Towards a Universal Temporal Ordering of Discrete Events for Bipedal Walking via the Optimal Control of Switched Systems

Ram Vasudevan

Department of Electrical Engineering and Computer Sciences
University of California, Berkeley

May 5th, 2011

Which is More Anthropomorphic?

Prior Work

Focus of work on bipedal robotics has been either on studying stability or minimizing some energy function.

	1 mode	2 mode	3 mode	4 mode	5 mode
w/o F, w/o K	McGeer 1990, Goswami 1996	–	–	–	–
w/o F, w/ K	Grizzle 2001	Ames 2006, McGeer 1990	–	–	–
w/ F, w/o K	Grizzle 2001	Tlalonini 2009	Schaub 2009, Tlalonini 2009	–	–
w/ F, w/ K	Grizzle 2001	Choi 2005, Tlalolini 2009	Schaub 2009	Braun 2009, Grizzle 2010	Sinnet 2009

Preview of Our Result

1. Experimentally show there exists a *universal* temporal ordering of discrete events for bipedal walking.
2. Construct a metric to determine the anthropomorphism of gait.
3. Develop an algorithm for the optimal control of constrained nonlinear switched dynamical systems which provably converges to local minima of our problems.

Hybrid Systems

Hybrid System

A hybrid system $\mathcal{H}$ is a tuple (Γ, D, U, G, R, FG) where

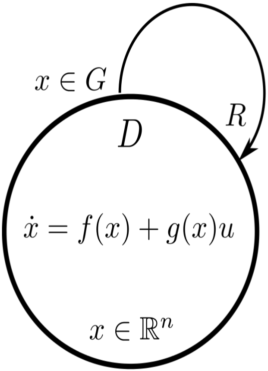
- $\Gamma = (V, E)$ is an oriented graph,
- $D = \{D_v\}_{v \in V}$ is a set of domains,
- $U = \{U_v\}_{v \in V}$ is a set of controls,
- $G = \{G_e\}_{e \in E}$ is a set of guards,
- $R = \{R_e\}_{e \in E}$ is a set of reset maps,
- $FG = \{(f_v, g_v)\}_{v \in D}$ is a set of vector fields.

Hybrid Systems

Hybrid System

A hybrid system $\mathcal{H}$ is a tuple (Γ, D, U, G, R, FG) where

- $\Gamma = (V, E)$ is an oriented graph,
- $D = \{D_v\}_{v \in V}$ is a set of domains,
- $U = \{U_v\}_{v \in V}$ is a set of controls,
- $G = \{G_e\}_{e \in E}$ is a set of guards,
- $R = \{R_e\}_{e \in E}$ is a set of reset maps,
- $FG = \{(f_v, g_v)\}_{v \in D}$ is a set of vector fields.

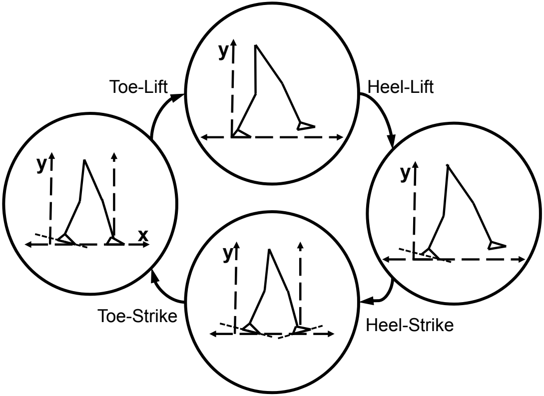


From Contact Points to a Hybrid System

The bipedal robot is modeled as a hybrid control system:

$$\mathcal{H} = (\Gamma, D, U, G, R, FG).$$

- Γ is an oriented graph.

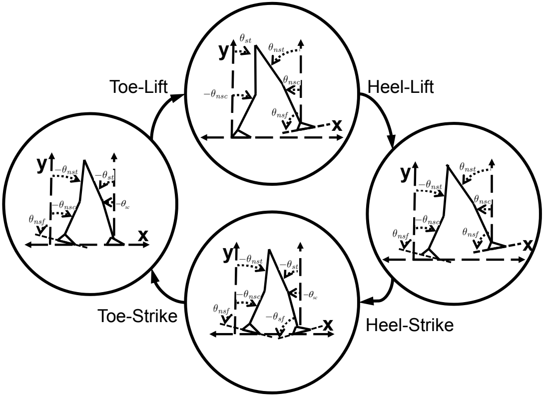


From Contact Points to a Hybrid System

The bipedal robot is modeled as a hybrid control system:

$$\mathcal{H} = (\Gamma, D, U, G, R, FG).$$

- D is the phase space of the configuration spaces for each discrete domain.

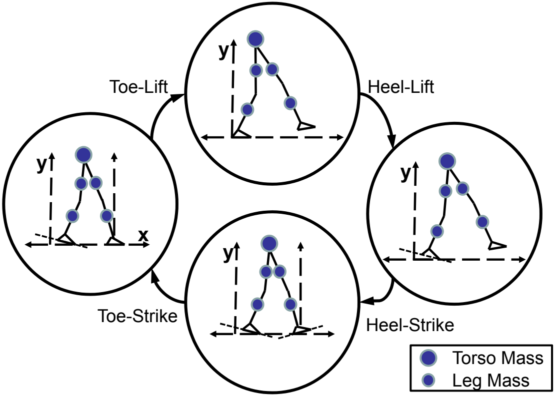


From Contact Points to a Hybrid System

The bipedal robot is modeled as a hybrid control system:

$$\mathcal{H} = (\Gamma, D, U, G, R, FG).$$

- *FG* are the control systems (f_i, g_i) obtained from the Lagrangians L_i on each domain.

