Trajectory Optimization, a brief introduction

Manoj Srinivasan
Mechanical Engineering
Ohio State University

Google: Manoj Srinivasan
Tutorial materials will also be made available at
http://movement.osu.edu

Tutorial at Dynamic Walking 2010
Media Lab, Massachusetts Institute of Technology
July 8, 2010

- Target audience
  - Basic familiarity with MATLAB
  - No prior familiarity with trajectory optimization (or even nonlinear optimization)
- Software requirements
  - MATLAB
  - MATLAB optimization toolbox (fmincon, fsolve) or SNOPT (free student version, say)

Some examples of trajectory optimization

- Trajectories to moon and other planets
- Trajectories for space shuttle reentry, airplanes, etc.
- Motions of industrial manipulators and other robots, including legged robots / animals
- Many mechanics problems (using some variant of the principle of least action, or potential/ free energy minimization)

Part 0

Nonlinear constrained optimization problems

finite dimensions

A made-up math problem

Example. Find values for the 5 variables x_1, x_2, x_3, x_4, and x_5 such that the function $f(x_1, x_2, x_3, x_4, x_5) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2$ is minimized and the following constraints are satisfied:

$$x_1 + x_2 + x_3 = 5 \quad (1)$$

$$x_3^2 + x_4 = 2 \quad (2)$$

$$x_1 \geq 0.3 \quad (3)$$

$$x_3 \leq 5 \quad (4)$$

$$x_4^2 + x_5^2 \leq 5 \quad (5)$$

Number of unknowns = 5.
Hence a "finite dimensional" optimization problem.

This made-up example is of the following general form

Minimize $f(x)$ such that:

OBJECTIVE FUNCTION

VARIOUS CONSTRAINTS

$$
\begin{aligned}
\mathrm{A}x &\leq \mathrm{B} && \text{LINEAR INEQUALITY} \\
\mathrm{C}x &= \mathrm{D} && \text{LINEAR EQUALITY} \\
l \leq\ & x \leq u && \text{SIMPLE BOUNDS} \\
h(x) &\leq 0 && \text{NONLINEAR INEQUALITY} \\
g(x) &= 0 && \text{NONLINEAR EQUALITY}
\end{aligned}
$$

$$
x = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ x_n \end{bmatrix}
$$

in which

$$
\begin{gathered}
x, l, u \in \mathbb{R}^n \\
A \in \mathbb{R}^{m_1 \times n},\ B \in \mathbb{R}^{m_1} \\
C \in \mathbb{R}^{m_2 \times n},\ D \in \mathbb{R}^{m_2} \\
h(x) \in \mathbb{R}^{m_3} \\
g(x) \in \mathbb{R}^{m_4}
\end{gathered}
$$

Nonlinear constrained optimization problem
(or) "Nonlinear Programming" problem

Minimize $f(x)$ such that:

OBJECTIVE FUNCTION

$$x = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ x_n \end{bmatrix}$$

in which

VARIOUS CONSTRAINTS

$$\begin{aligned} \mathrm{A}x &\leq \mathrm{B} && \text{LINEAR INEQUALITY} \\ \mathrm{C}x &= \mathrm{D} && \text{LINEAR EQUALITY} \\ l \leq\ & x \leq u && \text{SIMPLE BOUNDS} \\ h(x) &\leq 0 && \text{NONLINEAR INEQUALITY} \\ g(x) &= 0 && \text{NONLINEAR EQUALITY} \end{aligned}$$

$$x, l, u \in \mathbb{R}^n$$

$$A \in \mathbb{R}^{m_1 \times n},\ B \in \mathbb{R}^{m_1}$$

$$C \in \mathbb{R}^{m_2 \times n},\ D \in \mathbb{R}^{m_2}$$

$$h(x) \in \mathbb{R}^{m_3}$$

$$g(x) \in \mathbb{R}^{m_4}$$

References:
Numerical Optimization, Nocedal and Wright, 1999
Practical Methods of Optimization, Fletcher, 2000
Practical Optimization, Gill, Murray and Wright, 1982

MATLAB's fmincon can solve such nonlinear constrained optimization problems

fmincon(@objfun, x0, Aineq, Bineq, Aeq, Beq, LB, UB, @nonlcons, options, extra parameters)

objfun = function returning the objective function value, given the unknowns x
nonlcons = function returning the nonlinear inequality and equality constraints
Aineq, Bineq = matrices defining the linear inequality constraints
Aeq, Beq = matrices defining the linear equality constraints
LB, UB = vectors containing lower and upper bounds of x
extra parameters = if objfun and nonlcons need other input variables (other than x)
x0 = initial guess (initial seed) for the optimal solution
options = parameters set by 'optimset' that determine fmincon's behavior

(People like another software SNOPT better,
but we'll use fmincon here)

- See folder SimpleOptimizationProblem for a MATLAB solution of the simple optimization problem using fmincon.

- Another practise problem for fmincon:

Exercise. Find x_1, x_2, x_3 that maximize $f(x) = x_1x_2 + x_2x_3$ subject to the following nonlinear inequality constraints:

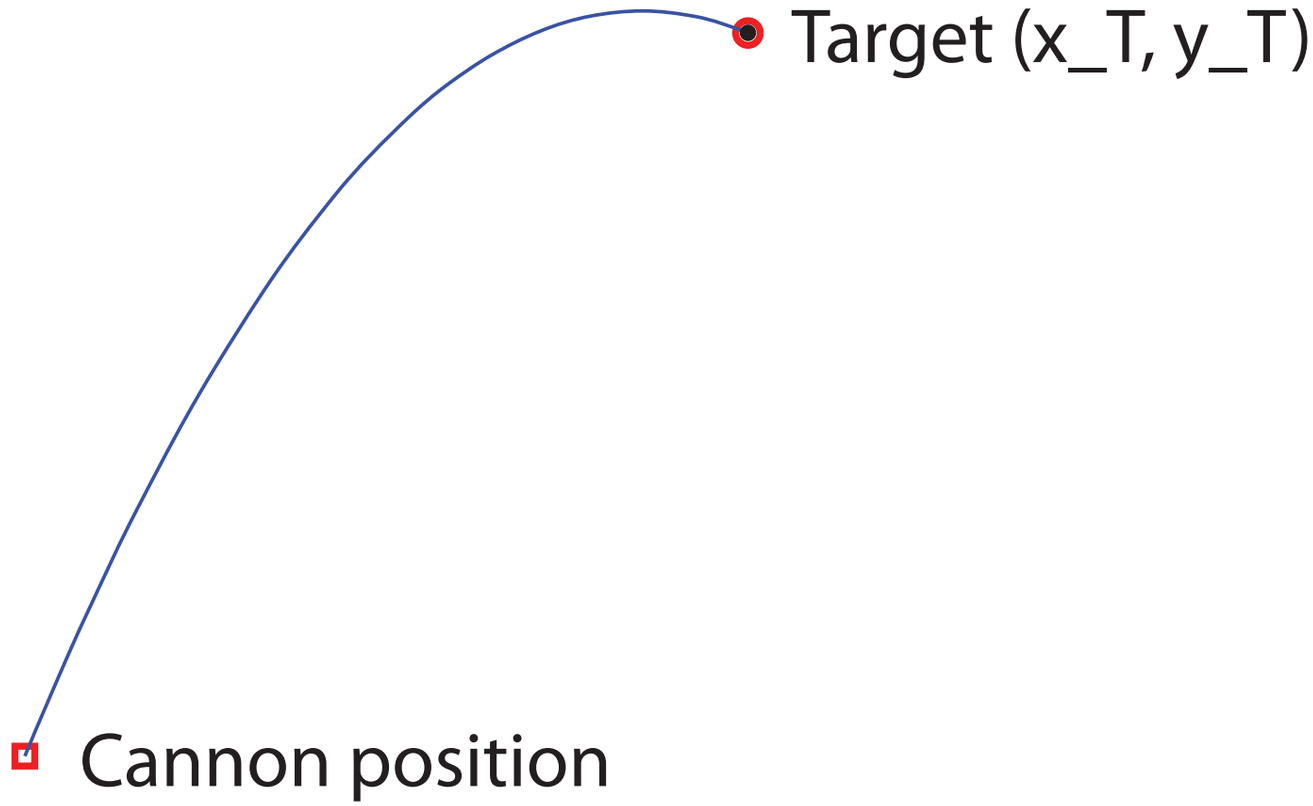
$$x_1^2 - x_2^2 + x_3^2 \leq 2 \quad (1)$$
$$x_1^2 + x_2^2 + x_3^2 \leq 10 \quad (2)$$

Note: Maximizing $f(x)$ is the same as minimizing $-f(x)$. This example is from the Wikipedia entry on "Nonlinear Programming".

Part I

Another baby optimization problem

Shooting a target using a cannon



Goal: Find launch conditions that minimize launch energy while hitting the target

We want to transform this question (shooting the target with minimum energy) into a "nonlinear programming problem"

What variables to use to describe the motion?

One "parameterization" of the problem

- 3 unknown variables to be found by optimization
  - time of flight tflight
  - initial velocity x component vx0
  - initial velocity y component vy0

Another parameterization of the problem

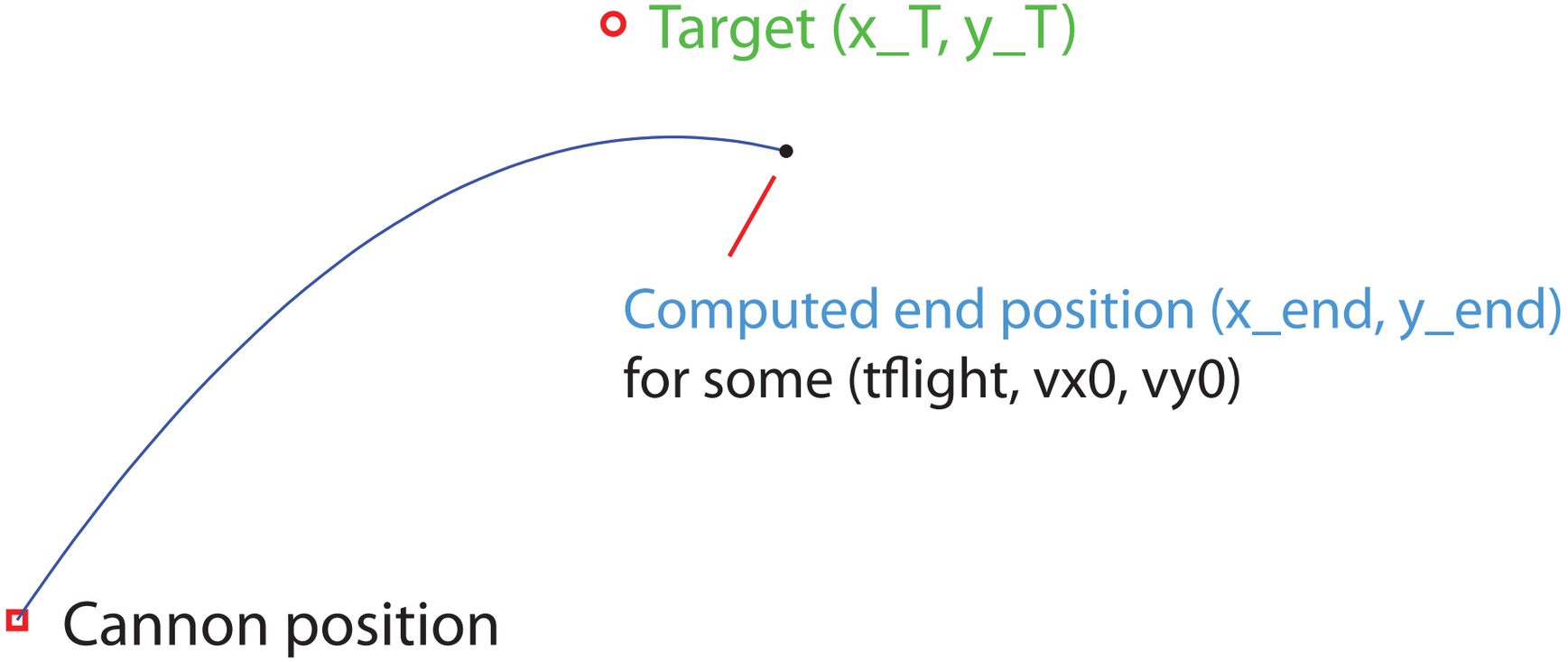
- 3 unknown variables to be found by optimization
  - time of flight tflight
  - initial velocity magnitude speed0
  - launch angle theta0

