Robotics and Animatronics in Disney

Lecture 1a: Humanoid Robot Dynamics

Katsu Yamane
kyamane@disneyresearch.com

Goals

- Introduce the basics of humanoid robot dynamics
  - Forward dynamics
  - Inverse dynamics
- Focus on issues specific to humanoid robots
  - Floating base
  - Contacts

General Robot Dynamics

General Robot Dynamics

- Force/torque ↔ acceleration
- cf. Statics: position/orientation only
  - Gravity, static balance, center of mass

Joint torques

Forward dynamics ⇩ ⇧ Inverse dynamics

Joint accelerations

Equation of Motion

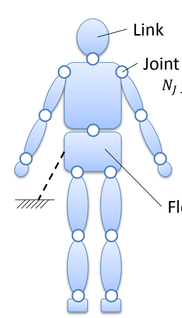
Equation of motion:

$$\underbrace{\boxed{M(\theta)\ddot{\theta}}}_{\text{Inertial forces}} + \underbrace{\boxed{c(\theta,\dot{\theta})}}_{\text{Centrifugal + Coriolis forces}} + \underbrace{\boxed{g(\theta)}}_{\text{Gravitational forces}} = \overbrace{\boxed{\tau_G}}^{\text{Generalized forces}}$$

$\theta, \tau_G \in \mathbb{R}^{N_G}$

N_G: degrees of freedom (DOF)
Number of variables required to uniquely determine the pose

Humanoid Robot Model



Link

Joint

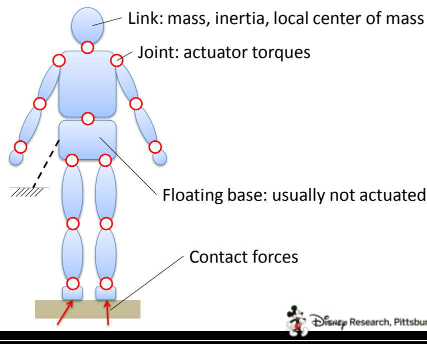
N_J joints → N_J DOF

$$N_G = N_J + 6$$

Floating base

3 translations + 3 rotations → 6 DOF

What Determines Humanoid Motion



Humanoid Robot Dynamics

- Many joints ($N_G > 30$)
- Contacts enforce kinematic constraints
- Floating base is not actuated
- Contact forces are subject to unilateral constraints
  - Normal force must be repelling (cannot pull each other)
  - Coulomb friction constraint

Humanoid Robot Dynamics

Equation of motion:

$$M(\theta)\ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta) = \tau_G = \overset{\text{Joint torques}}{\boxed{S^T \tau}} + \overset{\text{Contact forces}}{\boxed{J_C^T f_C}}$$

Unactuated base:

$$S^T = \begin{array}{c|c} & N_J \\ \hline 6 & \\ N_J & \end{array} \begin{pmatrix} 0 \\ \begin{matrix} 1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{matrix} \end{pmatrix}$$

Contact constraints:

$$J_C \dot{\theta} = 0 \quad \rightarrow \quad J_C \ddot{\theta} + \dot{J}_C \dot{\theta} = 0$$

How to Compute

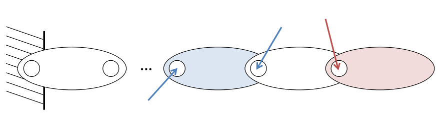
- Lagrangian mechanics
  - Gives analytical expression
  - Computationally expensive
- More efficient numerical algorithms

Numerical Inverse Dynamics

$$M(\theta)\ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta) = \tau_G$$

Newton-Euler Formulation [Orin et al. 1979]

- Computes τ_G for given $\theta, \dot{\theta}, \ddot{\theta}$
- Idea: the end link receives force/moment from only one joint

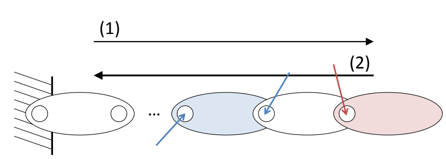


Newton-Euler Formulation

(1) Compute linear/angular accelerations of each link (forward kinematics)

→ Total force/moment applied to each link

(2) Compute joint force/moment



Forward Dynamics

$$M(\theta)\ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta) = \tau_G$$

- Given $\theta, \dot{\theta}, \tau_G \rightarrow$ compute $\ddot{\theta}$
- If M, c, g are known

$$\ddot{\theta} = M^{-1}(\tau_G - c - g)$$

Disney Research, Pittsburgh

Unit Vector Method

[Walker and Orin 1982]