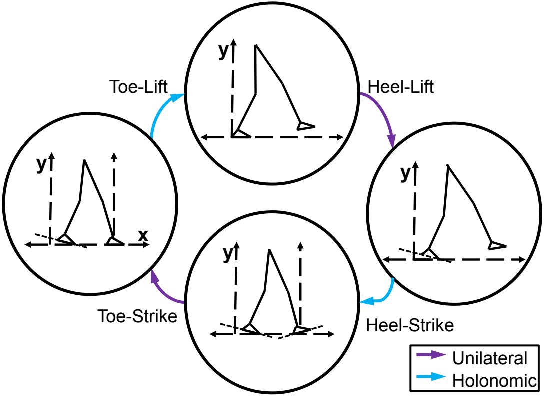


From Contact Points to a Hybrid System

The bipedal robot is modeled as a hybrid control system:

$$\mathcal{H} = (\Gamma, D, U, G, R, FG).$$

- G and R are obtained from kinematic and holonomic constraints.

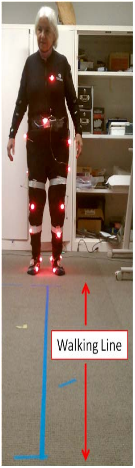


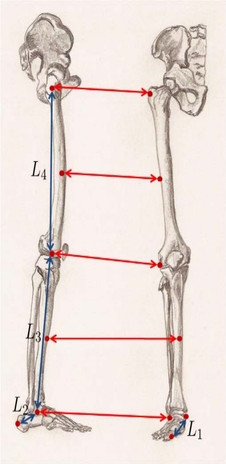
Hybrid Models for Bipedal Walking

- The sequence of contact points along with a Lagrangian that is intrinsic to the biped completely determines the hybrid model.
- The sequence of contact points can be arbitrarily complex, where do we focus our attention?
- Focus our attention on 6 contact points: left knee [*lk*], left heel [*lh*], left toe [*lt*], right knee [*rk*], right heel [*rh*], and right toe [*rt*].

The Setup

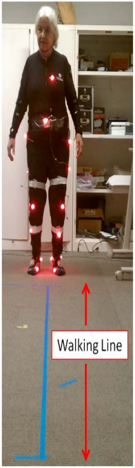
- Use a 12-camera motion capture system (480 fps, approximately 1mm accuracy), to record the 3D position of 19 LED sensors.
- To simplify the data analysis each subject was required to place their right foot at the starting point of the blue line at the start of the experiment and was required to repeat the experiment 12 times.

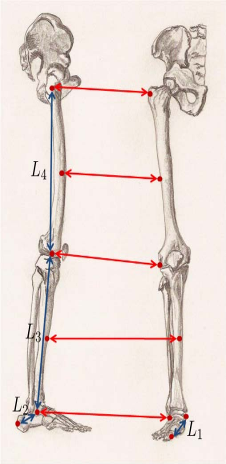




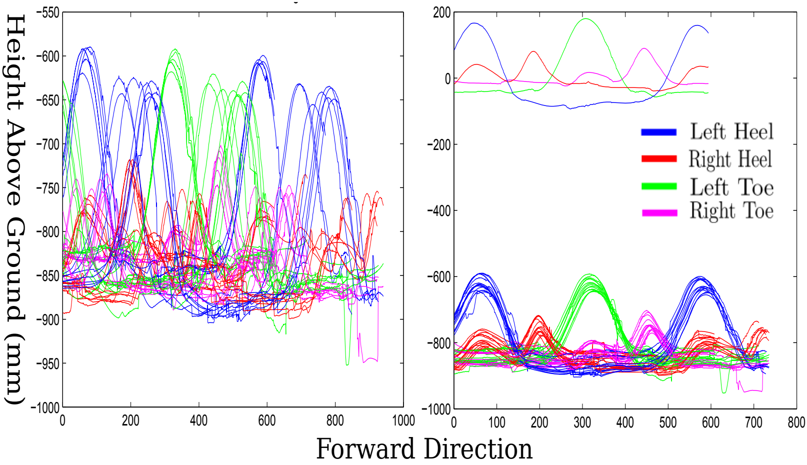
The Participants

	Sex	Age	Weight	Height	L_1	L_2	L_3	L_4
1	M	30	90.7	184	14.5	8.50	43.0	44.0
2	F	19	53.5	164	15.0	8.00	41.0	44.0
3	M	17	83.9	189	16.5	8.00	45.5	55.5
4	M	22	90.7	170	14.5	9.00	43.0	39.0
5	M	30	68.9	170	15.0	8.00	43.0	43.0
6	M	29	59.8	161	14.0	8.50	37.0	40.0
7	M	26	58.9	164	14.0	9.00	39.0	41.0
8	F	77	63.5	163	14.0	8.00	40.0	42.0
9	F	23	47.6	165	15.0	8.00	45.0	43.0



