

Sawyer 案例例程使用说明

1.1、每一次使用机器人之前必做步骤使能机器人。终端指令如下，如终端输出的结果显示“Robot Enabled”则表示已成功启动 Sawyer 机器人。

```
$ cd ros_ws  
  
$ ./intera.sh  
  
$ rosrn intera_interface enable_robot.py -e
```

1.2、禁用、启用 Sawyer 机器人案例。终端指令如下：

```
$ rosrn intera_interface enable_robot.py -d （禁用机器人）  
  
$ rosrn intera_interface enable_robot.py -e （启用机器人）
```

1.3、内置相机图像处理示例：

①、介绍：

摄像机显示示例演示了 Sawyer 机器人上摄像机的显示。用户可以选择是否显示未矫正的图像，以及“Canny”边缘检测图像。启动相机示例程序，您可以指定相机名称右手摄像机或头摄像机“（right_hand_camera 或 head_camera）”、使用原始图像（未整理）主题或使用流式处理“Canny”边缘检测图像。默认相机名称为“Head_Camera”。默认情况下，相机图像将在不进行“Canny”边缘检测的情况下进行校正。

②、案例运行使用命令：表示：启动头部摄像头，并使用经“opencv”矫正过的图像，不对图像进行平滑处理。

```
$ rosrn intera_examples camera_display.py
```

③、来启动该节点，另外还有 6 个参数：

- 1) -h 或 --help: 显示此帮助消息并退出。
- 2) -c 或 --camera: 是选择摄像头的选项，默认是“head_camera”可选“right_hand_camera”。
- 3) -r 或 --raw: 使用原始图像或是经“opencv”标定矫正后的图像，默认是使用经“opencv”标定矫正过图像，如果使用“-r”则选择原始图像：
- 4) -e 或 --edge: 是否使用“opencv”图像算子对图像进行灰度化和平滑处理，默认不处理，如果使用“-e”则进行图像平滑处理。
- 5) -g 或 --gain: “GAIN”设置相机增益（-1=自动）
- 6) -x 或 --exposure: “EXPOSURE”设置相机曝光（-1=自动）

④、节点案例运行使用命令：表示：使用手部摄像头，并使用有畸变的原图，对图像进

行平滑处理。

```
$ rosrn intera_examples camera_display.py -c right_hand_camera -r -e
```

⑤、源代码网站参考：

https://sdk.rethinkrobotics.com/intera/Camera_Image_Display_Example

1.4、显示屏背景图处理示例：

①、介绍：

显示屏背景图处理示例，演示如何将图像文件传递给机器人头部进行显示。该示例使用如何读取和转换图像文件“cv_bridge”将图像信息作为标准的“ROS”图像消息，发送到机器人头部屏幕，使用“xdisplay_image”将图像显示到 Sawyer 的头部显示。

②、案例运行使用命令：表示：头部显示的图片，使用的图片来自于“/home/bill/图片/3.png” -f <要发送的图像文件的路径>

```
$ rosrn intera_examples head_display_image.py -f /home/bill/图片/3.png
```

③、来启动该节点，另外还有 4 个参数：

1) -f 或 --file: “PATE” 为必要参数，用于显示的文件路径。

2) -l 或 --loop: 这是循环次数的选，默认是一次，如果选择“-l”则会持续显示该图片。

3) -r 或 --rate: 图片显示的频率，默认是 1.0。

4) -h 或 --help: 显示此帮助消息并退出。注意：其中最大屏幕分辨率为“1024x600”图像始终是以左上角为起点平铺，图像格式是“OpenCV-LoadImage()”支持的格式。

④、源代码网站参考：

https://sdk.rethinkrobotics.com/intera/Head_Display_Image_Example

1.5、头部显示屏转动示例：===

①、介绍：

Sawyer 的头部可以在平移模式下从左向右旋转。摇摄动作将 Sawyer 的“脸部”（也就是屏幕）摆动到可设定的角度。“head wobblers”示例旨在成为一个简单的演示，可以检查其代码以了解如何使用“head”并演示使用“intera_interface Head”类。

②、案例运行使用命令：表示：Sawyer 的头部将开始向左和向右平移到随机角度。

```
$ rosrn intera_examples head_wobbler.py
```

