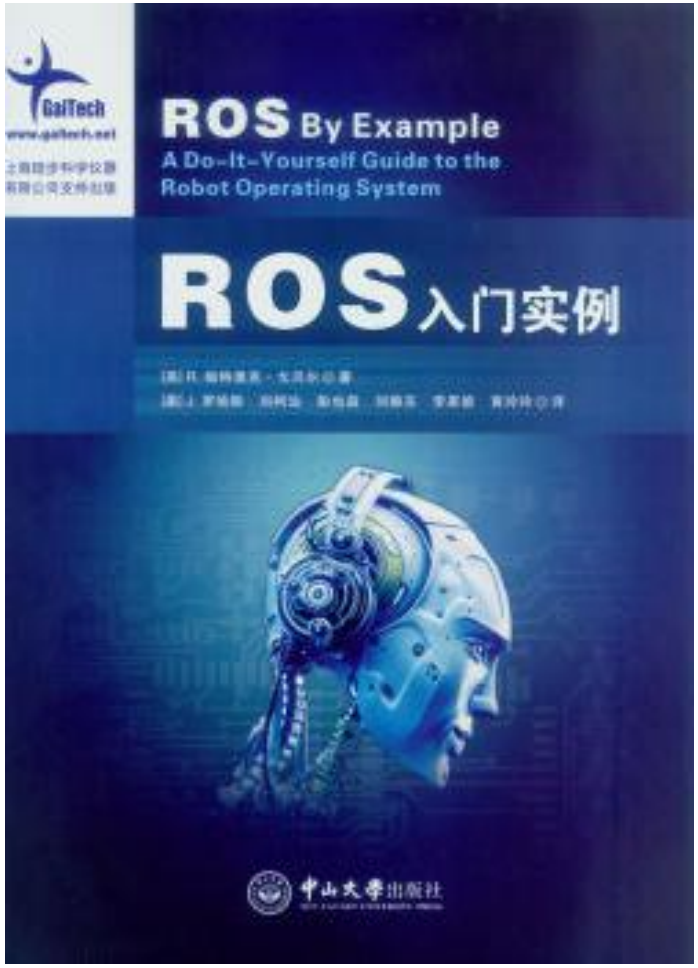
Baxter and Manipulation

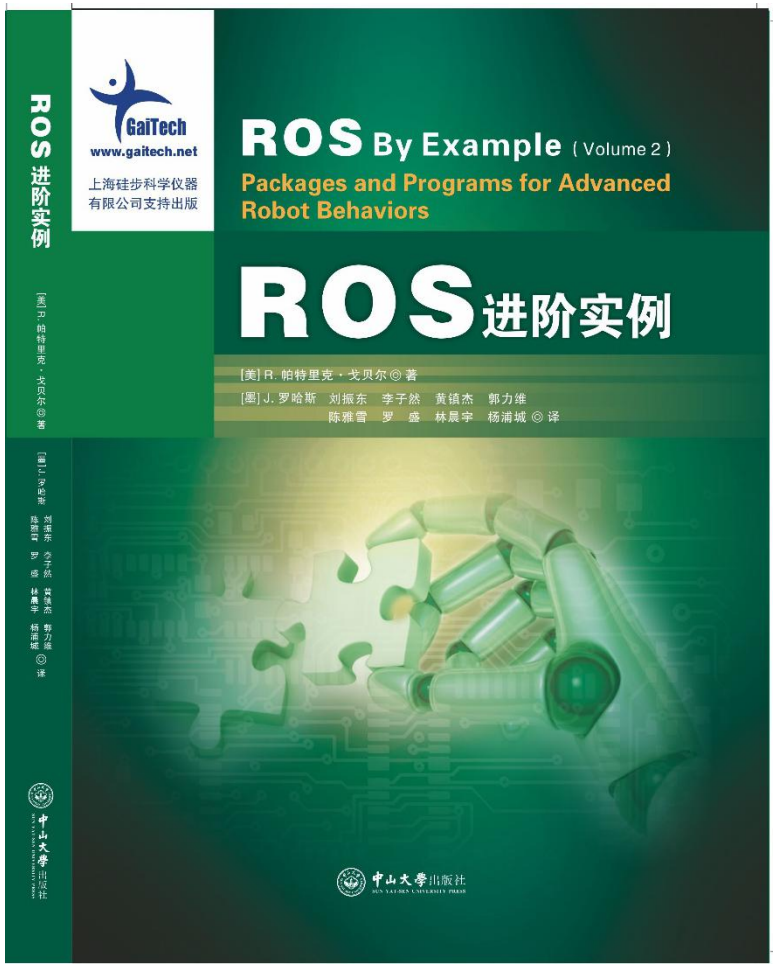
JUAN ROJAS
www.JuanRojas.net

BIOMIMETICS AND ROBOTICS LAB (BIRL)

GUANGDONG UNIVERSITY OF TECHNOLOGY (GDUT)