1. Set $\ddot{\theta} = 0$ and compute inverse dynamics $\rightarrow \tau_G = c + g$
2. Set i-th element of $\ddot{\theta}$ to 1 and the rest to 0
3. Compute inverse dynamics
   $\rightarrow \tau_G = m_i + c + g$ (m_i is the i-th column of M)
   $\rightarrow m_i = \tau_G - c - g$
4. Repeat step 2, 3 for $i = 1, \ldots, N_G \rightarrow M$

Cost: $O(N^2)$

Disney Research, Pittsburgh

$O(N)$ Forward Dynamics Algorithms

- Articulated-Body Algorithm [Featherstone 1987]
- Divide-and-Conquer Algorithms [Featherstone 1999]
- Assembly-Disassembly Algorithm [Yamane and Nakamura 2003]

- The last two can be parallelized

Disney Research, Pittsburgh

Inverse Dynamics of Humanoid Robots

Disney Research
Pittsburgh

Inverse Dynamics

$$M(\theta)\ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta) = S^T\tau + J_C^T f_C$$

- Given $\theta, \dot{\theta}, \ddot{\theta} \rightarrow$ compute τ and f_C
- Applications
  – Check if a motion is feasible
  – Trajectory optimization
- Problems
  – Joint accelerations may not satisfy contact constraints
  – Contact forces may not be feasible
  – Contact forces may not be unique

Disney Research, Pittsburgh

Inverse Dynamics

$$M(\theta)\ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta) = S^T\tau + J_C^T f_C$$

Use the structure of S^T:

$$\begin{array}{cc} & N_J \\ 6 & \begin{pmatrix} & 0 & \\ 1 & & \\ & 1 & \\ & & \ddots \\ & & & 1 \end{pmatrix} \\ N_J & \end{array}$$

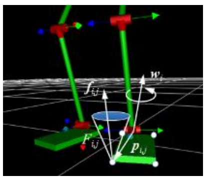
Top 6 rows $\quad M_1\ddot{\theta} + c_1 + g_1 = 0 + J_{C1}^T f_C$

The rest $\quad M_2\ddot{\theta} + c_2 + g_2 = \tau + J_{C2}^T f_C$

Disney Research, Pittsburgh

Computing the Contact Forces

- Constraints on contact force
  - Normal forces at all contact points must be repelling ↔ Center of pressure (CoP) or zero-moment point (ZMP) must be in the contact area
  - Friction

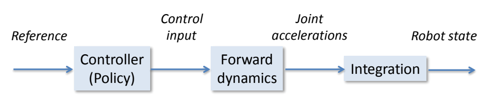


Computing the Contact Forces

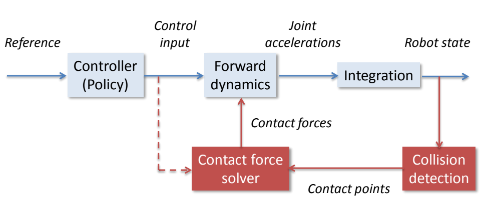
- Naïve solution: $f_C = J_{C1}^{\#T}\left(M_1\ddot{\theta} + c_1 + g_1\right)$
  - Contact forces may not satisfy the constraints
  - Weigh the contact friction and moment terms [Yamane and Hodgins 2009]
- Force data and quadratic programming [Yamane et al. 2005]
- Contact force optimization with geometric algorithms: Lecture 2 [Zheng and Yamane 2012]

Humanoid Robot Simulation with Contacts

Simulation without Contacts



Simulation with Contacts



Contact Force Solvers

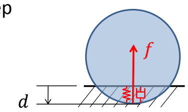
- Independent of joint torques
  - Spring-damper model
- Dependent on joint torques
  - Rigid-body (constraint-based) models
  - Impulse-based model

Independent of Joint Torques

Spring-damper (penalty-based) model

$$f = -k_p d - k_d \dot{d}$$

- Compute contact forces directly from current state
- Easy to implement
- Difficult to find appropriate parameters (k_p, k_d)
- Requires small integration timestep



Disney Research, Pittsburgh

Dependent on Joint Torques

Impulse-based models [Mirtich 1995]