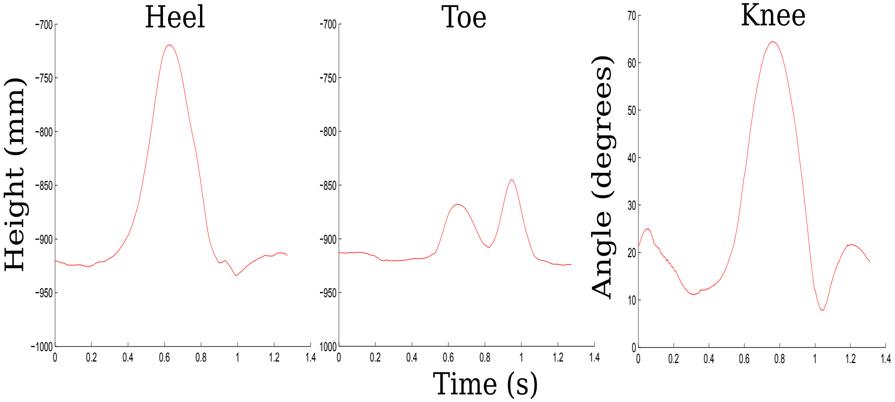


The Pre-Processing



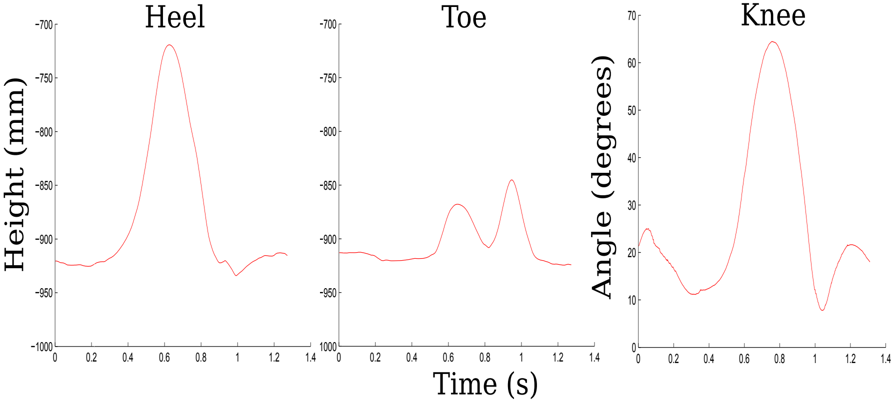
1. Interpolation $\longrightarrow$ 2. Rotation $\longrightarrow$ 3. Averaging

What About the Signal?



1. Treat each signal independently.
2. Treat contact point as constrained when signal is constant.

What About the Signal?



1. Treat each signal independently.
2. Treat contact point as constrained when signal is constant.

Detecting Constraint Enforcement

Let $y : [0, T] \to \mathbb{R}$ be a contact point sensor.

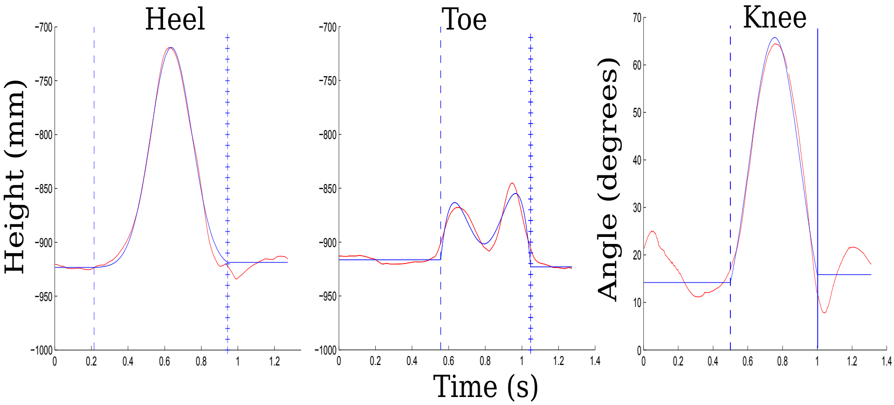
- Suppose we have a canonical walking function $s(t; \phi)$ we expect the contact point sensor to behave like:

$$c(t; \phi, \tau_l, \tau_s) = \begin{cases} \text{constant}_1(\phi) & \text{if} \quad t \leq \tau_l \\ s(t; \phi) & \text{if} \quad \tau_l < t < \tau_s \\ \text{constant}_2(\phi) & \text{if} \quad \tau_s \leq t \end{cases}$$

- Determine ϕ, τ_l, τ_s by solving an optimization problem:

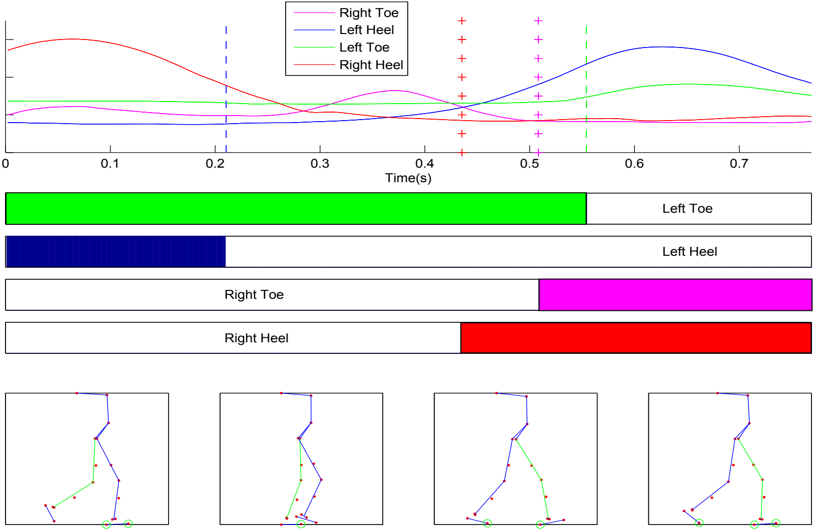
$$min_{\tau_l, \tau_s \in [0,T]} min_{\phi \in \mathbb{R}^k} \frac{1}{T} \sum_{t=0}^{T} |c(t; \phi, \tau_l, \tau_s) - y(t)|$$