ROS By Example in Chinese





Available at: Taobao, Amazon.cn, JD.com, Dangdang.com

Support Programs

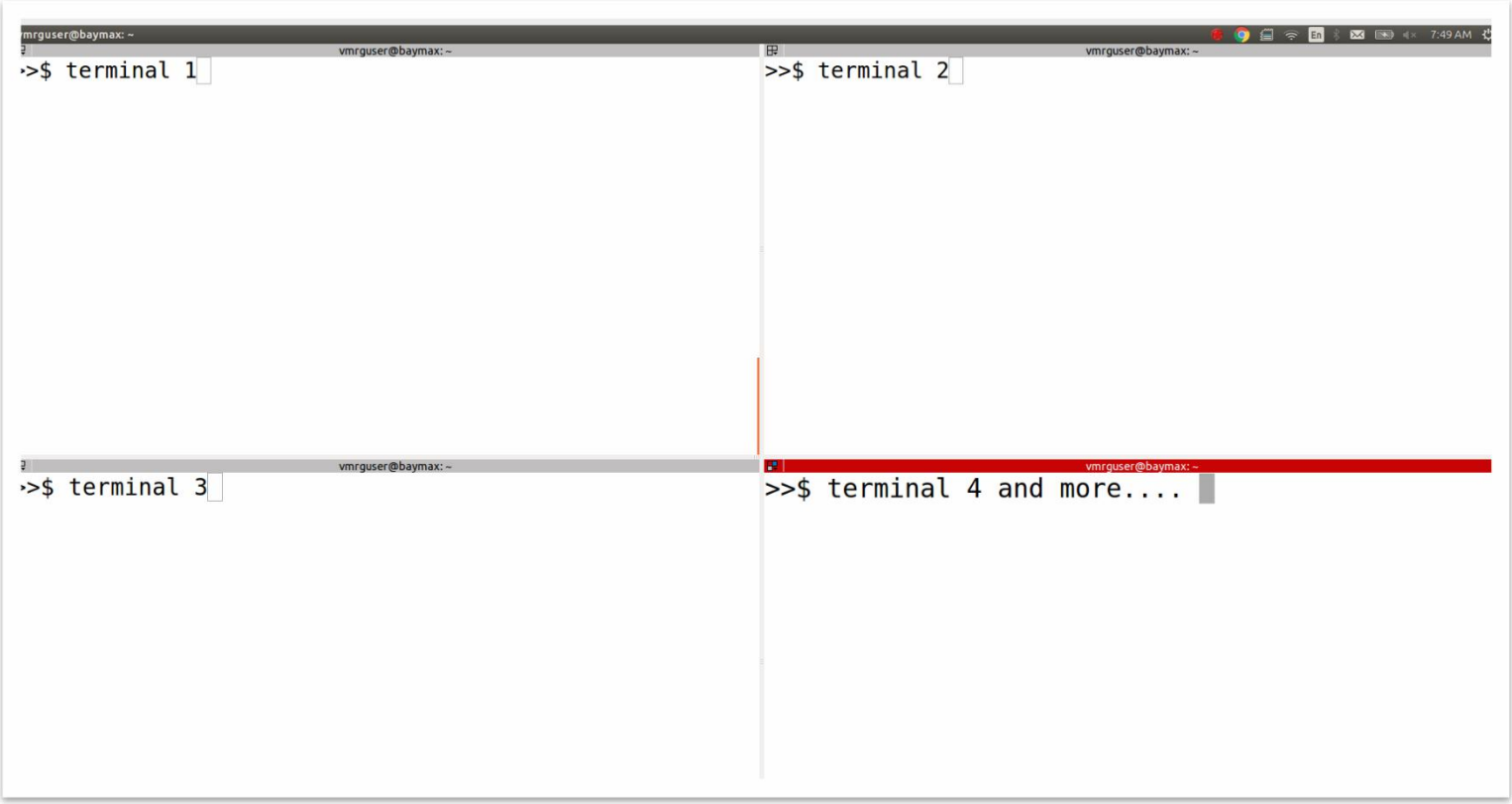
APT-GET

- Used to download programs in linux

```
sudo apt-get update
sudo apt-get upgrade
```

TERMINATOR

Great to run multiple terminals in the same window.



EMACS OR VIM

Extremely powerful editors and more.

- Powerful editor
- Strong integration with GDB/PDB

Live terminals

Easily expandable

emacs@baymax

File Edit Options Buffers Tools Gud Complete In/Out Signals Help

Set Breakpoint | p | Continue | Next Line | Step Line | Up Stack | Down Stack

```
1  #!/usr/bin/env python
2  import ipdb
3  import sys
4  import argparse
5  import rospy
6  from math import pi
7
8  import baxter_interface
9  from baxter_interface import CHECK_VERSION
10 from hand_action import GripperClient
11 from arm_action import (computerIK, computerApproachPose)
12
13 from pa_localization.msg import pa_location
14 from birl_recorded_motions import paHome_rightArm as rh
15 from copy import copy
16 import PyKDL
17 import tf
18
19 import threading
20
21 # Uses a thread class to block the subscription spin.
22 # Thread calls a subscriber pointing to topic pick_location, with msg type pa_location, a
   nd callback self.callback.
```

