Lecture 6: Whole Body Control for Humanoid Robots

Nicola Scianca

December 2024

introduction

- **Whole-Body Control** (WBC) aims to coordinate all joints of the robot to achieve desired motions while satisfying physical constraints
- the desired motions are specified as a set of **tasks**, such as the trajectory of the Center of Mass (CoM), the feet, the hands, etc...
- our goal is to find **joint torques** that realize these tasks, while also keeping the robot balanced

trajectory tracking

- on a **fully actuated fixed-base manipulator**, we can use its dynamics model to do **inverse dynamics**

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

  - q: the joint positions
  - $\dot{q}$: joint velocities
  - τ: joint torques
  - $M(q)$: inertia matrix
  - $n(q,\dot{q})$: other terms (centrifugal, Coriolis, gravity)

trajectory tracking

- using the dynamic model we can find the torque τ necessary to realize some joint acceleration $\ddot{q}_d$

$$\tau = M(q)\ddot{q}^d + n(q, \dot{q})$$

- the task given as an **end-effector trajectory**, specified by position $p_\mathrm{e}(t)$, velocity $v_\mathrm{e}(t)$, and acceleration $a_\mathrm{e}(t)$

- we can use a standard **feedforward + PD**

$$a_\mathrm{e} = J_\mathrm{e}\ddot{q} + \dot{J}_\mathrm{e}\dot{q} = \underbrace{a_\mathrm{e}^d}_{\text{fedforward}} + \underbrace{k_p(p_\mathrm{e}^d - p_\mathrm{e})}_{\text{proportional term}} + \underbrace{k_d(v_\mathrm{e}^d - v_\mathrm{e})}_{\text{derivative term}}$$

trajectory tracking

- we can find a suitable joint acceleration $\ddot{q}^d$ using the **pseudoinverse** of the task Jacobian $J_{\mathrm{e}}^{\dagger} = J_{\mathrm{e}}^T \left(J_{\mathrm{e}} J_{\mathrm{e}}^T\right)^{-1}$

$$\ddot{q}^d = J_{\mathrm{e}}^{\dagger}\left(a_{\mathrm{e}}^d + k_p(p_{\mathrm{e}}^d - p_{\mathrm{e}}) + k_d(v_{\mathrm{e}}^d - v_{\mathrm{e}}) - \dot{J}_{\mathrm{e}}\dot{q}\right)$$

- by substituting this joint acceleration inside the dynamic equation, we find the torques τ

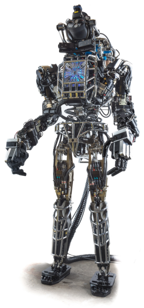
$$\tau = M(q) J_{\mathrm{e}}^{\dagger}\left(a_{\mathrm{e}}^d + k_p(p_{\mathrm{e}}^d - p_{\mathrm{e}}) + k_d(v_{\mathrm{e}}^d - v_{\mathrm{e}}) - \dot{J}_{\mathrm{e}}\dot{q}\right) + n(q, \dot{q})$$

humanoid robot

- if the robot is a humanoid, we have additional complications

$$M(q)\ddot{q} + n(q, \dot{q}) = \tau + \sum_i J_i^T(q) f_i$$

- $q = (q_\mathrm{f}, q_\mathrm{a})$ now includes not only the joint positions q_a, but also the position and orientation of the **floating base** q_f
- each contact force **contact forces** f_i affects the dynamics via the corresponding **contact Jacobian** J_i

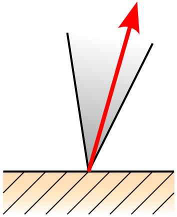


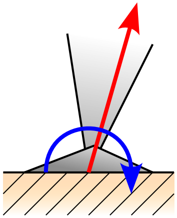
contact forces

- a point contact can only apply a **force**, therefore the contact Jacobian is the linear Jacobian of the end-effector position $(3 \times n)$

- a surface contact also applies a **torque** τ

- the stack of the corresponding force and torque $w = (f, \tau)$ is called the **contact wrench**, and the contact Jacobian is the Jacobian of the position and orientation of the end-effector $(6 \times n)$





contact constraint

- we must also keep in mind that the contact also imposes a **kinematic constraint** on the robot foot
- we express this by imposing that the velocity of the foot v_f, as well as its derivative, the acceleration a_f, must be zero

$$v_f = 0 \quad \Longrightarrow \quad a_f = J_f \ddot{q} + \dot{J}_f \dot{q} = 0$$