We'll use the first parameterization

- 3 unknown variables to be found by optimization
  - time of flight tflight
  - initial velocity x component vx0
  - initial velocity y component vy0

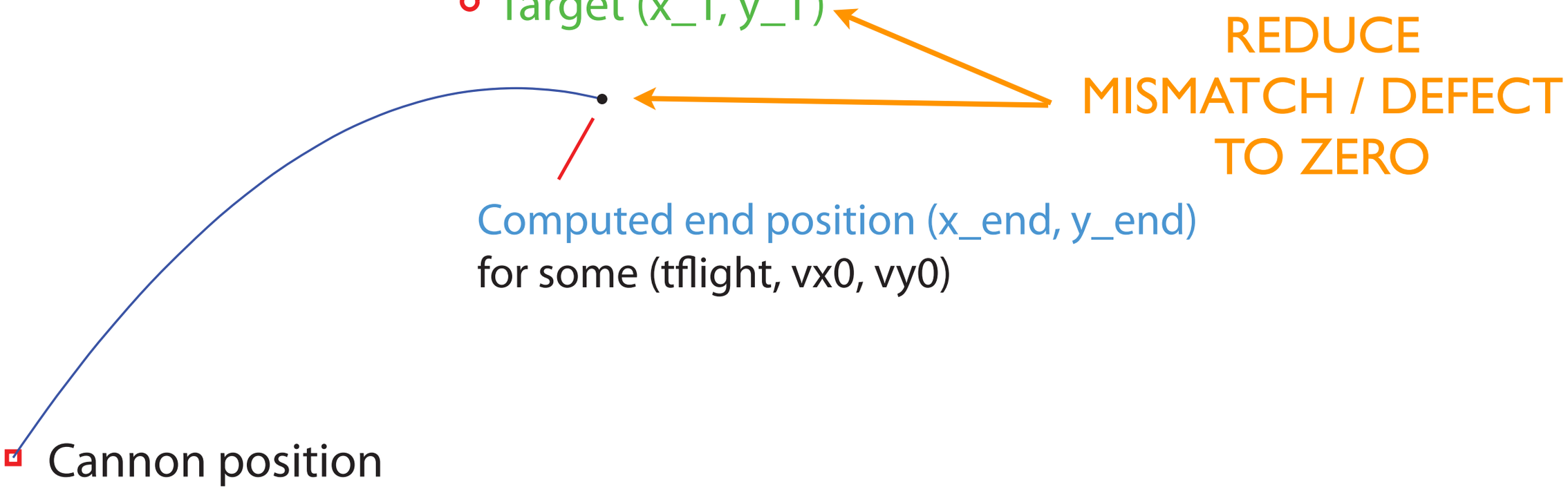
How to ensure hitting the target?

Computed flight trajectory for some (tflight, vx0, vy0)



How to ensure hitting the target?

Computed flight trajectory
for some (tflight, vx0, vy0)



Hitting the target

is equivalent to making

x_end = x_T and y_end = y_T

$$v_{\mathrm{x0}}\ t_{\mathrm{flight}} - x_{\mathrm{T}} = 0$$

$$v_{\mathrm{y0}}\ t_{\mathrm{flight}} - \frac{1}{2}\ g\ t_{\mathrm{flight}}^2 - y_{\mathrm{T}} = 0$$

TWO EQUALITY CONSTRAINTS
in the 3 unknown variables

Computed flight trajectory for some (tflight, vx0, vy0)

Target (x_T, y_T)

Computed end position (x_end, y_end)
for some (tflight, vx0, vy0)

Cannon position

What should the projectile minimize?
Objective function

Projectile energy = $v_{x0}^2 + v_{y0}^2$

(or some multiple or monotonic function of the above energy)

What is the optimal trajectory if you maximize the projectile energy?

Find $(t_{\text{tflight}}, v_{x0}, v_{y0})$ such that the function E:

$$E = v_{x0}^2 + v_{y0}^2 \tag{1}$$

is minimized, subject to the following equality constraints:

$$v_{x0}\ t_{\text{flight}} - x_T \quad = \quad 0 \tag{2}$$

$$v_{y0}\ t_{\text{flight}} - \frac{1}{2}\ g\ t_{\text{flight}}^2 - y_T \quad = \quad 0. \tag{3}$$

The following reasonable simple bounds may also be applied:

$$0.001\ (\text{say}) \quad \leq t_{\text{flight}} \quad < \infty \tag{4}$$

$$-\infty \quad < v_{x0} \quad < \infty \tag{5}$$

$$-\infty \quad < v_{y0} \quad < \infty \tag{6}$$

- Solution. See the folder CannonProblem for a MATLAB fmincon solution of this nonlinear optimization problem.

- Exercise. Use widely different initial guesses (initial seeds) and see if the optimization still converges.

- Exercise. Use different unknown variables to parameterize the problem and see if the convergence is similar. e.g., time of flight, launch angle and launch speed (that is, velocity magnitude), as noted earlier.

