

AdmittanceController

AdmittanceRule

update()

Convert to Cartesian deltas
in ik_base frame. (2)

Add Cartesian deltas to
current pose to get the
reference pose (aka target
pose)

Calculate pose error (3)
Pose_error = current - reference

Low-pass filter wrench & apply
gravity comp. Transform to
ik_base frame. (4)

Apply deadband to wrench

Calculate admittance rule (4)

$$a[i] = (1/m) * (w[i] - D * v[i] - K * x[i])$$

a[i] = acceleration component
m = mass
w[i] = wrench component
D = damping
v[i] = velocity component
K = spring constant
x[i] = pose error component

Integrate acceleration to relative desired
pose

Convert the relative Cartesian pose back
to delta-theta. Sum with current joint state
to get new joints for output

trajectory_msgs::JointTrajectoryPoint
Joint angle deltas

trajectory_msgs::JointTrajectoryPoint
Current joints from hardware
(1)

geometry_msgs::Wrench
Measured wrench
Can be in any frame.

Definitions:
ik_base frame- The root frame of the
URDF and MoveIt's working frame

Notes:

- (1) In open-loop mode, to allow joint offsets, the current joints are assumed equal to the previously-commanded joints.
- (2) It's important to do the admittance calculations in a stationary reference frame or vel/accel calcs will not be accurate. We make the assumption that ik_base frame is stationary.
- (3) In open-loop mode, the pose error is a sum of all admittance displacements since the last reset, because (current-reference) does not apply.
- (4) Major TODO: need to add the capability to transform the wrench to other frames.

Edit here:

https://docs.google.com/drawings/d/1AphphGxenALNeDEXFv_NQ_IOJpyi6_GLHUH2gTFJYMg/edit

trajectory_msgs::JointTrajectoryPoint
Output joint angles