- how do we make sure that **any motion** that the robot makes is compatible with this constraint?

contact constraint

- to make sure that the accelerations maintain the contact, we must ensure that they are in the **null-space** of the contact Jacobian

- we can use the **null-space projector**: a matrix that projects vectors onto the null space

- the null-space projector N for a Jacobian J is then defined as

$$N = I - J^\dagger J$$

where $J^\dagger = J^\top (JJ^\top)^{-1}$ is the pseudoinverse of J

dynamics model

$$M(q)\ddot{q} + n(q, \dot{q}) = S^T \tau + \sum_i J_i^T w_i$$

- $M(q)$ is the inertia matrix, $n(q, \dot{q})$ the nonlinear term collecting centrifugal, Coriolis and gravity terms
- S is a selection matrix, τ joint torques
- J_i is the i-th contact Jacobian, and w_i is the i-th contact wrench

balancing with single contact

- if we have a single contact, the contact interaction can be expressed as a single contact wrench w

$$M(q)\ddot{q} + n(q, \dot{q}) = S^T \tau + J^T w$$

remember that J is a $6 \times (6 + n_j)$ matrix

- we can split the above equation in two parts: the first six components which deal with the floating base, and the last n_j that deal with the actuated dofs

$$M_{\text{fb}}\ddot{q}_{\text{fb}} + n_{\text{fb}} = J_{\text{fb}}^T w$$
$$M_{\text{act}}\ddot{q}_{\text{act}} + n_{\text{act}} = \tau + J_{\text{act}}^T w$$

balancing with single contact

- we can use the first equation to recover w, because J_{fb} is a square invertible matrix

$$w = J_{\text{fb}}^{-T}\left(M_{\text{fb}}\ddot{q}_{\text{fb}} + n_{\text{fb}}\right)$$

- if we substitute this into the bottom equation, we get

$$M_{\text{act}}\ddot{q}_{\text{act}} + n_{\text{act}} = \tau + J_{\text{act}}^{T} J_{\text{fb}}^{-T}\left(M_{\text{fb}}\ddot{q}_{\text{fb}} + n_{\text{fb}}\right)$$

- now we need to find our desired accelerations

balancing with single contact

- to statically balance the robot, we must keep the projection of the CoM p_G within the footprint of the foot in contact with the ground

- therefore, we define a desired CoM position as $p_G^d = (p_f^x, p_f^y, p_f^h + h)$, where h is a vertical displacement

- joint accelerations that realize this task can be found as

$$\ddot{q}^d = J_G^{\dagger}\left(a_G^d + k_p(p_G^d - p_G) - k_d v_G - \dot{J}_G \dot{q}\right)$$

balancing with single contact

- however, if we just pseudoinvert the CoM Jacobian, we will probably find desired accelerations that make our robot fly
- that is because pseudoinverting the Jacobian doesn't know anything about the underactuation, so it will use all accelerations (even those related to unactuated joints) to realize the task
- to avoid this, we need to **project** these accelerations in the **null-space** of the contact Jacobian J_c

$$\ddot{q}^d = \left(I - J_c^\dagger J_c\right) J_G^\dagger \left(a_G^d - k_p p_G + k_d(v_G^d - v_G) - \dot{J}_G \dot{q}\right)$$

balancing with single contact

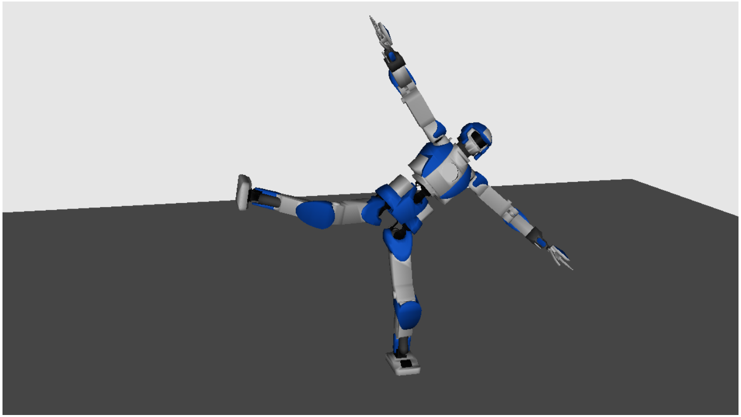
- now we can find our joint torques τ

$$\begin{pmatrix} \ddot{q}_{\text{fb}} \\ \ddot{q}_{\text{act}} \end{pmatrix} = \left(I - J_c^{\dagger} J_c\right) J_G^{\dagger} \left(a_G^d + k_p(p_G^d - p_G) - k_d v_G - \dot{J}_G \dot{q}\right)$$

