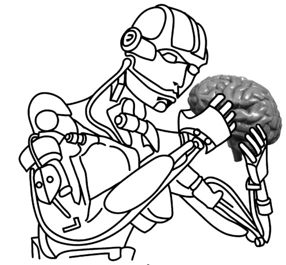
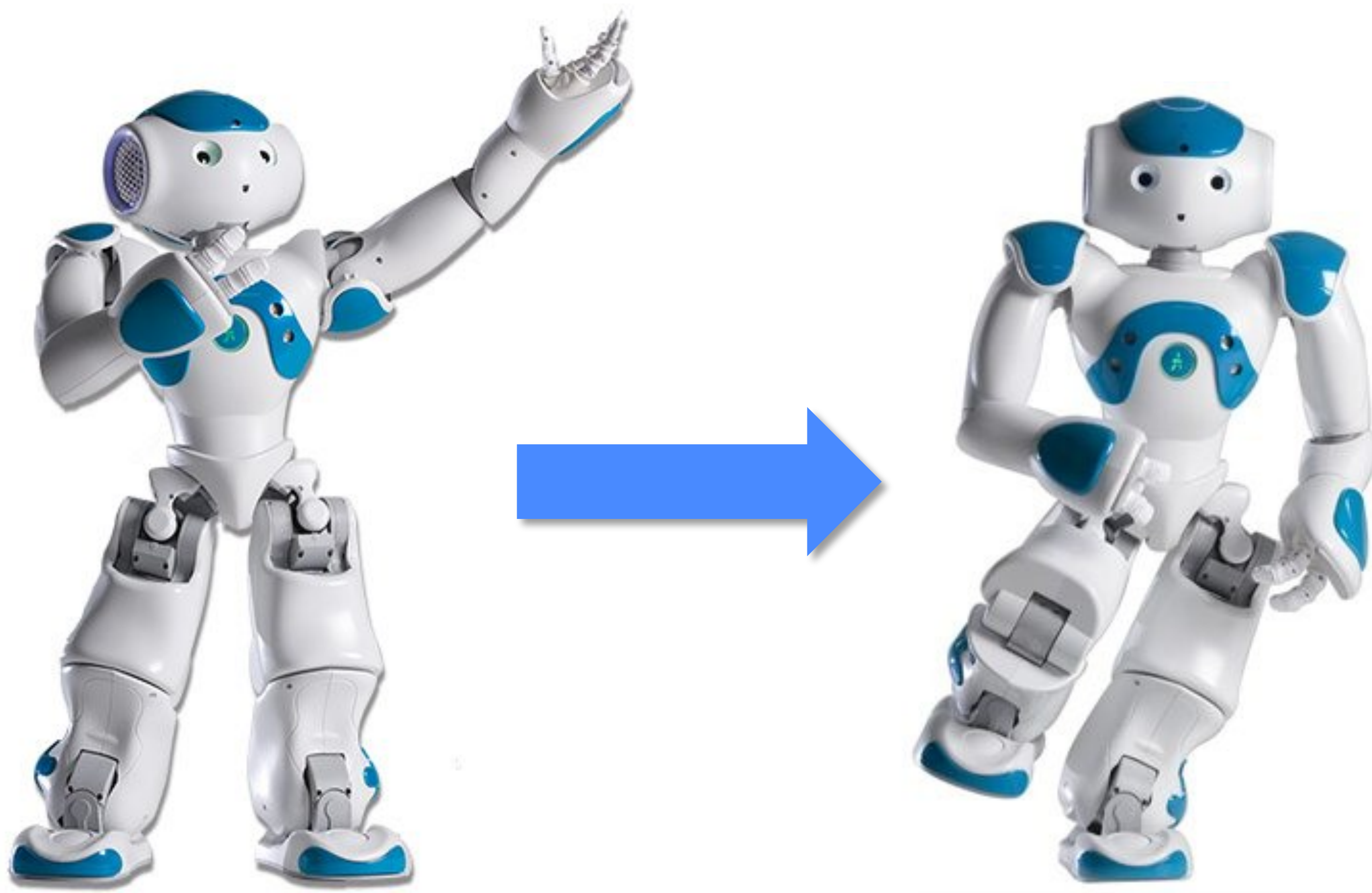
CS545—Floating Base Control

