr_hand::joints_data_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::joints_data_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::Definition< ::sr_hand::joints_data_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- joints_data.h

8.27 ros::message_traits::Definition< ::sr_hand::reverseKinematics_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::reverseKinematics_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::Definition< ::sr_hand::reverseKinematics_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- reverseKinematics.h

8.28 ros::message_traits::Definition< ::sr_hand::sendupdate_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::sendupdate_< ContainerAllocator > &)

template<class ContainerAllocator> struct ros::message_traits::Definition< ::sr_hand::sendupdate_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- sendupdate.h

8.29 shadowrobot::DiagnosticData Struct Reference

`#include <sr_articulated_robot.h>`

Data Fields

- std::string joint_name

  the name of the joint

- short level

  the level of alert: 0 = OK, 1 = WARNING, 2 = ERROR

- std::string flags

  a string containing flags: FORCE, TEMPERATURE, ... indicating the different cutouts / warning sent by the hand.

- int target_sensor_num

  the channel number of the target sensor

- double target

  the actual value of the target

- int position_sensor_num

  the channel number of the position sensor.

- double position

  the actual value of the position

- double temperature

  temperature value

- double current

  current value

- double force

  force value

8.29.1 Detailed Description

The information being published by the Diagnostic publisher

The documentation for this struct was generated from the following file:

- sr_articulated_robot.h

8.30 Grasp::Grasp Class Reference

Public Member Functions

- def **__init__**
- def **display_grasp**
- def **convert_to_xml**

Data Fields

- **grasp_name**
- **joints_and_positions**

The documentation for this class was generated from the following file:

- Grasp.py

8.31 grasps_interpoler::GraspInterpoler Class Reference

Public Member Functions

- def __init__
- def set_percentage_grasp1_to_grasp2
- def **interpolate**

Data Fields

- **percentage**
- **grasp_from**
- **grasp_to**
- **from_grasp1_to_grasp2**

8.31.1 Member Function Documentation

8.31.1.1 def grasps_interpoler::GraspInterpoler::__init__ (*self*, *grasp_from*, *grasp_to* = "current")

```
interpolate from one grasp to another

Keyword arguments
grasp_from -- goes from this grasp
grasp_to -- to this grasp (if not defined, then consider
the current position of the hand as a grasp)
```

8.31.1.2 def grasps_interpoler::GraspInterpoler::set_percentage_grasp1_to_grasp2 (*self*, *percentage*)

```
move to a given percentage between grasp1 and grasp2

Keyword arguments
percentage -- must be between 0 and 100
```

The documentation for this class was generated from the following file:

- grasps_interpoler.py

8.32 grasps_parser::GraspParser Class Reference

Public Member Functions

- def **__init__**
- def parse_tree
- def **write_grasp_to_file**
- def **refresh**

Data Fields

- **xml_tree**
- **grasps**
- **xml_path**

8.32.1 Detailed Description

```
Parses a XML file describing a grasp.
```

8.32.2 Member Function Documentation

8.32.2.1 def grasps_parser::GraspParser::parse_tree (*self*, *xml_filename* = `"grasps.xml"`)

```
parses a given tree, returns a Grasp

Keyword arguments:
xml_filename -- the filename where the grasp is defined
        if no filename is provided, then the default
        value is "grasps.xml"
```

The documentation for this class was generated from the following file:

- grasps_parser.py

8.33 shadowhand_ros::Joint Class Reference

Public Member Functions

- def **__init__**

Data Fields

- **name**
- **motor**
- **min**
- **max**

The documentation for this class was generated from the following file:

- shadowhand_ros.py

8.34 sr_hand::msg::_joint::joint Class Reference

Public Member Functions

- def __init__
- def serialize
- def deserialize
- def serialize_numpy
- def deserialize_numpy

Data Fields

- **joint_name**
- **joint_position**
- **joint_target**
- **joint_torque**
- **joint_temperature**
- **joint_current**
- **error_flag**

8.34.1 Member Function Documentation

8.34.1.1 def sr_hand::msg::_joint::joint::__init__ (*self*, *args*, *kwds*)

```
Constructor. Any message fields that are implicitly/explicitly
set to None will be assigned a default value. The recommend
use is keyword arguments as this is more robust to future message
changes.  You cannot mix in-order arguments and keyword arguments.

The available fields are:
   joint_name,joint_position,joint_target,joint_torque,joint_temperature,joint_current,error_flag

@param args: complete set of field values, in .msg order
@param kwds: use keyword arguments corresponding to message field names
to set specific fields.
```

8.34.1.2 def sr_hand::msg::_joint::joint::deserialize (*self*, *str*)

```
unpack serialized message in str into this message instance
@param str: byte array of serialized message
@type  str: str
```

8.34.1.3 def sr_hand::msg::_joint::joint::deserialize_numpy (*self*, *str*, *numpy*)

```
unpack serialized message in str into this message instance using numpy for array types
@param str: byte array of serialized message
@type  str: str
@param numpy: numpy python module
@type  numpy: module
```

8.34.1.4 def sr_hand::msg::_joint::joint::serialize (*self*, *buff*)

```
serialize message into buffer
@param buff: buffer
@type  buff: StringIO
```

8.34.1.5 def sr_hand::msg::_joint::joint::serialize_numpy (*self*, *buff*, *numpy*)

```
serialize message with numpy array types into buffer
@param buff: buffer
@type  buff: StringIO
@param numpy: numpy python module
@type  numpy module
```

The documentation for this class was generated from the following file:

- _joint.py

8.35 sr_hand::joint_< ContainerAllocator > Struct Template Reference

Public Types

- typedef joint_< ContainerAllocator > **Type**
- typedef std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > **_joint_name_type**
- typedef float **_joint_position_type**
- typedef float **_joint_target_type**
- typedef float **_joint_torque_type**
- typedef float **_joint_temperature_type**
- typedef float **_joint_current_type**
- typedef std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > **_error_flag_type**
- typedef boost::shared_ptr< ::sr_hand::joint_< ContainerAllocator > > **Ptr**
- typedef boost::shared_ptr< ::sr_hand::joint_< ContainerAllocator > const > **ConstPtr**

Public Member Functions

- **joint_** (const ContainerAllocator &_alloc)
- ROSCPP_DEPRECATED const std::string **__getDataType** () const
- ROSCPP_DEPRECATED const std::string **__getMD5Sum** () const
- ROSCPP_DEPRECATED const std::string **__getMessageDefinition** () const
- virtual ROSCPP_DEPRECATED uint8_t * **serialize** (uint8_t *write_ptr, uint32_t seq) const
- virtual ROSCPP_DEPRECATED uint8_t * **deserialize** (uint8_t *read_ptr)
- virtual ROSCPP_DEPRECATED uint32_t **serializationLength** () const

Static Public Member Functions

- static ROSCPP_DEPRECATED const std::string **__s_getDataType** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMD5Sum** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMessageDefinition** ()

Data Fields

- std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > **joint_name**
- float **joint_position**
- float **joint_target**
- float **joint_torque**
- float **joint_temperature**
- float **joint_current**
- std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > **error_flag**

template<class ContainerAllocator> struct sr_hand::joint_< ContainerAllocator >

The documentation for this struct was generated from the following file:

- joint.h

8.36 shadowrobot::JointControllerData Struct Reference

`#include <sr_articulated_robot.h>`

Public Member Functions

- **JointControllerData** (JointControllerData &jcd)
- **JointControllerData** (const JointControllerData &jcd)

Data Fields

- std::vector< Parameters > **data**

8.36.1 Detailed Description

A vector containing all the Parameters for a given controller.

The documentation for this struct was generated from the following file:

- sr_articulated_robot.h

8.37 shadowrobot::JointData Struct Reference

`#include <sr_articulated_robot.h>`

Public Member Functions

- **JointData** (JointData &jd)
- **JointData** (const JointData &jd)

Data Fields

- double **position**
- double **target**
- double **temperature**
- double **current**
- double **force**
- std::string **flags**
- int **jointIndex**
- double **min**
- double **max**
- short **isJointZero**

8.37.1 Detailed Description

This struct contains all the information regarding each joints.

The documentation for this struct was generated from the following file:

- sr_articulated_robot.h

8.38 sr_hand::msg::_joints_data::joints_data Class Reference

Public Member Functions

- def __init__
- def serialize
- def deserialize
- def serialize_numpy
- def deserialize_numpy

Data Fields

- **joints_list_length**
- **joints_list**

8.38.1 Member Function Documentation

8.38.1.1 def sr_hand::msg::_joints_data::joints_data::__init__ (*self*, *args*, *kwds*)

```
Constructor. Any message fields that are implicitly/explicitly
set to None will be assigned a default value. The recommend
use is keyword arguments as this is more robust to future message
changes.  You cannot mix in-order arguments and keyword arguments.

The available fields are:
   joints_list_length,joints_list

@param args: complete set of field values, in .msg order
@param kwds: use keyword arguments corresponding to message field names
to set specific fields.
```

8.38.1.2 def sr_hand::msg::_joints_data::joints_data::deserialize (*self*, *str*)

```
unpack serialized message in str into this message instance
@param str: byte array of serialized message
@type  str: str
```

8.38.1.3 def sr_hand::msg::_joints_data::joints_data::deserialize_numpy (*self*, *str*, *numpy*)

```
unpack serialized message in str into this message instance using numpy for array types
@param str: byte array of serialized message
@type  str: str
@param numpy: numpy python module
@type  numpy: module
```

8.38.1.4 def sr_hand::msg::_joints_data::joints_data::serialize (*self*, *buff*)

```
serialize message into buffer
@param buff: buffer
@type  buff: StringIO
```

8.38.1.5 def sr_hand::msg::_joints_data::joints_data::serialize_numpy (*self*, *buff*, *numpy*)

```
serialize message with numpy array types into buffer
@param buff: buffer
@type  buff: StringIO
@param numpy: numpy python module
@type  numpy module
```

The documentation for this class was generated from the following file:

- _joints_data.py

8.39 sr_hand::joints_data_< ContainerAllocator > Struct Template Reference

Public Types

- typedef joints_data_< ContainerAllocator > **Type**
- typedef int8_t **_joints_list_length_type**
- typedef std::vector< ::sr_hand::joint_< ContainerAllocator >, typename ContainerAllocator::template rebind< ::sr_hand::joint_< ContainerAllocator > >::other > **_joints_list_type**
- typedef boost::shared_ptr< ::sr_hand::joints_data_< ContainerAllocator > > **Ptr**
- typedef boost::shared_ptr< ::sr_hand::joints_data_< ContainerAllocator > const > **ConstPtr**

Public Member Functions

- **joints_data_** (const ContainerAllocator &_alloc)
- ROSCPP_DEPRECATED uint32_t **get_joints_list_size** () const
- ROSCPP_DEPRECATED void **set_joints_list_size** (uint32_t size)
- ROSCPP_DEPRECATED void **get_joints_list_vec** (std::vector< ::sr_hand::joint_< ContainerAllocator >, typename ContainerAllocator::template rebind< ::sr_hand::joint_< ContainerAllocator > >::other > &vec) const
- ROSCPP_DEPRECATED void **set_joints_list_vec** (const std::vector< ::sr_hand::joint_< ContainerAllocator >, typename ContainerAllocator::template rebind< ::sr_hand::joint_< ContainerAllocator > >::other > &vec)
- ROSCPP_DEPRECATED const std::string **__getDataType** () const
- ROSCPP_DEPRECATED const std::string **__getMD5Sum** () const
- ROSCPP_DEPRECATED const std::string **__getMessageDefinition** () const
- virtual ROSCPP_DEPRECATED uint8_t * **serialize** (uint8_t *write_ptr, uint32_t seq) const
- virtual ROSCPP_DEPRECATED uint8_t * **deserialize** (uint8_t *read_ptr)
- virtual ROSCPP_DEPRECATED uint32_t **serializationLength** () const

Static Public Member Functions

- static ROSCPP_DEPRECATED const std::string **__s_getDataType** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMD5Sum** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMessageDefinition** ()

