

EtherCAT Dextrous Hand

- User Manual -

Shadow Software team - software@shadowrobot.com

June 22, 2012



Contents

1	Overview	3
2	Installing our Software	4
2.1	From our apt repository (recommended)	4
2.1.1	Install a sr_config overlay for the EtherCAT Hand	4
2.2	From sources	4
2.2.1	Available rosininstall files	5
2.3	Import our custom Environment Variables	6
3	Navigating our code	7
3.1	Important Packages	7
3.2	Where to find the settings	7
3.2.1	Specifying system parameters	7
3.2.2	Specifying default controllers settings	7
4	Running our code	8
4.1	Starting the real hand	8
4.1.1	Setting up the Ethernet port	8
4.1.2	Starting the driver	8
4.2	Starting the simulated robot	9
4.3	Checking it works	9
5	Where to go next	10
6	What to do if it doesn't work	11

6.1	Check your ROS setup	11
6.2	Use the ROS community	11
6.3	Contact us	11
7	Changelog	12
7.1	Shadow Robot Stack	12

1 Overview

This user manual will get you started with using our EtherCAT hand or simulated robots. We'll guide you through installing the code in the **section 2**, show you some basic commands and code samples in **section 4**, and explain where to find things in **section 3**. This guide ends with some pointers to essential documentation you should read (**section 5**), and some troubleshooting tips (**section 6**).

If you have any questions, don't hesitate to contact us at: `software@shadowrobot.com`.

2 Installing our Software

For a typical user, we really recommend installing our software directly from the apt repository (see section 2.1). This will ensure that you keep the latest released software on your computer.

We assume you’ve already installed ROS electric, as detailed on [the ROS Wiki](#).

However if you plan to modify some parts of our code, you can use an overlay to install some of the sources from our publicly available Launchpad repository on top of your existing installation.

2.1 From our apt repository (recommended)

TODO: Mark fill this in!

2.1.1 Install a `sr_config` overlay for the EtherCAT Hand

If you have an EtherCAT Hand, you should create an overlay for the `sr_config` stack: this stack contains all the different configuration files which can be modified to reflect your specific system. You shouldn’t use the one installed by default as they are defined for the simulated hand.

To create this overlay, just follow the **section 2.2**.

2.2 From sources

If you need to modify one of the packages for your own use, you’ll want to first install the released version from the apt repository as explained in the **section 2.1**. This will ensure that you have all the code you need. Once you’ve identified in which stack the package you want to modify is, you can create an overlay¹ to use the source version.

To create an overlay, we’ll use the [rosinstall](#) utility. In this example, we’ll overlay the `sr_config` stack, using trunk version of the code hosted on launchpad using `bzr:lp:sr-config`.

- First we need to install the `rosinstall` utility if you haven’t done so already:

¹**Overlay:** use another version of a stack than the default installed one.

```
> sudo apt-get install python-setuptools bzip2
> sudo easy_install -U rosininstall vcstools
```

- We can now create a rosininstall file that will contain the information about the stacks we want to overlay (in our case we're overlaying the sr_config stack only):

shadow_robot.rosinstall

```
- bzip2: # VCS used (bzip2 in our case)
  uri: 'lp:sr-config' # public uri for the code
  local-name: sr_config # name under which we install the stack
```

- We're now going to create a workspace (in the ~/workspace directory), based on your currently installed ROS system (we're assuming you're using ROS electric), and use the rosininstall file we created to download the code into it.

```
> rosininstall ~/workspace /opt/ros/electric/setup.bash
> rosininstall ~/workspace shadow_robot.rosinstall
```

- To use the newly created environment, simply source the setup.bash created in the previous step (to have it sourced automatically when you open a new terminal, simply add it to your ~/.bashrc as show on the second line).

```
> source ~/workspace/setup.bash
> echo source ~/workspace/setup.bash >> ~/.bashrc
```

- Finally, you can build the whole stack in one go using the **rosmake** command:

```
> rosmake --rosdep-install --rosdep-yes sr_config
```

2.2.1 Available rosininstall files

If you want to install all our stacks in one go, you can use the **rosinstall** files available [here](#).

TODO: Mark fill this in! May be we should have a simpler shadow_robot.rosinstall which doesn't contain the unstable stacks (sr-demo / denso / object-manip / sr-manip)?