- Apply Poisson's collision model
- Good for handling frequent collisions
- Stacking represented by many microcollisions

Disney Research, Pittsburgh

Dependent on Joint Torques

Rigid-body (constraint-based) models

- Usually formulated as linear complementarity problem (LCP)
  - Acceleration/force
    [Lötstedt 1982] [Baraff 1989] [Baraff 1994]
  - Velocity/impulse after implicit integration
    [Anitescu and Potra 1997] [Stewart and Trinkle 2000]
- Comparison to penalty-based models
  - Numerically stable
  - Difficult to solve

Disney Research, Pittsburgh

Rigid-Body Models

Linear complimentarity problem:

For given M, q, obtain w, z that satisfy

$$\begin{cases} w = Mz + q \\ w \geq 0, z \geq 0, w^T z = 0 \; (w \geq 0 \perp z \geq 0) \end{cases}$$

Disney Research, Pittsburgh

Contacts as LCP

2D point mass, normal direction

$$m\dot{v} = f_N - mg \quad \text{(equation of motion)}$$

discretize and integrate

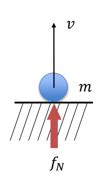
$$m(v_{t+1} - v_t) = (f_N - mg)\Delta t$$

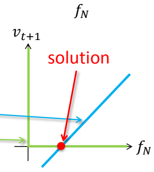
rearrange to LCP form

$$v_{t+1} = \frac{\Delta t}{m} f_N + v_t - g\Delta t$$

$$f_N \geq 0 \perp v_{t+1} \geq 0$$

(complementarity condition)



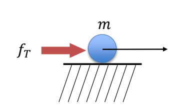


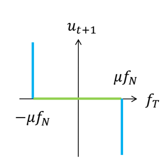
Disney Research, Pittsburgh

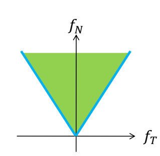
Contacts as LCP

Tangential direction (friction)

$$u_{t+1} = \frac{\Delta t}{m} f_T + u_t$$







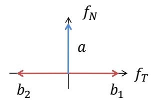
Complementarity condition for u_{t+1} and f_T?

Disney Research, Pittsburgh

Contacts as LCP

Additional parameters

- $\lambda \geq 0$: magnitude of u_{t+1}
- $a \geq 0$: magnitude of f_N
- $b_1, b_2 \geq 0$: magnitude of f_T in +/- directions ($f_T = b_1 - b_2$)



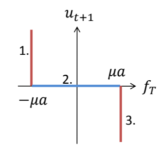
$$\mu a - \begin{pmatrix}1 & 1\end{pmatrix}\begin{pmatrix}b_1\\ b_2\end{pmatrix} \geq 0 \perp \lambda \geq 0$$

$$\begin{pmatrix}1\\ -1\end{pmatrix} u_{t+1} + \begin{pmatrix}1\\ 1\end{pmatrix} \lambda \geq 0 \perp \begin{pmatrix}b_1\\ b_2\end{pmatrix} \geq 0$$

Contacts as LCP

$$\mu a - \begin{pmatrix}1 & 1\end{pmatrix}\begin{pmatrix}b_1\\ b_2\end{pmatrix} \geq 0 \perp \lambda \geq 0$$

$$\begin{pmatrix}1\\ -1\end{pmatrix} u_{t+1} + \begin{pmatrix}1\\ 1\end{pmatrix} \lambda \geq 0 \perp \begin{pmatrix}b_1\\ b_2\end{pmatrix} \geq 0$$



1. $u_{t+1} > 0, f_T = -\mu a < 0 \quad \lambda = u_{t+1}, b_1 = 0, b_2 = \mu a$
2. $u_{t+1} = 0, -\mu a < f_T < \mu a \; \lambda = 0, b_1 - b_2 = f_T$
3. $u_{t+1} < 0, f_T = \mu a > 0 \; \lambda = -u_{t+1}, b_1 = \mu a, b_2 = 0$