-:--- pa_manipulation.py Top (2,0) Git-master (Python AC)

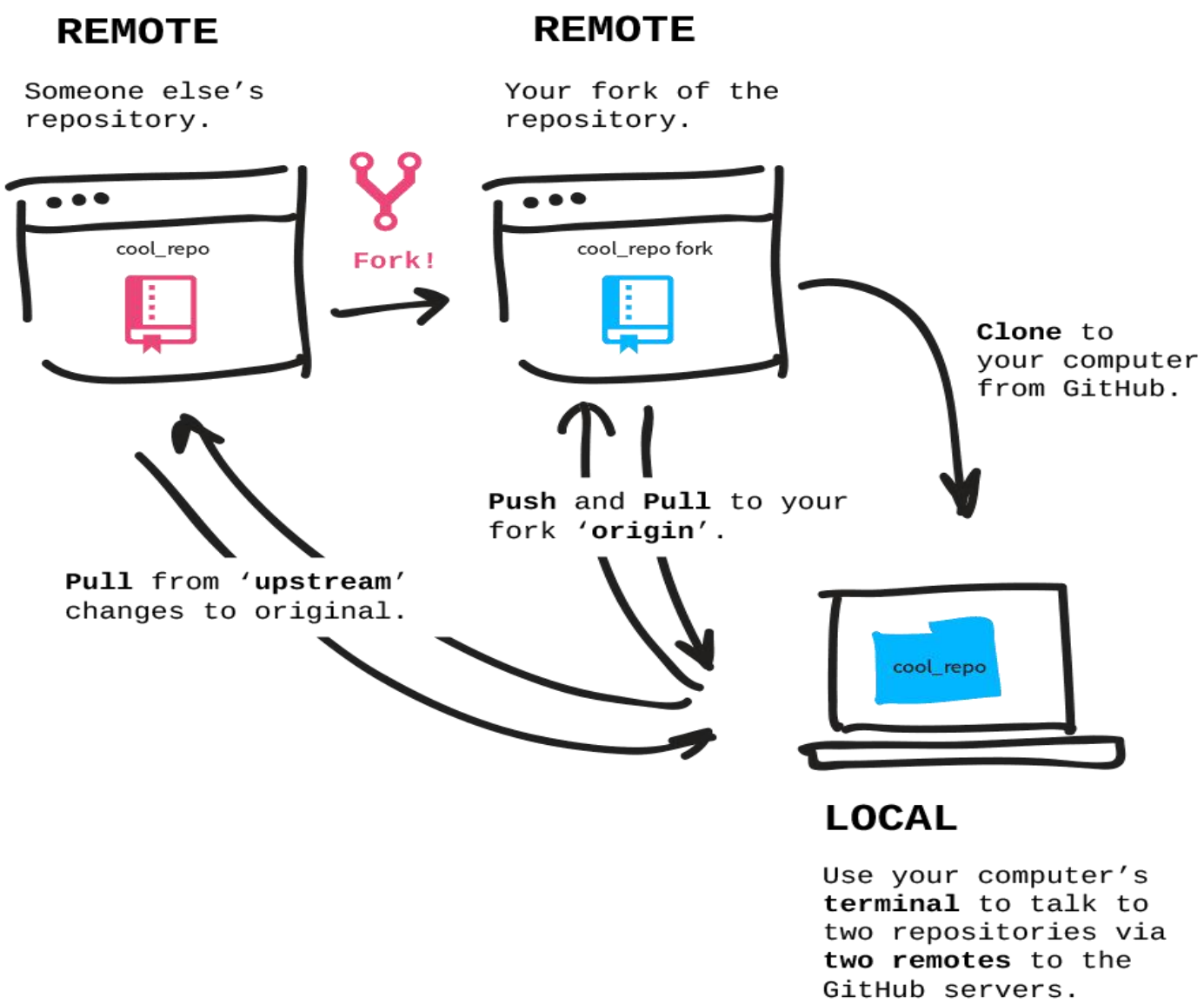
```
1 ls
2 >>$ ls
3 arm_action.py      get_pose_online.py   hand_action.py~        #pa_manipulation.py#
4 arm_action.py~     get_pose_online.py~  hand_action.pyc        pa_manipulation.py
5 arm_action.pyc     goOrigin.py          __init__.py            pa_manipulation.py~
6 endPose_calib.py   goOrigin.py~         pa_manipulation_2.py
7 endPose_calib.py~  hand_action.py       pa_manipulation_2.py~
8 >>$
```

```
29        self._lock=threading.Lock()
30        self._thread=threading.Thread(target=self.picking_location_listener)
31        self._thread.start()
32
33    def picking_location_listener(self):
34        rospy.Subscriber("pick_location",pa_location,self.callback)
35        rospy.spin()
36
37    def callback(self,msg):
38        self.lock()
39        self._pose=msg
40        self.unlock()
41
42    def getPose(self):
43        return self._pose
44
45    def lock(self):
46        self._lock.acquire()
47
48    def unlock(self):
49        self._lock.release()
50
51 def main():
```