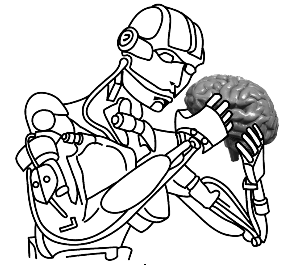


- Operational Space Control
  - A theoretically very clean approach to creating task space controllers
- Floating Base Control
  - How to deal with underactuated robots
  - Derived largely from operational space control
  - How it is used in the NAO simulator

Balancing Robots are Floating Base Systems



Operational Space Control



- Start with rigid body dynamics

$$\mathbf{B}\ddot{\mathbf{q}} + \mathbf{C} + \mathbf{g} = \tau$$

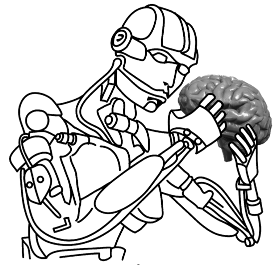
- The operational space coordinates (e.g., endeffector) are given through kinematics and differential kinematics

$$\mathbf{x} = \mathbf{f}\left(\mathbf{q}\right)$$

$$\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}}$$

$$\ddot{\mathbf{x}} = \mathbf{J}\ddot{\mathbf{q}} + \dot{\mathbf{J}}\dot{\mathbf{q}}$$

Operational Space Control



- Derive the operational space dynamics

$$\mathbf{B}\ddot{\mathbf{q}} + \mathbf{C} + \mathbf{g} = \tau$$

$$\mathbf{J}\mathbf{B}^{-1}\left(\mathbf{B}\ddot{\mathbf{q}} + \mathbf{C} + \mathbf{g}\right) = \mathbf{J}\mathbf{B}^{-1}\tau$$

$$\mathbf{J}\ddot{\mathbf{q}} + \mathbf{J}\mathbf{B}^{-1}\left(\mathbf{C} + \mathbf{g}\right) = \mathbf{J}\mathbf{B}^{-1}\tau$$

$$\ddot{\mathbf{x}} + \mathbf{J}\mathbf{B}^{-1}\mathbf{C} - \dot{\mathbf{J}}\dot{\mathbf{q}} + \mathbf{J}\mathbf{B}^{-1}\mathbf{g} = \mathbf{J}\mathbf{B}^{-1}\mathbf{J}^{T}\mathbf{F}$$

$$\left(\mathbf{J}\mathbf{B}^{-1}\mathbf{J}^{T}\right)^{-1}\ddot{\mathbf{x}} + \left(\mathbf{J}\mathbf{B}^{-1}\mathbf{J}^{T}\right)^{-1}\left(\mathbf{J}\mathbf{B}^{-1}\mathbf{C} - \dot{\mathbf{J}}\dot{\mathbf{q}}\right) + \left(\mathbf{J}\mathbf{B}^{-1}\mathbf{J}^{T}\right)^{-1}\mathbf{J}\mathbf{B}^{-1}\mathbf{g} = \mathbf{F}$$

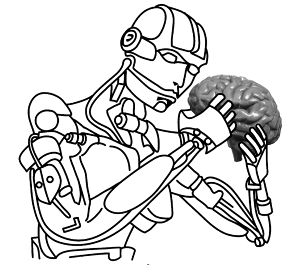
$$\bar{\mathbf{B}}\ddot{\mathbf{x}} + \bar{\mathbf{C}} + \bar{\mathbf{g}} = \mathbf{F}: \text{ Opertional space dynamics}$$

$$\bar{\mathbf{B}} = \left(\mathbf{J}\mathbf{B}^{-1}\mathbf{J}^{T}\right)^{-1}: \text{ Opertional space inertia matrix}$$

$$\bar{\mathbf{C}} = \left(\mathbf{J}\mathbf{B}^{-1}\mathbf{J}^{T}\right)^{-1}\left(\mathbf{J}\mathbf{B}^{-1}\mathbf{C} - \dot{\mathbf{J}}\dot{\mathbf{q}}\right): \text{ Opertional corriolis/centripedal forces}$$

$$\bar{\mathbf{g}} = \left(\mathbf{J}\mathbf{B}^{-1}\mathbf{J}^{T}\right)^{-1}\mathbf{J}\mathbf{B}^{-1}\mathbf{g}: \text{ Opertional space gravity forces}$$

