Dynamics

Without considering the external forces, the joint-space equations of motion can be written in the following canonical form:

$$\boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{\tau}, \tag{1}$$

where $\boldsymbol{H}(\boldsymbol{q})$ is the inertia matrix, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$ is the vector of centrifugal and Coriolis terms, $\boldsymbol{G}(\boldsymbol{q})$ is the gravity vector, and $\boldsymbol{\tau}$ is the joint torque vector. Each element of $\boldsymbol{H}(\boldsymbol{q})$ and $\boldsymbol{G}(\boldsymbol{q})$ is a complex function that depends on $\boldsymbol{q}$. Each element of $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is a complex function of both $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$. Once these dependencies are understood, the shorter symbols $\boldsymbol{H}$, $\boldsymbol{C}$, and $\boldsymbol{G}$ are used. While the analytical solution of them is almost impossible to derive (the equations will be several pages long at best), we have many numerical algorithms which can run efficiently in practice.

1.1 Recursive Newton-Euler Algorithm

The computation of $\boldsymbol{C}\dot{\boldsymbol{q}}$ and $\boldsymbol{G}$ can be done using inverse dynamics (ID) based on the recursive Newton-Euler algorithm (RNEA). RNEA is specifically to calculate ID of a kinematic tree (in comparison to closed-loop systems) and it is the simplest, most efficient algorithm for the job. It has a computational complexity of $O(n)$, which means that the amount of calculation grows linearly with the number of bodies, or joint variables, in the tree. RNEA

in body coordinates can be summarized as follows:

$$\boldsymbol{v}_0 = \boldsymbol{0}, \tag{2}$$

$$\boldsymbol{a}_0 = \boldsymbol{g}, \tag{3}$$

$$\boldsymbol{v}_{\mathrm{J}i} = \boldsymbol{S}_i \dot{\boldsymbol{q}}_i, \tag{4}$$

$$\boldsymbol{a}_{\mathrm{J}i} = \boldsymbol{S}_i \ddot{\boldsymbol{q}}_i, \tag{5}$$

$$\boldsymbol{v}_i = {}^i\boldsymbol{X}_{\lambda(i)} \boldsymbol{v}_{\lambda(i)} + \boldsymbol{v}_{\mathrm{J}i}, \tag{6}$$

$$\boldsymbol{a}_i = {}^i\boldsymbol{X}_{\lambda(i)} \boldsymbol{a}_{\lambda(i)} + \boldsymbol{a}_{\mathrm{J}i} + \boldsymbol{v}_i \times \boldsymbol{v}_{\mathrm{J}i}, \tag{7}$$

$$\boldsymbol{f}_i^B = \boldsymbol{I}_i \boldsymbol{a}_i + \boldsymbol{v}_i \times^* \boldsymbol{I}_i \boldsymbol{v}_i, \tag{8}$$

$$\boldsymbol{f}_i = \boldsymbol{f}_i^B + \sum_{j \in \mu(i)} {}^i\boldsymbol{X}_j^* \boldsymbol{f}_j, \tag{9}$$

$$\boldsymbol{\tau}_i = \boldsymbol{S}_i^\top \boldsymbol{f}_i, \tag{10}$$

where $\boldsymbol{g} = [0, 0, -9.81]^\top$ m/s^2 is the gravitational acceleration, $\boldsymbol{S}_i$ is the joint motion subspace vector which is equal to $[0, 0, 1, 0, 0, 0]^\top$ for revolute joint, $\boldsymbol{v}_i$ and $\boldsymbol{a}_i$ are the spatial velocity and acceleration of body i, $\boldsymbol{v}_{\mathrm{J}i}$ and $\boldsymbol{a}_{\mathrm{J}i}$ are the spatial velocity and acceleration across joint i, $\boldsymbol{I}_i$ is the spacial inertia, $\boldsymbol{f}_i^B$ is the net force acting on body i, $\boldsymbol{f}_i$ is the force transmitted from body $\lambda(i)$ to body i across joint i,

$${}^i\boldsymbol{X}_{\lambda(i)} = \begin{bmatrix} {}^i\boldsymbol{R}_{\lambda(i)} & \boldsymbol{0} \\ -{}^i\boldsymbol{R}_{\lambda(i)} \widehat{{}^i\boldsymbol{p}_{\lambda(i)}} & {}^i\boldsymbol{R}_{\lambda(i)} \end{bmatrix} \Rightarrow {}^i\boldsymbol{X}_{\lambda(i)}^* = {}^i\boldsymbol{X}_{\lambda(i)}^{-\top} = \begin{bmatrix} {}^i\boldsymbol{R}_{\lambda(i)} & -{}^i\boldsymbol{R}_{\lambda(i)} \widehat{{}^i\boldsymbol{p}_{\lambda(i)}} \\ \boldsymbol{0} & {}^i\boldsymbol{R}_{\lambda(i)} \end{bmatrix} \tag{11}$$

is the coordinate transform from $\lambda(i)$ to i, $\lambda(i)$ is the parent of body i, $\mu(i)$ is the set of children of body i, and $\times$ with its dual $\times^*$ is the spatial cross product.