Data Fields

- int8_t **joints_list_length**
- std::vector< ::sr_hand::joint_< ContainerAllocator >, typename ContainerAllocator::template rebind< ::sr_hand::joint_< ContainerAllocator > >::other > **joints_list**

template<class ContainerAllocator> struct sr_hand::joints_data_< ContainerAllocator >

The documentation for this struct was generated from the following file:

- joints_data.h

8.40 ros::message_traits::MD5Sum< ::sr_hand::cartesian_data_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::cartesian_data_< ContainerAllocator > &)

Static Public Attributes

- static const uint64_t **static_value1** = 0x0efe699b2863e30aULL
- static const uint64_t **static_value2** = 0x84ffabfdca8dfb19ULL

template<class ContainerAllocator> struct ros::message_traits::MD5Sum< ::sr_hand::cartesian_data_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- cartesian_data.h

8.41 ros::message_traits::MD5Sum< ::sr_hand::cartesian_position_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::cartesian_position_< ContainerAllocator > &)

Static Public Attributes

- static const uint64_t **static_value1** = 0xcfc3f1750a36a668ULL
- static const uint64_t **static_value2** = 0xeb93288ed1128f42ULL

template<class ContainerAllocator> struct ros::message_traits::MD5Sum< ::sr_hand::cartesian_position_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- cartesian_position.h

8.42 ros::message_traits::MD5Sum< ::sr_hand::command_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::command_< ContainerAllocator > &)

Static Public Attributes

- static const uint64_t **static_value1** = 0xfb5780bd888e4fbcULL
- static const uint64_t **static_value2** = 0x42b3012bc46be2faULL

template<class ContainerAllocator> struct ros::message_traits::MD5Sum< ::sr_hand::command_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- command.h

8.43 ros::message_traits::MD5Sum< ::sr_hand::config_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::config_< ContainerAllocator > &)

Static Public Attributes

- static const uint64_t **static_value1** = 0xaad3efafb735fbacULL
- static const uint64_t **static_value2** = 0x1153457b0e7c6e24ULL

template<class ContainerAllocator> struct ros::message_traits::MD5Sum< ::sr_hand::config_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- config.h

8.44 ros::message_traits::MD5Sum< ::sr_hand::contrlr_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::contrlr_< ContainerAllocator > &)

Static Public Attributes

- static const uint64_t **static_value1** = 0x7b2aa60305c5a9abULL
- static const uint64_t **static_value2** = 0x0ff05803e4d2ab85ULL

template<class ContainerAllocator> struct ros::message_traits::MD5Sum< ::sr_hand::contrlr_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- contrlr.h

8.45 ros::message_traits::MD5Sum< ::sr_hand::joint_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::joint_< ContainerAllocator > &)

Static Public Attributes

- static const uint64_t **static_value1** = 0xd30d28ea91e78da0ULL
- static const uint64_t **static_value2** = 0x630191a4cbf1d55bULL

template<class ContainerAllocator> struct ros::message_traits::MD5Sum< ::sr_hand::joint_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- joint.h

8.46 ros::message_traits::MD5Sum< ::sr_hand::joints_data_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::joints_data_< ContainerAllocator > &)

Static Public Attributes

- static const uint64_t **static_value1** = 0xa696419a80afab19ULL
- static const uint64_t **static_value2** = 0xd9681a582e8368efULL

template<class ContainerAllocator> struct ros::message_traits::MD5Sum< ::sr_hand::joints_data_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- joints_data.h

8.47 ros::message_traits::MD5Sum< ::sr_hand::reverseKinematics_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::reverseKinematics_< ContainerAllocator > &)

Static Public Attributes

- static const uint64_t **static_value1** = 0xdb455ae8ce3e7703ULL
- static const uint64_t **static_value2** = 0x63971b1e8e35fac4ULL

template<class ContainerAllocator> struct ros::message_traits::MD5Sum< ::sr_hand::reverseKinematics_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- reverseKinematics.h

8.48 ros::message_traits::MD5Sum< ::sr_hand::sendupdate_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- static const char * **value** ()
- static const char * **value** (const ::sr_hand::sendupdate_< ContainerAllocator > &)

Static Public Attributes

- static const uint64_t **static_value1** = 0x78ecf46ea823fec4ULL
- static const uint64_t **static_value2** = 0x46125c0ac5bfee40ULL

template<class ContainerAllocator> struct ros::message_traits::MD5Sum< ::sr_hand::sendupdate_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- sendupdate.h

8.49 shadowrobot::Parameters Struct Reference

`#include <sr_articulated_robot.h>`

Public Member Functions

- **Parameters** (Parameters &param)
- **Parameters** (const Parameters &param)

Data Fields

- std::string name

  name of the parameter

- std::string value

  value of the parameter

8.49.1 Detailed Description

Description of a parameter for a controller.

The documentation for this struct was generated from the following file:

- sr_articulated_robot.h

8.50 ros::message_operations::Printer< ::sr_hand::cartesian_data_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream >
  static void **stream** (Stream &s, const std::string &indent, const ::sr_hand::cartesian_data_< ContainerAllocator > &v)

template<class ContainerAllocator> struct ros::message_operations::Printer< ::sr_hand::cartesian_data_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- cartesian_data.h

8.51 ros::message_operations::Printer< ::sr_hand::cartesian_position_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream >
  static void **stream** (Stream &s, const std::string &indent, const ::sr_hand::cartesian_position_< ContainerAllocator > &v)

template<class ContainerAllocator> struct ros::message_operations::Printer< ::sr_hand::cartesian_position_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- cartesian_position.h

8.52 ros::message_operations::Printer< ::sr_hand::command_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream >
  static void **stream** (Stream &s, const std::string &indent, const ::sr_hand::command_< Container-Allocator > &v)

template<class ContainerAllocator> struct ros::message_operations::Printer< ::sr_-hand::command_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- command.h

8.53 ros::message_operations::Printer< ::sr_hand::config_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream >
  static void **stream** (Stream &s, const std::string &indent, const ::sr_hand::config_< ContainerAllocator > &v)

template<class ContainerAllocator> struct ros::message_operations::Printer< ::sr_hand::config_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- config.h

8.54 ros::message_operations::Printer< ::sr_hand::contrlr_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream >
  static void **stream** (Stream &s, const std::string &indent, const ::sr_hand::contrlr_< ContainerAllocator > &v)

template<class ContainerAllocator> struct ros::message_operations::Printer< ::sr_hand::contrlr_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- contrlr.h

8.55 ros::message_operations::Printer< ::sr_hand::joint_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream >
  static void **stream** (Stream &s, const std::string &indent, const ::sr_hand::joint_< ContainerAllocator > &v)

template<class ContainerAllocator> struct ros::message_operations::Printer< ::sr_hand::joint_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- joint.h

8.56 ros::message_operations::Printer< ::sr_hand::joints_data_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream >
  static void **stream** (Stream &s, const std::string &indent, const ::sr_hand::joints_data_< Container-Allocator > &v)

template<class ContainerAllocator> struct ros::message_operations::Printer< ::sr_hand::joints_-data_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- joints_data.h

8.57 ros::message_operations::Printer< ::sr_hand::reverseKinematics_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream >
  static void **stream** (Stream &s, const std::string &indent, const ::sr_hand::reverseKinematics_< ContainerAllocator > &v)

template<class ContainerAllocator> struct ros::message_operations::Printer< ::sr_hand::reverseKinematics_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- reverseKinematics.h

8.58 ros::message_operations::Printer< ::sr_hand::sendupdate_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream >
  static void **stream** (Stream &s, const std::string &indent, const ::sr_hand::sendupdate_< ContainerAllocator > &v)

template<class ContainerAllocator> struct ros::message_operations::Printer< ::sr_hand::sendupdate_< ContainerAllocator > >

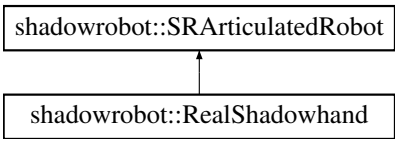
The documentation for this struct was generated from the following file:

- sendupdate.h

8.59 shadowrobot::RealShadowhand Class Reference

`#include <real_shadowhand.h>`

Inheritance diagram for shadowrobot::RealShadowhand:



Public Member Functions

- RealShadowhand ()
- ~RealShadowhand ()

  Destructor.

- virtual short sendupdate (std::string joint_name, double target)
- virtual JointData getJointData (std::string joint_name)
- virtual JointsMap getAllJointsData ()
- virtual short setContrl (std::string contrlr_name, JointControllerData ctrlr_data)
- virtual JointControllerData getContrl (std::string contrlr_name)
- virtual short setConfig (std::vector< std::string > myConfig)
- virtual void getConfig (std::string joint_name)
- virtual std::vector< DiagnosticData > getDiagnostics ()

Protected Member Functions

- void initializeMap ()

8.59.1 Detailed Description

The real shadowhand class is a class used to access the C code of the Dextrous Hand.

8.59.2 Constructor & Destructor Documentation

8.59.2.1 shadowrobot::RealShadowhand::RealShadowhand ()

Constructor for the real Shadowhand: initialize the connection to the robot and also initialize the joint_map containing the mapping between joint_names and their data.

8.59.3 Member Function Documentation

8.59.3.1 SRArticulatedRobot::JointsMap shadowrobot::RealShadowhand::getAllJointsData () [virtual]

Get the data for all the joints.

Returns

a mapping between the joints names and the information for each joint.

Implements shadowrobot::SRArticulatedRobot.

8.59.3.2 void shadowrobot::RealShadowhand::getConfig (std::string *joint_name*) [virtual]

Get the config of the palm

Todo

Not implemented yet

Parameters

joint_name

Implements shadowrobot::SRArticulatedRobot.

8.59.3.3 JointControllerData shadowrobot::RealShadowhand::getContrl (std::string *contrlr_name*) [virtual]

Get the controller parameters for a given controller name.

Parameters

contrlr_name the name of the controller.

Returns

The parameters of this controller

Implements shadowrobot::SRArticulatedRobot.

8.59.3.4 std::vector< DiagnosticData > shadowrobot::RealShadowhand::getDiagnostics () [virtual]

Get the diagnostics for the whole articulated robot.

Returns

A vector containing all the diagnostics for the robot (motor information, etc...)

Implements shadowrobot::SRArticulatedRobot.

8.59.3.5 JointData shadowrobot::RealShadowhand::getJointData (std::string *joint_name*) [virtual]

Get the joint data for a specific joint.

See also

JointData

Parameters

joint_name The name of the joint, as specified in joints_map.

Returns

The information regarding this joint.

Implements shadowrobot::SRArticulatedRobot.

8.59.3.6 void shadowrobot::RealShadowhand::initializeMap () [protected]

the parameters for the motors

8.59.3.7 short shadowrobot::RealShadowhand::sendupdate (std::string *joint_name*, double *target*) [virtual]

Send a new target to a given joint on the robot. The command will be issued to the robot which will move if the controllers are started and tuned. The target is truncated to the correct range.

Parameters

joint_name The Joint in joints_map you wish to send the target to.

target The target in degree

Returns

0 if success ; -1 if error

Implements shadowrobot::SRArticulatedRobot.

8.59.3.8 short shadowrobot::RealShadowhand::setConfig (std::vector< std::string > *myConfig*) [virtual]

Set the config of the palm

Todo

Not implemented yet

Parameters

myConfig

Returns

Implements shadowrobot::SRArticulatedRobot.

8.59.3.9 short shadowrobot::RealShadowhand::setContrl (std::string *contrlr_name*, JointControllerData *ctrlr_data*) [virtual]

Set the controller parameters for a given controller name.

Parameters

contrlr_name The name of the controller to setup.

ctrlr_data The data to pass to this controller.

Returns

0 if success.

Implements shadowrobot::SRArticulatedRobot.