End of part 1

Part 2:

A simple trajectory optimization problem (using "single shooting")

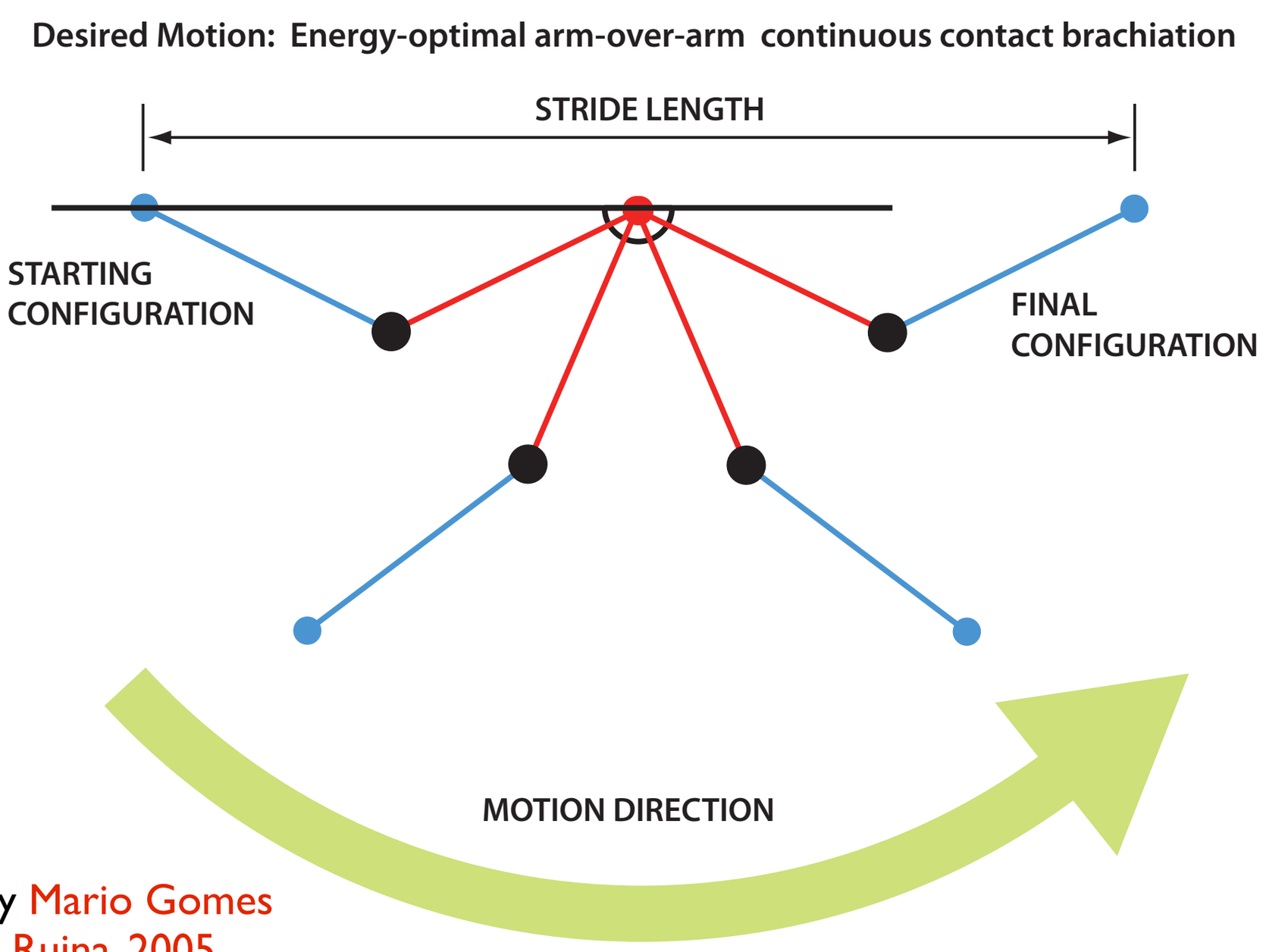
Will briefly mention other methods toward the end. 1) Multiple shooting 2) Direct collocation

Brachiation

Brachiation = locomotion of apes, swinging with their arms on trees, or other handholds, etc.