Functions to Fit

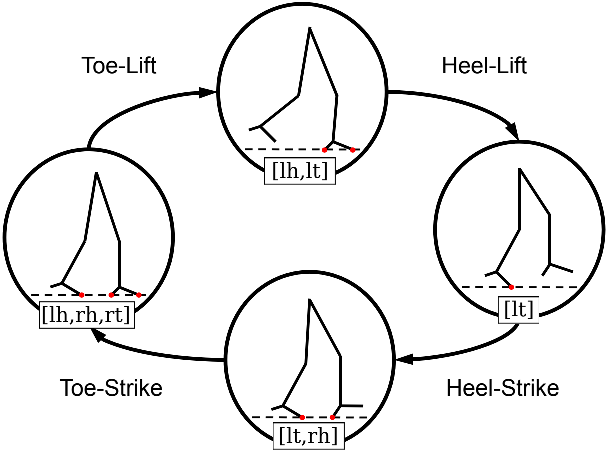


Persistence

Determining a Temporal Ordering of Discrete Events



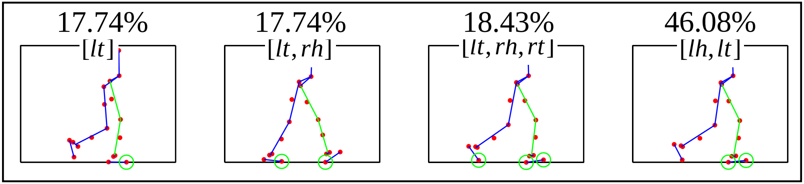
The Universal Temporal Ordering w/o Knee-Lock



There was a common temporal ordering of discrete events for all human subjects in the case w/o knee-lock.

Walking Cycles

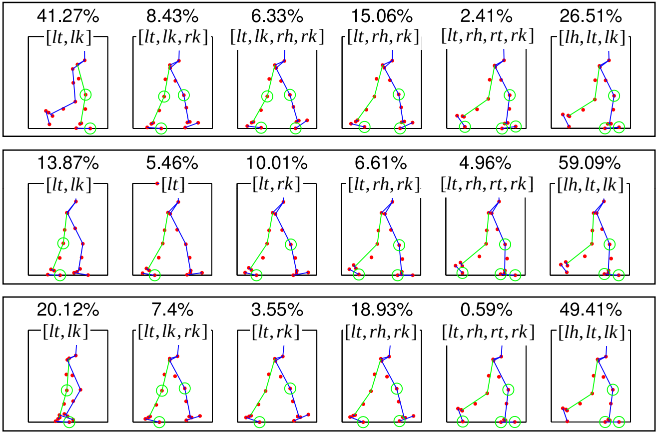
The difference between each subject is found in the amount of time spent in each mode, which we call a walking cycle:



This can be represented as a weighted graph (α, ℓ):

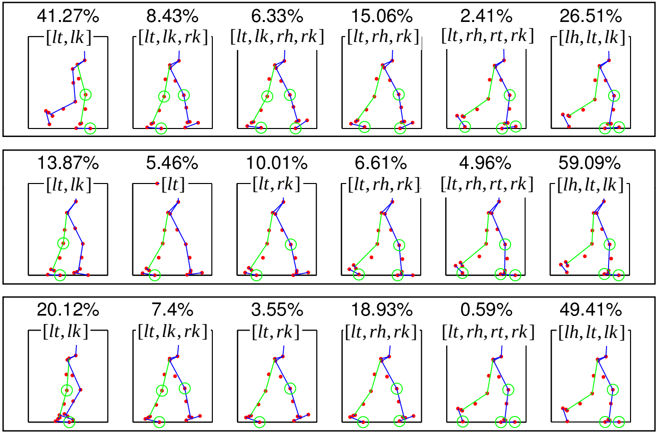
$$\begin{array}{rcccccccc} \ell & : & [lt] & \rightarrow & [lt, rh] & \rightarrow & [lt, rh, rt] & \rightarrow & [lh, lt] \\ \alpha(\ell) & : & 17.74\% & \rightarrow & 17.74\% & \rightarrow & 18.43\% & \rightarrow & 46.08\% \end{array}$$

The Universal Temporal Ordering w/ Knee Lock



1. 7 distinct temporal orderings!
2. Issues probably arise due to poor knee fitting, but they still seem to have a lot in common.
3. Can we measure how much they have in common?

The Universal Temporal Ordering w/ Knee Lock



1. 7 distinct temporal orderings!
2. Issues probably arise due to poor knee fitting, but they still seem to have a lot in common.
3. Can we measure how much they have in common?

Comparing Weighted Graphs

We are after a weighted graph metric that satisfies the following properties:

1. penalizes those walking cycles that do not have domains in common
2. penalizes those walking cycles that do not have transitions in common

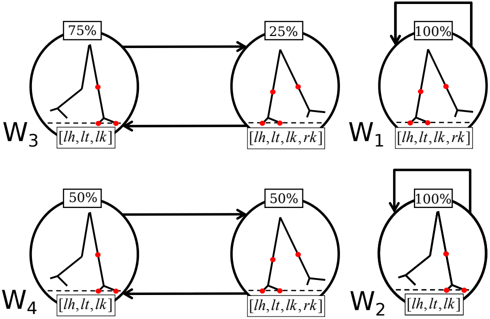
An example of such a metric is the cut metric.

Comparing Weighted Graphs

We are after a weighted graph metric that satisfies the following properties:

1. penalizes those walking cycles that do not have domains in common
2. penalizes those walking cycles that do not have transitions in common

An example of such a metric is the cut metric.

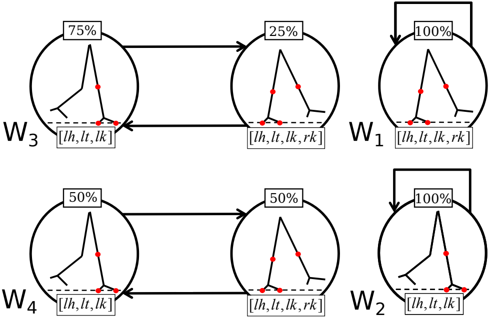


Comparing Weighted Graphs

We are after a weighted graph metric that satisfies the following properties:

1. penalizes those walking cycles that do not have domains in common
2. penalizes those walking cycles that do not have transitions in common

An example of such a metric is the cut metric.



