Manual for package "careobot_tuberlin" with Care-O-Bot 3

Autors: Mukul Suhas Bandodkar,
M. K. Hasan &
Shaon Debnath

Technische Universität Berlin, Germany

Last modified on Wednesday 2nd August, 2017

Contents

Chapter 1: Introduction

This manual is divided into three main parts. The first part (chapter 2), It will give you an overall idea about COB3 hardware component and about careobot_tuberlin package including the nodes. In second part (chapter 3), it shows all command you have to put to run this package. In part 3 (chapter 4), we have mentioned about some errors that can arise during installing the packages or to run the code and their solution.

Chapter 2: Manual

NOTE: In this manual we will show how to manipulate care o bot3 in Gazebo and Rviz simulation environment using careobot_tuberlin package. This package also can be used on real Care-O-Bot3. But before use on real robot please go through the Care-O-Bot 3 Manual [1] to understand emergency stop, emergency stop button, how to connect with robot, how to login etc.

Also these following user manual requires some basic knowledge about

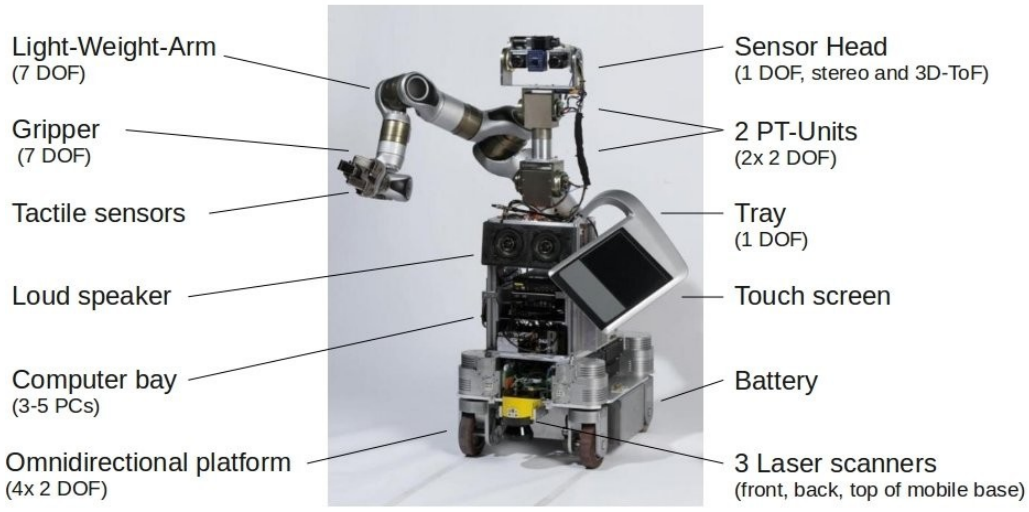
- Linux/Ubuntu
- Source code management with git
- ROS usage

If you are missing some of this requirements or feel uncomfortable with what you are doing, please interrupt and ask somebody to help you before continuing.

[1] https://github.com/ipa320/setup/blob/master/manual/Care-O-bot_manual.pdf

a. Hardware Overview

From official web site[1] and [2], you can get knowledge about technical data of care-o-bot 3. Also you can see the distribution of the different Care-O-bots at [3].
An overview of the robot hardware is shown in the following picture.



1. http://www.care-o-bot-research.org/care-o-bot-3/technical-data
2. http://www.care-o-bot-research.org/care-o-bot-3/components
3. http://www.ros.org/wiki/Robots/Care-O-bot/distribution

b. Software overview

In our package careobot_tuberlin , we defined a launch file which have called all necessary packages to run before start manipulating cob3. In the following we are describing the launch file and other nodes that are used in this package.

1. **myrobot_launch_sim.launch:** This launch file creates a simulation with kitchen environment, Care-O-Bot 3 and an object on the table. To test in real Robot myrobot_launch.launch can be used. More over these, this launch file launches all those topics which will be required to manipulate COB properly. The other packages we are launching here are: 2d_nav_ros_dwa.launch and move_group.launch.

2. **start_node:** This node is used to initialize all the components of COB3. i.e torso, head, arm, gripper, base.To initialize or manipulate all component we need to communicate with several rostopic like geomatry_msgs/twist , arm_navigation_msg, tf.transform etc. and we need to pass some rosparam. Start_node communicate with them through simple_script_server. After initializing the components using simple_script_server, arm and tray is taken to folded position, which are safe before navigating the base around. Then it should ask for it destination. We have set parameter for Table but you also can put any other coordinate in x,y format. After taking location in terminal, this node call move base to navigate COB3

3. **grasp_detect_node:** This node is responsible for detecting grasping point of an object. We have used GPD packages to detect grasping point. As we did not finalized this code, this node is not available in our package. But anyone can start with this node by launching gpd tutorial1.launch , once this package is launched, cob3 will start to give grasping point. As an improvement you can detect the grasping point of a specific object.