Operational Space Control



- Operational Space Control Law

Assume a desired operatational space trajectory:

$$\ddot{\mathbf{x}}_{ref} = \ddot{\mathbf{x}}_d + \mathbf{K}_d\left(\dot{\mathbf{x}}_d - \dot{\mathbf{x}}\right) + \mathbf{K}_p\left(\mathbf{x}_d - \mathbf{x}\right)$$

such that the appropriate force to realize this trajectory is:

$$\mathbf{F} = \bar{\mathbf{B}}\ddot{\mathbf{x}}_{ref} + \bar{\mathbf{C}} + \bar{\mathbf{g}}$$

convert to joint space:

$$\tau = \mathbf{J}^T\mathbf{F}$$

$$\tau = \mathbf{B}\mathbf{B}^{-1}\mathbf{J}^T\left(\bar{\mathbf{B}}\ddot{\mathbf{x}}_{ref} + \bar{\mathbf{C}} + \bar{\mathbf{g}}\right)$$

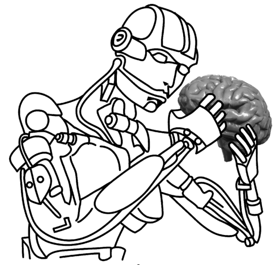
$$\tau = \mathbf{B}\bar{\mathbf{J}}\left(\ddot{\mathbf{x}}_{ref} - \dot{\mathbf{J}}\dot{\mathbf{q}} + \mathbf{J}\mathbf{B}^{-1}\left(\mathbf{C} + \mathbf{g}\right)\right)$$

Add null space forces by:

$$\tau = \mathbf{B}\bar{\mathbf{J}}\left(\ddot{\mathbf{x}}_{ref} - \dot{\mathbf{J}}\dot{\mathbf{q}} + \mathbf{J}\mathbf{B}^{-1}\left(\mathbf{C} + \mathbf{g}\right)\right) + \left(\mathbf{I} - \mathbf{J}^T\bar{\mathbf{J}}\right)\tau_{null}$$

$\bar{\mathbf{J}} = \mathbf{B}^{-1}\mathbf{J}^T\left(\mathbf{J}\mathbf{B}^{-1}\mathbf{J}^T\right)^{-1}$: inertia weighted pseudo inverse

Floating Base Control



- Treat Robot as a "Space Robot"

$$\tilde{\mathbf{B}}\ddot{\tilde{\mathbf{q}}} + \tilde{\mathbf{C}} + \tilde{\mathbf{g}} = \mathbf{S}^T \tau$$

where

$$\mathbf{S} = \begin{bmatrix} \mathbf{I}^n & 0^6 \end{bmatrix} \text{: selector matrix, like operational space Jacobian}$$

$$\tilde{\mathbf{q}} = \begin{bmatrix} \mathbf{q} \\ \mathbf{x}_B \end{bmatrix}$$

- If the robot is in contact with the world, add constraint forces:

$$\tilde{\mathbf{B}}\ddot{\tilde{\mathbf{q}}} + \tilde{\mathbf{C}} + \tilde{\mathbf{g}} = \mathbf{S}^T \tau + \tilde{\mathbf{J}}_c^T \mathbf{f}_{ext}$$

$$\tilde{\mathbf{J}}_c \dot{\tilde{\mathbf{q}}} = 0$$

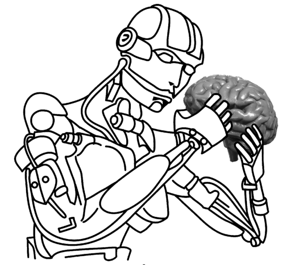
A Possible Floating Base Control Law

- Write the joint space dynamics from operational space control derivations:

$$\left(\mathbf{S}\tilde{\mathbf{B}}^{-1}\mathbf{S}^{T}\right)\ddot{\mathbf{q}}+\overline{\mathbf{S}}^{T}\left(\tilde{\mathbf{C}}+\tilde{\mathbf{g}}-\mathbf{J}_{c}^{T}\mathbf{f}_{ext}\right)=\boldsymbol{\tau}$$

- This is essentially an inverse dynamics control law, but it requires reliable force measurement at the constraint points