Cut Distance between pairs:

	W_1	W_2	W_3	W_4
W_1	0.000	3.000	2.500	2.000
W_2	3.000	0.000	1.500	2.000
W_3	2.500	1.500	0.000	0.625
W_4	2.000	2.000	0.625	0.000

The Optimal Walking Cycle and the Human Based Cost

Optimal Walking Cycle

Letting $\mathcal{L} = \bigcup_{i=1}^{N} \ell_i$ be the graph obtained by combining all N cycles ℓ_i, we define the *optimal walking cycle* denoted (α^*, ℓ^*) by:

$$\underset{(\alpha,\ell)\in\mathbb{R}^{|\ell|}\times\mathcal{L}}{\operatorname{argmin}} \frac{1}{N}\sum_{i=1}^{N} d(\alpha, \ell, \alpha_i, \ell_i)$$

Human Based Cost (HBC)

Given a biped with walking cycle (α_r, ℓ_r), the *human-based cost (HBC) of walking* is:

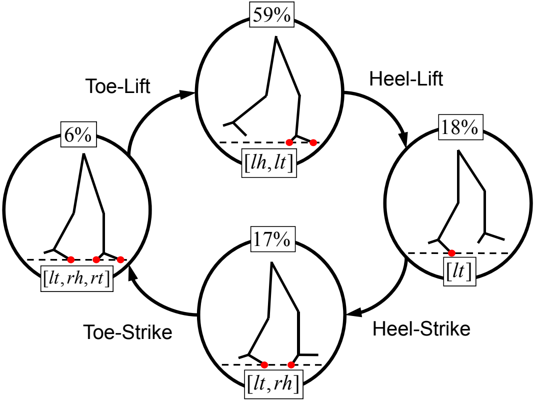
$$\mathcal{H}(R) = d(\alpha_r, \ell_r, \alpha^*, \ell^*).$$

The Optimal Walking Cycle and the Human Based Cost

Optimal Walking Cycle

Letting $\mathcal{L} = \bigcup_{i=1}^{N} \ell_i$ be the graph obtained by combining all N cycles ℓ_i, we define the *optimal walking cycle* denoted (α^*, ℓ^*) by:

$$\underset{(\alpha,\ell)\in\mathbb{R}^{|\ell|}\times\mathcal{L}}{\operatorname{argmin}} \frac{1}{N}\sum_{i=1}^{N} d(\alpha, \ell, \alpha_i, \ell_i)$$



Human Based Cost (HBC)

Given a biped with walking cycle (α_r, ℓ_r), the *human-based cost (HBC) of walking* is:

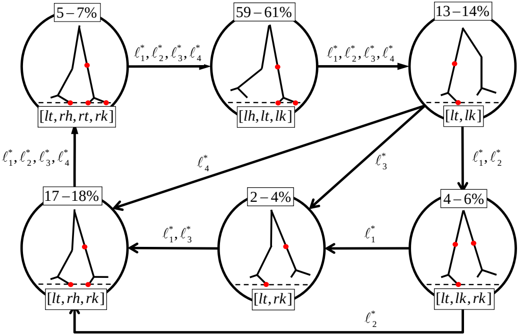
$$\mathcal{H}(R) = d(\alpha_r, \ell_r, \alpha^*, \ell^*).$$

The Optimal Walking Cycle and the Human Based Cost

Optimal Walking Cycle

Letting $\mathcal{L} = \bigcup_{i=1}^{N} \ell_i$ be the graph obtained by combining all N cycles ℓ_i, we define the *optimal walking cycle* denoted (α^*, ℓ^*) by:

$$\underset{(\alpha,\ell)\in\mathbb{R}^{|\ell|}\times\mathcal{L}}{\text{argmin}} \frac{1}{N}\sum_{i=1}^{N} d(\alpha,\ell,\alpha_i,\ell_i)$$



Human Based Cost (HBC)

Given a biped with walking cycle (α_r, ℓ_r), the *human-based cost (HBC) of walking* is:

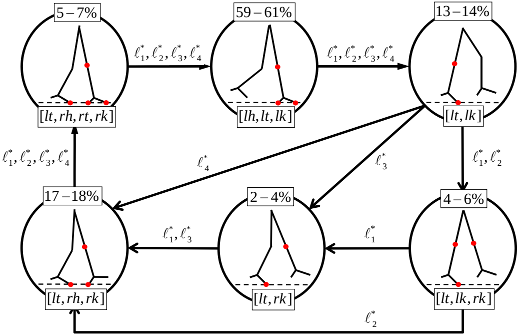
$$\mathcal{H}(R) = d(\alpha_r, \ell_r, \alpha^*, \ell^*).$$

The Optimal Walking Cycle and the Human Based Cost

Optimal Walking Cycle

Letting $\mathcal{L} = \bigcup_{i=1}^{N} \ell_i$ be the graph obtained by combining all N cycles ℓ_i, we define the *optimal walking cycle* denoted (α^*, ℓ^*) by:

$$\underset{(\alpha,\ell)\in\mathbb{R}^{|\ell|}\times\mathcal{L}}{\operatorname{argmin}} \frac{1}{N}\sum_{i=1}^{N} d(\alpha, \ell, \alpha_i, \ell_i)$$