Now that we have an ID calculation function, **ID**, which performs the calculation

$$\boldsymbol{\tau} = \mathbf{ID}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}). \tag{12}$$

On comparing this expression with (1), it follows immediately that

$$\boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{G} = \mathbf{ID}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \mathbf{0}), \tag{13}$$

$$\boldsymbol{G} = \mathbf{ID}(\boldsymbol{q}, \mathbf{0}, \mathbf{0}). \tag{14}$$

The ith column of $\boldsymbol{H}$ can also be computed as follows:

$$\boldsymbol{H}_i = \mathbf{ID}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\delta}_i) - \mathbf{ID}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \mathbf{0}), \tag{15}$$

where $\boldsymbol{\delta}_i$ is a column vector having a one in its ith element and zeros elsewhere. The algorithm has the advantage of being simple and straightforward. However, it is not the most efficient way to calculate $\boldsymbol{H}$. The fastest algorithm for that job is the composite-rigid-body algorithm (CRBA), which is the subject of the following section.

1.2 Composite-Rigid-Body Algorithm

CRBA has a computational complexity of $O(nd)$, where d is the depth of the kinematic tree. CRBA in body coordinates can be summarized as follows:

$$\boldsymbol{I}_i^c = \boldsymbol{I}_i + \sum_{j \in \mu(i)} {}^i\boldsymbol{X}_j^* \, \boldsymbol{I}_j^c \, {}^j\boldsymbol{X}_i, \tag{16}$$

$${}^{\lambda(i)}\boldsymbol{F}_i = {}^{\lambda(i)}\boldsymbol{X}_j^* \, {}^j\boldsymbol{F}_i, \tag{17}$$

$${}^i\boldsymbol{F}_i = \boldsymbol{I}_i^c \, \boldsymbol{S}_i, \tag{18}$$

$$\boldsymbol{H}_{ij} = \begin{cases} {}^j\boldsymbol{F}_i^\top \boldsymbol{S}_j & \text{if } i \in \nu(j), \\ \boldsymbol{H}_{ji} & \text{if } j \in \nu(i), \\ \mathbf{0} & \text{otherwise,} \end{cases} \tag{19}$$

where $\boldsymbol{I}_i^c$ is the inertia of the subtree rooted at body i, treated as a single composite rigid body (this is where the algorithm gets its name), ${}^j\boldsymbol{F}_i$ is the value of $\boldsymbol{I}_i^c \boldsymbol{S}_i$ expressed in body j coordinates (i.e., ${}^j\boldsymbol{F}_i = {}^j\boldsymbol{X}_i^* \boldsymbol{I}_i^c \boldsymbol{S}_i$), and $\nu(i)$ is the set of bodies in the subtree starting at body i (or the set of all bodies supported by joint i).

1.3 Centroidal Momentum Dynamics

The control of centroidal momentum is important for humanoid robots, which consists of the net linear momentum $\boldsymbol{l}_G$ as well as the net angular momentum $\boldsymbol{k}_G$ about the robot CoM. While the linear part has a well-known relationship with the CoM velocity $\boldsymbol{v}_G$:

$$\boldsymbol{l}_G = M\boldsymbol{v}_G, \tag{20}$$

the angular component is abstract as it represents the sum of the angular momenta of each individual body. As a result, properly designing the angular momentum trajectory is still an open problem. Fortunately, biomechanics studies of human walking have shown that although humans have large nonzero angular momenta for individual bodies in the limbs, yet, the neuro-control system coordinates significant inter-segmental momentum cancelations, regulating their centroidal angular momentum to near zero.

To effectively control the centroidal momentum, the centroidal momentum matrix (CMM) $\boldsymbol{A}_G(\boldsymbol{q})$ and its derivative $\dot{\boldsymbol{A}}_G(\boldsymbol{q})$ are required, which relate the centroidal momentum $\boldsymbol{h}_G = \left[\boldsymbol{k}_G^\top, \boldsymbol{l}_G^\top\right]^\top$ and its rate of change $\dot{\boldsymbol{h}}_G$ to the generalized velocity and acceleration of the robot:

$$\boldsymbol{h}_G = \boldsymbol{A}_G\dot{\boldsymbol{q}}, \tag{21}$$

$$\dot{\boldsymbol{h}}_G = \boldsymbol{A}_G\ddot{\boldsymbol{q}} + \dot{\boldsymbol{A}}_G\dot{\boldsymbol{q}}. \tag{22}$$

The computation of $\boldsymbol{A}_G$ and $\dot{\boldsymbol{A}}_G\dot{\boldsymbol{q}}$ can be done from knowledge of the floating-base kinemat-

ics, the inertia matrix $\boldsymbol{H}$, and the the Coriolis term $\boldsymbol{C}\dot{\boldsymbol{q}}$ alone:

$$\boldsymbol{I}_b^c = \boldsymbol{S}_b \boldsymbol{H} \boldsymbol{S}_b^\top = \begin{bmatrix} \bar{\boldsymbol{I}}_b^c & M\widehat{{}^b\boldsymbol{p}_G} \\ -M\widehat{{}^b\boldsymbol{p}_G} & M\mathbb{I} \end{bmatrix}, \tag{23}$$

$${}^b\boldsymbol{X}_G^\top = \begin{bmatrix} \boldsymbol{R}_b & -\boldsymbol{R}_b\widehat{{}^b\boldsymbol{p}_G} \\ \boldsymbol{0} & \boldsymbol{R}_b \end{bmatrix}, \tag{24}$$

$$\boldsymbol{A}_G = {}^b\boldsymbol{X}_G^\top \boldsymbol{S}_b \boldsymbol{H}, \tag{25}$$

$$\dot{\boldsymbol{A}}_G \dot{\boldsymbol{q}} = {}^b\boldsymbol{X}_G^\top \boldsymbol{S}_b \boldsymbol{C}\dot{\boldsymbol{q}}, \tag{26}$$

where $\boldsymbol{S}_b$ is the base selection matrix.