The documentation for this class was generated from the following files:

- real_shadowhand.h
- real_shadowhand.cpp

8.60 sr_hand::msg::_reverseKinematics::reverseKinematics Class Reference

Public Member Functions

- def __init__
- def serialize
- def deserialize
- def serialize_numpy
- def deserialize_numpy

Data Fields

- **finger_name**

8.60.1 Member Function Documentation

8.60.1.1 def sr_hand::msg::_reverseKinematics::reverseKinematics::__init__ (*self*, *args*, *kwds*)

```
Constructor. Any message fields that are implicitly/explicitly
set to None will be assigned a default value. The recommend
use is keyword arguments as this is more robust to future message
changes.  You cannot mix in-order arguments and keyword arguments.

The available fields are:
   finger_name

@param args: complete set of field values, in .msg order
@param kwds: use keyword arguments corresponding to message field names
to set specific fields.
```

8.60.1.2 def sr_hand::msg::_reverseKinematics::reverseKinematics::deserialize (*self*, *str*)

```
unpack serialized message in str into this message instance
@param str: byte array of serialized message
@type  str: str
```

8.60.1.3 def sr_hand::msg::_reverseKinematics::reverseKinematics::deserialize_numpy (*self*, *str*, *numpy*)

```
unpack serialized message in str into this message instance using numpy for array types
@param str: byte array of serialized message
@type  str: str
@param numpy: numpy python module
@type  numpy: module
```

8.60.1.4 def sr_hand::msg::_reverseKinematics::reverseKinematics::serialize (*self*, *buff*)

```
serialize message into buffer
@param buff: buffer
@type  buff: StringIO
```

8.60.1.5 def sr_hand::msg::_reverseKinematics::reverseKinematics::serialize_numpy (*self*, *buff*, *numpy*)

```
serialize message with numpy array types into buffer
@param buff: buffer
@type  buff: StringIO
@param numpy: numpy python module
@type  numpy module
```

The documentation for this class was generated from the following file:

- _reverseKinematics.py

8.61 sr_hand::reverseKinematics_< ContainerAllocator > Struct Template Reference

Public Types

- typedef reverseKinematics_< ContainerAllocator > **Type**
- typedef std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > **_finger_name_type**
- typedef boost::shared_ptr< ::sr_hand::reverseKinematics_< ContainerAllocator > > **Ptr**
- typedef boost::shared_ptr< ::sr_hand::reverseKinematics_< ContainerAllocator > const > **ConstPtr**

Public Member Functions

- **reverseKinematics_** (const ContainerAllocator &_alloc)
- ROSCPP_DEPRECATED const std::string **__getDataType** () const
- ROSCPP_DEPRECATED const std::string **__getMD5Sum** () const
- ROSCPP_DEPRECATED const std::string **__getMessageDefinition** () const
- virtual ROSCPP_DEPRECATED uint8_t * **serialize** (uint8_t *write_ptr, uint32_t seq) const
- virtual ROSCPP_DEPRECATED uint8_t * **deserialize** (uint8_t *read_ptr)
- virtual ROSCPP_DEPRECATED uint32_t **serializationLength** () const

Static Public Member Functions

- static ROSCPP_DEPRECATED const std::string **__s_getDataType** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMD5Sum** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMessageDefinition** ()

Data Fields

- std::basic_string< char, std::char_traits< char >, typename ContainerAllocator::template rebind< char >::other > **finger_name**

template<class ContainerAllocator> struct sr_hand::reverseKinematics_< ContainerAllocator >

The documentation for this struct was generated from the following file:

- reverseKinematics.h

8.62 sr_hand::msg::_sendupdate::sendupdate Class Reference

Public Member Functions

- def __init__
- def serialize
- def deserialize
- def serialize_numpy
- def deserialize_numpy

Data Fields

- **sendupdate_length**
- **sendupdate_list**

8.62.1 Member Function Documentation

8.62.1.1 def sr_hand::msg::_sendupdate::sendupdate::__init__ (*self*, *args*, *kwds*)

```
Constructor. Any message fields that are implicitly/explicitly
set to None will be assigned a default value. The recommend
use is keyword arguments as this is more robust to future message
changes.  You cannot mix in-order arguments and keyword arguments.

The available fields are:
   sendupdate_length,sendupdate_list

@param args: complete set of field values, in .msg order
@param kwds: use keyword arguments corresponding to message field names
to set specific fields.
```

8.62.1.2 def sr_hand::msg::_sendupdate::sendupdate::deserialize (*self*, *str*)

```
unpack serialized message in str into this message instance
@param str: byte array of serialized message
@type  str: str
```

8.62.1.3 def sr_hand::msg::_sendupdate::sendupdate::deserialize_numpy (*self*, *str*, *numpy*)

```
unpack serialized message in str into this message instance using numpy for array types
@param str: byte array of serialized message
@type  str: str
@param numpy: numpy python module
@type  numpy: module
```

8.62.1.4 def sr_hand::msg::_sendupdate::sendupdate::serialize (*self*, *buff*)

```
serialize message into buffer
@param buff: buffer
@type  buff: StringIO
```

8.62.1.5 def sr_hand::msg::_sendupdate::sendupdate::serialize_numpy (*self*, *buff*, *numpy*)

```
serialize message with numpy array types into buffer
@param buff: buffer
@type  buff: StringIO
@param numpy: numpy python module
@type  numpy module
```

The documentation for this class was generated from the following file:

- _sendupdate.py

8.63 sr_hand::sendupdate_< ContainerAllocator > Struct Template Reference

Public Types

- typedef sendupdate_< ContainerAllocator > **Type**
- typedef int8_t **_sendupdate_length_type**
- typedef std::vector< ::sr_hand::joint_< ContainerAllocator >, typename ContainerAllocator::template rebind< ::sr_hand::joint_< ContainerAllocator > >::other > **_sendupdate_list_type**

- typedef boost::shared_ptr< ::sr_hand::sendupdate_< ContainerAllocator > > **Ptr**
- typedef boost::shared_ptr< ::sr_hand::sendupdate_< ContainerAllocator > const > **ConstPtr**

Public Member Functions

- **sendupdate_** (const ContainerAllocator &_alloc)
- ROSCPP_DEPRECATED uint32_t **get_sendupdate_list_size** () const
- ROSCPP_DEPRECATED void **set_sendupdate_list_size** (uint32_t size)
- ROSCPP_DEPRECATED void **get_sendupdate_list_vec** (std::vector< ::sr_hand::joint_< ContainerAllocator >, typename ContainerAllocator::template rebind< ::sr_hand::joint_< ContainerAllocator > >::other > &vec) const
- ROSCPP_DEPRECATED void **set_sendupdate_list_vec** (const std::vector< ::sr_hand::joint_< ContainerAllocator >, typename ContainerAllocator::template rebind< ::sr_hand::joint_< ContainerAllocator > >::other > &vec)
- ROSCPP_DEPRECATED const std::string **__getDataType** () const
- ROSCPP_DEPRECATED const std::string **__getMD5Sum** () const
- ROSCPP_DEPRECATED const std::string **__getMessageDefinition** () const
- virtual ROSCPP_DEPRECATED uint8_t * **serialize** (uint8_t *write_ptr, uint32_t seq) const
- virtual ROSCPP_DEPRECATED uint8_t * **deserialize** (uint8_t *read_ptr)
- virtual ROSCPP_DEPRECATED uint32_t **serializationLength** () const

Static Public Member Functions

- static ROSCPP_DEPRECATED const std::string **__s_getDataType** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMD5Sum** ()
- static ROSCPP_DEPRECATED const std::string **__s_getMessageDefinition** ()

Data Fields

- int8_t **sendupdate_length**
- std::vector< ::sr_hand::joint_< ContainerAllocator >, typename ContainerAllocator::template rebind< ::sr_hand::joint_< ContainerAllocator > >::other > **sendupdate_list**

template<class ContainerAllocator> struct sr_hand::sendupdate_< ContainerAllocator >

The documentation for this struct was generated from the following file:

- sendupdate.h

8.64 ros::serialization::Serializer< ::sr_hand::cartesian_data_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream , typename T >
  static void **allInOne** (Stream &stream, T m)

Data Fields

- **ROS_DECLARE_ALLINONE_SERIALIZER**

template<class ContainerAllocator> struct ros::serialization::Serializer< ::sr_hand::cartesian_data_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- cartesian_data.h

8.65 ros::serialization::Serializer< ::sr_hand::cartesian_position_-< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream , typename T >
  static void **allInOne** (Stream &stream, T m)

Data Fields

- **ROS_DECLARE_ALLINONE_SERIALIZER**

template<class ContainerAllocator> struct ros::serialization::Serializer< ::sr_hand::cartesian_-position_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- cartesian_position.h

8.66 ros::serialization::Serializer< ::sr_hand::command_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream , typename T >
  static void **allInOne** (Stream &stream, T m)

Data Fields

- **ROS_DECLARE_ALLINONE_SERIALIZER**

template<class ContainerAllocator> struct ros::serialization::Serializer< ::sr_hand::command_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- command.h

8.67 ros::serialization::Serializer< ::sr_hand::config_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream , typename T >
  static void **allInOne** (Stream &stream, T m)

Data Fields

- **ROS_DECLARE_ALLINONE_SERIALIZER**

template<class ContainerAllocator> struct ros::serialization::Serializer< ::sr_hand::config_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- config.h

8.68 ros::serialization::Serializer< ::sr_hand::contrlr_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream , typename T >
  static void **allInOne** (Stream &stream, T m)

Data Fields

- **ROS_DECLARE_ALLINONE_SERIALIZER**

template<class ContainerAllocator> struct ros::serialization::Serializer< ::sr_hand::contrlr_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- contrlr.h

8.69 ros::serialization::Serializer< ::sr_hand::joint_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream , typename T >
  static void **allInOne** (Stream &stream, T m)

Data Fields

- **ROS_DECLARE_ALLINONE_SERIALIZER**

template<class ContainerAllocator> struct ros::serialization::Serializer< ::sr_hand::joint_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- joint.h

8.70 ros::serialization::Serializer< ::sr_hand::joints_data_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream , typename T >
  static void **allInOne** (Stream &stream, T m)

Data Fields

- **ROS_DECLARE_ALLINONE_SERIALIZER**

template<class ContainerAllocator> struct ros::serialization::Serializer< ::sr_hand::joints_data_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- joints_data.h

8.71 ros::serialization::Serializer< ::sr_hand::reverseKinematics_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream , typename T >
  static void **allInOne** (Stream &stream, T m)

Data Fields

- **ROS_DECLARE_ALLINONE_SERIALIZER**

template<class ContainerAllocator> struct ros::serialization::Serializer< ::sr_hand::reverseKinematics_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- reverseKinematics.h

8.72 ros::serialization::Serializer< ::sr_hand::sendupdate_< ContainerAllocator > > Struct Template Reference

Static Public Member Functions

- template<typename Stream , typename T >
  static void **allInOne** (Stream &stream, T m)

Data Fields

- **ROS_DECLARE_ALLINONE_SERIALIZER**

template<class ContainerAllocator> struct ros::serialization::Serializer< ::sr_hand::sendupdate_< ContainerAllocator > >

The documentation for this struct was generated from the following file:

- sendupdate.h

8.73 shadowhand_ros::ShadowHand_ROS Class Reference

Public Member Functions

- def __init__
- def **create_grasp_interpoler**
- def callback
- def callback_arm
- def init_actual_joints
- def check_hand_presence
- def set_shadowhand_data_topic
- def set_sendupdate_topic
- def sendupdate_from_dict
- def sendupdate
- def sendupdate_arm_from_dict
- def sendupdate_arm
- def valueof
- def has_arm
- def record_step_to_file
- def **save_hand_position_to_file**
- def read_all_current_positions
- def read_all_current_targets
- def read_all_current_arm_positions
- def read_all_current_arm_targets
- def resend_targets
- def callVisualisationService

Data Fields

- **allJoints**
- **handJoints**
- **armJoints**
- **lastMsg**
- **lastArmMsg**
- **cyberglove_pub**
- **cyberglove_sub**
- **cybergrasp_pub**
- **sub**
- **pub**
- **sub_arm**
- **pub_arm**
- **isFirstMessage**
- **isFirstMessageArm**
- **isReady**
- **liste**
- **hasarm**
- **dict_pos**
- **dict_tar**
- **dict_arm_pos**
- **dict_arm_tar**