A Better Floating Base Control Law



- Decompose the constraint matrix by a QR decomposition:

$$\mathbf{J}_c^T = \mathbf{Q}\begin{bmatrix} \mathbf{R} \\ 0 \end{bmatrix}$$

$\mathbf{Q}$ is orthonormal $\mathbf{Q}^T\mathbf{Q} = \mathbf{Q}\mathbf{Q}^T = \mathbf{I}$

$\mathbf{R}$ is upper triangular of rank k (k is the number of constraints)

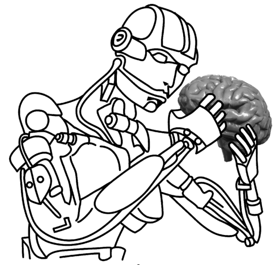
With some algebra, a control law becomes:

$$\tau = \left(\mathbf{S}_u\mathbf{Q}^T\mathbf{S}^T\right)^{\#}\mathbf{S}_u\mathbf{Q}^T\left(\tilde{\mathbf{B}}\ddot{\mathbf{q}}_{ref} + \tilde{\mathbf{C}} + \tilde{\mathbf{g}}\right)$$

$$\mathbf{S}_u = \begin{bmatrix} \mathbf{0}_{(n+6-k)\times k} & \mathbf{I}_{(n+6-k)\times(n+6-k)} \end{bmatrix}$$

- This control does not require knowledge of the external forces, and this is what is implemented for the NAO

Theory of COG Inverse Kinematics



$$\text{COG: } \mathbf{x}_{cog} = \frac{1}{\sum_{i=1}^{n} m_i} \sum_{i=1}^{n} m_i \mathbf{x}_{i,cog}$$

$$\text{COG Jacobian: } \mathbf{J}_{cog} = \frac{\partial \mathbf{x}_{cog}}{\partial \boldsymbol{\theta}} = \frac{1}{\sum_{i=1}^{n} m_i} \sum_{i=1}^{n} m_i \frac{\partial \mathbf{x}_{i,cog}}{\partial \boldsymbol{\theta}}$$

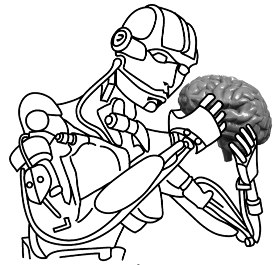
$$\text{Floating Base COG Jacobian: } \mathbf{J}_{cog,float} = \begin{bmatrix} \mathbf{J}_{cog} & \mathbf{J}_{base} \end{bmatrix}$$

$$\text{Constraints from standing on 2 feet: } \mathbf{J}_{feet,float} \begin{bmatrix} \dot{\boldsymbol{\theta}} \\ \dot{\mathbf{x}}_{base} \\ \boldsymbol{\omega}_{base} \end{bmatrix} = 0 \qquad \text{(no slipping)}$$

$$\text{Null Space Projection for Constraints: } \mathbf{N}_c = \left(\mathbf{I} - \mathbf{J}^{\#}_{feet,float} \mathbf{J}_{feet,float} \right)$$

$$\text{Constraint COG Jacobian: } \mathbf{J}_{cog,const} = \mathbf{J}_{cog} \mathbf{N}_c$$

Inverse Kinematics with Constraint COG Jacobian



- Given: Desired trajectory of COG

$$\mathbf{x}_{cog,des}, \dot{\mathbf{x}}_{cog,des}$$

- Reference COG velocity

$$\dot{\mathbf{x}}_{cog,ref} = k_p\left(\mathbf{x}_{cog,des} - \mathbf{x}_{cog}\right) + \dot{\mathbf{x}}_{cog,des}$$

- IK Solution

$$\begin{bmatrix} \dot{\boldsymbol{\theta}}_{des} \\ \dot{\mathbf{x}}_{base,des} \\ \boldsymbol{\omega}_{base,des} \end{bmatrix} = \mathbf{J}^{\#}{}_{cog,const}\dot{\mathbf{x}}_{cog,ref} \qquad \boldsymbol{\theta}_{des}(t+1) = \dot{\boldsymbol{\theta}}_{des}(t)\Delta t + \boldsymbol{\theta}_{des}(t)$$

Implentation In SL

- balance_task.cpp is the skeleton to use
- All important variables are pre-computed and commented