4. **move_arm:** This node is responsible for moving the arm to the grasping point. Before moving the arm, we need to place the arm in pregrasp position, which will make it easier to plan for join of arm. And also, we need to cylopn the gripper.Once arm reached to pregrasp position with opened gripper, we need to plan all the joint position according to the grasping point, so that we can move the arm from pregrasp position to grasping point position. We have used moveit_group to calculate the value for each joint against grasping point. While it gets plan, COB3 will start to move its arm towards the grasping point.

5. **mygrasp_node:** This node is used for Grasping the object, hold it, put it back on the tray and navigate to initial position.To grasp the object we have again used simple_script_server. To grasp the object gipper needed to be cylclosed and arm need to be position “home” at first, then it can move toward tray. And then arm should be folded before navigate to initial position.

Chapter 3: Run careobot_tuberlin package

a. Prerequisites

Ubuntu 14.04: To run this project, as we described before we will need ubuntu 14.04 LTS (Trusty), since care o bot 3 does not support since indigo. And Ros-Indigo supports on ubuntu 14.04

ROS-Indigo Installation:

Setup your sources: [http://wiki.ros.org/indigo/Installation/Ubuntu]

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

Setup your key:

```
sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80 --recv-key 421C365BD9FF1F717815A3895523BAEEB01FA116
```

Installation:

```
sudo apt-get update
sudo apt-get install ros-indigo-desktop-full
```

Initialize Rosdep

```
sudo rosdep init
rosdep update
```

Environment setup:

```
echo "source /opt/ros/indigo/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

Getting ROS Install

```
sudo apt-get install python-rosinstall
```

Ros Environment check

```
$ printenv | grep ROS
```

Create ROS Work Space[http://wiki.ros.org/catkin/Tutorials/create_a_workspace]

```
$ mkdir -p ~/catkin_ws/src
$ cd ~/catkin_ws/
$ catkin_make
```

Source Path

```
$ source devel/setup.bash
OR
echo "source /home/shaon/catkin_ws/devel/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

Now check your path

```
$ echo $ROS_PACKAGE_PATH
It should look like
/home/youruser/catkin_ws/src:/opt/ros/indigo/share
```

Care O Bot Packages Installation [http://wiki.ros.org/Robots/Care-O-bot/indigo]:

Setup source list

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

Setup your keys

```
sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80 --recv-key 421C365BD9FF1F717815A3895523BAEEB01FA116
```

Installation:

```
sudo apt-get update
sudo apt-get install ros-indigo-care-o-bot
```

Environment setup:
Though it should be already setup, but if its needed run

```
echo "source /opt/ros/indigo/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

Sometimes all the packages do not download with this automatically. In those cases you will need to install them manually. For these:

```
sudo apt-get install ros-indigo-cob*
```

(If installation get error, Please see the section Error and Solution)

Moveit Package installation [4]:

```
sudo apt-get install ros-indigo-moveit
```

Setup is done! Now you can test your setup by running default care o bot project
To run this project,

```
export ROBOT=cob3-2
export ROBOT_ENV=ipa-kitchen
roslaunch cob_bringup_sim robot.launch
```

If it is running properly careobot_tuberlin package is ready to run.

GPD installation:

(NOTE: We need this package to detect the objects and it's grasping point. As we could not finalized it, so grasp_detect_node is not available in our package. So you can skip this part. But if you install this and run gpd tutorial1.launch, cob will start to detect the grasping point)

Object Detection commands

Install gpd from https://github.com/atenpas/gpd, to do this we needed to install prerequisites packages

Install PCL
****
Install PCL from http://pointclouds.org/downloads/linux.html

```
sudo add-apt-repository ppa:v-launchpad-jochen-sprickerhof-de/pcl
sudo apt-get update
sudo apt-get install libpcl-all
```

Eigen:
download Eigen 3.3.3 from (ref http://eigen.tuxfamily.org/index.php?title=Main_Page)

http://bitbucket.org/eigen/eigen/get/3.3.3.tar.bz2

Install caffe:
To install it we needed to install general dependemcies, Cuda, ATLAS first

General Dependencies:

```
sudo apt-get install libprotobuf-dev libleveldb-dev libsnappy-dev libopencv-dev libhdf5-serial-dev protobuf-compiler
sudo apt-get install --no-install-recommends libboost-all-dev
```

Install Cuda:

Download Cuda from https://developer.nvidia.com/cuda-downloads
and install