- **sendupdate_lock**
- **grasp_parser**
- **rootPath**
- **grasp_interpoler**

8.73.1 Detailed Description

```
This is a python library used to easily access the shadow hand ROS interface.
```

8.73.2 Member Function Documentation

8.73.2.1 def shadowhand_ros::ShadowHand_ROS::__init__ (*self*)

```
Builds the library, creates the communication node in ROS and
initializes the hand publisher and subscriber to the default
values of shadowhand_data and sendupdate
```

8.73.2.2 def shadowhand_ros::ShadowHand_ROS::callback (*self*, *data*)

```
@param data: The ROS message received which called the callback
If the message is the first received, initializes the dictionnaries
Else, it updates the lastMsg
```

8.73.2.3 def shadowhand_ros::ShadowHand_ROS::callback_arm (*self*, *data*)

```
@param data: The ROS message received which called the callback
If the message is the first received, initializes the dictionnaries
Else, it updates the lastMsg
```

8.73.2.4 def shadowhand_ros::ShadowHand_ROS::callVisualisationService (*self*, *callList* = 0, *reset* = 0)

```
@param callList: dictionnary mapping joint names to information that should be displayed
@param reset: flag used to tell if the parameters should be replaced by the new ones or just added to the
Calls a ROS service to display various information in Rviz
```

8.73.2.5 def shadowhand_ros::ShadowHand_ROS::check_hand_presence (*self*)

```
@return : true if the hand if detected
```

8.73.2.6 def shadowhand_ros::ShadowHand_ROS::has_arm (*self*)

```
@return : True if an arm is detected on the roscore
```

8.73.2.7 def shadowhand_ros::ShadowHand_ROS::init_actual_joints (*self*)

```
Initializes the library with just the fingers actually connected
```

8.73.2.8 def shadowhand_ros::ShadowHand_ROS::read_all_current_arm_positions (*self*)

```
@return: dictionnary mapping joint names to actual positions
Read all the positions in the lastMsg
```

8.73.2.9 def shadowhand_ros::ShadowHand_ROS::read_all_current_arm_targets (*self*)

```
@return: dictionnary mapping joint names to actual targets
Read all the targets in the lastMsg
```

8.73.2.10 def shadowhand_ros::ShadowHand_ROS::read_all_current_positions (*self*)

```
@return: dictionnary mapping joint names to actual positions
Read all the positions in the lastMsg
```

8.73.2.11 def shadowhand_ros::ShadowHand_ROS::read_all_current_targets (*self*)

```
@return: dictionnary mapping joint names to current targets
Read all the targets in the lastMsg
```

8.73.2.12 def shadowhand_ros::ShadowHand_ROS::record_step_to_file (*self*, *filename*, *grasp_as_xml*)

```
@param filename: name (or path) of the file to save to
@param grasp_as_xml: xml-formatted grasp
Write the grasp at the end of the file, creates the file if does not exist
```

8.73.2.13 def shadowhand_ros::ShadowHand_ROS::resend_targets (*self*)

```
Resend the targets read in the lastMsg to the hand
```

8.73.2.14 def shadowhand_ros::ShadowHand_ROS::sendupdate (*self*, *jointName*, *angle* = 0)

```
@param jointName: Name of the joint to update
@param angle: Target of the joint, 0 if not set
Sends a new target for the specified joint
```

8.73.2.15 def shadowhand_ros::ShadowHand_ROS::sendupdate_arm (*self*, *jointName*, *angle* = 0)

```
@param jointName: Name of the joint to update
@param angle: Target of the joint, 0 if not set
Sends a new target for the specified joint
```

8.73.2.16 def shadowhand_ros::ShadowHand_ROS::sendupdate_arm_from_dict (*self*, *dicti*)

```
@param dicti: Dictionnary containing all the targets to send, mapping the name of the joint to the value o
Sends new targets to the hand from a dictionnary
```

8.73.2.17 def shadowhand_ros::ShadowHand_ROS::sendupdate_from_dict (*self*, *dicti*)

```
@param dicti: Dictionnary containing all the targets to send, mapping the name of the joint to the value o
Sends new targets to the hand from a dictionnary
```

8.73.2.18 def shadowhand_ros::ShadowHand_ROS::set_sendupdate_topic (*self*, *topic*)

```
@param topic: The new topic to be set as the hand subscribing topic
Set the library to publish to a new topic
```

8.73.2.19 def shadowhand_ros::ShadowHand_ROS::set_shadowhand_data_topic (*self*, *topic*)

```
@param topic: The new topic to be set as the hand publishing topic
Set the library to listen to a new topic
```

8.73.2.20 def shadowhand_ros::ShadowHand_ROS::valueof (*self*, *jointName*)

```
@param jointName: Name of the joint to read the value
@return: 'NaN' if the value is not correct, the actual position of the joint else
```

The documentation for this class was generated from the following file:

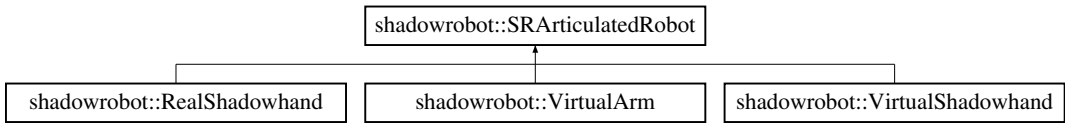
- shadowhand_ros.py

8.74 shadowrobot::SRArticulatedRobot Class Reference

This is a parent class for the different types of Shadowhand we can have. It makes it possible to swap from a virtual to a real hand while using the same utilities to interact with the hand. One (or more) ROS subscriber and publisher can then share the same instance of a Shadowhand object to update the information contained in this object.

`#include <sr_articulated_robot.h>`

Inheritance diagram for shadowrobot::SRArticulatedRobot:



Public Types

- typedef std::map< std::string, JointData > JointsMap
- typedef std::map< std::string, enum controller_parameters > ParametersMap

Public Member Functions

- SRArticulatedRobot ()
- ~SRArticulatedRobot ()
- virtual short sendupdate (std::string joint_name, double target)=0
- virtual JointData getJointData (std::string joint_name)=0
- virtual JointsMap getAllJointsData ()=0
- virtual short setContrl (std::string contrlr_name, JointControllerData ctrlr_data)=0
- virtual JointControllerData getContrl (std::string contrlr_name)=0
- virtual short setConfig (std::vector< std::string > myConfig)=0
- virtual void getConfig (std::string joint_name)=0
- virtual std::vector< DiagnosticData > getDiagnostics ()=0

Protected Attributes

- JointsMap joints_map

  A mapping between the joint names and the information regarding those joints.

- ParametersMap parameters_map

  A mapping between the parameter names and their values.

- boost::mutex **joints_map_mutex**
- boost::mutex **parameters_map_mutex**
- boost::mutex **controllers_map_mutex**

8.74.1 Detailed Description

This is a parent class for the different types of Shadowhand we can have. It makes it possible to swap from a virtual to a real hand while using the same utilities to interact with the hand. One (or more) ROS subscriber and publisher can then share the same instance of a Shadowhand object to update the information contained in this object.

8.74.2 Member Typedef Documentation

8.74.2.1 typedef std::map<std::string, JointData> shadowrobot::SRArticulatedRobot::JointsMap

See also

joints_map

8.74.2.2 typedef std::map<std::string, enum controller_parameters> shadowrobot::SRArticulatedRobot::ParametersMap

See also

parameters_map

8.74.3 Constructor & Destructor Documentation

8.74.3.1 shadowrobot::SRArticulatedRobot::SRArticulatedRobot () `[inline]`

empty constructor.

8.74.3.2 shadowrobot::SRArticulatedRobot::~SRArticulatedRobot () `[inline]`

empty destructor.

8.74.4 Member Function Documentation

8.74.4.1 virtual JointsMap shadowrobot::SRArticulatedRobot::getAllJointsData () `[pure virtual]`

Get the data for all the joints.

Returns

a mapping between the joints names and the information for each joint.

Implemented in shadowrobot::RealShadowhand, shadowrobot::VirtualArm, and shadowrobot::VirtualShadowhand.

8.74.4.2 virtual void shadowrobot::SRArticulatedRobot::getConfig (std::string *joint_name*) `[pure virtual]`

Get the config of the palm

Todo

Not implemented yet

Parameters

joint_name

Implemented in shadowrobot::RealShadowhand, shadowrobot::VirtualArm, and shadowrobot::VirtualShadowhand.

8.74.4.3 virtual JointControllerData shadowrobot::SRArticulatedRobot::getContrl (std::string *contrlr_name*) [pure virtual]

Get the controller parameters for a given controller name.

Parameters

contrlr_name the name of the controller.

Returns

The parameters of this controller

Implemented in shadowrobot::RealShadowhand, shadowrobot::VirtualArm, and shadowrobot::VirtualShadowhand.

8.74.4.4 virtual std::vector<DiagnosticData> shadowrobot::SRArticulatedRobot::getDiagnostics () [pure virtual]

Get the diagnostics for the whole articulated robot.

Returns

A vector containing all the diagnostics for the robot (motor information, etc...)

Implemented in shadowrobot::RealShadowhand, shadowrobot::VirtualArm, and shadowrobot::VirtualShadowhand.

8.74.4.5 virtual JointData shadowrobot::SRArticulatedRobot::getJointData (std::string *joint_name*) [pure virtual]

Get the joint data for a specific joint.

See also

JointData

Parameters

joint_name The name of the joint, as specified in joints_map.

Returns

The information regarding this joint.

Implemented in shadowrobot::RealShadowhand, shadowrobot::VirtualArm, and shadowrobot::VirtualShadowhand.

8.74.4.6 virtual short shadowrobot::SRArticulatedRobot::sendupdate (std::string *joint_name*, double *target*) [pure virtual]

Generic method called to pass a new target to an articulated robot.

Parameters

joint_name The Joint in joints_map you wish to send the target to.

target The target in degree

Returns

0 if success ; -1 if error

Implemented in shadowrobot::RealShadowhand, shadowrobot::VirtualArm, and shadowrobot::VirtualShadowhand.

8.74.4.7 virtual short shadowrobot::SRArticulatedRobot::setConfig (std::vector< std::string > *myConfig*) [pure virtual]

Set the config of the palm

Todo

Not implemented yet

Parameters

myConfig

Returns

Implemented in shadowrobot::RealShadowhand, shadowrobot::VirtualArm, and shadowrobot::VirtualShadowhand.

8.74.4.8 virtual short shadowrobot::SRArticulatedRobot::setContrl (std::string *contrlr_name*, JointControllerData *ctrlr_data*) [pure virtual]

Set the controller parameters for a given controller name.

Parameters

contrlr_name The name of the controller to setup.

ctrlr_data The data to pass to this controller.

Returns

0 if success.

Implemented in shadowrobot::RealShadowhand, shadowrobot::VirtualArm, and shadowrobot::VirtualShadowhand.

The documentation for this class was generated from the following file:

- sr_articulated_robot.h

8.75 shadowrobot::SRDiagnosticer Class Reference

Public Member Functions

- SRDiagnosticer (boost::shared_ptr< SRArticulatedRobot > sr_art_robot, hardware_types hw_type)
- ~SRDiagnosticer ()

  Destructor.

- void publish ()

8.75.1 Constructor & Destructor Documentation

8.75.1.1 shadowrobot::SRDiagnosticer::SRDiagnosticer (boost::shared_ptr< SRArticulatedRobot > *sr_art_robot*, hardware_types *hw_type*)

Constructor initializing the ROS node, and setting the topic to which it publishes. The frequency at which this node will publish data is set by a parameter, read from ROS parameter server.

Parameters

sh A SRArticulatedRobot object, where the information to be published comes from.

hw_type The type of hardware we are publishing diagnostics about.

8.75.2 Member Function Documentation

8.75.2.1 void shadowrobot::SRDiagnosticer::publish ()

The callback method which is called at a given frequency. Gets the data from the shadowhand / shadowarm object.

