Advanced Robotics

Inverse dynamics control

Subramanian Ramamoorthy
School of Informatics
University of Edinburgh

References

These slides are adapted, with permission, from lecture notes of Andrea Del Prete.

See also Ch 11 of Lynch and Park, Modern Robotics.

Summary on rigid body motion

- The general form for the dynamics equation of articulated robots is:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) = \tau$$

- Assuming robot is fully actuated and that $\mathbf{v}_{\mathbf{q}} = \dot{\mathbf{q}}$
- This describes dynamics in the *configuration* space
- As with geometry / kinematics, we are often mainly interested in the *task* space

Outline

- ❑ Inverse dynamics control in the configuration space
- ❑ Task-space inverse dynamics

Reminder on inverse dynamics

- Given **q**, $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$, compute torque commands τ that achieve desired acceleration $\ddot{\mathbf{q}}^d$

- Given a reference $\mathbf{q}^r(t)$ find $\tau(t)$ such that resulting $\mathbf{q}(\tau(t))$ follows $\mathbf{q}^r(t)$

- We assume we can measure **q** and $\dot{\mathbf{q}}$

Reminder on inverse dynamics

- Given **q**, $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$, compute torque commands τ that achieve desired acceleration $\ddot{\mathbf{q}}^d$.
- Given a reference $\mathbf{q}^r(t)$ find $\tau(t)$ such that resulting $\mathbf{q}(\tau(t))$ follows $\mathbf{q}^r(t)$
- We assume we can measure **q** and $\dot{\mathbf{q}}$
- We set $\tau = \mathbf{M}\ddot{\mathbf{q}}^d + \mathbf{h}$, and now we must compute desired $\ddot{\mathbf{q}}^d$

$$\ddot{\mathbf{q}}^d = \ddot{\mathbf{q}}^r$$

Inverse dynamics control in a nutshell

- Given $\mathbf{q}$, $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$, compute torque commands τ that achieve desired acceleration $\ddot{\mathbf{q}}^d$.
- Given a reference $\mathbf{q}^r(t)$ find $\tau(t)$ such that resulting $\mathbf{q}(\tau(t))$ follows $\mathbf{q}^r(t)$
- We assume we can measure $\mathbf{q}$ and $\dot{\mathbf{q}}$
- We set $\tau = \mathbf{M}\ddot{\mathbf{q}}^d + \mathbf{h}$, and now we must compute desired $\ddot{\mathbf{q}}^d$

$$\ddot{\mathbf{q}}^d = \ddot{\mathbf{q}}^r$$?

Inverse dynamics control in a nutshell

- Given **q**, $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$, compute torque commands τ that achieve desired acceleration $\ddot{\mathbf{q}}^d$.

- Given a reference $\mathbf{q}^r(t)$ find $\tau(t)$ such that resulting $\mathbf{q}(\tau(t))$ follows $\mathbf{q}^r(t)$

- We assume we can measure **q** and $\dot{\mathbf{q}}$

- We set $\tau = \mathbf{M}\ddot{\mathbf{q}}^d + \mathbf{h}$, and now we must compute desired $\ddot{\mathbf{q}}^d$

$$\ddot{\mathbf{q}}^d = \underbrace{\ddot{\mathbf{q}}^r}_{\ddot{\mathbf{e}}} - \mathbf{K}_p(\underbrace{\mathbf{q} - \mathbf{q}^r}_{\mathbf{e}}) - \mathbf{K}_v(\underbrace{\dot{\mathbf{q}} - \dot{\mathbf{q}}^r}_{\dot{\mathbf{e}}})$$

Simpler control laws for manipulator

$$\tau = \underbrace{-K_d\dot{\mathbf{e}} - K_p\mathbf{e}}_{\text{PD}} + \mathrm{g}(\mathrm{q})$$

gravity torque

Even simpler is PID control:

$$\tau = -K_d\dot{\mathbf{e}} - K_p\mathbf{e} + \int_0^t K_i e(s)ds$$

Where integral replaces gravity compensation

All these control laws are stable. In theory, ID control > PD + gravity > PID

Inverse Dynamics control as optimisation problem

- As for inverse kinematics, we can write a least square problem:

$$(\tau^*, \ddot{q}^*) = \underset{\tau, \ddot{\mathbf{q}}}{\textbf{argmin}} \; ||\ddot{\mathbf{q}} - \ddot{\mathbf{q}}^d||^2$$

