Shadow Robot Documentation

Release 1.4.0

Ugo Cupcic

Feb 06, 2019

Contents

To get the latest version of our software, you can install it from source. We're doing our best to keep the indigo-devel branch stable.

CHAPTER 1

Workspaces

We created a one liner script that installs everything for you: - ros and dependencies - creates the proper workspaces hierarchy (we are pulling a few dependencies from source) - compiles everything

You simply need to run (replace `-w ~{{ros_user}}/projects/shadow_robot/base` with the path you want to install the sources in, make sure you keep the `~{{ros_user}}` if you want the project in your home folder or that you point to a directory with write permission for your user):

```
curl -L bit.ly/dev-machine | bash -s -- -w ~{{ros_user}}/projects/shadow_robot/base
```

The output is quite verbose:

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   177  100   177    0     0    465      0 --:--:-- --:--:-- --:--:--   465
100  4146  100  4146    0     0   6281      0 --:--:-- --:--:-- --:--:--  6281
================================================================
|       Installing Shadow Robot development environment
```

[...]

```
--------------------
 | Running Ansible |
 --------------------

sudo password:


PLAY [ros-hydro-desktop-precise64] ********************************************
skipping: no hosts match
```

[...]

```
PLAY RECAP ********************************************************************
changed: [localhost] => {"changed": true, "cmd": "usermod -a -G dialout 'ugo' ",
 ↪"delta": "0:00:00.009358", "end": "2016-01-04 08:33:52.390686", "item": "", "rc": 0,
 ↪ "start": "2016-01-04 08:33:52.381328", "stderr": "", "stdout": ""}
```

(continues on next page)

(continued from previous page)

```
localhost                  : ok=38   changed=26   unreachable=0    failed=0


 ------------------------------------------------
 | Install complete, please restart the machine |
 ------------------------------------------------
```

CHAPTER 2

Updating your workspace

For a quick update of the main workspace, we added an alias: simply run `ws_update` from the console to get the latest source code.

To fully update the different workspaces (both the shadow code and the dependencies), you can rerun the install script. **WARNING: If you use the same workspace path, it'll completely overwrite it**

CHAPTER 3

Installing for a real robot

3.1 Configuration

If you're installing the code for a real robot, you simply need to add an option to the line above to pull the proper `sr-config` branch. `sr-config` contains the parameters specific to your hand (calibration, controller tuning etc...).

You can check which branch is installed on the computer provided by Shadow by running (on the machine provided with your hand):

```
roscd sr_ethercat_hand_config
git branch
```

The highlighted branch is the one that is currently used. Let's assume it's `shadowrobot_1234` for the following instructions.

3.2 ROS Indigo

On the newly installed computer with Ubuntu Trusty you will need to pull the same configuration branch:

```
curl -L bit.ly/dev-machine | bash -s -- -w ~/projects/shadow_robot/base -c␣
↪shadowrobot_1234
```

3.3 ROS Kinetic

Please install Ubuntu Xenial and use the following command for ROS Kinetic:

```
bash <(curl -Ls https://raw.githubusercontent.com/shadow-robot/sr-build-tools/master/
↪ansible/deploy.sh) -r sr-build-tools -b master -i data/shadow_robot-kinetic.
↪rosinstall -v kinetic -t mongodb,pyassimp,hand_e_icons - shadowrobot_1234
```

After successfull command execution please run:

```
echo 'source $HOME/workspace/shadow_robot-kinetic/base/devel/setup.bash' >> ~/.bashrc
```

3.4 Notice

Note: the etherCAT configuration has evolved quite a bit in the latest years. If the config is not working for you, get in touch and we'll help you migrate the configuration.

CHAPTER 4

Installing for a real robot using Docker container

4.1 Docker installation

Your machine should have Ubuntu Trusty or Xenial installed.
Please install Docker using the following

instructions.

4.2 ROS Indigo Docker container

Pull ROS Indigo container