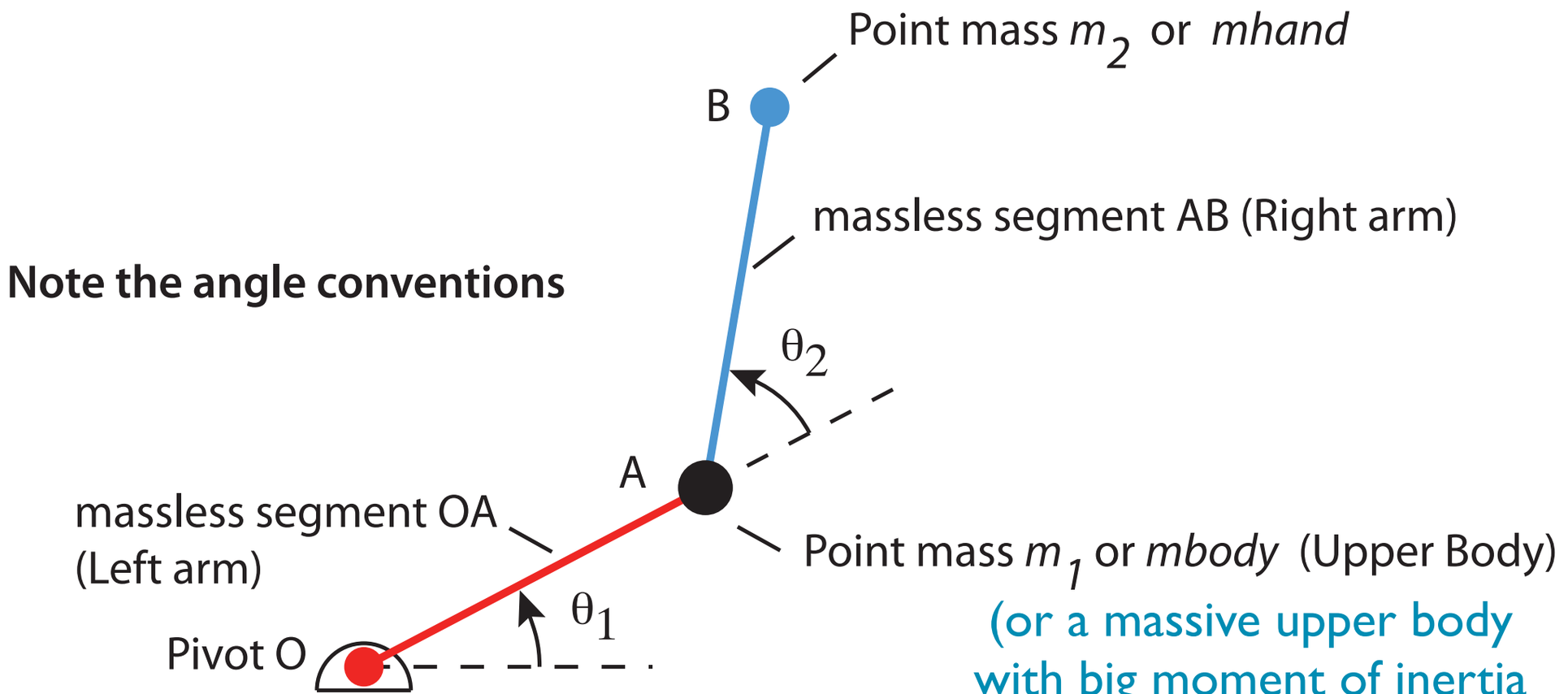
Video: John Bertram, U Calgary

For this tutorial, desired type of motion of the animal



See article by Mario Gomes and Andy Ruina, 2005

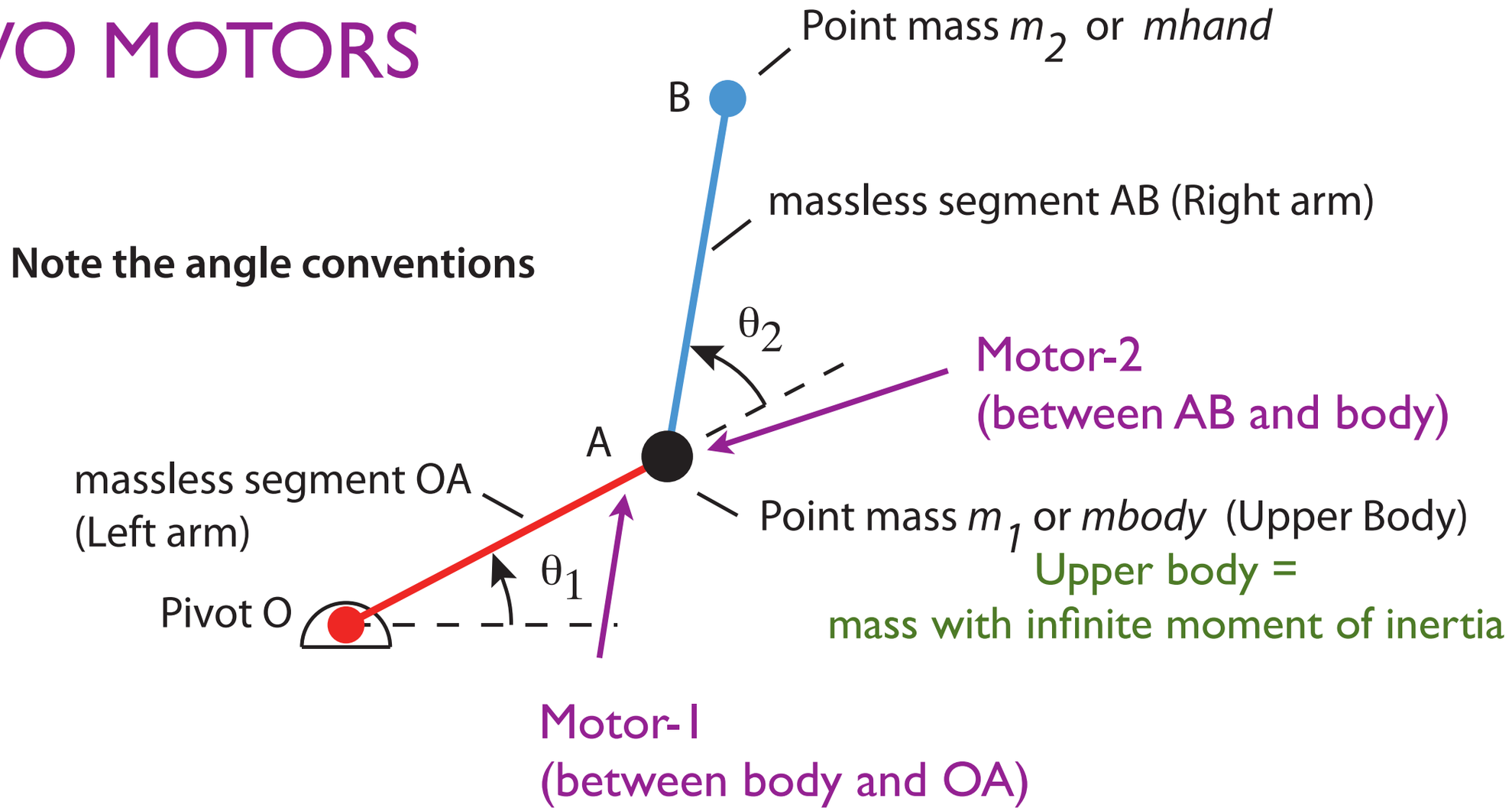
Mechanical Model of the animal (could be a biped, swinging ape, or a planar arm)



(or a massive upper body with big moment of inertia with shoulders at the center of mass, with zero)

Mechanical Model of the animal (could be a biped, swinging ape, or a planar arm)