Subject to $$\tau = \mathbf{M}\ddot{\mathbf{q}} + \mathbf{h}$$

- The optimal solution to this is exactly the ID control law if we set

$$\ddot{\mathbf{q}}^d = \ddot{\mathbf{q}}^r - \mathbf{K}_p(\mathbf{q} - \mathbf{q}^r) - \mathbf{K}_v(\dot{\mathbf{q}} - \dot{\mathbf{q}}^r)$$

- So there may be no real advantage here, but the more general framing is useful for more complex problems

Least Square Problem (LSP) (reminder)

- LSP taxonomy:
  - An L_2 norm cost $||Ax - b||^2$
  - Possibly linear inequality / equality constraints (Cx <= d ; D x = x)
- LSPs are a sub-class of convex Quadratic Problems (QPs) which have:
  - Quadratic cost $x^T H x + h^T x$, with H >= 0
  - Possibly linear inequality / equality constraints (Cx <= d ; D x = x)
- LSPs and QPs can be solved **extremely** fast with off-the-shelf software
  => compatible with real-time control loops (~ 1 KHz)

Main advantage of optimisation is constraints

- e.g., adding torque limits is much more straightforward:

$$(\tau^*, \ddot{q}^*) = \underset{\tau, \ddot{\mathbf{q}}}{\textbf{argmin}} \; ||\ddot{\mathbf{q}} - \ddot{\mathbf{q}}^d||^2$$

Subject to

$$\tau = \mathbf{M}\ddot{\mathbf{q}} + \mathbf{h}$$

$$\tau^- \leq \tau \leq \tau^+$$

Main advantage of optimisation is constraints

- Assuming constant aceleration at each time step,

$$\dot{\mathbf{q}}(t + \Delta t) = \dot{\mathbf{q}}(t) + \Delta t \ddot{\mathbf{q}}$$

- Joint velocities constraints:

$$(\tau^*, \ddot{q}^*) = \underset{\tau, \ddot{\mathbf{q}}}{\textbf{argmin}} \; ||\ddot{\mathbf{q}} - \ddot{\mathbf{q}}^d||^2$$

Subject to

$$\tau = \mathbf{M}\ddot{\mathbf{q}} + \mathbf{h}$$

$$\tau^- \leq \tau \leq \tau^+$$

$$\dot{\mathbf{q}}(t)^- \leq \dot{\mathbf{q}}(t) + \Delta t \ddot{\mathbf{q}} \leq \dot{\mathbf{q}}(t)^+$$

Main advantage of optimisation is constraints

- Likewise for joint limits:

$$\mathbf{q}(t+\Delta t) = \mathbf{q}(t) + \Delta t\dot{\mathbf{q}}(t) + \frac{1}{2}\Delta t^2 \Delta t\ddot{\mathbf{q}}$$

- However, we need caution, as this can result in high accelerations
  - Incompatible with torque / current constraints
  - Leads to infeasible problems (i.e. no solutions may exist)
  - These issues are addressed in the research literature, but we will not discuss them further here

Task space inverse dynamics

- Joint space ID control expects reference $\mathbf{q}^r(t)$
- What if we only have reference **end-effector trajectory** $\mathbf{x}^r(t)$?
    - Option 1: compute corresponding $\mathbf{q}^r(t)$ then apply ID control
        - Issue 1: this is the inverse geometry problem, non-linear problem with infinity of solutions
        - Issue 2: Tracking $\mathbf{q}^r(t)$ is **sufficient** but not necessary to track $\mathbf{x}^r(t)$

          This means that perturbations that affect $\mathbf{q}^r(t)$ but not the Forward Geometry FG(**q**) are rejected
    - What might an option 2 be?