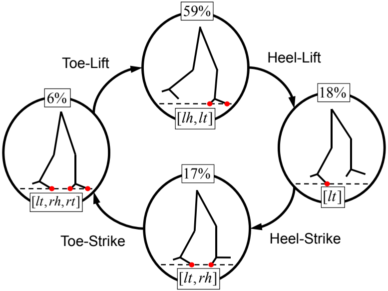


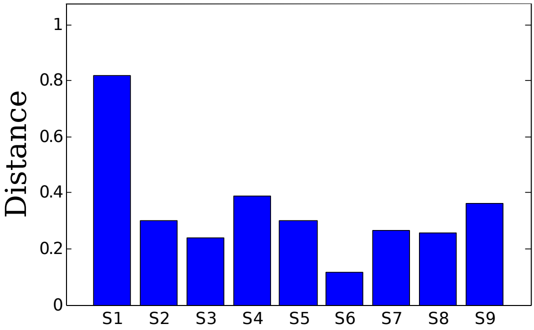
Human Based Cost (HBC)

Given a biped with walking cycle (α_r, ℓ_r), the *human-based cost (HBC) of walking* is:

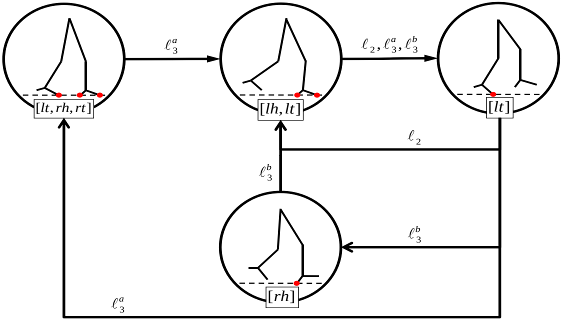
$$\mathcal{H}(R) = d(\alpha_r, \ell_r, \alpha^*, \ell^*).$$

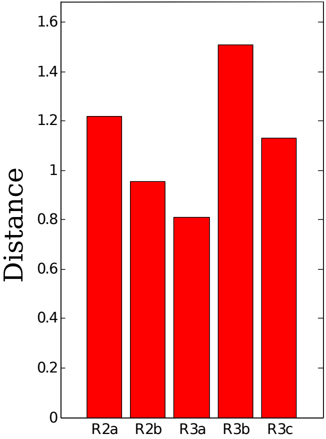
The HBC: w/o Knee-Lock





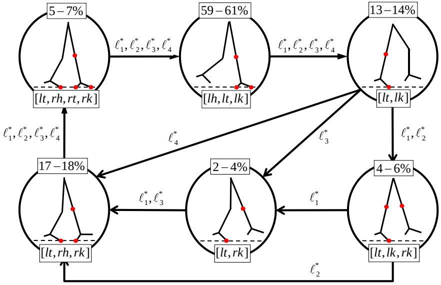
The HBC: w/o Knee-Lock

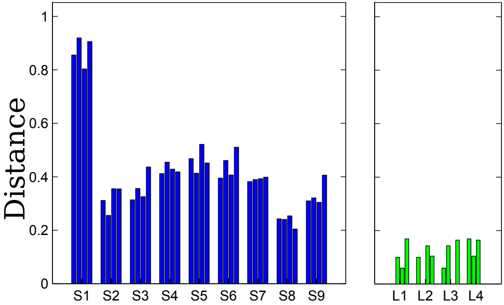




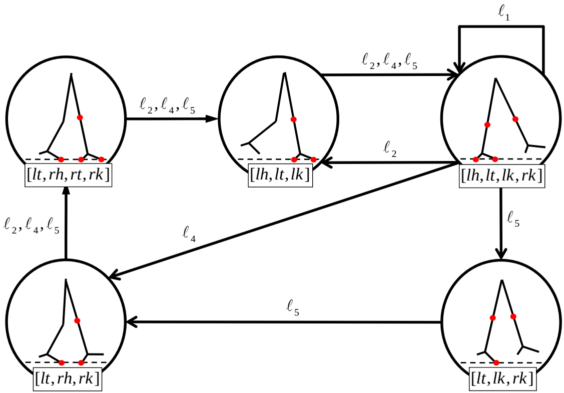
Models are drawn from Tlaloni et al. 2009 and Schaub et al. 2009.

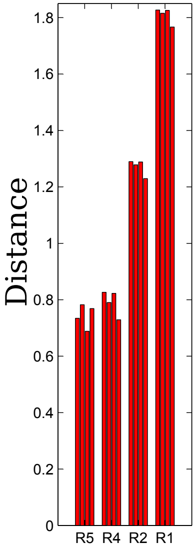
The HBC: w/ Knee-Lock





The HBC: w/ Knee-Lock





Models are drawn from Goswami et al. 1996, Ames et al. 2009, and Sinnet et al. 2009.

Mini-Conclusion

A Teaser

Definition: Switched Dynamical System

- Let $\mathcal{Q} = \{1, \ldots, Q\}$ be the set of modes.
- Let $\{f_q\}_{q \in \mathcal{Q}}$ be a set of vector fields, $f_q : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$.
- Consider a system governed by the following differential equation:

$$\dot{x}(t) = f_{\pi(t)}\big(x(t), u(t)\big), \quad x(0) = x_0$$

  where $u : [0, \infty), \to \mathbb{R}^m$, and $\pi : [0, \infty) \to \mathcal{Q}$.
- Let NF denote the mode in which the trajectories stop evolving, i.e. $f_{\text{NF}}(x, u) = 0$.

Relaxing Constraint Satisfaction

Prior Work: Switched System Optimization

Modeling the Optimization Problem

- Given a fixed initial condition, $x_0 \in \mathbb{R}^n$, the trajectory of a "classical" continuous dynamical system is determined by a continuous-valued input, u.
- Given a fixed initial condition, $x_0 \in \mathbb{R}^n$, the trajectory of a switched dynamical system is determined by a continuous-valued input, u, and a discrete-valued input, π.

Idea

We need to *encode* the two types of inputs in a way that allows for the application of as much existing optimal control theory as possible.