```
sudo dpkg -i cuda-repo-ubuntu1404-8-0-local-ga2_8.0.61-1_amd64.deb
sudo apt-get update
sudo apt-get install cuda
```

Then patch

BLAS:

Install ATLAS by sudo apt-get install libatlas-base-dev or install OpenBLAS by sudo apt-get install libopenblas-dev or MKL for better CPU performance.

Remaining Dependencies , 14.04

```
sudo apt-get install libgflags-dev libgoogle-glog-dev liblmdb-dev
```

Install OpenCV:

Please follow the instruction from
http://www.pyimagesearch.com/2015/06/22/install-opencv-3-0-and-python-2-7-on-ubuntu/

Install caffe

```
git clone https://github.com/BVLC/caffe.git && cd caffe
git checkout 923e7e8b6337f610115ae28859408bc392d13136
```

Now compile it

```
cp Makefile.config.example Makefile.config
# Adjust Makefile.config (for example, if using Anaconda Python, or if cuDNN is desired)
make all
make test
make runtest
```

Install GPG:

```
cd <location_of_your_workspace>
git clone https://github.com/atenpas/gpg.git

cd gpg
mkdir build && cd build
cmake ..
make
sudo make install
```

Download GPD:

```
cd <location_of_your_workspace/src>
git clone https://github.com/atenpas/gpd.git

change Caffe PATH cmakelist.txt of GPD like
SET(CAFFE_DIR "/home/shaon/catkin_ws/src/caffe/build")

cd <location_of_your_workspace>
catkin_make
```

b. Run command for careobot_tuberlin

Download the package careobot_tuberlin from gitlab [https://shaondebnath@gitlab.tubit.tu-berlin.de/aac-hrc/robot-grasping.git]

Keep it in your catkin workspace example /catkin_ws/src/

To run this project,

```
export ROBOT=cob3-2
export ROBOT_ENV=ipa-kitchen
roslaunch careobot_tuberlin myrobot_launch_sim.launch
```

We have to wait some times until all required packages running. at end of all package running you should see "odiom received"

Now we can run our node in other terminal

```
rosrun careobot_tuberlin start_node.py
```

It will initialize all COB3 components and will move to it's component to initial position. Then it will as you for the location and will wait for input. Enter "table" (or any other x,y position). COB3 will navigate to table and it will run the next script move_arm.py, it will again ask for your input. Enter "milk". Grasping point for milk is given manually. COB3 arm will be moved from folded position to grasping point position via pregrasp position. After reaching this position mygrasp_node.py will be running. Through this node cob3 will grasp the object, take it in hold position keep it on its tray and will navigate back to its initial position.

Chapter 3: Errors and Solution

During installation or run the packages you may encounter many errors. Here we are some common errors with solution which we faced during this package creation

Error 1:
During installation of ros-indigo-cob*

```
sudo apt-get install ros-indigo-cob*
```

some errors can be appeared like:
"Problem for installation because of broken package: ros-indigo-libopengm_0.6.10-0trusty-20170313-063520-0700_amd64.deb"

This error occurs during installation of cob_extern, But other packages installed properly. We need to solve it as, it can be cause of installation failure for future installation of any other packages.
To solve it we need to remove ros-indigo-cob-extern (we do not need this package now, but it can be installed later if needed)

```
sudo apt-get remove ros-indigo-cob-extern
```

```
sudo rm /var/cache/apt/archives/ros-indigo-libopengm_0.6.10-0trusty-20170313-063520-0700_amd64.deb
```

Error 2:

When you will try to manipulate gripper, you may get some rospy error. To solve it you will need to modify simple_script_server.

Please update simple_script_server.py (find it at /opt/ros/indigo/lib/python2.7/dist-packages/simple_script_server/) and search for def calculate_point_time

Update code[6]:

```
def calculate_point_time(self, component_name, start_pos, end_pos, default_vel):
```

```
        try:
```

```
                d_max = max(list(abs(numpy.array(start_pos) - numpy.array(end_pos))))
```

```
                point_time = d_max / default_vel
        except ValueError as e:
                print "Value Error", e
                print "Likely due to mimic joints. Using default point_time: 3.0 [sec]"
                point_time = 3.0            # use default point_time
                return point_time
```

Error 3: During the nodes running it may ask you for same input more than 1 time (example location or object). It happens because of parsing by simple_script_server. To solve this issue command 2 lines from Parse function inside simple_script_server.py

```
def Parse(self):
------- some codes-------
-------some codes----

# run script in simulation mode
            self.sss = simple_script_server(parse=True)
            #self.Initialize()
            #self.Run()
```

Error 3:
Though dwa planner should be installed automatically with ros package but some time it does not installed. We need it to navigate cob.

```
sudo apt-get install ros-indigo-dwa-local-planner
```