Once you've downloaded the code using the **rosinstall** commands described previously, you can build all our stacks:

```
> rosmake --rosdep-install --rosdep-yes shadow_robot \
  shadow_robot_ethercat sr_contrib sr_visualization \
  sr_teleop
```

2.3 Import our custom Environment Variables

As described in **section 3.2.1**, we're using a file to define some useful environment variables. To make sure the variables are imported in your setup, you need to source it in your `~/.bashrc`, after sourcing the `ros setup.bash`. It should look like this:

`~/.bashrc`

```
#source the ros setup.bash
/opt/ros/electric/setup.bash
#or the one created in your workspace
# if you used an overlay
#source ~/workspace/setup.bash

#now source our additional environment variables
source `rosstack find sr_config`/bashrc/env_variables.bashrc
```

3 Navigating our code

3.1 Important Packages

3.2 Where to find the settings

3.2.1 Specifying system parameters

3.2.2 Specifying default controllers settings

4 Running our code

4.1 Starting the real hand

4.1.1 Setting up the Ethernet port

The first time you plug an EtherCAT Hand, you probably need to check on which Ethernet port the hand is connected (you probably have 2 network ports in your computer, one of them would typically be connected to the network, the other one to the hand).

An easy way to see which port the hand is using is to unplug the hand network cable from your computer and then plug it back in. You can now type `dmesg` and you should see a line saying which port has just been plugged in, for example, in the lines below, you see that `eth1` is used by the hand:

```
[16661.901920] e1000e: eth1 NIC Link is Down
[16666.460518] e1000e: eth1 NIC Link is Up 1000 Mbps Full Duplex
```

Now that we know which Ethernet port is used by the hand (`eth1` in our example), we can edit the interface in the system parameters, as shown in **section 3.2.1**:

- Make sure you've created your overlay for the `sr_config` stack as described in the **section 2.1.1**.
- Let's open the configuration file:

```
> roscd sr_config
> cd bashrc
> gedit env_variables.bashrc
```

- Set the `ETHERCAT_PORT` to the correct value (it is `eth1` by default)
- Save and restart your terminal to make sure the new options have been taken into account.

4.1.2 Starting the driver

You can now start the driver. To be able to run the driver, you need to elevate the permissions to root access:

```
#elevate permissions
> sudo -s
> roslaunch sr_edc_launch sr_edc.launch
```

4.2 Starting the simulated robot

To start the simulated robot you can simply run:

```
> roslaunch sr_hand gazebo_hand_motor.launch
```

Or if you want a simulated Shadow Arm as well as our hand:

```
> roslaunch sr_hand gazebo_arm_and_hand_motor.launch
```

4.3 Checking it works

5 Where to go next

The [ROS wiki](#) is probably the best place to start learning about ROS. We strongly recommend going through the excellent [ROS Start Guide](#), especially the set of [tutorials](#).

6 What to do if it doesn't work

6.1 Check your ROS setup

You can use very simple examples to check different things:

- Check you can access our stacks (if this command doesn't work, make sure you've installed the code properly and sourced the generated setup.bash as described in **section 2**):

```
> roscd shadow_robot
```

- Check you can publish and subscribe to a simple topic. You'll need to start each of the following commands in a separate terminal. The first command is used to start the rosmaster. The second one will publish some data on the `test` topic while the third command subscribes to the topic and print the output on the screen. If this test works, you should see the incoming data streaming in your third terminal.

```
#in terminal 1, start the rosmaster
> roscore
#in terminal 2 publish data at 5Hz on /test
> rostopic pub /test std_msgs/Float64 -r 5 1.0
#in terminal 3, subscribe to the /test topic
# you should see the streaming data
# stop it with Ctrl+c
> rostopic echo /test
data: 1.0
-----
data: 1.0
-----
data: 1.0
```

- If all hope is lost, run the [roswtf](#) utility.

6.2 Use the ROS community

A good thing to remember is that you're probably not the first one having run into this problem. You should check the ROS community website: answers.ros.org and post your question if you can't find a relevant answer.

6.3 Contact us

Don't hesitate to contact us: software@shadowrobot.com.

7 Changelog

7.1 Shadow Robot Stack

This is the log for the Shadow Robot stack.