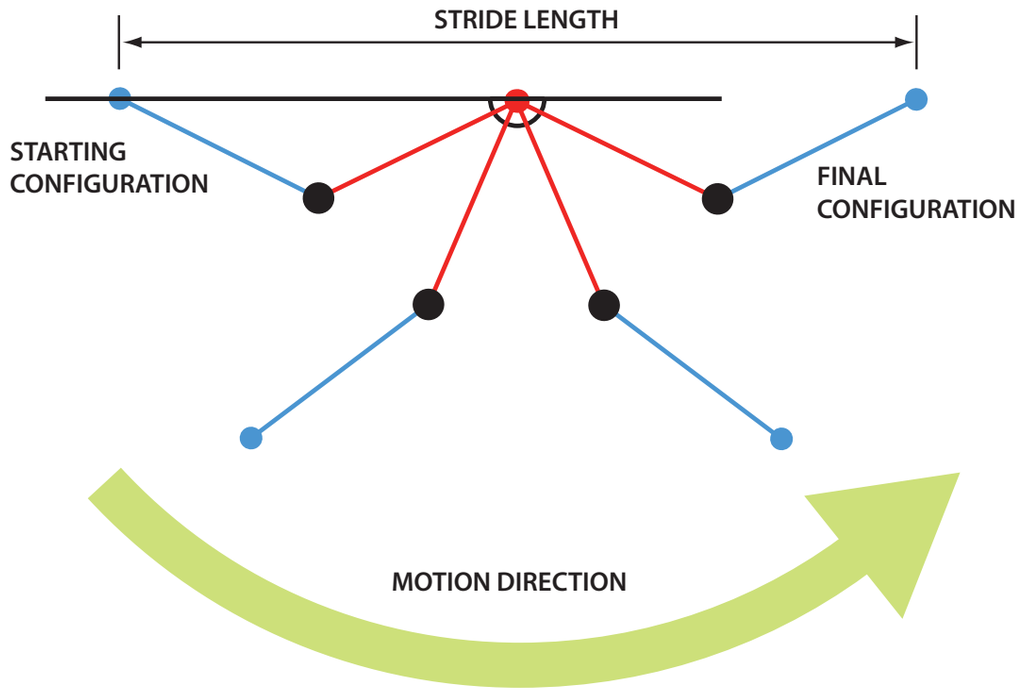
TWO MOTORS



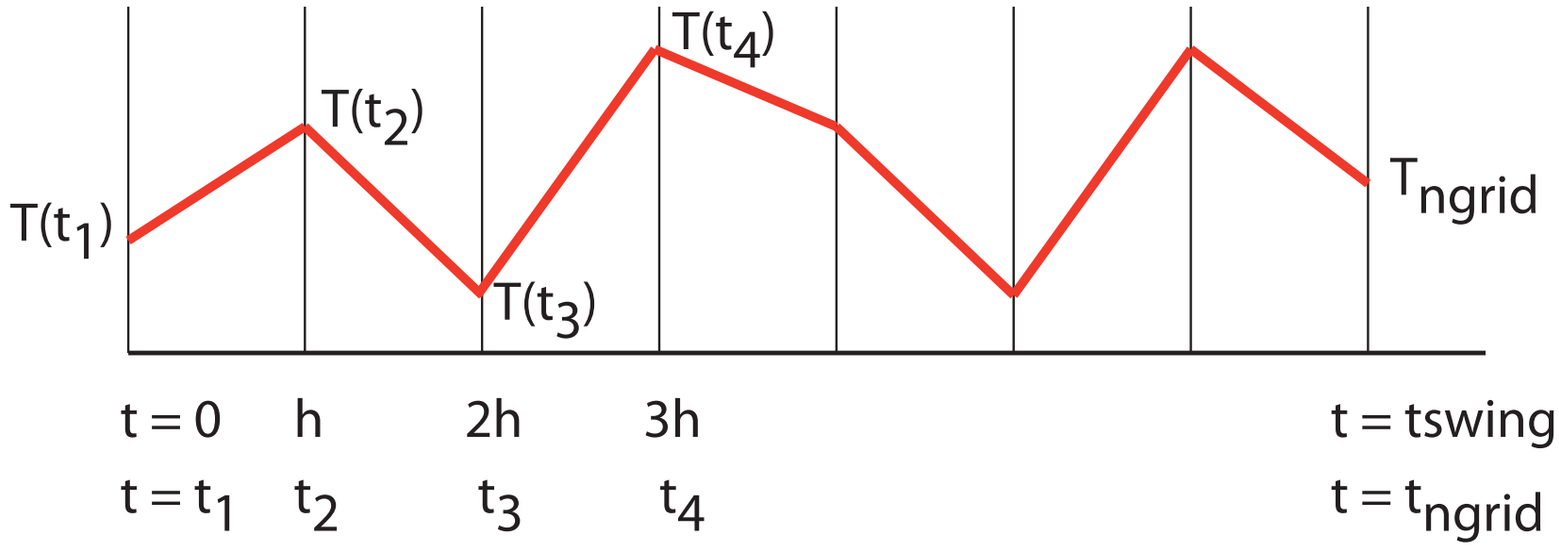
What set of variables is sufficient to describe the motion? (the "parameterization"):

- Initial conditions (4 numbers)
- The time of swing (1 number)
- Torques as functions of time (Infinitely many numbers)

Desired Motion: Energy-optimal arm-over-arm continuous contact brachiation



We make the problem finite dimensional by using a piecewise linear discretization of the joint torque (t)



Number of grid points = ngrid.

Each torque(t) is defined by ngrid variables.

Aside: why piecewise linear?

I like piece-wise linear because the torque can change abruptly if and when necessary for optimality (e.g., in bang-bang control).

But you might want to use higher order polynomials if

1) the objective function depends on higher derivatives of the torque (e.g., when there is a spring in series)

2) the control can only change smoothly, as when there are bounds on the derivatives of the torque.

Total number of unknowns

- time of swing = 1
- Initial conditions = 4 (theta1, theta1dot, theta2, theta2dot)
- torques = 2 x ngrid.
- Total = 5+ 2 x ngrid.