$$\tau = M_{\text{act}} \ddot{q}_{\text{act}} + n_{\text{act}} - J_{\text{act}}^T J_{\text{fb}}^{-T} \left(M_{\text{fb}} \ddot{q}_{\text{fb}} + n_{\text{fb}}\right)$$

- to avoid self motions, we usually need to resolve the **redundancy**, which we can do by specifying a reference joint configuration as an additional kinematic task (see example code)

example: balancing with single contact



contact wrench constraints

- in the previous example, we always managed to generate a contact wrench that is **realizable**
- in general, this is not guaranteed: in fact, if contacts start changing, algebraic manipulations will not necessarily give us realizable contact wrenches
- we must use an approach that allows us to enforce **constraints**

contact wrench constraints

- the requirements [Caron et al., 2015] that a contact wrench must satisfy are
  - **unilaterality**: the force must push **away** from the contact surface
  - **no slipping**: if we want to avoid relative motion of the contact surfaces, we must ensure sufficient **friction**
  - **no tilting**: if we want to maintain a surface contact, we must impose constraints on the **ZMP**

Caron et al.,"Stability of surface contacts for humanoid robots: Closed-form formulae of the Contact Wrench Cone for rectangular support areas", ICRA, 2015

contact wrench constraints

- the **unilaterality** constraint is relatively straightforward: it simply requires us to enforce that

$$f_z \geq 0$$

- we can write this as a constraint on the contact wrench as

$$Aw \leq b$$

with

$$A = \begin{pmatrix} 0 & 0 & -1 & 0 & 0 & 0 \end{pmatrix}, \qquad b = \begin{pmatrix} 0 \end{pmatrix}$$

contact wrench constraints

- to achieve **no slipping**, the force must be within the **friction cone**, i.e., the **tangential force** f_t must satisfy

$$|f_t| \leq \mu f_z \quad \Longrightarrow \quad \sqrt{f_x^2 + f_y^2} \leq \mu f_z$$

  where μ is the static friction coefficient of the contact

- to make it linear, we approximate it using a smaller $\bar{\mu} < \mu$

$$-\bar{\mu} f_z \leq f_x \leq \bar{\mu} f_z$$
$$-\bar{\mu} f_z \leq f_y \leq \bar{\mu} f_z$$

- on the contact wrench, we can write it as $Aw \leq b$, with

$$A = \begin{pmatrix} 1 & 0 & \bar{\mu} & 0 & 0 & 0 \\ -1 & 0 & -\bar{\mu} & 0 & 0 & 0 \\ 0 & 1 & \bar{\mu} & 0 & 0 & 0 \\ 0 & -1 & -\bar{\mu} & 0 & 0 & 0 \end{pmatrix}, \qquad b = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

contact wrench constraints

- since the foot has a limited surface, we must ensure that the point of application of the equivalent ground reaction force (the ZMP of that particular contact surface) is within the convex hull of that contact surface
- let's analyze what is happening on a foot surface: our wrench w represents the action of a force applied at the foot center $+$ a torque
- but where is the **equivalent** ground reaction force applied on the foot surface?

contact wrench constraints

- let's compute the horizontal moments with respect to a generic point on the ground p_o

$$f_z(x_f - x_o) - \tau_y = M_o$$
$$f_z(y_f - y_o) + \tau_x = M_o$$

- if p_o is a ZMP, by definition $M_o = 0$

$$f_z(x_f - x_z) - \tau_y = 0$$
$$f_z(y_f - y_z) + \tau_x = 0$$

- therefore, the position of the ZMP is

$$x_z - x_f = -\frac{\tau_y}{f_z}$$
$$y_z - y_f = \frac{\tau_x}{f_z}$$

contact wrench constraints

- if the foot has a rectangular shape, with size $(2d_x, 2d_y)$, to keep the ZMP within the foot surface, we impose that

$$|x_z - x_f| \leq d_x$$
$$|y_z - y_f| \leq d_y$$

- this is equivalent to

$$-d_x \leq -\frac{\tau_y}{f_z} \leq d_x$$
$$-d_y \leq \frac{\tau_x}{f_z} \leq d_y$$