Putting Everything Together

Unknowns: $z = (a \quad b_1 \quad b_2 \quad \lambda)^T$

Eliminate v_{t+1} and u_{t+1} using the equation of motion

$$w = \begin{pmatrix}\Delta t/m & 0 & 0 & 0\\ 0 & \Delta t/m & -\Delta t/m & 1\\ 0 & -\Delta t/m & \Delta t/m & 1\\ \mu & -1 & -1 & 0\end{pmatrix} z + \begin{pmatrix}v_t - g\Delta t\\ u_t\\ -u_t\\ 0\end{pmatrix}$$

$$w \geq 0 \perp z \geq 0$$

Solving LCPs

Pivot-based method

Optimization

Iterative method

Solving LCPs

Pivot-based method

- Identify inactive (or vanishing) constraints
- 2^npossibilities for n contact points
- No "intermediate" solution
- General LCPs [Lemke and Howson 1964] [Cottle and Dantzig 1968] [Murty 1988]
- Application to contacts [Lloyd 2005]
- Numerical robustness [Yamane and Nakamura 2008]
- Heuristics for contacts [Baraff 1989] [Baraff 1994]

Solving LCPs

Pivot-based method $\quad w = Mz + q, w \geq 0 \perp z \geq 0$

Divide w, z into two parts: $\begin{pmatrix}w_{\tilde{\alpha}}\\ w_{\tilde{\beta}}\end{pmatrix} = M\begin{pmatrix}z_\alpha\\ z_\beta\end{pmatrix} + q$

such that $w_{\tilde{\alpha}}$ and z_α contain the same set of indices

Pivoted equation: $\begin{pmatrix}z_\alpha\\ w_{\tilde{\beta}}\end{pmatrix} = M'\begin{pmatrix}w_{\tilde{\alpha}}\\ z_\beta\end{pmatrix} + q'$

If $q' \geq 0$, the solution is $\begin{pmatrix}z_\alpha\\ w_{\tilde{\beta}}\end{pmatrix} = q'$, $\begin{pmatrix}w_{\tilde{\alpha}}\\ z_\beta\end{pmatrix}$=0

(Confirm $w \geq 0 \perp z \geq 0$)

Lemke Algorithm [Lemke and Howson 1964]

- One of pivot-based methods
- Introduce an auxiliary variable z_0
- $w = \bar{M}\binom{z}{z_0} + q$ $\quad \bar{M} = (M \quad c)$, $c = (1 \quad 1 \quad \ldots \quad 1)^T$
- Overview
  1. Swap w_r and z_0 such that $q' \geq 0$
  2. Continue swapping keeping $q' \geq 0$
  3. Terminate when z_0 comes back to right-hand side

Disney Research, Pittsburgh

Solving LCPs

Convert to a quadratic program [Lötstedt 1982]

$$\begin{cases} w = Mz + q \\ w \geq 0 \perp z \geq 0 \end{cases}$$

$$\updownarrow$$

Minimize $z^T(Mz + q)$

Subject to $Mz + q \geq 0, w \geq 0, z \geq 0$

Disney Research, Pittsburgh

Solving LCPs

Non-smooth Newton method [Ralph 1994] [Ferris et al. 1998]

- Fischer-Burmeister function $\phi(a, b) = \sqrt{a^2 + b^2} - a - b$
  $\rightarrow \phi = 0$ iff $a \geq 0 \perp b \geq 0$
- Apply Newton method with proved convergence
- Application to nonlinear contact model [Todorov 2010]

Disney Research, Pittsburgh

Solving LCPs

Iterative method [Jourdan 1998] [Kokkevis 2004]

- Extension of iterative algorithms for solving linear equations (e.g., Gauss-Seidel method)
- Relatively easy to implement
- Can stop at any iteration
- Guaranteed to converge only when M is SPD, which is not the case with frictional contacts
- But it does converge in many practical cases

Disney Research, Pittsburgh

Collision Detection

- Input
  - Polygon approximation
  - Parameteric surface representation
- Outputs
  - Contact point locations
  - Contact normal
- Sometimes required (c.f. spring-damper contact model)
  - Distance/depth

Disney Research, Pittsburgh

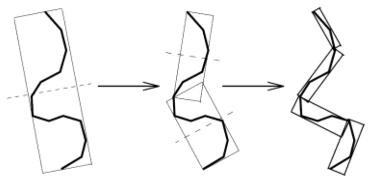
Collision Detection Algorithms

- Mostly rigid bodies (no deformation)
- Algorithms for general polygon models
  - Oriented bounding box (OBB)
  - Many useful libraries from UNC Gamma Group
    http://gamma.cs.unc.edu/research/collision/
- Algorithms for parameteric surface representation
  - Geometry-based algorithms → Lecture 2