Objective function

Integral torque-squared

$$\int \left(T_1^2 + T_2^2\right) dt \tag{1}$$

Absolute work cost (try this and see how the optimization often fails because of the nonsmoothness)

$$\int \left(\, |P_1| + |P_2| \, \right) dt \tag{2}$$

where $P_1 = T_1\dot{\theta}_1$ and $T_2\dot{\theta}_2$.

Smoothed version of absolute work cost

$$\int \left(\sqrt{P_1^2 + \epsilon^2} + \sqrt{P_2^2 + \epsilon^2} \right) dt \tag{3}$$

where ϵ is a small number.

Equality constraints

- The hand B starts and ends at the ceiling with zero.

$$y_B = 0,\ y_B(tswing) = 0$$

- All the angular velocities are equal to zero initially and finally.

$$\dot{\theta}_1(0) = 0,\ \dot{\theta}_2(0) = 0$$
$$\dot{\theta}_1(\text{tswing}) = 0,\ \dot{\theta}_2(\text{tswing}) = 0$$

Inequality constraints

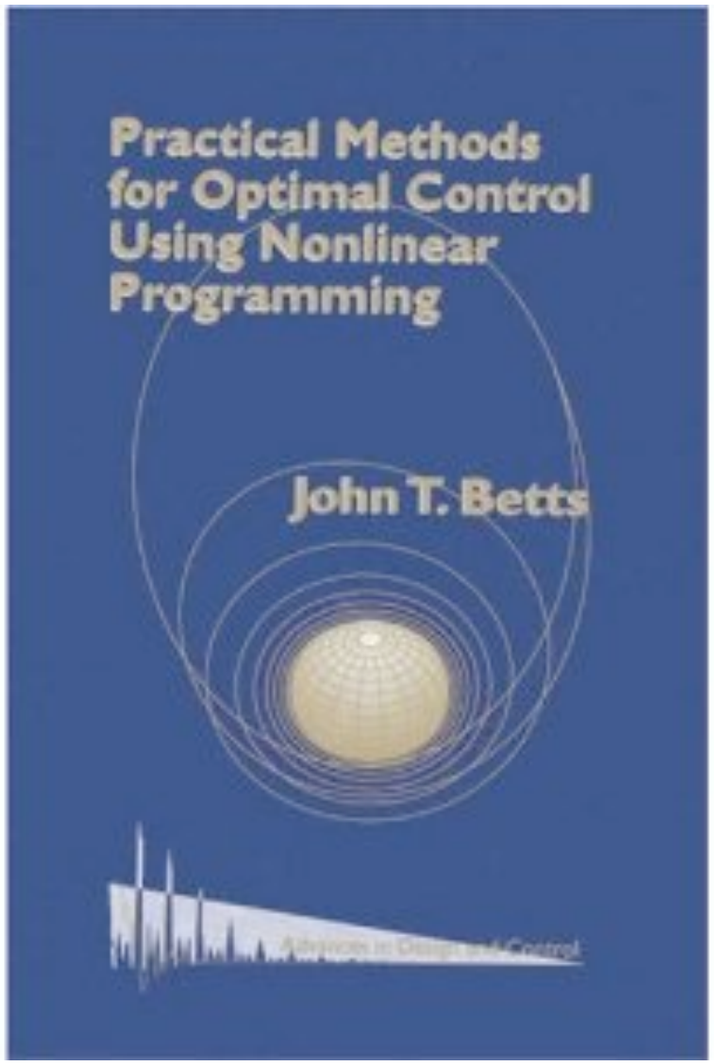
- The hand B starts at the left of the pivot O and ends at the right.

$$x_B(0) \quad < \quad 0$$
$$x_B(\text{tswing}) \quad > \quad 0$$

- That's it, really.
- You can add other inequality constraints to guide the solution to have properties that you prefer.

On to MATLAB

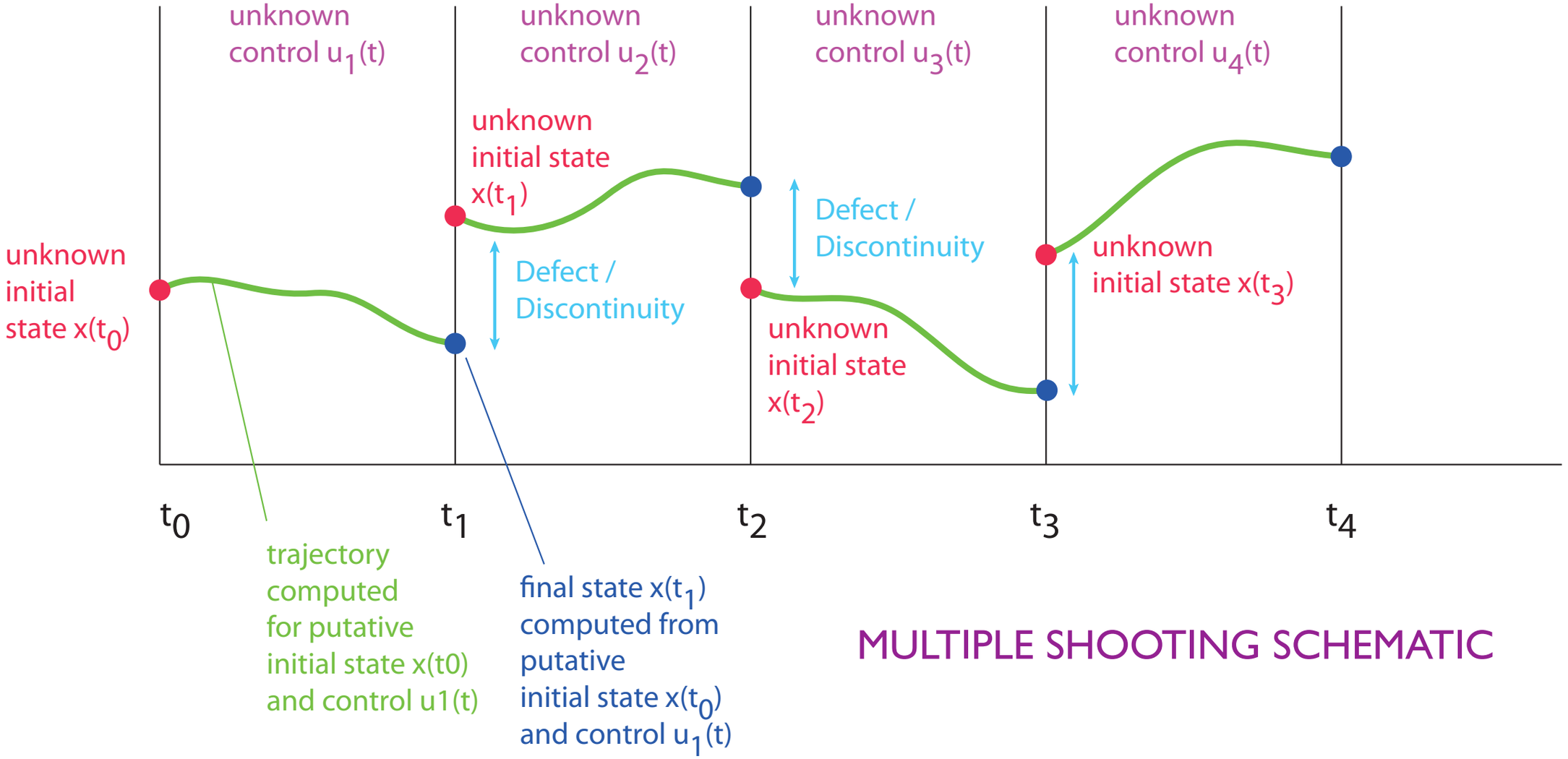
but before that ...