The documentation for this class was generated from the following files:

- sr_diagnosticer.h
- sr_diagnosticer.cpp

8.76 test::SRLibrary Class Reference

Public Member Functions

- def **__init__**

Data Fields

- **mutex**
- **pub**

8.76.1 Detailed Description

```
This is a python library used to easily access the shadow robot ROS interface.
```

The documentation for this class was generated from the following file:

- test.py

8.77 shadowrobot::SRPublisher Class Reference

`#include <sr_publisher.h>`

Public Member Functions

- SRPublisher (boost::shared_ptr< SRArticulatedRobot > sr_art_robot)
- ~SRPublisher ()

  Destructor.

- void publish ()

8.77.1 Detailed Description

This class publishes data concerning the shadowhand /shadowarm, like position, targets, forces, ... on different topics. To publish those data, just call the publish() function.

The data are published on three different topics. Two of them are joint_states topics (joint names, joint angles, joint velocities, joint efforts), which are then transformed into coordinates by two different robot_state_publisher (a ROS package). This is useful to visualize the data in rviz (part of ROS). A third topic is \/prefix\/shadowhand_data. The messages published on this last topic are better formatted for our hardware.

8.77.2 Constructor & Destructor Documentation

8.77.2.1 shadowrobot::SRPublisher::SRPublisher (boost::shared_ptr< SRArticulatedRobot > *sr_art_robot*)

Constructor initializing the ROS node, and setting the topic to which it publishes. The frequency at which this node will publish data is set by a parameter, read from ROS parameter server.

Parameters

sh A Shadowhand or Shadowarm object, where the information to be published comes from.

8.77.3 Member Function Documentation

8.77.3.1 void shadowrobot::SRPublisher::publish ()

The callback method which is called at a given frequency. Gets the data from the shadowhand/shadowarm object.

The documentation for this class was generated from the following files:

- sr_publisher.h
- sr_publisher.cpp

8.78 shadowrobot::SRSubscriber Class Reference

`#include <sr_subscriber.h>`

Public Member Functions

- SRSubscriber (boost::shared_ptr< SRArticulatedRobot > sr_art_robot)
- SRSubscriber (boost::shared_ptr< SRArticulatedRobot > sr_art_robot, KDL::Tree tree)
- ~SRSubscriber ()

  Destructor.

8.78.1 Detailed Description

This ROS subscriber is used to issue commands to the hand / arm, from sending a set of targets, to changing the controller parameters.

8.78.2 Constructor & Destructor Documentation

8.78.2.1 shadowrobot::SRSubscriber::SRSubscriber (boost::shared_ptr< SRArticulatedRobot > *sr_art_robot*)

Constructor initializing the ROS node, and setting the topic to which it subscribes.

Parameters

sh A Shadowhand object, where the information to be published comes from.

8.78.2.2 shadowrobot::SRSubscriber::SRSubscriber (boost::shared_ptr< SRArticulatedRobot > *sr_art_robot*, KDL::Tree *tree*)

Constructor initializing the ROS node, and setting the topic to which it subscribes.

Parameters

sh A Shadowhand object, where the information to be published comes from.

tree The kinematic tree of the hand. Used to compute reverse kinematics.

The documentation for this class was generated from the following files:

- sr_subscriber.h
- sr_subscriber.cpp

8.79 shadowrobot::SRTactileSensorPublisher Class Reference

Public Member Functions

- SRTactileSensorPublisher ()

  Constructor.

- ~SRTactileSensorPublisher ()

  Destructor.

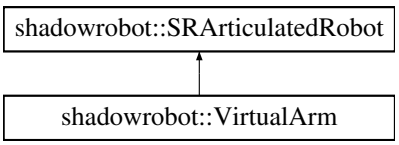
- void **publish** ()

The documentation for this class was generated from the following files:

- sr_tactile_sensor_pub.h
- sr_tactile_sensor_pub.cpp

8.80 shadowrobot::VirtualArm Class Reference

Inheritance diagram for shadowrobot::VirtualArm:



Public Member Functions

- VirtualArm ()
- ~VirtualArm ()

  destructor

- virtual short sendupdate (std::string joint_name, double target)
- virtual JointData getJointData (std::string joint_name)
- virtual JointsMap getAllJointsData ()
- virtual short setContrl (std::string contrlr_name, JointControllerData ctrlr_data)
- virtual JointControllerData getContrl (std::string ctrlr_name)
- virtual short setConfig (std::vector< std::string > myConfig)
- virtual void getConfig (std::string joint_name)
- virtual std::vector< DiagnosticData > getDiagnostics ()

Protected Member Functions

- void initializeMap ()
- double toRad (double deg)
- double toDegrees (double rad)

8.80.1 Constructor & Destructor Documentation

8.80.1.1 shadowrobot::VirtualArm::VirtualArm ()

Initializes the necessary mappings with a static list of names.

8.80.2 Member Function Documentation

8.80.2.1 SRArticulatedRobot::JointsMap shadowrobot::VirtualArm::getAllJointsData () [virtual]

Get the data for all the joints.

Returns

a mapping between the joints names and the information for each joint.

Implements shadowrobot::SRArticulatedRobot.

8.80.2.2 void shadowrobot::VirtualArm::getConfig (std::string *joint_name*) [virtual]

Get the config of the palm

Todo

Not implemented yet

Parameters

joint_name

Implements shadowrobot::SRArticulatedRobot.

8.80.2.3 JointControllerData shadowrobot::VirtualArm::getContrl (std::string *contrlr_name*) [virtual]

Get the controller parameters for a given controller name.

Parameters

contrlr_name the name of the controller.

Returns

The parameters of this controller

Implements shadowrobot::SRArticulatedRobot.

8.80.2.4 std::vector< DiagnosticData > shadowrobot::VirtualArm::getDiagnostics () [virtual]

Generates a set of random data to be published by the diagnostic publisher, but keep the position and target as they are (those are updated by the sendupdate() function).

Returns

A vector containing all the diagnostics for the hand (motor information, etc...)

Implements shadowrobot::SRArticulatedRobot.

8.80.2.5 JointData shadowrobot::VirtualArm::getJointData (std::string *joint_name*) [virtual]

In the virtual arm, getJointData() simply fetches the data from a given joint in the joints_map. As the targets and positions are up to date, there's no need of reading other values here for those two values. However, we generate random values for the other data (temperature, current, etc...)

Parameters

joint_name The name of the joint, as specified in joints_map.

Returns

The information regarding this joint.

Implements shadowrobot::SRArticulatedRobot.

8.80.2.6 void shadowrobot::VirtualArm::initializeMap () [protected]

Initialise a mapping for the joints.

8.80.2.7 short shadowrobot::VirtualArm::sendupdate (std::string *joint_name*, double *target*) [virtual]

This function will set the target of the object to the given angle. It will also set the position to this target.

Todo

This could be improved by implementing a control algorithm in this theoretic arm.

Parameters

joint_name The Joint in joints_map you wish to send the target to.
target The target in degree

Returns

0 if success ; -1 if error

Implements shadowrobot::SRArticulatedRobot.

8.80.2.8 short shadowrobot::VirtualArm::setConfig (std::vector< std::string > *myConfig*) [virtual]

Set the config of the palm

Todo

Not implemented yet

Parameters

myConfig

Returns

Implements shadowrobot::SRArticulatedRobot.

8.80.2.9 short shadowrobot::VirtualArm::setContrl (std::string *contrlr_name*, JointControllerData *ctrlr_data*) [virtual]

Set the controller parameters for a given controller name.

Parameters

contrlr_name The name of the controller to setup.
ctrlr_data The data to pass to this controller.

Returns

0 if success.

Implements shadowrobot::SRArticulatedRobot.

8.80.2.10 double shadowrobot::VirtualArm::toDegrees (double *rad*) [inline, protected]

Convert an angle in radian to an angle in degrees.

Parameters

deg the angle in rads

Returns

the value in degrees.

8.80.2.11 double shadowrobot::VirtualArm::toRad (double *deg*) [inline, protected]

Convert an angle in degree to an angle in radians.

Parameters

deg the angle in degrees

Returns

the value in rads.

The documentation for this class was generated from the following files:

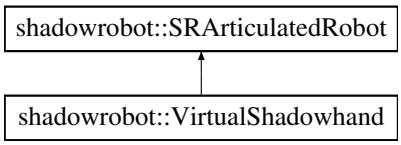
- virtual_arm.h
- virtual_arm.cpp

8.81 shadowrobot::VirtualShadowhand Class Reference

The Virtual Shadowhand can be used as a simulator. As both the real hand and the virtual hand are children from the shadowhand class, using a virtual or a real hand doesn't change anything in the way you call them in your programs.

`#include <virtual_shadowhand.h>`

Inheritance diagram for shadowrobot::VirtualShadowhand:



Public Member Functions

- VirtualShadowhand ()
- ~VirtualShadowhand ()

  destructor

- virtual short sendupdate (std::string joint_name, double target)
- virtual JointData getJointData (std::string joint_name)
- virtual JointsMap getAllJointsData ()
- virtual short setContrl (std::string contrlr_name, JointControllerData ctrlr_data)
- virtual JointControllerData getContrl (std::string ctrlr_name)
- virtual short setConfig (std::vector< std::string > myConfig)
- virtual void getConfig (std::string joint_name)
- virtual std::vector< DiagnosticData > getDiagnostics ()

Protected Types

- typedef std::map< std::string, JointControllerData > ControllersMap

Protected Member Functions

- void initializeMap ()
- double toRad (double deg)
- double toDegrees (double rad)

Protected Attributes

- ControllersMap controllers_map

  Contains the mapping between the controller names and their data.

8.81.1 Detailed Description

The Virtual Shadowhand can be used as a simulator. As both the real hand and the virtual hand are children from the shadowhand class, using a virtual or a real hand doesn't change anything in the way you call them in your programs.

8.81.2 Member Typedef Documentation

8.81.2.1 typedef std::map<std::string, JointControllerData> shadowrobot::VirtualShadowhand::ControllersMap [protected]

See also

controllers_map

8.81.3 Constructor & Destructor Documentation

8.81.3.1 shadowrobot::VirtualShadowhand::VirtualShadowhand ()

Initializes the necessary mappings with a static list of names.

8.81.4 Member Function Documentation

8.81.4.1 SRArticulatedRobot::JointsMap shadowrobot::VirtualShadowhand::getAllJointsData () [virtual]

Get the data for all the joints.

Returns

a mapping between the joints names and the information for each joint.

Implements shadowrobot::SRArticulatedRobot.

8.81.4.2 void shadowrobot::VirtualShadowhand::getConfig (std::string *joint_name*) [virtual]

Get the config of the palm

Todo

Not implemented yet

Parameters

joint_name

Implements shadowrobot::SRArticulatedRobot.

8.81.4.3 JointControllerData shadowrobot::VirtualShadowhand::getContrl (std::string *contrlr_name*) [virtual]

Get the controller parameters for a given controller name.

Parameters

contrlr_name the name of the controller.

Returns

The parameters of this controller

Implements shadowrobot::SRArticulatedRobot.

8.81.4.4 std::vector< DiagnosticData > shadowrobot::VirtualShadowhand::getDiagnostics () [virtual]

Generates a set of random data to be published by the diagnostic publisher, but keep the position and target as they are (those are updated by the sendupdate() function).

Returns

A vector containing all the diagnostics for the hand (motor information, etc...)

Implements shadowrobot::SRArticulatedRobot.

8.81.4.5 JointData shadowrobot::VirtualShadowhand::getJointData (std::string *joint_name*) [virtual]

In the virtual hand, getJointData() simply fetches the data from a given joint in the joints_map. As the targets and positions are up to date, there's no need of reading other values here for those two values. However, we generate random values for the other data (temperature, current, etc...)

Parameters

joint_name The name of the joint, as specified in joints_map.

Returns

The information regarding this joint.

Implements shadowrobot::SRArticulatedRobot.