Disney Research, Pittsburgh

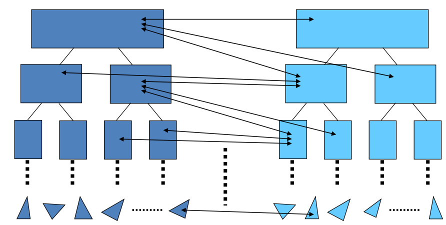
OBB-Based Collision Detection

1. Triangulate the polygons
2. Recursively partition the polygon mesh into oriented bounding boxes (OBBTree [Gottschalk et al. 1996])



OBB-Based Collision Detection

3. Recursively check collisions between OBBs

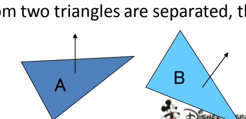


OBB-Based Collision Detection

4. Check collisions between triangles
   1) Check if each vertex of triangle B is above or below the plane including triangle A. If all vertices are on the same side, A and B are not colliding.
   2) Check if an edge of B is passing A.
   3) Check if projection of a vertex of B is inside A

      basic operation: project the vertices onto a vector

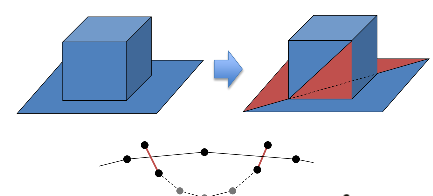
      if the projections from two triangles are separated, then there is no collision



OBB-Based Collision Detection

Problems

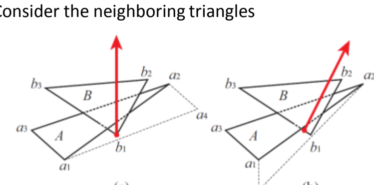
- Publicly available codes do not give normal vector
- No global shape information



Contact Normal

Extension [Yamane and Nakamura 2006]

- Consider all possible displacements to separate the triangles
- Choose the one with the smallest depth
- Consider the neighboring triangles



Available Libraries/Software

Game engines

- ODE http://ode.org/
  - approximated friction model
  - rigid bodies + constraints: artificial parameters to maintain constraints
- PhysX: LCP (iterative solver) + penalty-based? http://www.nvidia.com/object/physx_new.html
  - works on PPU (Physics Processing Unit)
- Havok http://www.havok.com/

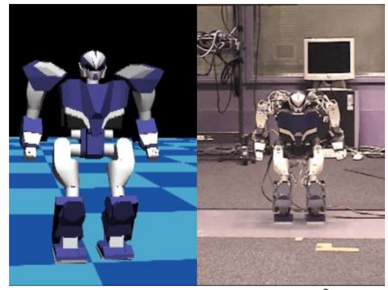
Available Libraries/Software

General dynamics simulation

- SD/Fast: http://www.sdfast.com/
  - generates the code for a specific model
- OpenHRP3: iterative and Lemke solvers
  http://www.openrtp.jp/openhrp3/en/index.html
  - Nice UI but heavy
  - Link to necessary source code
- Webots
  http://www.cyberbotics.com/

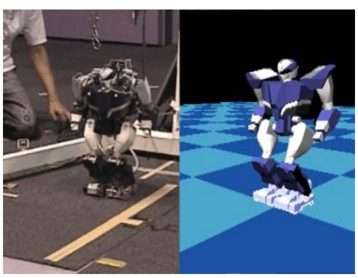
Comparison with Experiments

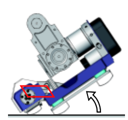
[Yamane, Nakamura, Yamamoto IROS 2008]



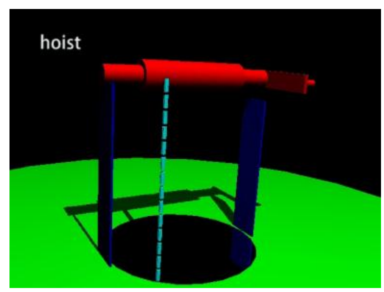
Comparison with Experiments

with closed loop (toe joint)





Non-Humanoid Examples



Discussion

- Real robots are different from simulation models
- A controller that works in simulation does not always work on real robot
- Is simulation useful at all?
  - Simulation gives baseline (ideal) results
  - Compare experiments with simulation
  - Compare different controllers/parameters