③、源代码网站参考：

https://sdk.rethinkrobotics.com/intera/Head_Movement_Example

1.6、头部灯光闪烁测试示例：

①、介绍：

使用默认参数：“head_green_light”运行此示例，观察 Saywer 机器人头部的指示灯是否闪烁。使用灯光名称“lights.list_all_lights ()”来更改要切换的灯光。

②、案例运行使用命令：表示：来测试头顶的亮灯情况，进行两次的绿灯亮和灭的显示。

```
$ rosrn intera_examples lights_blink.py
```

③、来启动该节点,另外还有 2 个参数：

1) -h 或 --help: 显示此帮助消息并退出。

2) -l 或 --light_name: “LIGHT_NAME”要使用的灯光组件的名称（默认：头部绿灯）

④、源代码网站参考：

https://sdk.rethinkrobotics.com/intera/Lights_Blinking_Example

1.7、键盘操控夹爪：

①、介绍：

使用键盘或操纵杆控制 Sawyer 的抓手。可以控制和感测位置、速度、保持力和移动力。支持“Logitech”和“Xbox”游戏控制器。

②、案例运行使用命令：

```
$ rosrn intera_examples gripper_keyboard.py
```

③、来启动该节点，另外还有 2 个参数：

1) -h 或 --help: 显示此帮助消息并退出。

2) -l 或 --limb: 运行抓取键盘示例的肢体，默认的是 right。

④、启动后输入“？”号，会出现键盘按键的对应操作提示：

```
Initializing node...
Getting robot state...
[INFO] [1563267908.159049]: Gripper deadzone set to 0.001
[INFO] [1563267908.159460]: Enabling robot...
[INFO] [1563267908.162915]: Robot Enabled
Controlling grippers. Press ? for help, Esc to quit.
key bindings:
  Esc: Quit
  ?: Help
  c: calibrate
  q: close
  u: decrease position by 12.5%
  -: decrease velocity by 12.5%
  i: increase position by 12.5%
  +: increase velocity by 12.5%
  o: open
  r: reboot
  s: stop
```

⑤、源代码网站参考：

https://sdk.rethinkrobotics.com/intera/Gripper_Example

1.8、腕部操控夹爪：

①、介绍：

该程序演示了使用夹持器、袖带、灯界面控制夹持器和导航灯的关闭和打开动作。该“main()”函数创建一个“GripperConnect”类的实例，它将夹子按钮连接到各种回调函数。

②、案例运行使用命令：表示：来通过 Sawyer 电动夹爪的按钮来控制夹爪的开闭。

```
$ rosrund intera_examples gripper_cuff_control.py
```

③、来启动该节点，另外还有 4 个参数：

1) -h 或 --help: 帮助提示。

2) -g {'right'}: 可选参数，默认为“right”。

3) -n 或 --no-lights: 可选参数，输入“-n”表示不开启与图 1 中“Zero-G”键相连的亮灯（图 1 中“Training Cuff Light”），即按住手臂的零重力按钮（图 1 中“Zero-G”）时蓝灯或红灯不亮

4) -v 或 --verbose: 可选参数，输入“-v”则打印“debug”信息。

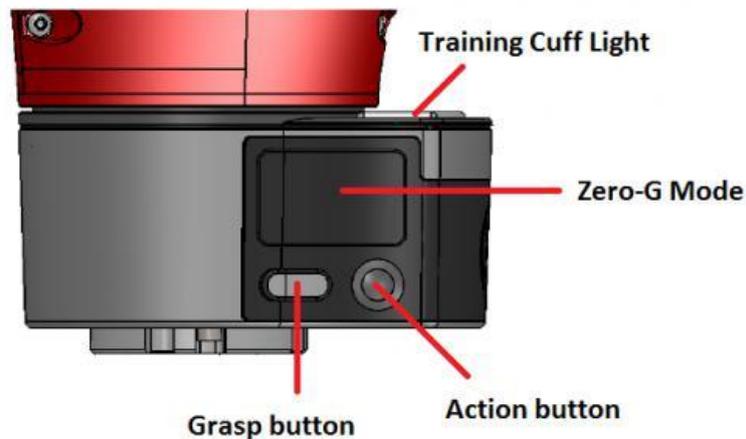


图 1 Sawyer 按钮示意图

注意：启动后，按住长型的“Dash(Grasp button)”键则关闭夹爪，长按住圆形的“Circle(Action button)”按钮，则开启夹爪。

④、源代码网站参考：

https://sdk.rethinkrobotics.com/intera/Gripper_Cuff_Control_Example

1.9、反向运动学示例：

①、介绍：

采用两种逆向运动学“IK”解算方式，分别是简单的“SolvePositionIK”和高阶的“SolvePositionIK-Request”，此示例演示了 Sawyer 中可用的“IK”服务的用法。该“main()”函数记录要在其上调用“IK”服务的关节，然后调用该“ik_test()”函数。在这里初始化“Rospy”节点，初始化默认笛卡尔坐标以及调用“IK”服务。服务“-