8.81.4.6 void shadowrobot::VirtualShadowhand::initializeMap () [protected]

Initialize a mapping for the joints as well as a mapping for the controllers.

the parameters for the motors

8.81.4.7 short shadowrobot::VirtualShadowhand::sendupdate (std::string *joint_name*, double *target*) [virtual]

This function will set the target of the object to the given angle. It will also set the position to this target.

Todo

This could be improved by implementing a control algorithm in this theoretic hand.

Parameters

joint_name The Joint in joints_map you wish to send the target to.

target The target in degree

Returns

0 if success ; -1 if error

Implements shadowrobot::SRArticulatedRobot.

8.81.4.8 short shadowrobot::VirtualShadowhand::setConfig (std::vector< std::string > *myConfig*) [virtual]

Set the config of the palm

Todo

Not implemented yet

Parameters

myConfig

Returns

Implements shadowrobot::SRArticulatedRobot.

8.81.4.9 short shadowrobot::VirtualShadowhand::setContrl (std::string *contrlr_name*, JointControllerData *ctrlr_data*) [virtual]

Set the controller parameters for a given controller name.

Parameters

contrlr_name The name of the controller to setup.

ctrlr_data The data to pass to this controller.

Returns

0 if success.

Implements shadowrobot::SRArticulatedRobot.

8.81.4.10 double shadowrobot::VirtualShadowhand::toDegrees (double *rad*) [inline, protected]

Convert an angle in radian to an angle in degrees.

Parameters

deg the angle in rads

Returns

the value in degrees.

8.81.4.11 double shadowrobot::VirtualShadowhand::toRad (double *deg*) [inline, protected]

Convert an angle in degree to an angle in radians.

Parameters

deg the angle in degrees

Returns

the value in rads.

The documentation for this class was generated from the following files:

- virtual_shadowhand.h
- virtual_shadowhand.cpp

Chapter 9

File Documentation

9.1 link_joints.cpp File Reference

This is an example to show how to get data from the hand, read the position for a specific joint and send this as the target to another joint.

```
#include <ros/ros.h>
#include <string>
#include <sr_hand/joints_data.h>
#include <sr_hand/joint.h>
#include <sr_hand/sendupdate.h>
```

Functions

- void callback (const sr_hand::joints_dataConstPtr &msg)
- int main (int argc, char **argv)

Variables

- std::string parent_name = "FFJ3"

  the name of the parent joint

- std::string child_name = "MFJ3"

  the name of the child joint to link to the parent

- ros::Subscriber **sub**
- ros::Publisher **pub**

9.1.1 Detailed Description

This is an example to show how to get data from the hand, read the position for a specific joint and send this as the target to another joint.

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Thu Jul 8 16:57:22 2010

To test this program, just start the hand, rviz visualizer, the control GUI and this example (in 4 different consoles):

```
roslaunch sr_hand srh_motor.launch
roslaunch sr_hand rviz_motor.launch
rosrun sr_control_gui __init__.py
rosrun sr_hand link_joints
```

If you move the joint slider for FFJ3, then MFJ3 will move as well.

9.1.2 Function Documentation

9.1.2.1 void callback (const sr_hand::joints_dataConstPtr & *msg*)

The callback function is called each time a message is received on the topic /srh/shadowhand_data

Parameters

msg message of type sr_hand::joints_data

if it's the parent joint, read the target, and send it to the child.

9.1.2.2 int main (int *argc*, char ** *argv*)

The main: initialise a ros node, a subscriber and a publisher

Parameters

argc

argv

Returns

0 on success

init the subscriber and subscribe to the topic /srh/shadowhand_data, using the callback function callback()

init the publisher on the topic /srh/sendupdate publishing messages of the type sr_hand::sendupdate.

9.2 real_shadowhand.cpp File Reference

```
#include <robot/robot.h>
#include <robot/hand.h>
#include <robot/hand_protocol.h>
#include <ros/ros.h>
#include "sr_hand/hand/real_shadowhand.h"
```

9.2.1 Detailed Description

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Tue May 25 17:50:42 2010

9.3 real_shadowhand.h File Reference

The real shadowhand is the ROS interface to Shadow Robot robotic hand.

```
#include "sr_hand/hand/sr_articulated_robot.h"
```

Data Structures

- class shadowrobot::RealShadowhand

9.3.1 Detailed Description

The real shadowhand is the ROS interface to Shadow Robot robotic hand.

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Tue May 25 17:51:10 2010

Todo

Make sure it works with the motor hand.

9.4 real_shadowhand_node.cpp File Reference

Contains the main for the real Shadow Dextrous Hand. We start the publishers / subscribers in this node. They all share the same RealShadowhand object, this way the subscriber can update the hand properties, while the publishers publish up to date data. The diagnostics and the other publisher are started in two different threads, to allow them to be published at different frequencies.

```
#include <kdl_parser/kdl_parser.hpp>
#include <ros/ros.h>
#include <kdl/tree.hpp>
#include <boost/thread.hpp>
#include <boost/smart_ptr.hpp>
#include "sr_hand/sr_subscriber.h"
#include "sr_hand/sr_publisher.h"
#include "sr_hand/sr_diagnosticer.h"
#include "sr_hand/hand/real_shadowhand.h"
```

Functions

- void **run_diagnotics** (boost::shared_ptr< SRDiagnosticer > shadowhand_diag)
- void **run_publisher** (boost::shared_ptr< SRPublisher > shadowhand_pub)
- int main (int argc, char **argv)

9.4.1 Detailed Description

Contains the main for the real Shadow Dextrous Hand. We start the publishers / subscribers in this node. They all share the same RealShadowhand object, this way the subscriber can update the hand properties, while the publishers publish up to date data. The diagnostics and the other publisher are started in two different threads, to allow them to be published at different frequencies.

Author

Ugo Cupcic <ugo@ugo-kubuntu.local>

Date

Wed Apr 7 15:37:06 2010

9.4.2 Function Documentation

9.4.2.1 int main (int *argc*, char ** *argv*)

The main function initializes the links with the robot, initializes this ROS subscriber and sets the different callbacks. This ROS subscriber will listen for new commands and send them to the robot.

Parameters

argc

argv

Returns

-1 if error linking with the robot (i.e. robot code not started)

9.5 sr_diagnosticer.cpp File Reference

The goal of this ROS publisher is to publish relevant data concerning the hand at a regular time interval. Those data are (not exhaustive): positions, targets, temperatures, currents, forces, error flags, ...

```
#include <ros/ros.h>
#include <diagnostic_msgs/DiagnosticStatus.h>
#include <diagnostic_msgs/DiagnosticArray.h>
#include <vector>
#include <string>
#include <sstream>
#include <boost/smart_ptr.hpp>
#include "sr_hand/sr_diagnosticer.h"
```

9.5.1 Detailed Description

The goal of this ROS publisher is to publish relevant data concerning the hand at a regular time interval. Those data are (not exhaustive): positions, targets, temperatures, currents, forces, error flags, ...

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Thu Mar 25 15:36:41 2010

9.6 sr_diagnosticer.h File Reference

The Diagnosticer is a ROS publisher which publishes diagnostic data regarding the Dextrous Hand or the Shadow Robot Arm, on the \/diagnostics topic. The diagnostics can be viewed using the robot_monitor package.

```
#include <ros/ros.h>
#include <boost/smart_ptr.hpp>
#include "sr_hand/hand/sr_articulated_robot.h"
```

Data Structures

- class shadowrobot::SRDiagnosticer

Enumerations

- enum **hardware_types** { shadowrobot::sr_hand_hardware, shadowrobot::sr_arm_hardware }

  An enum containing the different types of hardware the diagnosticer is publishing data about.

9.6.1 Detailed Description

The Diagnosticer is a ROS publisher which publishes diagnostic data regarding the Dextrous Hand or the Shadow Robot Arm, on the \/diagnostics topic. The diagnostics can be viewed using the robot_monitor package.

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Tue May 25 17:51:10 2010

To view the diagnostics, just rosmake robot_monitor, and then rosrun robot_monitor robot_monitor.

9.7 sr_publisher.cpp File Reference

The goal of this ROS publisher is to publish relevant data concerning the hand at a regular time interval. Those data are (not exhaustive): positions, targets, temperatures, currents, forces, error flags, ...

```
#include <ros/ros.h>
#include <sr_hand/joints_data.h>
#include <sr_hand/joint.h>
#include <sensor_msgs/JointState.h>
#include <vector>
#include <string>
#include <sstream>
#include "sr_hand/sr_publisher.h"
```

9.7.1 Detailed Description

The goal of this ROS publisher is to publish relevant data concerning the hand at a regular time interval. Those data are (not exhaustive): positions, targets, temperatures, currents, forces, error flags, ...

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Thu Mar 25 15:36:41 2010

9.8 sr_publisher.h File Reference

This class reads and publishes data concerning the shadowhand / shadowarm. To publish those data, just call the publish() function.

```
#include <boost/smart_ptr.hpp>
#include <ros/ros.h>
#include "sr_hand/hand/sr_articulated_robot.h"
```

Data Structures

- class shadowrobot::SRPublisher

9.8.1 Detailed Description

This class reads and publishes data concerning the shadowhand / shadowarm. To publish those data, just call the publish() function.

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Thu Apr 22 10:26:41 2010

9.9 sr_subscriber.cpp File Reference

The role of this ROS subscriber is to receive commands messages (sent by a publisher) and pass them to the hand after translating them to the correct format.

```
#include <iostream>
#include <kdl_parser/kdl_parser.hpp>
#include "sr_hand/sr_subscriber.h"
```

9.9.1 Detailed Description

The role of this ROS subscriber is to receive commands messages (sent by a publisher) and pass them to the hand after translating them to the correct format.

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Thu Mar 25 15:34:37 2010

9.10 sr_subscriber.h File Reference

This ROS subscriber is used to issue commands to the hand / arm, from sending a set of targets, to changing the controller parameters.

```
#include <ros/ros.h>
#include <vector>
#include <string>
#include <map>
#include <boost/smart_ptr.hpp>
#include <kdl/tree.hpp>
#include <kdl/treefksolverpos_recursive.hpp>
#include <kdl/treeiksolvervel_wdls.hpp>
#include <kdl/treeiksolverpos_nr_jl.hpp>
#include <sr_hand/joints_data.h>
#include <sr_hand/joint.h>
#include <sr_hand/contrlr.h>
#include <sr_hand/sendupdate.h>
#include <sr_hand/config.h>
#include <sr_hand/reverseKinematics.h>
#include "sr_hand/hand/sr_articulated_robot.h"
```

Data Structures

- class shadowrobot::SRSubscriber

9.10.1 Detailed Description

This ROS subscriber is used to issue commands to the hand / arm, from sending a set of targets, to changing the controller parameters.

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Thu Apr 22 10:26:41 2010

9.11 sr_tactile_sensor_pub.cpp File Reference

The goal of this ROS publisher is to publish relevant data concerning the hand at a regular time interval. Those data are (not exhaustive): positions, targets, temperatures, currents, forces, error flags, ...

```
#include <ros/ros.h>
#include <shadowhand/joints_data.h>
#include <shadowhand/joint.h>
#include <sensor_msgs/JointState.h>
#include "cybergrasp/cybergraspforces.h"
#include <vector>
#include <string>
#include <sstream>
#include <robot/robot.h>
#include <robot/hand.h>
#include <robot/hand_protocol.h>
#include "sr_hand/sr_tactile_sensor_pub.h"
```

9.11.1 Detailed Description

The goal of this ROS publisher is to publish relevant data concerning the hand at a regular time interval. Those data are (not exhaustive): positions, targets, temperatures, currents, forces, error flags, ...

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Thu Mar 25 15:36:41 2010

9.12 sr_tactile_sensor_pub.h File Reference

This class reads and publishes data concerning the shadowhand. To publish those data, just call the publish() function.

```
#include <ros/ros.h>
```

Data Structures

- class shadowrobot::SRTactileSensorPublisher

9.12.1 Detailed Description

This class reads and publishes data concerning the shadowhand. To publish those data, just call the publish() function.