Modeling the Discrete-Valued Input

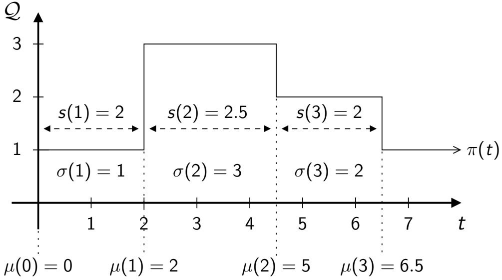
1. A mode sequence, σ, is an element in the Mode Sequence Space:

$$\Sigma = \bigcup_{N=1}^{\infty} \left\{ \sigma \in (\mathcal{Q} \cup \mathrm{NF})^{\mathbb{N}} \,\middle|\, \sigma(j) \in \mathcal{Q}\ j \leq N,\ \sigma(j) = \mathrm{NF}\ j > N \right\}.$$

2. A transition time sequence, s, represents the time spent in each mode in σ and is an element in the Switching Time Sequence Space:

$$\mathcal{S} = \left\{ s \in l^1 \,\middle|\, s(j) \geq 0 \ \ \forall j \leq N,\ s(j) = 0\ \forall j > N \right\}.$$

Define $\mu(i) = \sum_{k=1}^{i} s(k)$, and $\mu_f = \sum_{k=1}^{\infty} s(k)$.



Continuous Input and Waypoint Spaces

1. Let the Continuous Input Space be:

$$\mathcal{U} = \left\{ u \in L^2([0,\infty), \mathbb{R}^m) \mid u(t) \in U, \ \forall t \in [0,\infty) \right\},$$

   where $U \subset \mathbb{R}^m$ is a compact, connected set containing the origin.
2. Let the Waypoint Space be $\mathbb{N}^W$, where W is equal to the number of waypoints.
   - Associates each waypoint to a particular element of a modal sequence.
   - Specifically it gives an index into the modal sequence space.
   - Utility becomes clear only after considering implementation of the algorithm.

Optimization Space

- Given $\sigma \in \Sigma$, let $\#\sigma = \max\{j \in \mathbb{N} \mid \sigma(j) \neq \text{NF}\}$, i.e. $\#\sigma$ is the number of non-trivial modes in the sequence.
- Combine the four spaces together to define an Optimization Space as:

$$\mathcal{X} = \big\{(\sigma, s, u, w) \in \Sigma \times \mathcal{S} \times \mathcal{U} \times \mathbb{N}^W \mid \\ s(k) = 0 \ \forall k > \#\sigma, \text{and } w(i) \leq \#\sigma \ \forall i\big\},$$

- Denote $\xi \in \mathcal{X}$ by a 4–tuple $\xi = (\sigma, s, u, w)$.
- Meterize the optimization space by letting:

$$d(\xi_x, \xi_y) = \mathbb{1}\{\sigma_x \neq \sigma_y\} + \|s_x - s_y\|_{l^1} + \|u_x - u_y\|_2 + \mathbb{1}\{w_x \neq w_y\},$$

where $\|\cdot\|_{l^1}$ is the l^1–norm and $\|\cdot\|_2$ is the L^2–norm.

Optimization Problem

- Given a $\xi \in \mathcal{X}$ and an initial condition, x_0, the corresponding trajectory, $x^{(\xi)}(t)$, is the unique solution to:

$$\dot{x}^{(\xi)}(t) = f_{\pi(t;\xi)}\big(x^{(\xi)}(t), u(t)\big), \quad \forall t \in (0, \mu_f]$$
$$x^{(\xi)}(0) = x_0,$$

- Let $J : \mathcal{X} \to \mathbb{R}$ be the cost function:

$$J(\xi) = \int_0^{\mu_f} L\big(x^{(\xi)}(t), u(t)\big)\, dt + \sum_{i=1}^{W} \phi_i\big(x^{(\xi)}(\mu(w(i)))\big) + \phi\big(x^{(\xi)}(\mu_f)\big),$$

  where each of the ϕ_i's is a waypoint.
- Let $h_j : \mathbb{R}^n \to \mathbb{R}$, $j = 1, \ldots, N_c$, be the state constraints, i.e. we want $x(t) \in \{y \in \mathbb{R}^n \mid h_j(y) \leq 0,\ \forall j\}$, $\forall t \in [0, \mu_f]$.
- Compactly, describe all of the constraints via a constraint function:

$$\psi(\xi) = \max_{\substack{j=1,\ldots,N_c \\ t \in [0,\mu_f]}} h_j\big(x^{(\xi)}(t)\big)$$

Assumptions

1. The functions L and f_q are Lipschitz and differentiable in x and u for all q $\in \mathcal{Q}$. In addition, the derivatives of these functions with respect to x and u are also Lipschitz.
2. The functions ϕ_i, ϕ, and h_j are Lipschitz and differentiable in x for all i $\in \{1, \ldots, W\}$ and j $\in \mathcal{J}$. In addition, the derivatives of these functions with respect to x are also Lipschitz.

Optimization Problem

Multiple Waypoint Switched Hybrid Optimal Control Problem

$$\begin{aligned} &\min_{\xi \in \mathcal{X}} \; J(\xi) \\ \text{s.t.} \quad &\psi(\xi) \leq 0 \end{aligned}$$

Required Properties of Our Algorithm

Initializing an algorithm at a feasible point is non-trivial, therefore having an algorithm capable of coping with infeasibility is critical.

1. *Phase I/Phase II*: If the initialization is infeasible, find a feasible point and then minimize the cost.
2. *Stay Feasible*: Once a feasible point is found make sure to stay feasible.

Optimal Control Algorithm

- Numerical methods for "classical" optimal control, are able to simultaneously optimize over the input and initial condition.
- Given a fixed mode sequence, σ and a fixed waypoint sequence w, our problem is transformed into a "classical" optimal control problem via the time-free transformation, wherein:

  optimization over time spent in each mode is transformed into optimization over the initial condition with the addition of states with null flows.