ExternalTools/right/PositionKinematicsNode/IKService”

②、案例运行使用命:

从“RSDK Shell”中,运行“inverse_kinematics”(运行反向运动学)包中的“ik_service_client.py”演示,该测试将为“IK”服务调用给定臂的硬编码笛卡尔姿势。如果解决方案是 Sawyer 手臂的有效配置,则响应将使用找到的联合解决方案打印到屏幕上。

```
$ rosrn intera_examples ik_service_client.py
```

③、源代码网站参考:

https://sdk.rethinkrobotics.com/intera/IK_Service_Example

1.10、键盘操控关节运动:

①、介绍:

使用开发机器人人员的键盘来控制 Sawyer 手臂的角关节位置。

②、案例运行使用命: 表示: 通过键盘控制 Sawyer 关节的运动,数字从“1到9”、字母都是第一行从“Q到I”,关闭进退都是数字字母上下对应的,“I和Q”对应同一个关节、“8和1”对应同一个关节,“9”是单独表示夹爪的标定,数字增加、字母减少。

```
$ rosrn intera_examples joint_position_keyboard.py
```

③、来启动该节点,另外还有2个参数:

1) -h 或 --help: 可选,帮助信息。

2) -l {right}: 可选,选择手臂 Sawyer 默认右手臂。

④、输入节点命令后,按下“?”号出现键盘控制提示。

```
key bindings:
Esc: Quit
?: Help
9: right gripper calibrate
8: right gripper close
i: right gripper open
q: right_j0 decrease
1: right_j0 increase
w: right_j1 decrease
2: right_j1 increase
e: right_j2 decrease
3: right_j2 increase
r: right_j3 decrease
4: right_j3 increase
t: right_j4 decrease
5: right_j4 increase
y: right_j5 decrease
6: right_j5 increase
u: right_j6 decrease
7: right_j6 increase
```

⑤、源代码网站参考:

https://sdk.rethinkrobotics.com/intera/Joint_Position_Example

1.11、控制关节到达路径点示例:

①、介绍:

此示例演示了位置控制器在简单关节位置移动的用法。该“main()”函数调用“record()”用户根据需要记录航路点的方法。然后“playback()”调用该方法，该方法通过记录的关节位置演示平滑的运动回放。

②、案例运行使用命令：表示：此时执行一个连续动作，并按下 Sawyer 手臂上的“OK”键，便会将这个连续动作记录：多次记录动作，会将前面已记录的动作覆盖，默认为记录一次动作：当终端上出现至少一个“Waypoint Recorded”时，按下“Rethink”按钮 Sawyer 便会循环执行上一次记录的动作，直至按下“ctrl+c”终止。

```
$ rosrun intera_examples joint_position_waypoints.py
```

③、来启动该节点,另外还有 3 个参数：

- 1) -h: 可选，帮助信息。
- 2) -s: 可选，速度信息，0.0 - 1.0，默认 0.3。
- 3) -a: 可选，精度，默认 0.04)。

④、节点使用命令：表示：机器人以“0.1”的速率和“0.1”的精度运行。此时执行一个连续动作，并按下 Sawyer 手臂上的“OK”键，便会将这个连续动作记录：多次记录动作，会将前面已记录的动作覆盖，默认为记录一次动作：当终端上出现至少一个“Waypoint Recorded”时，按下“Rethink”按钮 Sawyer 便会循环执行上一次记录的动作，直至按下“ctrl+c”终止。

```
$ rosrun intera_examples joint_position_waypoints.py -s 0.1 -a 0.1
```