- on the contact wrench w, we can write as $Aw \leq b$, with

$$A = \begin{pmatrix} 0 & 0 & -d_x & 0 & -1 & 0 \\ 0 & 0 & -d_x & 0 & 1 & 0 \\ 0 & 0 & -d_y & 1 & 0 & 0 \\ 0 & 0 & -d_y & -1 & 0 & 0 \end{pmatrix}, \qquad b = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

contact wrench constraints

- there is a final condition that requires that $\tau_{\min} \leq \tau_z \leq \tau_{\max}$, with

$$\tau_{\min} = -\bar{\mu}(d_x + d_y)f_z + |d_y f_x - \bar{\mu}\tau_x| + |d_x f_y - \bar{\mu}\tau_y|$$
$$\tau_{\max} = +\bar{\mu}(d_x + d_y)f_z - |d_y f_x + \bar{\mu}\tau_x| - |d_x f_y + \bar{\mu}\tau_y|$$

- this condition is more complex to apply, but it also can be written as a set of linear constraints on the contact wrench

whole-body QP

- if we collect all our constraints, we can set up a Quadratic Programming (QP), whose solution gives us feasible torques and contact forces

$$\min_{\ddot{q},\tau,w_i} \sum_j \alpha_j \left(J_j\ddot{q} + \dot{J}\dot{q} - a_j^d - k_p(p_j^d - p_j) - k_v(v_j^d - v_j) \right)^2$$

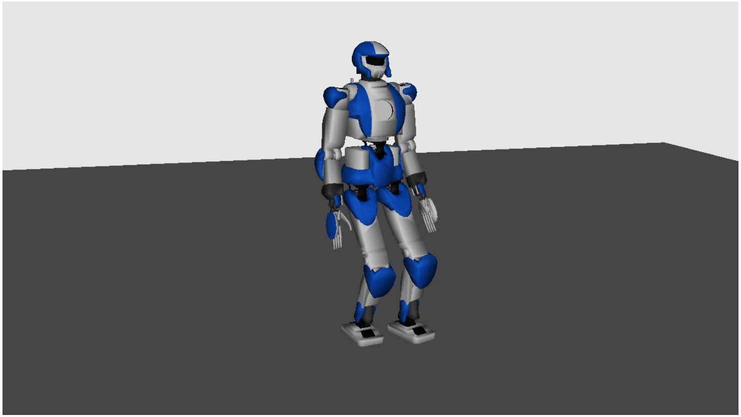
$$\begin{aligned} \text{s.t.} \quad & M\ddot{q} + n = S\tau + \sum_i J_i^T w_i \\ & J_{c,i}\ddot{q} + \dot{J}_{c,i}\dot{q} = 0 \\ & Aw_j \leq b \end{aligned}$$

- $\sum_j$ sums over all the task that we want our robot to achieve, with weights α_j
- constraints include **dynamics**, **contact constraints**, and **contact wrench constraints**
- we should include regularization terms for τ and w_j

whole-body QP

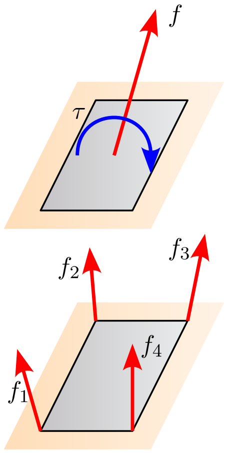
- the tasks typically include
  - tracking a center of mass trajectory
  - tracking trajectories of the feet for stepping
  - tracking an angular momentum reference
  - tracking a torso reference orientation
  - tracking a reference joint configuration, to resolve redundancy
  - tracking hand trajectories, for manipulation tasks

example: whole-body QP for walking



four-forces contact model

- the conditions we need to impose on the contact wrenches (in particular the last one) are a bit hard to manage
- a different approach is to use a **contact model** in which we specify that each contact is mediated by a predefined number of forces, and no torque (usually 4 forces)



four-forces contact model

- on each of these four forces we need to impose the usual force constraints: **unilaterality** and **friction cone**
- we don't need to worry about the ZMP: imposing unilaterality of the four forces is **equivalent** to imposing that the ZMP is inside the convex hull of their points of application
- some people prefer this approach to using contact wrenches, but they are equivalent