```
docker pull shadowrobot/dexterous-hand:indigo
```

Container created via

```
docker run -it --privileged --name hand_e_indigo_real_hw --network=host -e DISPLAY -e␣
↪QT_X11_NO_MITSHM=1 -e LOCAL_USER_ID=$(id -u) -v /tmp/.X11-unix:/tmp/.X11-unix:rw␣
↪shadowrobot/dexterous-hand:indigo
```

(you don't need to run "docker run" every time, as the container is persistent)

To start the container again please execute

```
docker start hand_e_indigo_real_hw
```

4.3 ROS Kinetic Docker container

Pull ROS Kinetic container

```
docker pull shadowrobot/dexterous-hand:kinetic
```

Container created via

```
docker run -it --privileged --name hand_e_kinetic_real_hw --network=host -e DISPLAY -
↪e QT_X11_NO_MITSHM=1 -e LOCAL_USER_ID=$(id -u) -v /tmp/.X11-unix:/tmp/.X11-unix:rw␣
↪shadowrobot/dexterous-hand:kinetic
```

(you don't need to run "docker run" every time, as the container is persistent)

To start the container again please execute

```
docker start hand_e_kinetic_real_hw
```

4.4 Configuration

If you're installing the code for a real robot, you simply need to pull the proper `sr-config` branch. `sr-config` contains the parameters specific to your hand (calibration, controller tuning etc...).

You can check which branch is installed on the computer provided by Shadow by running (on the machine provided with your hand):

```
roscd sr_ethercat_hand_config
git branch
```

The highlighted branch is the one that is currently used. Let's assume it's `shadowrobot_1234` for the following instructions.

In Docker container's console window type the following commands

```
roscd sr_ethercat_hand_config
git fetch
git checkout shadowrobot_1234
```

Now you are ready to use Docker container with your hand.

4.5 Launch

Launch the right hand in PWM mode (safe in case of uncalibrated hand or untested sensors)

```
roslaunch sr_ethercat_hand_config sr_rhand.launch
```

To close the container use CTRL-d or

```
exit
```

CHAPTER 5

Starting the robots (simulated / real)

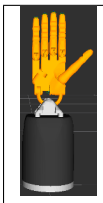
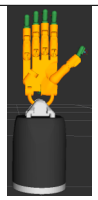
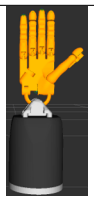
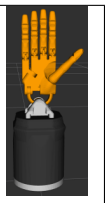
5.1 Shadow hand only

5.1.1 Simulation

To start our hand, simply do:

```
roslaunch sr_robot_launch srhand.launch
```

This will launch the five finger hand (shadowhand_motor) by default . If you want to launch another hand, these are the hands available:

										
1	2	3	4	5	6	7	8	9	10	11

	Right	Left
1	shadowhand_motor.urdf.xacro	shadowhand_left_motor.urdf.xacro
2	shadowhand_motor_biotac.urdf.xacro	shadowhand_left_motor_biotac.urdf.xacro
3	shadowhand_motor_ff_biotac.urdf.xacro	
4	shadowhand_motor_btsp.urdf.xacro	
5	shadowhand_motor_ellipsoid.urdf.xacro	
6	shadowhand_motor_th_ff_rf_ellipsoid.urdf.xacro	
7	shadowhand_muscle.urdf.xacro	shadowhand_left_muscle.urdf.xacro
8	shadowhand_muscle_biotac.urdf.xacro	
9	shadowhand_lite.urdf.xacro	
10	shadowhand_extra_lite.urdf.xacro	
11	shadowhand_motor_plus.urdf.xacro	shadowhand_left_motor_plus.urdf.xacro

To start the simulation of a shadow hand, you can run:

```
roslaunch sr_robot_launch srhand.launch robot_description:=`rospack find sr_
↪description`/robots/shadowhand_motor.urdf.xacro
```

- The `robot description` param can be changed to start any of the available Shadow hands shown in the table.
- If it is a left hand, `hand_id:=lh` should be added. For example:

```
roslaunch sr_robot_launch srhand.launch robot_description:=`rospack find sr_
↪description`/robots/shadowhand_left_motor.urdf.xacro hand_id:=lh
```

- Moveit will enable advanced behaviour (inverse kinematics, planning, collision detectection, etc. . .), but if it is not needed, you can set `use_moveit:=false`
- To start the dexterous hand plus (shadowhand_motor_plus.urdf.xacro), you can add the hand_type like this:

```
roslaunch sr_robot_launch srhand.launch hand_type:=hand_e_plus
```

5.1.2 Real hand

To start a real hand, you can run:

```
roslaunch sr_ethercat_hand_config sr_rhand.launch
```

It has the specific configuration to launch your hand, including the ethernet port, the hand serial and robot description.

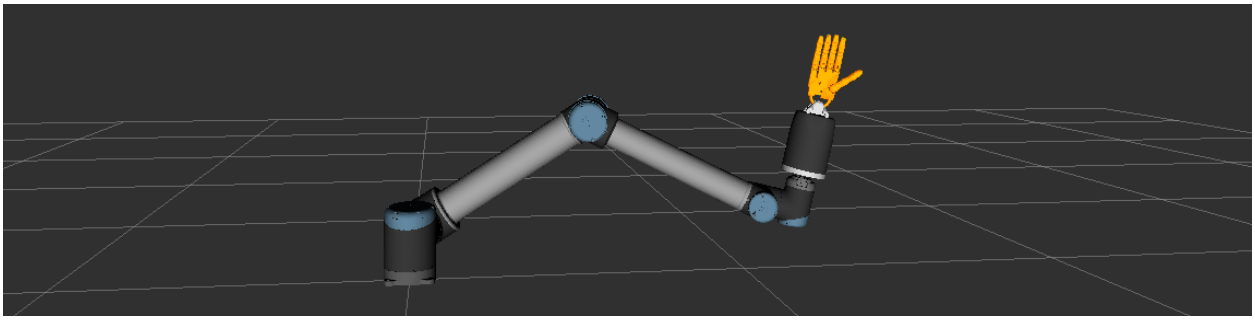
5.2 Shadow hand with UR10 arm

5.2.1 Simulation

To start the simulation of the hand and arm, you can run:

```
roslaunch sr_robot_launch sr_right_ur10arm_hand.launch
```

or, for the left hand and arm



```
roslaunch sr_robot_launch sr_left_ur10arm_hand.launch
```

5.2.2 Real Robots

To start the real robots, do:

```
roslaunch sr_robot_launch sr_right_ur10arm_hand.launch sim:=false hand_serial:=1178
```

or, for the left hand and arm

```
roslaunch sr_robot_launch sr_left_ur10arm_hand.launch sim:=false hand_serial:=1178
```

To find the hand serial you can launch the command without the `hand_serial` argument and then check the program output. You should see something like:

```
Trying to read mapping for: /hand/mapping/1178
```

In this case 1178 is the serial number of the hand.

To change the hand mapping, you can set the `mapping_path` argument. For example adding:

```
mapping_path:=`rospack find sr_edc_launch`/mappings/default_mappings/rh_E_v3.yaml
```

To change the ethernet port used for your hand, you can add the `eth_port` argument, such as:

```
eth_port:=eth6
```

Real Robots, using the normal (not limited) joint range

By default the URDF used for the UR10 arm uses a limited range for the joints, as that helps moveit find a planning solution. But as that restricts the robot movements, the user might want to start the robots with the full joint range. To do that:

```
roslaunch sr_robot_launch sr_right_ur10arm_hand.launch sim:=false robot_
↪description:=`rospack find sr_multi_description`/urdf/right_srhand_ur10.urdf.xacro␣
↪hand_serial:=1178
```

or, for the left hand and arm

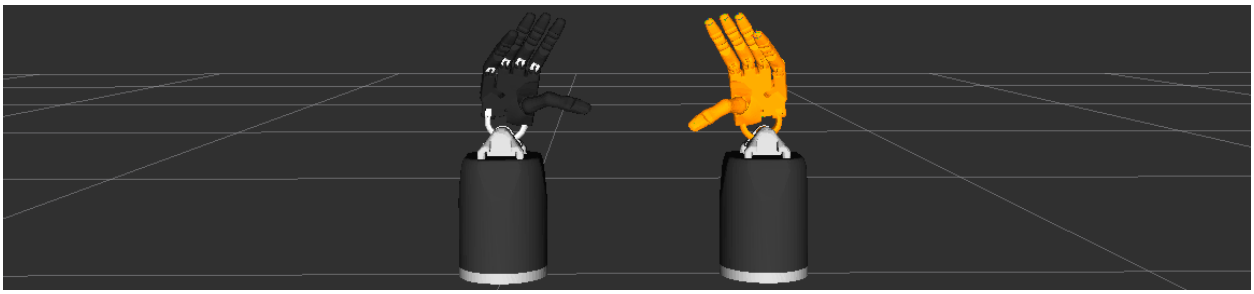
```
roslaunch sr_robot_launch sr_left_ur10arm_hand.launch sim:=false robot_
↪description:=`rospack find sr_multi_description`/urdf/left_srhand_ur10.urdf.xacro␣
↪hand_serial:=1178
```

Hand with tactile sensors

If your hand has biotacs sensors, simply append `_biotacs` to the `robot_description:=` and to the `robot_config_file:=` as seen below:

```
robot_description:=`rospack find sr_multi_description`/urdf/right_srhand_ur10_joint_
↪limited_biotacs.urdf.xacro robot_config_file:=`rospack find sr_multi_moveit_config`/
↪config/robot_configs/right_sh_ur10_biotac.yaml
```

5.3 Bimanual system



5.3.1 Simulation

To start the simulation of a bimanual system, you can run:

```
roslaunch sr_robot_launch sr_bimanual.launch use_moveit:=true
```

5.3.2 Real Robots

To start the real robots, do:

```
roslaunch sr_robot_launch sr_bimanual.launch sim:=false rh_serial:=1290 lh_
↪serial:=1338
```

CHAPTER 6

High Level Interface

This package contains libraries that provide a high level interface to easily control the different robots supported by Shadow Robot. They encapsulate the functionality provided by different ROS packages, specially the moveit_commander, enabling their access throughout a more simplified interface.

The following classes can be found in this package

- SrRobotCommander - base class
- SrArmCommander - arm management class
- SrHandCommander - hand management class

In order to access the full functionality of the hand, you can also interface with it directly through the ROS interface.

CHAPTER 7

Examples

We are providing a high level wrapper around our robots to make the interfacing easier for non-ROS people. The sr_example package contains examples on how to use it.

Use cases are given for:

- Hand
- Arm
- Hand and arm

To run these examples, the robot should be launched first (instructions for this can be found here)

In order to access the full functionality of the hand, you can interface with it directly through the ROS interface. You can find python examples in the advanced folder and c++ examples in the src folder.

Description of each example can be found in the comment at the top of each script.

CHAPTER 8

Shadow Robot's Documentation

Note: This documentation is deprecated. Please refer to the following link for the most up to date documentation of our Dexterous Hand.

8.1 Overview

This is the starting point for our core software packages built around our robots. We'll focus more on the simulated side of things here (as you'll get a full training when receiving one of our robots), but all the code that runs on our simulated robot also runs on the real hardware.

Our code is split into different repositories:

- sr_common: This repository contains the bare minimum for communicating with the Shadow Hand from a remote computer (urdf models and messages).
- sr_core: These are the core packages for the Shadow Robot hardware and simulation.
- sr_interface: This repository contains the high level interface and its dependencies for interacting simply with our robots.
- sr_tools: This repository contains more advanced tools that might be needed in specific use cases.
- sr_visualization: This repository contains the various rqt_gui plugins we developed.
- sr_config: This repository contains the customer specific configuration for the Shadow Robot Hand.

8.2 Getting Started

8.2.1 Installation Instructions

To install our software please take a look at these install steps.

8.2.2 Running the software

You can start our different robots (one hand - full hand, Lite, Extra Lite, Left, Right, . . . -, two hands, hand and UR10, etc. . .) in both simulation and with the real hardware using the launch files in sr_robot_launch

8.2.3 Connecting to the robots

Depending on your level of expertise with ROS and our software stack, as well as the use case there are two main ways to interface with our hand:

- for a high level, low expertise access to our hand, please refer to the robot commander interface.
- for power users, you can obviously access the different ROS interfaces directly.