⑤、源代码网站参考：

https://sdk.rethinkrobotics.com/intera/Joint_Position_Waypoint_Example

1-12、控制关节的力矩弹簧示例：

①、介绍：

关节扭矩弹簧示例演示了使用扭矩控制器将扭矩命令应用于机器人关节。在“main()”函数中，来自服务器的动态重新配置被初始化并传递给“jointsprings”类的对象。在初始化时，对象“limb”为传递的特定边创建类的实例，并从动态重配置服务器捕获默认的弹簧和阻尼常量。该“main()”函数调用该方法，“move_to_neutral()”将肢体发送到中立位置。然后，“the attach_spring()”方法捕获肢体的初始关节角度，并调用“update_forces()”基于胡克定律计算必要扭矩的方法，并将其作为联合命令发布。

②、案例运行使用命令：表示：实现力矩的控制,运行完命令之后，Sawyer 会先运行到“neutral”姿态,进入力矩的空档，此时 Sawyer 是处于零重力状态的。可以用手去推动，但是会有 Sawyer 会带有一点回力。

```
$ rosrun intera_examples joint_torque_springs.py
```

③、来启动该节点，另外还有 2 个参数：

1) -h 或 help: 可选，帮助信息。

2) -l {right}: 可选，选择手臂，Sawyer 默认右手臂。

④、源代码网站参考：

https://sdk.rethinkrobotics.com/intera/Joint_Torque_Springs_Example

1-13、关节轨迹回放示例：

①、概观：

关节轨迹动作界面是机器人手臂运动控制中常用的一种 ROS 方法。

(trajectory_controller) 及其对应的关节轨迹动作服务器是支持该动作接口的“intera_interface”实现。该示例示出了用于启动关节轨迹动作服务器以及将描述带时间戳的关节位置目标的文本文件解析为要命令到关节轨迹动作服务器的关节轨迹动作调用的用法。

②、介绍：

此示例演示了“Joint Trajectory Action server”命令原始关节位置命令的用法。

“main()、Trajectory”类的创建和实例并调用该“parse_file()”方法。此方法负责解析输入文件并将其打包为请求消息“Joint Trajectory Action server”。“start()”然后调用该方法，该方法将请求消息发送到动作服务器。最后，调用“wait()”，验证轨迹执行是否成功。动作服务器“-robot/limb/right/follow_joint_trajectory”。

③、案例运行使用命令：

如 1-14、1-15 讲解所说

4)、源代码网站参考：

https://sdk.rethinkrobotics.com/intera/Joint_Trajectory_Playback_Example

1-14、记录关节运动示例：

①、案例运行使用命令：表示：就会记录你控制 Sawyer 执行的连续动作并保存在“test_recorder”文件中，如果不指定路径，就默认保存在工作区间“catkin_ws”文件夹下，如果指定绝对路径，就保存在绝对路径下。如果文件已经在相同的路径下保存了一份，再次记录就会覆盖之前的内容。注意：您可以在录制时使用机器人的袖口按钮打开和关闭抓取器：椭圆形（下）=关闭（Close），圆形（上）=打开（Open）。

```
$ rosrn intera_examples joint_recorder.py -f test_recorder
```

②、来启动该节点，另外还有 3 个参数：

1) -h 或 --help: 帮助信息。

2) -r RECORDRATE: 记录速递，缺省 100。

3) -f FILENAME: 必选参数,保存记录文件的路径。

③、记录的文件内容格式为：

```
Time、right_j0、right_j1、right_j2、right_j3、right_j4、right_j5、right_j6、right_gripper  
0.2791、0.3556、0.54189、...
```

④、源代码网站参考：

https://sdk.rethinkrobotics.com/intera/Joint_Trajectory_Playback_Example

1-15、重演关节运动轨迹：

①、案例运行使用命令：表示：循环执行 3 次 “test_recorder” 文件里保存的运动轨迹。

```
$ rosrun intera_examples joint_position_file_playback.py -f test_recorder -n 3
```

②、来启动该节点，另外还有 4 个参数：

- 1) -f: 必选参数，已记录关节运动的路径文件名。
- 2) -h: 可选参数，帮助信息。
- 3) -l: 可选参数，选择手臂，默认为右臂。
- 4) -n: 可选参数，循环播放次数，默认为 1 次。

③、源代码网站参考：

https://sdk.rethinkrobotics.com/intera/Joint_Trajectory_Playback_Example

1-16、关节轨迹运行服务测试：

①、案例运行使用命令：表示：首先通过节点来启动关节控制器。

```
$ rosrun intera_interface joint_trajectory_action_server.py
```