Task space Inverse Dynamics: option 2

- End-effector control. Feeback directly effector configuration

$$\dot{\mathcal{V}}^d = \dot{\mathcal{V}}^r - K_d(\mathcal{V} - \mathcal{V}^r) - K_p(\mathbf{x} - \mathbf{x}^r)$$

Task space Inverse Dynamics: option 2

- End-effector control. Feeback directly effector configuration

$$\dot{\mathcal{V}}^d = \dot{\mathcal{V}}^r - K_d(\mathcal{V} - \mathcal{V}^r) - K_p(\mathbf{x} - \mathbf{x}^r)$$

- Let's differentiate $\mathcal{V}$:

$$\mathcal{V} = \mathbf{J}\dot{\mathbf{q}}$$

$$\dot{\mathcal{V}} = \mathbf{J}\ddot{\mathbf{q}} + \dot{\mathbf{J}}\dot{\mathbf{q}}$$

Task space Inverse Dynamics: option 2

- End-effector control. Feeback directly effector configuration

$$\dot{\mathcal{V}}^d = \dot{\mathcal{V}}^r - K_d(\mathcal{V} - \mathcal{V}^r) - K_p(\mathbf{x} - \mathbf{x}^r)$$

- Let's differentiate $\mathcal{V}$:

$$\mathcal{V} = \mathbf{J}\dot{\mathbf{q}}$$

$$\dot{\mathcal{V}} = \mathbf{J}\ddot{\mathbf{q}} + \dot{\mathbf{J}}\dot{\mathbf{q}}$$

- As a result, desired acceleration should be

$$\ddot{\mathbf{q}}^d = \mathbf{J}^+(\dot{\mathcal{V}}^d - \dot{\mathbf{J}}\dot{\mathbf{q}})$$

Task space Inverse Dynamics: option 2

- End-effector control. Feeback directly effector configuration

$$\dot{\mathcal{V}}^d = \dot{\mathcal{V}}^r - K_d(\mathcal{V} - \mathcal{V}^r) - K_p(\mathbf{x} - \mathbf{x}^r)$$

- Let's differentiate $\mathcal{V}$:

$$\mathcal{V} = \mathbf{J}\dot{\mathbf{q}}$$

$$\dot{\mathcal{V}} = \mathbf{J}\ddot{\mathbf{q}} + \dot{\mathbf{J}}\dot{\mathbf{q}}$$

- As a result, desired acceleration should be

$$\ddot{\mathbf{q}}^d = \mathbf{J}^+(\dot{\mathcal{V}}^d - \dot{\mathbf{J}}\dot{\mathbf{q}})$$

Again, the torques are obtained straightforwardly as $\tau = \mathbf{M}\ddot{\mathbf{q}}^d + \mathbf{h}$

Option 2 is often preferred

+ Gains defined in Cartesian space

+ No pre-computations

+ Online specification of reference trajectory

- More complex controller

Generalising the notion of task

- Not all tasks are just a matter of tracking end-effector trajectories

- Task = a control objective (as in examples at the start of the control lecture)

- A task can be described as a function *e* to minimise error (as in optimal control)

    - Denote e as measuring the **error** between the **real** and **reference** outputs

$$\underbrace{e(\mathbf{x}, \mathbf{u}, t)}_{\text{error}} = \underbrace{y(\mathbf{u}, t)}_{\text{measure}} - \underbrace{y^*(t)}_{\text{reference}}$$

- A large variety of such tasks can then fit into ID control. Relevant ones for your labs are postural tasks (tracking a reference configuration) and force control tasks, e.g. for contact interactions.

A very short note on contacts (for the lab)

- We have seen that

$$\tau = \mathbf{M}\ddot{\mathbf{q}} + \mathbf{h}$$

- What if we introduce contacts?

We can write

$$\tau = \mathbf{M}\ddot{\mathbf{q}} + \mathbf{h} + \mathbf{J}^T\mathbf{f}_c$$

Where $\mathbf{f}_c$ is a 6D contact force. To control your robot for lifting the cube, you can set a desired $\mathbf{f}_c$ on both effectors and use the control laws we have used before to compute the appropriate torques.

If equiped with a force sensor, you could also implement a PI control to track the error accurately.

More info on: https://scaron.info/robotics/joint-torques-and-jacobian-transpose.html