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Thu Apr 22 10:26:41 2010

9.13 sr_tactile_sensor_pub_node.cpp File Reference

```
#include <ros/ros.h>
#include "sr_hand/sr_tactile_sensor_pub.h"
```

Functions

- int main (int argc, char **argv)

9.13.1 Detailed Description

Author

Ugo Cupcic <ugo@ugo-kubuntu.local>

Date

Wed Apr 7 15:37:06 2010

9.13.2 Function Documentation

9.13.2.1 int main (int *argc*, char ** *argv*)

The main function initializes the links with the robot, initializes this ROS publisher regularly publishes data regarding the finger tips tactile sensors

Parameters

argc

argv

Returns

-1 if error linking with the robot (i.e. robot code not started)

9.14 virtual_arm.cpp File Reference

The virtual arm can be used as a simulator. It modelizes the Shadow Robot muscle arm.

```
#include "sr_hand/hand/virtual_arm.h"
#include <time.h>
#include <ros/ros.h>
```

9.14.1 Detailed Description

The virtual arm can be used as a simulator. It modelizes the Shadow Robot muscle arm.

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Tue Jun 29 14:56:10 2010

9.15 virtual_arm.h File Reference

The virtual arm can be used as a simulator. It modelizes the Shadow Robot muscle arm.

```
#include "sr_hand/hand/sr_articulated_robot.h"
```

Data Structures

- class shadowrobot::VirtualArm

9.15.1 Detailed Description

The virtual arm can be used as a simulator. It modelizes the Shadow Robot muscle arm.

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Tue Jun 29 14:56:10 2010

9.16 virtual_arm_node.cpp File Reference

Contains the main for the virtual arm. We start the publishers / subscribers in this node. They all share the same VirtualArm object, this way the subscriber can update the arm properties, while the publishers publish up to date data. The diagnostics and the other publisher are started in two different threads, to allow them to be published at different frequencies.

```
#include <kdl_parser/kdl_parser.hpp>
#include <ros/ros.h>
#include <kdl/tree.hpp>
#include <boost/thread.hpp>
#include <boost/smart_ptr.hpp>
#include "sr_hand/sr_subscriber.h"
#include "sr_hand/sr_publisher.h"
#include "sr_hand/sr_diagnosticer.h"
#include "sr_hand/hand/virtual_arm.h"
```

Functions

- void **run_diagnotics** (boost::shared_ptr< SRDiagnosticer > shadowhand_diag)
- void **run_publisher** (boost::shared_ptr< SRPublisher > shadowhand_pub)
- int main (int argc, char **argv)

9.16.1 Detailed Description

Contains the main for the virtual arm. We start the publishers / subscribers in this node. They all share the same VirtualArm object, this way the subscriber can update the arm properties, while the publishers publish up to date data. The diagnostics and the other publisher are started in two different threads, to allow them to be published at different frequencies.

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Tue Jun 29 14:56:10 2010

9.16.2 Function Documentation

9.16.2.1 int main (int *argc*, char ** *argv*)

The main function initialises this ROS subscriber and sets the different callbacks. This ROS subscriber will listen for new commands and send them to the virtual robot.

Parameters

argc

argv

Returns

0 on success

9.17 virtual_shadowhand.cpp File Reference

```
#include "sr_hand/hand/virtual_shadowhand.h"
#include <time.h>
#include <ros/ros.h>
```

9.17.1 Detailed Description

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Tue May 25 17:50:42 2010

9.18 virtual_shadowhand.h File Reference

The Virtual Shadowhand can be used as a simulator. As both the real hand and the virtual hand are children from the shadowhand class, using a virtual or a real hand doesn't change anything in the way you call them in your programs.

```
#include "sr_hand/hand/sr_articulated_robot.h"
```

Data Structures

- class shadowrobot::VirtualShadowhand

  The Virtual Shadowhand can be used as a simulator. As both the real hand and the virtual hand are children from the shadowhand class, using a virtual or a real hand doesn't change anything in the way you call them in your programs.

9.18.1 Detailed Description

The Virtual Shadowhand can be used as a simulator. As both the real hand and the virtual hand are children from the shadowhand class, using a virtual or a real hand doesn't change anything in the way you call them in your programs.

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Tue May 25 17:51:10 2010

9.19 virtual_shadowhand_node.cpp File Reference

Contains the main for the virtual hand. We start the publishers / subscribers in this node. They all share the same VirtualHand object, this way the subscriber can update the hand properties, while the publishers publish up to date data. The diagnostics and the other publisher are started in two different threads, to allow them to be published at different frequencies.

```
#include <kdl_parser/kdl_parser.hpp>
#include <ros/ros.h>
#include <kdl/tree.hpp>
#include <boost/thread.hpp>
#include <boost/smart_ptr.hpp>
#include "sr_hand/sr_subscriber.h"
#include "sr_hand/sr_publisher.h"
#include "sr_hand/sr_diagnosticer.h"
#include "sr_hand/hand/virtual_shadowhand.h"
```

Functions

- void **run_diagnotics** (boost::shared_ptr< SRDiagnosticer > shadowhand_diag)
- void **run_publisher** (boost::shared_ptr< SRPublisher > shadowhand_pub)
- int main (int argc, char **argv)

9.19.1 Detailed Description

Contains the main for the virtual hand. We start the publishers / subscribers in this node. They all share the same VirtualHand object, this way the subscriber can update the hand properties, while the publishers publish up to date data. The diagnostics and the other publisher are started in two different threads, to allow them to be published at different frequencies.

Author

Ugo Cupcic <ugo@shadowrobot.com>, Contact <contact@shadowrobot.com>

Date

Wed Apr 7 15:37:06 2010

9.19.2 Function Documentation

9.19.2.1 int main (int *argc*, char ** *argv*)

The main function initializes the links with the robot, initializes this ROS subscriber and sets the different callbacks. This ROS subscriber will listen for new commands and send them to the robot.

Parameters

argc

argv

Returns

0 on success

Chapter 10

Example Documentation

10.1 /examples/shadowhand_publisher.py

This is a python example on how to interact with this ROS subscriber to send commands to the hand.

10.2 /examples/shadowhand_subscriber.py

This is a python example on how to interact with this ROS publisher to read data published by the hand.

10.3 link_joints.cpp

10.4 link_joints.py

Index

~SRArticulatedRobot
    shadowrobot::SRArticulatedRobot, 119
__init__
    grasps_interpoler::GraspInterpoler, 64
    shadowhand_ros::ShadowHand_ROS, 115
    sr_hand::msg::_cartesian_data::cartesian_data, 27
    sr_hand::msg::_cartesian_position::cartesian_-position, 30
    sr_hand::msg::_command::command, 33
    sr_hand::msg::_config::config, 36
    sr_hand::msg::_contrlr::contrlr, 40
    sr_hand::msg::_joint::joint, 67
    sr_hand::msg::_joints_data::joints_data, 73
    sr_hand::msg::_-reverseKinematics::reverseKinematics, 99
    sr_hand::msg::_sendupdate::sendupdate, 102

callback
    link_joints.cpp, 138
    shadowhand_ros::ShadowHand_ROS, 115
callback_arm
    shadowhand_ros::ShadowHand_ROS, 115
callVisualisationService
    shadowhand_ros::ShadowHand_ROS, 115
check_hand_presence
    shadowhand_ros::ShadowHand_ROS, 115
ControllersMap
    shadowrobot::VirtualShadowhand, 132

deserialize
    sr_hand::msg::_cartesian_data::cartesian_data, 27
    sr_hand::msg::_cartesian_position::cartesian_-position, 30
    sr_hand::msg::_command::command, 33
    sr_hand::msg::_config::config, 36
    sr_hand::msg::_contrlr::contrlr, 40
    sr_hand::msg::_joint::joint, 67
    sr_hand::msg::_joints_data::joints_data, 73
    sr_hand::msg::_-reverseKinematics::reverseKinematics, 99
    sr_hand::msg::_sendupdate::sendupdate, 102
deserialize_numpy
    sr_hand::msg::_cartesian_data::cartesian_data, 27
    sr_hand::msg::_cartesian_position::cartesian_-position, 30
    sr_hand::msg::_command::command, 33
    sr_hand::msg::_config::config, 36
    sr_hand::msg::_contrlr::contrlr, 40
    sr_hand::msg::_joint::joint, 67
    sr_hand::msg::_joints_data::joints_data, 73
    sr_hand::msg::_-reverseKinematics::reverseKinematics, 99
    sr_hand::msg::_sendupdate::sendupdate, 102

getAllJointsData
    shadowrobot::RealShadowhand, 95
    shadowrobot::SRArticulatedRobot, 119
    shadowrobot::VirtualArm, 127
    shadowrobot::VirtualShadowhand, 132
getConfig
    shadowrobot::RealShadowhand, 96
    shadowrobot::SRArticulatedRobot, 119
    shadowrobot::VirtualArm, 127
    shadowrobot::VirtualShadowhand, 132
getContrl
    shadowrobot::RealShadowhand, 96
    shadowrobot::SRArticulatedRobot, 120
    shadowrobot::VirtualArm, 128
    shadowrobot::VirtualShadowhand, 132
getDiagnostics
    shadowrobot::RealShadowhand, 96
    shadowrobot::SRArticulatedRobot, 120
    shadowrobot::VirtualArm, 128
    shadowrobot::VirtualShadowhand, 133
getJointData
    shadowrobot::RealShadowhand, 96
    shadowrobot::SRArticulatedRobot, 120
    shadowrobot::VirtualArm, 128
    shadowrobot::VirtualShadowhand, 133
Grasp::Grasp, 63
grasps_interpoler::GraspInterpoler, 64
    __init__, 64
    set_percentage_grasp1_to_grasp2, 64
grasps_parser::GraspParser, 65

parse_tree, 65

has_arm
shadowhand_ros::ShadowHand_ROS, 115

init_actual_joints
shadowhand_ros::ShadowHand_ROS, 115
initializeMap
shadowrobot::RealShadowhand, 97
shadowrobot::VirtualArm, 128
shadowrobot::VirtualShadowhand, 133

JointsMap
shadowrobot::SRArticulatedRobot, 119

link_joints.cpp, 137
callback, 138
main, 138

main
link_joints.cpp, 138
real_shadowhand_node.cpp, 141
sr_tactile_sensor_pub_node.cpp, 151
virtual_arm_node.cpp, 154
virtual_shadowhand_node.cpp, 158

