Advanced Robotic Manipulation Handout

CS327A (Spring 2017)

Solution Set #4

Problem 1 - Centralized co-operative manipulation

Solution: The solution code to this problem is available under `cs327a/hw4_sol/p1-main-sol.cpp`. From within `cs327a`, run

```
$ git stash
$ git pull --rebase
$ git stash pop
$ pushd ../build && cmake -DCMAKE_BUILD_TYPE=Release .. && make && popd
$ ./hw4-p1-sol
```

Problem 3 - Whole body control

(b) Constraint controller design

The constraint considered in this problem is to maintain a safe distance from the red sphere. One way to do so is to create an artificial potential field.

Let the surface of the sphere be located at a distance d from the surface of the closest link, link i on the PUMA. Also let $x_{c,i}$ be the location of the point on link i closest to the sphere. Finally, let c be the location of the center of the sphere.

Design a potential function $U_c(d)$ whose gradient produces a repulsive force field in the direction $n_c = (c - x_{c,i})/\|c - x_{c,i}\|$. Refer to page 4 on the "Lecture10 - Elastic planning" handout for a possible design. Choose a cut-off distance d_0 such that the potential as well as its gradient are zero for $d \geq d_0$.

The control torques to maintain a safe distance from the sphere can then be written as

$$\Gamma_c = J_c^T \nabla U_c + J_c^T \Lambda_c (-k_{vc} J_c \dot{q})$$

where the additional term on the right is to damp motions in the constraint direction. J_c is the constraint Jacobian given by

$$J_c = n_c^T J_v (x_{c,i})$$

where $J_v (x_{c,i})$ is the linear velocity Jacobian at the point $x_{c,i}$ on link i.

We consider the constraint "active" if $d < d_0$ when the sphere gets too close to the PUMA, else we consider it "inactive". When the constraint is "inactive", we set $\Gamma_c = 0$.

Solution: One possible potential field is as follows:

$$U_c = \frac{1}{2}\eta\left(\frac{1}{d} - \frac{1}{d_0}\right)^2$$

where for the purpose of this problem, we consider $\eta = 0.3Nm^3$ and $d_0 = 0.095m$.

(c) Task controller design

For this problem, we consider the task of maintaining the end-effector of the PUMA along the following desired trajectory:

$$\begin{aligned} x_{ee,d} &= 0.7 \\ y_{ee,d} &= 0.2 + 0.4\sin\left(\frac{2\pi t}{6}\right) \\ z_{ee,d} &= 0.5 \end{aligned}$$

Design a PD controller such that the end-effector follows the above trajectory while not violating the constraint. That is, if the constraint is "active", the task control torque should not produce any acceleration in the constraint direction. In other words, we require that

$$J_c A^{-1}\Gamma_{t|c} = 0$$

when the constraint is "active".

The following template might be useful:

$$\Gamma_{t|c} = J_{t|c}^T \Lambda_{t|c} F_{motion}^*$$

where $J_{t|c}$ is the constraint consistent task Jacobian. If the constraint is "active", then

$$J_{t|c} = J_t N_c$$

else, if the constraint is "inactive"

$$J_{t|c} = J_t.$$

The constraint consistent task space inertia $\Lambda_{t|c}$ is given by $\left(J_{t|c}A^{-1}J_{t|c}^T\right)^{-1}$.

Solution: We design a dynamically decoupled PD controller for the end-effector task as follows:

$$\Gamma_{t|c} = J_{t|c}^T\left(\Lambda_{t|c}(-k_{px}(\mathbf{x}_{ee} - \mathbf{x}_{ee,d}) - k_{vx}\dot{\mathbf{x}}_{ee})\right)$$

where $\dot{\mathbf{x}}_{ee} = J_t\dot{q}$ is the end-effector velocity. Note that the end-effector velocity is still measured with respect to the task Jacobian J_t and not with respect to the constraint consistent task Jacobian $J_{t|c}$.

Optional: The above controller will work as expected, but it can be made better. Currently, we allow the constraint torques Γ_c to disturb the task by producing accelerations given by $J_t A^{-1}\Gamma_c$

which are non-zero as long as the constraint is "active". Try to modify F^*_{motion} to compensate for these accelerations.

Solution: By the current design, the instantaneous task acceleration, assuming a perfect model estimate is given by:

$$\begin{aligned}\ddot{\mathbf{x}}_{ee} &= J_t\ddot{q} + \dot{J}_t\dot{q} \\ &= J_t\left(A^{-1}\Gamma_c + A^{-1}\Gamma_{t|c} + A^{-1}\Gamma_{p|t|c} - A^{-1}b\right) + \dot{J}_t\dot{q}\end{aligned}$$

Since $A^{-1}\Gamma_{p|t|c} = 0$ by design, we get

$$= J_tA^{-1}\Gamma_c + J_tA^{-1}N_c^TJ_t^T\Lambda_{t|c}F^*_{motion} - J_tA^{-1}b + \dot{J}_t\dot{q}$$

which, since $J_tA^{-1}N_c^TJ_t^T = \Lambda^{-1}_{t|c}$ when the task is not singular, yields

$$= J_tA^{-1}\Gamma_c + F^*_{motion} - J_tA^{-1}b + \dot{J}_t\dot{q}$$

Thus, ignoring the disturbance from centrifugal and Coriolis forces, we still have an additional task acceleration component $J_tA^{-1}\Gamma_c$ from the constraint torques. If we want to compensate for them, we can redesign F^*_{motion} to be

$$F^*_{motion} = -k_{px}(\mathbf{x}_{ee} - \mathbf{x}_{ee,d}) - k_{vx}\dot{\mathbf{x}}_{ee} - J_tA^{-1}\Gamma_c$$

(d) Posture controller design

Finally, design a posture PD controller $\Gamma_{p|t|c}$ that holds the following joint angles:

$$q_d = \begin{bmatrix}0.00\\5.90\\3.70\\1.57\\1.75\\3.14\end{bmatrix}.$$

However, ensure that the posture control torques produce no acceleration either in the constraint or in the task co-ordinates. That is, we require

$$J_cA^{-1}\Gamma_{p|t|c} = 0 \quad \text{and} \quad J_tA^{-1}\Gamma_{p|t|c} = 0$$

Solution: To ensure that the posture control torques do not affect the end-effector task and the constraint, we have to filter them through the shared torque null space for the constraint and the task Jacobians. The following design achieves exactly this:

$$\Gamma_{p|t|c} = N_c^TN_{t|c}^T\left(A(-k_{pj}(q - q_d) - k_{vj}\dot{q})\right)$$

where $N_c = I - \bar{J}_cJ_c$ and $N_{t|c} = I - \bar{J}_{t|c}J_{t|c}$ are the dynamically consistent null space projection matrices for J_c and $J_{t|c}$ respectively.

(e) Implementation

Now implement your controller in the portion marked as `FILL ME IN` in `cs327a/hw4/p3-main.cpp`. Once you have your controller running, try moving the sphere around to test its robustness. Write a brief summary of the observed behavior.

Solution: The solution code to this problem is available under `cs327a/hw4_sol/p3-main-sol.cpp`. From within `cs327a`, run

```
$ pushd ../build && cmake -DCMAKE_BUILD_TYPE=Release .. && make && popd
$ ./hw4-p3-sol
```

You should observe that the robot changes its posture to avoid the obstacle while keeping the end-effector on the desired trajectory. However if the obstacle is in such a position that the task is unachievable, the controller prioritizes avoiding the obstacle over performing the task, as designed. Also, if the obstacle is removed from the vicinity of the robot, the robot goes back to a desirable posture with the elbow pointing up.