-:--- pa_manipulation.py 9% (29,0) Git-master (Python AC)

```
1 Current directory is ~/ros/indigo/birl_baxter_ws/src/birl_demos/pick_n_place_demo/pa_demo/s
  cripts/pa_demo/
2 > /home/vmrguser/ros/indigo/birl_baxter_ws/src/birl_demos/pick_n_place_demo/pa_demo/scripts
  /pa_demo/pa_manipulation.py(2)<module>()
3 -> import ipdb
4 (Pdb)
```

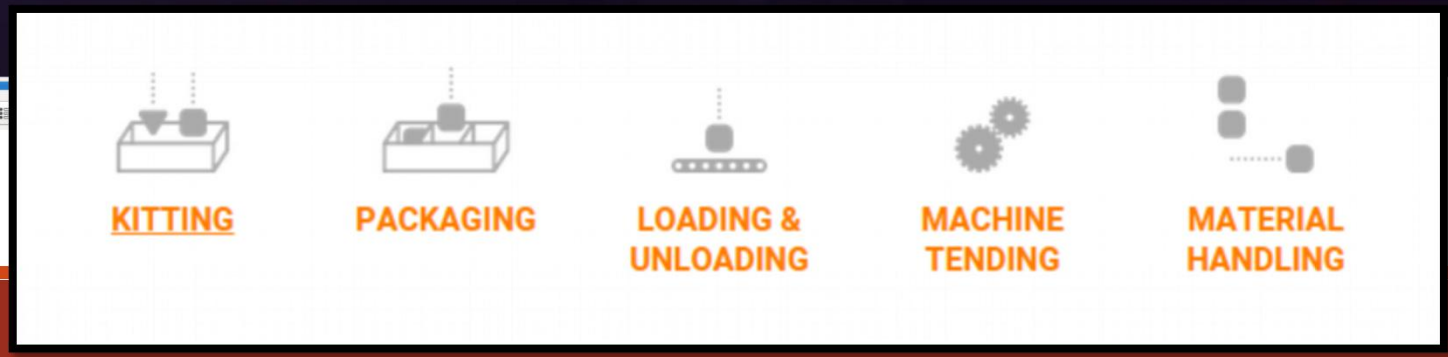
GIT


REMOTE
Someone else's repository.
cool_repo
Fork!
REMOTE
Your fork of the repository.
cool_repo fork
Clone to your computer from GitHub.
Push and Pull to your fork 'origin'.
Pull from 'upstream' changes to original.
cool_repo
LOCAL
Use your computer's terminal to talk to two repositories via two remotes to the GitHub servers.

Getting to know Baxter


baxter
rethink robotics

00:37 00:48


KITTING
PACKAGING
LOADING & UNLOADING
MACHINE TENDING
MATERIAL HANDLING

Baxter's Arms

7 Degrees of Freedom (DoF)

7vs6 DoF = wider mobility.

Series Elastic Actuators

Spring between motor/gear:

1. Stable, low-noise Force Control.
2. Compliant.
3. Measure Torque at each joint.

Programming Layers

API

- Python interface for Baxter.
- Interface interacts with ROS.
- Goal to facilitate programming.

SDK

- Defines ROS:
  messages, topics, services, action libs.
- Also provides command line tools.

Getting the Baxter Code

- Open source @ sdk.rethinkrobotics.com/wiki/Workstation_Setup

Contents [hide]

Baxter's SDK

- As part of the SDK, Rethink has defined:
  - Topics: /robot/limb/….
    /robot/head/…
  - Message Types:
    baxter_core_msgs/
  - Parameters:
    /baxter_emulator/left_gripper_type
  - Services:
    /ExternalTools/PositionKinematicsNode/IKService
  - Action Libs:
    /robot/limb/<limb>/follow_joint_trajectory/feedback
    /robot/limb/<limb>/follow_joint_trajectory/result
    /robot/limb/<limb>/follow_joint_trajectory/status
  - User Tools
    rosrun baxter_tools ….

Getting Baxter Started

Setting the Baxter environment:

```
>> roscd          (ROS_WORKSPACE=/your_fav_ws_path)
>> ./baxter.sh (sim for simulator)
```

- Starting the Simulator:

```
>> roslaunch baxter_gazebo baxter_world.launch
```

- For real Baxter, you can check for automatic connection:

```
>> roslaunch baxter_gazebo baxter_world.launch
```

```
[baxter - http://011405P0002.local:11311] >>$ rostopic list
```

Baxter's Arm and Head Joints

The 7 DoF arms and Head pan consists of joints states, including:

- Position – joint angles (radians)
- Velocities – joint velocities (rad/s)
- Effort – torque exerted at each joint (Nm)

Topic

/robot/joint_states

Message Type:

sensor_msgs/JointState

Baxter's Arms: Control Modes

Arms can be controlled in 4 different modes. Top 3:

- Position Control – controller moves to target joint angles
- Velocity Control – controller moves to target joint velocities
- Torque Control – controller moves to target joint torques

Switch modes by pub commands (pos,vel,effort) @ > 5Hz

```
/robot/limb/<side>/joint_command        (baxter_core_msgs/JointCommand.msg)
```

Message Type: baxter_core_msgs/JointCommand

```
int32 POSITION_MODE=1, int32 VELOCITY_MODE=2,
int32 TORQUE_MODE=3, int32 RAW_POSITION_MODE=4
int32 mode,
float64[] command
string[] names
```

Move Arm Manually…

```
rostopic pub -r 1000
/robot/limb/right/joint_command
baxter_core_msgs/JointCommand
'{mode: 1, command: [0.1744], names: ['right_s0']}'
```

Publish to joint_command

- Manually test right position/velocity control.

- Simple Position Control Command

```
rostopic pub -r 10   /robot/limb/right/joint_command
baxter_core_msgs/JointCommand   '{mode: 1,    command: [-1.0], names:
['right_s0']}'
```

- Simple Velocity Control Command

```
rostopic pub -r 10   /robot/limb/right/joint_command
baxter_core_msgs/JointCommand   '{mode: 2,    command: [-0.01], names:
['right_s0']}'
```

EndPointState

Provides the following at the end-effector:

- Pose (m)
  (position, orientation)
- Twist (m/s)
  (lin vel, angular vel)
- Wrench (N/m)
  (forces, torques)

/robot/limb/<side>/endpoint_state (baxter_core_msgs-EndpointState)

Baxter API

API

What is the API?

A new layer of code (based on python) is built on top of ROS.

Instead of having to:

- Publish or subscribe
- Call services

Call one of the API methods and

- read/write data through function arguments.

API is organized according to:

- Modules
  - Sub-modules.

The Baxter Interface: Python Module

baxter_interface

- This module consists of sub-modules to help interact with different parts of the robot.
- Each sub-module consists of a class of the same name. baxter_interface::limb::Limb
- The class is a wrapper around ROS communications.

Sub-Modules (Interfaces)

Robot Enable	Limb	Head	Camera
Gripper	Navigator	Digital IO	Analog IO

Limb

Limb is the *class* within the limb sub-module.

- Queries the joint state
- Switches between control modes
- Sends Joint Commands (pos, vel, torque)

```
from baxter_interface import Limb
```

```
right_arm = Limb('right')
left_arm = Limb('left')
```

Topics

/robot/joint_states
/robot/limb/<side>/joint_command

Limb Class Overview

The methods below consider position only but…
The same routines exist for velocity and effort.

[str]	`joint_names(self)` Return the names of the joints for the spec
float	`joint_angle(self, joint)` Return the requested joint angle.
dict({str:float})	`joint_angles(self)` Return all joint angles.
dict({str:Limb.Point,str:Limb.Quaternion})	`endpoint_pose(self)` Return Cartesian endpoint pose {position, orientation}.
	`set_joint_positions(self, positions, raw=False)` Commands the joints of this limb to the specified positions.
	`move_to_neutral(self, timeout=15.0)` Command the joints to the center of their joint ranges

BAXTER REPO

https://github.com/RethinkRobotics/baxter

Create ROS Workspace

```
$ mkdir -p ~/ros_ws/src
# ros_ws (short for ROS Workspace)
```

Source ROS Setup

```
$ source /opt/ros/indigo/setup.bash
```

Build and Install

```
$ cd ~/ros_ws
$ catkin_make
$ catkin_make install
```

Install SDK Dependencies

```
$ sudo apt-get update
$ sudo apt-get install git-core python-argparse python-wstool python-vcstools python-rosdep ros-indigo-control-msgs ros-indigo-joystick-drivers
```

Install Baxter SDK

Using the wstool workspace tool, we will checkout all required Baxter Github Repositories into your ROS workspace source directory.

```
$ cd ~/ros_ws/src
$ wstool init .
$ wstool merge https://raw.githubusercontent.com/RethinkRobotics/baxter/master/baxter_sdk.rosinstall
$ wstool update
```