ParametersMap
shadowrobot::SRArticulatedRobot, 119
parse_tree
grasps_parser::GraspParser, 65
publish
shadowrobot::SRDiagnosticer, 122
shadowrobot::SRPublisher, 124

read_all_current_arm_positions
shadowhand_ros::ShadowHand_ROS, 115
read_all_current_arm_targets
shadowhand_ros::ShadowHand_ROS, 116
read_all_current_positions
shadowhand_ros::ShadowHand_ROS, 116
read_all_current_targets
shadowhand_ros::ShadowHand_ROS, 116
real_shadowhand.cpp, 139
real_shadowhand.h, 140
real_shadowhand_node.cpp, 141
main, 141
RealShadowhand
shadowrobot::RealShadowhand, 95
record_step_to_file
shadowhand_ros::ShadowHand_ROS, 116
resend_targets
shadowhand_ros::ShadowHand_ROS, 116
ros::message_operations::Printer< ::sr_hand::cartesian_data_< ContainerAllocator > >, 86
ros::message_operations::Printer< ::sr_hand::cartesian_position_< ContainerAllocator > >, 87
ros::message_operations::Printer< ::sr_hand::command_< ContainerAllocator > >, 88
ros::message_operations::Printer< ::sr_hand::config_< ContainerAllocator > >, 89
ros::message_operations::Printer< ::sr_hand::contrlr_< ContainerAllocator > >, 90
ros::message_operations::Printer< ::sr_hand::joint_< ContainerAllocator > >, 91
ros::message_operations::Printer< ::sr_hand::joints_data_< ContainerAllocator > >, 92
ros::message_operations::Printer< ::sr_hand::reverseKinematics_< ContainerAllocator > >, 93
ros::message_operations::Printer< ::sr_hand::sendupdate_< ContainerAllocator > >, 94
ros::message_traits::DataType< ::sr_hand::cartesian_data_< ContainerAllocator > >, 44
ros::message_traits::DataType< ::sr_hand::cartesian_position_< ContainerAllocator > >, 45
ros::message_traits::DataType< ::sr_hand::command_< ContainerAllocator > >, 46
ros::message_traits::DataType< ::sr_hand::config_< ContainerAllocator > >, 47
ros::message_traits::DataType< ::sr_hand::contrlr_< ContainerAllocator > >, 48
ros::message_traits::DataType< ::sr_hand::joint_< ContainerAllocator > >, 49
ros::message_traits::DataType< ::sr_hand::joints_data_< ContainerAllocator > >, 50
ros::message_traits::DataType< ::sr_hand::reverseKinematics_< ContainerAllocator > >, 51
ros::message_traits::DataType< ::sr_hand::sendupdate_< ContainerAllocator > >, 52
ros::message_traits::Definition< ::sr_hand::cartesian_data_< ContainerAllocator > >, 53
ros::message_traits::Definition< ::sr_hand::cartesian_position_< ContainerAllocator > >, 54

ros::message_traits::Definition< ::sr_hand::command_< ContainerAllocator > >, 55
ros::message_traits::Definition< ::sr_hand::config_< ContainerAllocator > >, 56
ros::message_traits::Definition< ::sr_hand::contrlr_< ContainerAllocator > >, 57
ros::message_traits::Definition< ::sr_hand::joint_< ContainerAllocator > >, 58
ros::message_traits::Definition< ::sr_hand::joints_data_< ContainerAllocator > >, 59
ros::message_traits::Definition< ::sr_hand::reverseKinematics_< ContainerAllocator > >, 60
ros::message_traits::Definition< ::sr_hand::sendupdate_< ContainerAllocator > >, 61
ros::message_traits::MD5Sum< ::sr_hand::cartesian_data_< ContainerAllocator > >, 76
ros::message_traits::MD5Sum< ::sr_hand::cartesian_position_< ContainerAllocator > >, 77
ros::message_traits::MD5Sum< ::sr_hand::command_< ContainerAllocator > >, 78
ros::message_traits::MD5Sum< ::sr_hand::config_< ContainerAllocator > >, 79
ros::message_traits::MD5Sum< ::sr_hand::contrlr_< ContainerAllocator > >, 80
ros::message_traits::MD5Sum< ::sr_hand::joint_< ContainerAllocator > >, 81
ros::message_traits::MD5Sum< ::sr_hand::joints_data_< ContainerAllocator > >, 82
ros::message_traits::MD5Sum< ::sr_hand::reverseKinematics_< ContainerAllocator > >, 83
ros::message_traits::MD5Sum< ::sr_hand::sendupdate_< ContainerAllocator > >, 84
ros::serialization::Serializer< ::sr_hand::cartesian_data_< ContainerAllocator > >, 105
ros::serialization::Serializer< ::sr_hand::cartesian_position_< ContainerAllocator > >, 106
ros::serialization::Serializer< ::sr_hand::command_< ContainerAllocator > >, 107
ros::serialization::Serializer< ::sr_hand::config_< ContainerAllocator > >, 108
ros::serialization::Serializer< ::sr_hand::contrlr_< ContainerAllocator > >, 109
ros::serialization::Serializer< ::sr_hand::joint_< ContainerAllocator > >, 110
ros::serialization::Serializer< ::sr_hand::joints_data_< ContainerAllocator > >, 111
ros::serialization::Serializer< ::sr_hand::reverseKinematics_< ContainerAllocator > >, 112
ros::serialization::Serializer< ::sr_hand::sendupdate_< ContainerAllocator > >, 113

sendupdate
  shadowhand_ros::ShadowHand_ROS, 116
  shadowrobot::RealShadowhand, 97
  shadowrobot::SRArticulatedRobot, 120
  shadowrobot::VirtualArm, 129
  shadowrobot::VirtualShadowhand, 133
sendupdate_arm
  shadowhand_ros::ShadowHand_ROS, 116
sendupdate_arm_from_dict
  shadowhand_ros::ShadowHand_ROS, 116
sendupdate_from_dict
  shadowhand_ros::ShadowHand_ROS, 117
serialize
  sr_hand::msg::_cartesian_data::cartesian_data, 28
  sr_hand::msg::_cartesian_position::cartesian_position, 30
  sr_hand::msg::_command::command, 33
  sr_hand::msg::_config::config, 36
  sr_hand::msg::_contrlr::contrlr, 40
  sr_hand::msg::_joint::joint, 67
  sr_hand::msg::_joints_data::joints_data, 73
  sr_hand::msg::_reverseKinematics::reverseKinematics, 99
  sr_hand::msg::_sendupdate::sendupdate, 102
serialize_numpy
  sr_hand::msg::_cartesian_data::cartesian_data, 28
  sr_hand::msg::_cartesian_position::cartesian_position, 31
  sr_hand::msg::_command::command, 33
  sr_hand::msg::_config::config, 36
  sr_hand::msg::_contrlr::contrlr, 40
  sr_hand::msg::_joint::joint, 68
  sr_hand::msg::_joints_data::joints_data, 73
  sr_hand::msg::_reverseKinematics::reverseKinematics, 99
  sr_hand::msg::_sendupdate::sendupdate, 102
set_percentage_grasp1_to_grasp2

grasps_interpoler::GraspInterpoler, 64
set_sendupdate_topic
shadowhand_ros::ShadowHand_ROS, 117
set_shadowhand_data_topic
shadowhand_ros::ShadowHand_ROS, 117
setConfig
shadowrobot::RealShadowhand, 97
shadowrobot::SRArticulatedRobot, 121
shadowrobot::VirtualArm, 129
shadowrobot::VirtualShadowhand, 134
setContrl
shadowrobot::RealShadowhand, 97
shadowrobot::SRArticulatedRobot, 121
shadowrobot::VirtualArm, 129
shadowrobot::VirtualShadowhand, 134
shadowhand_ros::Joint, 66
shadowhand_ros::ShadowHand_ROS, 114
__init__, 115
callback, 115
callback_arm, 115
callVisualisationService, 115
check_hand_presence, 115
has_arm, 115
init_actual_joints, 115
read_all_current_arm_positions, 115
read_all_current_arm_targets, 116
read_all_current_positions, 116
read_all_current_targets, 116
record_step_to_file, 116
resend_targets, 116
sendupdate, 116
sendupdate_arm, 116
sendupdate_arm_from_dict, 116
sendupdate_from_dict, 117
set_sendupdate_topic, 117
set_shadowhand_data_topic, 117
valueof, 117
shadowrobot::DiagnosticData, 62
shadowrobot::JointControllerData, 71
shadowrobot::JointData, 72
shadowrobot::Parameters, 85
shadowrobot::RealShadowhand, 95
getAllJointsData, 95
getConfig, 96
getContrl, 96
getDiagnostics, 96
getJointData, 96
initializeMap, 97
RealShadowhand, 95
sendupdate, 97
setConfig, 97
setContrl, 97
shadowrobot::SRArticulatedRobot, 118
~SRArticulatedRobot, 119
getAllJointsData, 119
getConfig, 119
getContrl, 120
getDiagnostics, 120
getJointData, 120
JointsMap, 119
ParametersMap, 119
sendupdate, 120
setConfig, 121
setContrl, 121
SRArticulatedRobot, 119
shadowrobot::SRDiagnosticer, 122
publish, 122
SRDiagnosticer, 122
shadowrobot::SRPublisher, 124
publish, 124
SRPublisher, 124
shadowrobot::SRSubscriber, 125
SRSubscriber, 125
shadowrobot::SRTactileSensorPublisher, 126
shadowrobot::VirtualArm, 127
getAllJointsData, 127
getConfig, 127
getContrl, 128
getDiagnostics, 128
getJointData, 128
initializeMap, 128
sendupdate, 129
setConfig, 129
setContrl, 129
toDegrees, 129
toRad, 130
VirtualArm, 127
shadowrobot::VirtualShadowhand, 131
ControllersMap, 132
getAllJointsData, 132
getConfig, 132
getContrl, 132
getDiagnostics, 133
getJointData, 133
initializeMap, 133
sendupdate, 133
setConfig, 134
setContrl, 134
toDegrees, 134
toRad, 134
VirtualShadowhand, 132
sr_diagnosticer.cpp, 143
sr_diagnosticer.h, 144
sr_hand::cartesian_data_, 29
sr_hand::cartesian_position_, 32
sr_hand::command_, 35
sr_hand::config_, 38
sr_hand::contrlr_, 42

sr_hand::joint_, 69
sr_hand::joints_data_, 75
sr_hand::msg::_cartesian_data, 17
sr_hand::msg::_cartesian_data::cartesian_data, 27
__init__, 27
deserialize, 27
deserialize_numpy, 27
serialize, 28
serialize_numpy, 28
sr_hand::msg::_cartesian_position, 18
sr_hand::msg::_cartesian_position::cartesian_-position, 30
__init__, 30
deserialize, 30
deserialize_numpy, 30
serialize, 30
serialize_numpy, 31
sr_hand::msg::_command, 19
sr_hand::msg::_command::command, 33
__init__, 33
deserialize, 33
deserialize_numpy, 33
serialize, 33
serialize_numpy, 33
sr_hand::msg::_config, 20
sr_hand::msg::_config::config, 36
__init__, 36
deserialize, 36
deserialize_numpy, 36
serialize, 36
serialize_numpy, 36
sr_hand::msg::_contrlr, 21
sr_hand::msg::_contrlr::contrlr, 40
__init__, 40
deserialize, 40
deserialize_numpy, 40
serialize, 40
serialize_numpy, 40
sr_hand::msg::_joint, 22
sr_hand::msg::_joint::joint, 67
__init__, 67
deserialize, 67
deserialize_numpy, 67
serialize, 67
serialize_numpy, 68
sr_hand::msg::_joints_data, 23
sr_hand::msg::_joints_data::joints_data, 73
__init__, 73
deserialize, 73
deserialize_numpy, 73
serialize, 73
serialize_numpy, 73
sr_hand::msg::_reverseKinematics, 24
sr_hand::msg::_reverseKinematics::reverseKinematics, 99
__init__, 99
deserialize, 99
deserialize_numpy, 99
serialize, 99
serialize_numpy, 99
sr_hand::msg::_sendupdate, 25
sr_hand::msg::_sendupdate::sendupdate, 102
__init__, 102
deserialize, 102
deserialize_numpy, 102
serialize, 102
serialize_numpy, 102
sr_hand::reverseKinematics_, 101
sr_hand::sendupdate_, 104
sr_publisher.cpp, 145
sr_publisher.h, 146
sr_subscriber.cpp, 147
sr_subscriber.h, 148
sr_tactile_sensor_pub.cpp, 149
sr_tactile_sensor_pub.h, 150
sr_tactile_sensor_pub_node.cpp, 151
main, 151
SRArticulatedRobot
shadowrobot::SRArticulatedRobot, 119
SRDiagnosticer
shadowrobot::SRDiagnosticer, 122
SRPublisher
shadowrobot::SRPublisher, 124
SRSubscriber
shadowrobot::SRSubscriber, 125

test::SRLibrary, 123
toDegrees
shadowrobot::VirtualArm, 129
shadowrobot::VirtualShadowhand, 134
toRad
shadowrobot::VirtualArm, 130
shadowrobot::VirtualShadowhand, 134

valueof
shadowhand_ros::ShadowHand_ROS, 117
virtual_arm.cpp, 152
virtual_arm.h, 153
virtual_arm_node.cpp, 154
main, 154
virtual_shadowhand.cpp, 156
virtual_shadowhand.h, 157
virtual_shadowhand_node.cpp, 158
main, 158
VirtualArm
shadowrobot::VirtualArm, 127
VirtualShadowhand

shadowrobot::VirtualShadowhand, 132