Practical Methods for Optimal Control
Using Nonlinear Programming
John T. Betts

Good and/but somewhat technical book on the various issues related to numerical optimal control -- good discussion of "Direct Collocation" Methods.

Multiple Shooting



MULTIPLE SHOOTING SCHEMATIC

- Break up the trajectory into multiple segments, each with its own unknown initial state and unknown control function u(t)
- Minimize the total cost subject to all original constraints + new constraints that ensure that the state is continuous across different segments. That is, we want the defect / discontinuity to be zero.

Direct collocation

- Very similar to Multiple Shooting except:
    - each time-segment has only one ODE-integration step (typically using an implicit ODE scheme)
    - the number of segments is many more so that ODEs can be integrated with sufficient accuracy and stability

Other resources

Chris Atkeson's webpage:
http://www.cs.cmu.edu/~cga/walking/grad.html

Some softwares that do the "trajectory optimization" transcription for you:

- SOCS - Sparse Optimal Control Software (Boeing, John Betts et al)
- DIRCOL - Direct Collocation (von Stryk et al, TU-Darmstadt)
- TOMLAB's PROPT
- Katja Mombaur et al.?

Random extra slides

Why not use "continuous time" necessary conditions of optimality?

E.g., Pontryagin's maximum/minimum principle, etc.

- "No one" seems to be doing this any more
- Gives two point boundary value problems for the state, with potentially unpredictably switching controls -- seems harder to solve than discretizing and then optimizing (Betts' terminology).

What type of optimization algorithm to use?

- **Smooth methods** (those that assume and use first and second derivative continuity e.g., fmincon, SNOPT -- Sequential Quadratic Programs) are usually faster than those that do not. The downside is that the problem needs to be sufficiently smooth.

- Genetic Algorithms, Simulated Annealing, Nelder-Mead (Downhill) Simplex Method. etc. Probably too slow without combining them with derivative-based methods.

- Might want to use some Direct Search methods (derivative free) if your problem is really non-smooth and you do not want to smooth things.

Ideally, use software that always respect bound constraints

So we can enforce desirable things like "time durations" are always positive, muscles are always tensional, etc.

- SNOPT, etc
- Latest MATLAB's fmincon, when using certain algorithms (interior-point, sqp, etc)

fmincon assumes continuity of second derivatives

- If the optimization problem is somehow non-smooth, then fmincon may not converge or might give up at non-optima.
- So make sure your evaluations are smooth

Smoothness of function evaluation

- If the algorithms stop at a non-minimum, it might be getting stuck at some point where the derivative estimates are bad (so the algorithm does not know which way to do).

Avoid using event detection to stop integration exactly at the collision, before swing time is up

- This would mean that some of the control parameters have zero effect on the motion
- The motion will depend non-smoothly on the control parameters
- Additional non-smoothness introduced because of the interaction between the collision and the ODE integration steps

Numerical derivatives

- Need to have good smooth numerical derivatives
    - Finite differencing (need to be careful with the differencing step size and how that interacts with function evaluation accuracy)
    - Automatic Differentiation ... (Algorithmic Differentiation)

Getting derivatives

- Finite differencing (automatic in MATLAB) -- usually good enough
- Automatic Differentiation (sometimes called Algorithmic Differentiation)

Use ode45 with high accuracy

- Or a constant step-size method (if your ordinary differential equations are sufficiently well-behaved i.e., not stiff)

See discussion of Consistency vs Accuracy in Betts (2001).

Alternate parameterization for shooting

- discretize thetadotdot.
- Integrate to find thetadot and theta
- Solve a linear system to find the joint torques (inverse dynamics).
- Compute cost and constraints
- advantages: more direct control over the motion -- the angles and angle rates are linear functions of the unknowns. Easy to give initialinitail seeds and constraints on angles, etc are linear constraints.

Advantage of Multiple shooting or Direct collocation

- Makes the problem less nonlinear (sort of) and more well-conditioned ...

- You can judiciously use of
    - inequality and bound constraints to rule out motions you do not care for
    - use an initial seed of states that most closely resembles the motion one is looking for.