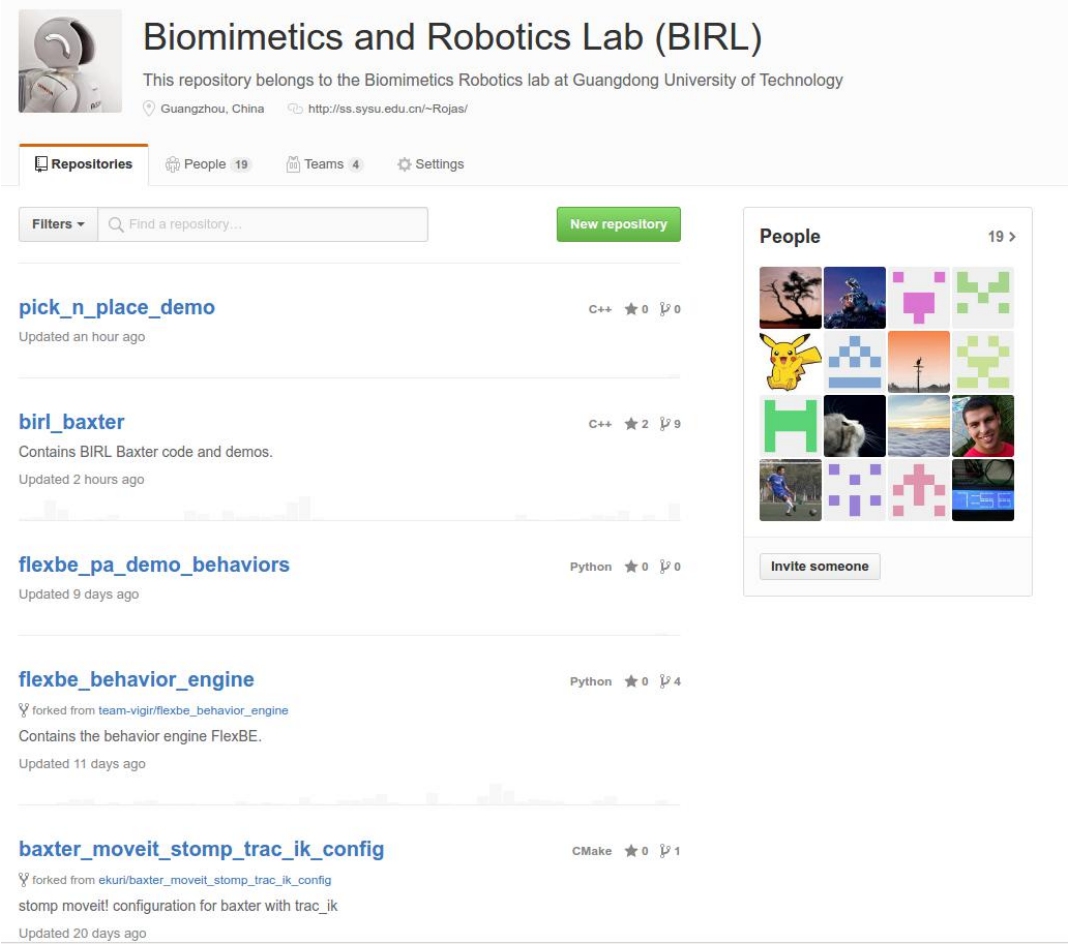
Build and Install

```
$ cd ~/ros_ws
$ catkin_make
$ catkin_make install
```

BIRL BAXTER REPO

https://github.com/birlrobotics/birl_baxter/wiki

BIRL Robotics GitHub Repo



BAXTER EXAMPLES

Baxter Examples

http://sdk.rethinkrobotics.com/wiki/Examples

Movement

Joint Position Waypoints Example - The basic example for joint position moves. Hand-over-hand teach and recording a number of

Joint Position Keyboard Example - This example demonstrates numerous joint position control.

Joint Position Example - Joystick, keyboard and file record/playback examples using joint position control of Baxter's arms.

Joint Torque Springs Example - Joint torque control example applying virtual spring torques.

Joint Velocity Wobbler Example - Simple demo that moves the arm with sinusoidal joint velocities.

Joint Velocity Puppet Example - Simple demo which mirrors moves of one arm on the other

Inverse Kinematics Service Example - Basic use of Inverse Kinematics solver service.

Simple Joint Trajectory Example - Simple demo using the joint trajectory interface.

Simple Joint Trajectory Example - Simple

Joint Trajectory Playback Example - Trajectory playback using the joint trajectory interface.

Head Movement Example - Simple demo moving and nodding the head.

Head Action Client Example - A demo to showcase the functionality of the head trajectory action server.

Gripper Example - Joystick and Keyboard control for the grippers.

Gripper Cuff Control Example - Simple cuff-interaction control with Zero-G mode.

Robot Configuration

URDF Configuration Example - A simple ROS node that shows how to add segment and joint subtrees to the robot's model.

Simulator

IK Pick and Place Demo - An intermediate example for combining Inverse Kinematics Service calls with Arm movement, gripper a

Input and Output

Camera Control Example - Demonstrates usage for listing, opening, and closing the available cameras.

View Cameras Example - Simple tool for viewing camera feed on development machine.

Screen Display - Example tool for displaying image files (png, jpeg) on the Head Screen.

