6) Basic Takktile actions

The next tutorial is 7) Writing a basic Takktile script

The ReFlex fingers (both Takktile and SF) can be position controlled, velocity controlled, or force controlled. In addition, the Takktile hand can be configured so that fingers stop on contact with an object. Make sure the hand has been calibrated before using any of these commands.

Position control

As seen in the previous tutorial, the fingers can be sent a set of position commands.

```
roslaunch reflex reflex_takktile.launch
rostopic pub /reflex_takktile/command_position reflex_msgs/PoseCommand "f1: 1.0
f2: 1.0
f3: 1.0
preshape: 0.0"
```

NOTE: The fingers will stop when overloaded in this state, as mentioned in the Safety Overloads section of the previous tutorial. When tactile stops are enabled, a finger in this state will halt its motion upon contact.

Velocity control

Velocity control can be used to make the motors open or close at a given speed. This could be used to slowly creep in until contact is made, or it could just be a simple way of quickly clamping down on an object within the hand. The fingers will close when commanded with positive velocity, open when commanded with negative. The motors will stop when they reach their travel limits, whether that is closed or open.

```
# Assuming code is still running
rostopic pub /reflex_takktile/command_velocity reflex_msgs/VelocityCommand "f1: 5.0
f2: 1.0
f3: 0.25
preshape: 0.0"
rostopic pub /reflex_takktile/command_velocity reflex_msgs/VelocityCommand "f1: -0.
f2: -1.0
f3: -5.0
preshape: 0.0"
```

NOTE: The fingers will stop when overloaded in this state, as mentioned in the Safety Overloads section of the previous tutorial. When tactile stops are enabled, a finger in this state will halt its motion upon contact.

Combined position and velocity control

Combining position and velocity control is simple, the motors can be commanded to reach a given position with a given velocity. This allows you flexibility in the tasks you attempt with the hand.

```
# Assuming code is still running
rostopic pub /reflex_takktile/command reflex_msgs/Command "pose:
  f1: 3.0
  f2: 2.0
  f3: 1.0
  preshape: 0.0
velocity:
  f1: 0.5
  f2: 1.0
  f3: 5.0
  preshape: 0.0"
rostopic pub /reflex_takktile/command reflex_msgs/Command "pose:
  f1: 0.0
  f2: 2.0
  f3: 1.0
```

```
  preshape: 0.0
velocity:
  f1: 0.5
  f2: 5.0
  f3: 1.0
  preshape: 0.0"
```

NOTE: The easiest way to enter this type of command in terminal is to type it up through Command, regularly tab-completing the longer sections, then hit tab to auto-fill the portion with f1, f2, etc. It is a pain to type that in by hand. You can also add a -1 right after pub to make the command send once and end, but then it won't tab-complete. I usually make the whole command and then go back to add -1. Such is the rostopic terminal tool.

Rough force control

How to send a command

An interesting feature in the Takktile and SF code is force control, the motors can be commanded to a force and a control loop in the code tries to follow that command. To get out of this mode, just send another type of command, like those shown above.

```
# Assuming code is still running
rostopic pub /reflex_takktile/command_motor_force reflex_msgs/ForceCommand "f1: 200
f2: 200.0
f3: 200.0
preshape: 0.0"
```

NOTE: Whether or not tactile stops are enabled, a finger in this state will NOT halt its motion upon contact, because the idea of force control is to provide relatively constant pressure while in contact. The finger will loosen when overloaded, leading to oscillations if the commanded force is too close to the overload threshold. More details about the overload threshold can be found in the Safety Overloads section of the previous tutorial

NOTE: The control code attempts to keep the force on the actuating motor constant, which means that actual applied force will vary depending on where along the finger the force is applied. This is because the finger acts as a variable lever arm. It could perhaps be compensated for using tactile data to locate the object on the finger.

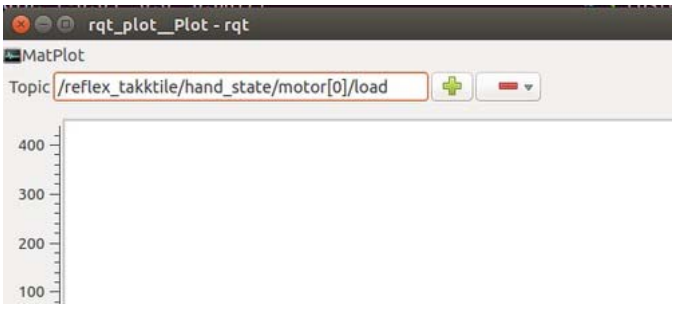
How to monitor the current motor loads

ROS has a simple tool for plotting data being published on topics, rqt_plot. If you want to see the current motor loads, run the plot tool:

```
rqt_plot
```

You'll see a place to enter a topic to plot. To live plot the motor loads, enter the following topics:

```
/reflex_takktile/hand_state/motor[0]/load
/reflex_takktile/hand_state/motor[1]/load
/reflex_takktile/hand_state/motor[2]/load
/reflex_takktile/hand_state/motor[3]/load
```



Using tactile stops

When the hand is running, if you view the visualizer or echo the /hand_state topic you can see whether the sensors are in contact. When we enable tactile stops, contact will stop the motor from moving. Let's try enabling tactile stops, zeroing the tactile sensors, and closing the hand.

```
rosservice call /reflex_takktile/enable_tactile_stops
rosservice call /reflex_takktile/calibrate_tactile
rostopic pub /reflex_takktile/command_velocity reflex_msgs/VelocityCommand "f1: 1.0
```

```
f2: 1.0
f3: 1.0
preshape: 0.0"
```

If you touch the fingers while they are closing, they should register contact and stop. If things aren't working as expected, try running the visualizer or echoing the hand_state topic to view whether the tactile sensors are in contact.

If you want to disable tactile stops so that the hand won't react to contact, just call the following service:

```
rosservice call /reflex_takktile/disable_tactile_stops
```

Finally, remember that you can reset the thresholds on each sensor to determine which sensors can cause the hand to stop and what level they register contact at. The tutorial on configuring the pressure sensors will recap how to do that.

Keep going on to 7) Writing a basic Takktile script

Add a New Comment