②、来启动该节点，另外还有 4 个参数：

- 1) -h 或 help: 可选参数，帮助信息。
- 2) -l {right}: 可选参数，选择手臂，默认右臂。
- 3) -r RATE: 可选参数，速率，默认 100。
- 4) -m {position_w_id,position,velocity}: 可选参数，轨迹执行的控制模式，默认 “position_w_id”。

③、然后通过节点，来执行已经记录的轨迹运动。

```
$ rosrun intera_examples joint_trajectory_file_playback.py
```

④、来启动该节点，另外还有 4 个参数：

- 1) -h 或 help: 可选参数，帮助信息。
- 2) -l {right}: 可选参数，选择手臂，默认右臂。
- 3) -n 或 loops: 可选参数，循环播放次数，默认为 1 次。
- 4) -f 或 file: 必选参数，已记录关节运动的路径文件名。

⑤、“-m position_w_id”节点命令使用方法：使能机器人以后先通过终端输入①命令来启动服务器，再打开新的终端使能机器人以后通过终端输入②命令来执行运动轨迹。注意：来执行运动轨迹时，如提示一些错误信息，说某些关节位置超出错误门限，比如：

“Exceeded Error Threshold on right_j4:0.20672507”，多测试几次的错误信息，基本都是某个关节超过“0.2”以上就是会出错，因此“position_w_id”模式下的运动轨迹应该时小幅度运动时的精细动作的轨迹模式，某些关节超出范围便会不执行。

```
$ rosrn intera_interface joint_trajectory_action_server.py -m position_w_id
```

```
$ rosrn intera_examples joint_trajectory_file_playback.py -f test_recorder
```

⑥、“-m position”节点命令使用方法：使能机器人以后先通过终端输入①命令来启动服务器，再打开新的终端使能机器人以后通过终端输入②命令来执行运动轨迹。注意：来执行“test_recorder”文件保存的运动轨迹，在这种模式下，运动轨迹基本全部回放，和录制时的动作一样。

```
$ rosrn intera_interface joint_trajectory_action_server.py -m position
```

```
$ rosrn intera_examples joint_trajectory_file_playback.py -f test_recorder
```

⑦、“-m velocity”节点命令使用方法：使能机器人以后先通过终端输入①命令来启动服务器，再打开新的终端使能机器人以后通过终端输入②命令来执行运动轨迹。注意：来执行“test_recorder”文件保存的运动轨迹时，只会将动作大致执行，并没有将关节运动记录点全部回放。

```
$ rosrn intera_interface joint_trajectory_action_server.py -m velocity
```

```
$ rosrn intera_examples joint_trajectory_file_playback.py -f test_recorder
```

5)、源代码网站参考：

https://sdk.rethinkrobotics.com/intera/Joint_Trajectory_Playback_Example

1-17、设置交互控制示例：

①、介绍：

“sdk/intera 5.2”以上版本更新中的新功能！此交互选项设置示例演示如何使用交互控制模式在当前“ARM”配置中生成所需的交互行为。

②、案例运行使用命：表示：启动后将打印出默认的交互控制命令消息，如下图所示：默认情况下，交互控制命令消息将以“10Hz”发布。

```
$ rosrn intera_examples set_interaction_options.py
```

```
[INFO] [1563246112.549082]: header:
  seq: 1
  stamp:
    secs: 1563246112
    nsecs: 548079967
  frame_id: "base"
interaction_control_active: True
K_impedance: [1300.0, 1300.0, 1300.0, 30.0, 30.0, 30.0]
max_impedance: [True, True, True, True, True, True]
D_impedance: []
K_nullspace: [5.0, 10.0, 5.0, 10.0, 5.0, 10.0, 5.0]
force_command: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
interaction_frame:
  position:
    x: 0
    y: 0
    z: 0
  orientation:
    x: 0
    y: 0
    z: 0
    w: 1
endpoint_name: "right_hand"
in_endpoint_frame: False
disable_damping_in_force_control: False
disable_reference_resetting: False
interaction_control_mode: [1, 1, 1, 1, 1, 1]
rotations_for_constrained_zeroG: False
```

③、节点使用命令：表示通过传递“set_interaction_options.py the-h”、“help”参数，查看命令行上的可用参数：

```
$ rosrn intera_examples set_interaction_options.py -h
```

④、源代码网站参考：

https://sdk.rethinkrobotics.com/intera/Set_Interaction_Options_Example

https://sdk.rethinkrobotics.com/intera/Interaction_Control_Tutorial