I/O Example - Flash the lights on the digital outputs.

baxter_baxter_tasks

https://github.com/birlrobotics/birl_baxter_tasks

Random Pick and Place w/ Smach

- Open 5 terminator windows and run the ./baxters.sh sim script in all of them.
- Launch gazebo along with all the URDFs

```
$ roslaunch birl_baxter_description pick_n_place_box_gazebo.launch
```

- Run a service server to serve client calls.

```
$ rosrun birl_baxter_description pick_n_place_box_gazebo.launch
```

- Open the smach viewer:

```
$ rosrun smach_viewer smach_viewer.py
```

- Run the Joint Trajectory Action Server:

```
$ rosrun birl_sim_examples pick_n_place_joint_trajectory_smach.py
```

- Try different clients to perform tasks:

```
$ rosrun birl_sim_examples pick_n_place_srv_client_random_smach.py
```

Random Pick and Place w/ Smach

- Open 5 terminator windows and run the ./baxters.sh sim script in all of them.
- Launch gazebo along with all the URDFs

```
$ roslaunch birl_baxter_description pick_n_place_box_gazebo.launch
```

- Run a service server to serve client calls.

```
$ rosrun birl_baxter_description pick_n_place_box_gazebo.launch
```

- Open the smach viewer:

```
$ rosrun smach_viewer smach_viewer.py
```

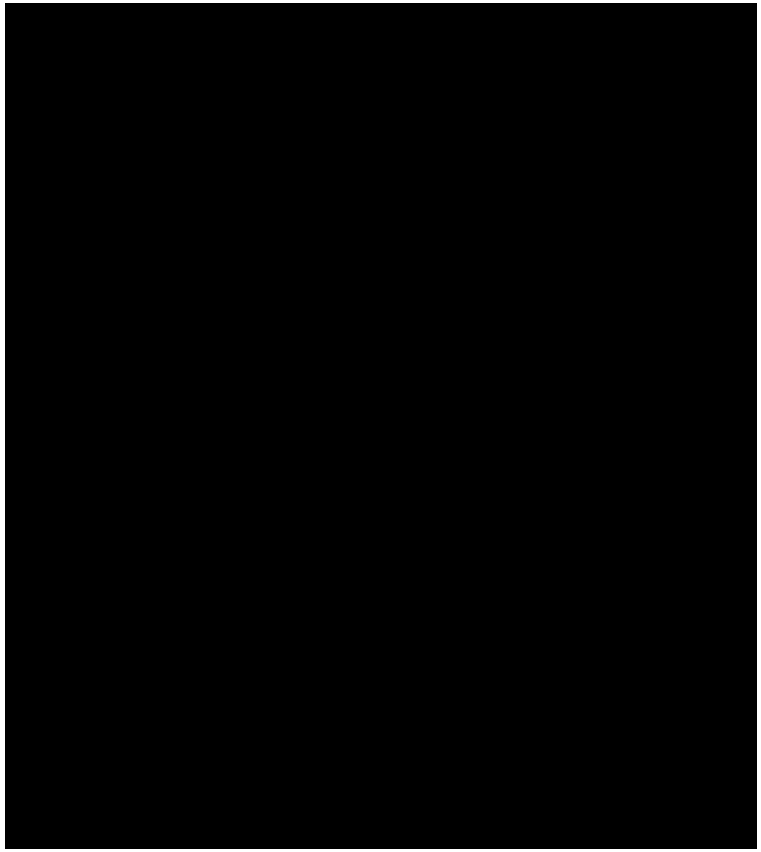
- Run the Joint Trajectory Action Server:

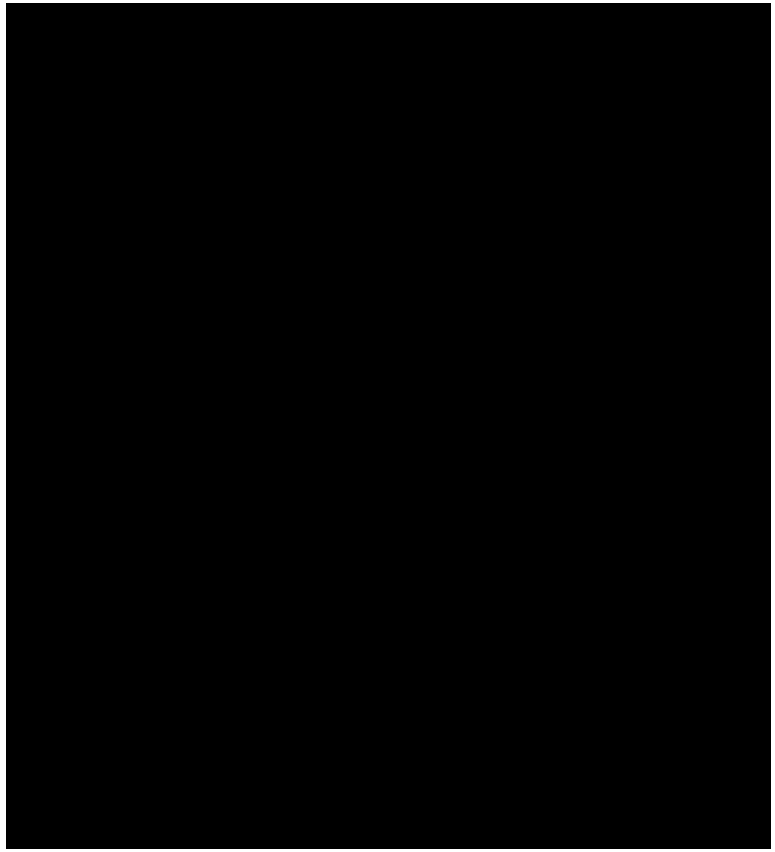
```
$ rosrun birl_sim_examples pick_n_place_joint_trajectory_smach.py
```

- Try different clients to perform tasks:

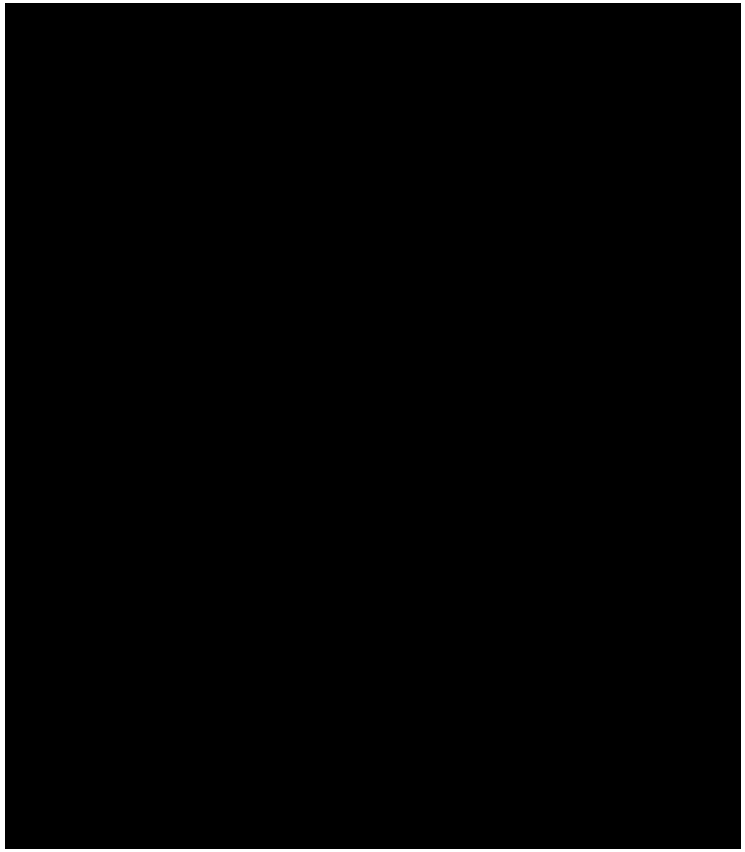
```
$ rosrun birl_sim_examples pick_n_place_joint_trajectory_smach.py
```

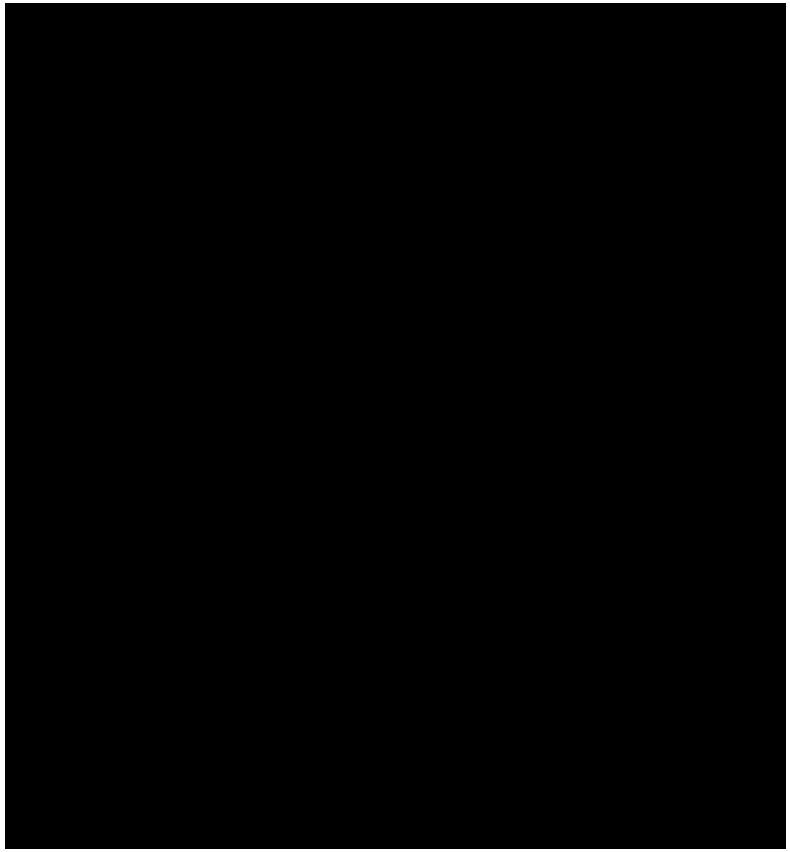
Advanced: Pick and Place with Anomaly Recovery





Advanced: Open and Close Drawer with Anomaly Recovery





questions