Algorithm for the Discrete Input

- We cannot define gradients in our optimization space since there is no notion of locality in the mode sequence space.
- Define a "variation" to the discrete control input by inserting a new mode in the mode sequence for a short interval of time and computing the change in the cost and constraints due to this variation.

Local Minima for the Switched System Problem

Whenever the first-order approximation of the cost (constraint) is constant with respect to this variation when initialized at a feasible (infeasible) point we are at an extrema.

Algorithm for the Discrete Input

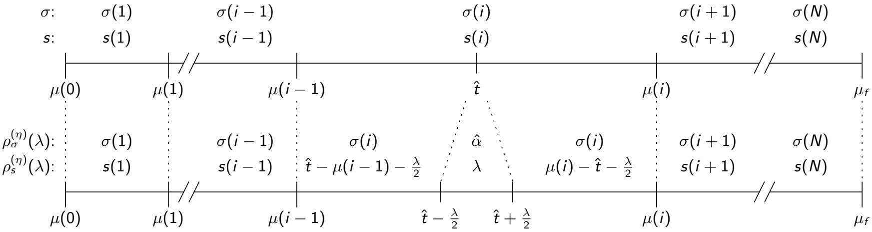
- We cannot define gradients in our optimization space since there is no notion of locality in the mode sequence space.
- Define a "variation" to the discrete control input by inserting a new mode in the mode sequence for a short interval of time and computing the change in the cost and constraints due to this variation.

Local Minima for the Switched System Problem

Whenever the first-order approximation of the cost (constraint) is constant with respect to this variation when initialized at a feasible (infeasible) point we are at an extrema.

Variation ρ

- Let $\mathcal{H} = \mathcal{Q} \times [0, \mu_f] \times U$
- Given a 3–tuple $\eta = (\hat{\alpha}, \hat{t}, \hat{u}) \in \mathcal{H}$ and $\xi \in \mathcal{X}$, define a variation $\rho^{(\eta)} : [0, \infty) \to \mathcal{X}$ that inserts a mode $\hat{\alpha}$, at time $\hat{t}$, with continuous-valued input $\hat{u}$, for a period of time of length λ which is equal to its argument.



First-Order Approximation of Cost and Constraints

The variation of J and ψ:

$$\frac{dJ\big(\rho^{(\eta)}(\lambda)\big)}{d\lambda}\Big|_{\lambda=0} = \lim_{\lambda\downarrow 0} \frac{J\big(\rho^{(\eta)}(\lambda)\big) - J(\xi)}{\lambda}$$

$$\frac{d\psi\big(\rho^{(\eta)}(\lambda)\big)}{d\lambda}\Big|_{\lambda=0} = \lim_{\lambda\downarrow 0} \frac{\psi\big(\rho^{(\eta)}(\lambda)\big) - \psi(\xi)}{\lambda}$$

1. If $\frac{dJ(\rho^{(\eta)}(\lambda))}{d\lambda}\Big|_{\lambda=0} < 0$, then the new mode sequence "locally" reduces the cost.
2. If $\psi(\xi) = 0$ and $\frac{d\psi(\rho^{(\eta)}(\lambda))}{d\lambda}\Big|_{\lambda=0} < 0$, then the new mode sequence remains feasible.

Bi-Level Optimization Scheme

Stage 1

Fix a mode sequence, σ, and waypoint sequence, w, find either a locally optimal transition time sequence, s, and continuous control, u.

Stage 2

If for all variations, ρ, that insert a new mode, the first-order approximation is constant, then the algorithm is at a local minima, so terminate.

Otherwise choose the variation that produces the steepest descent, and repeat Stage 1 using the mode sequence created by the variation.

Stage 1: Methodology

- Given a discrete mode sequence, σ and a waypoint sequence, w, we need to find a transition time sequence, s, and continuous-valued input, u, that minimize the cost, J.
  1. Transform optimization over the transition time sequence, s, and the continuous-valued input, u, into an optimization over initial conditions and inputs on a family of $\#\sigma$ continuous systems.
  2. Use any numerical method for optimal control like *SNOPT* or *NPSOL*.
- Let $\hat{a} : \mathcal{S} \times \mathcal{U} \rightarrow \mathcal{S} \times \mathcal{U}$ denote the algorithm that implements Stage 1.

Stage 1: Sketch of Time-Free Transformation

- Let z be the solution of:

$$\frac{dz(t)}{dt} = f\big(z(t), u(t)\big), \quad z(0) = z_0, \quad \forall t \in [t_1, t_2]$$

- Define $\tau(t) = \frac{t-t_1}{t_2-t_1}$, $\tilde{u}\big(\tau(t)\big) = u(t)$ for each $t \in [0, \mu_f]$. Let $(\tilde{z}, r)$ be the solution of:

$$\frac{d\tilde{z}(\tau)}{d\tau} = r(\tau) f\big(\tilde{z}(\tau), \tilde{u}(\tau)\big), \quad \tilde{z}(0) = z_0, \quad \forall \tau \in [0, 1]$$
$$\frac{dr(\tau)}{d\tau} = 0, \quad r(0) = t_2 - t_1, \quad \forall \tau \in [0, 1]$$

then $\tilde{z}\big(\tau(t)\big) = z(t)$ for each t. In the $\tilde{z}$ formulation, the time interval is an initial condition.

Stage 1: Formulation

Given $x_0 \in \mathbb{R}^n$, $\sigma \in \Sigma$, and $w \in \mathbb{N}^W$, Stage 1 solves:

Time-Free Transformation

$$\min_{(s_k)_1^{\#\sigma} \subset \mathbb{R}_+, \; (\tilde{u}_k)_1^{\#\sigma} \subset \mathcal{U}} \sum_{k=1}^{\#\sigma} \gamma_k(1) + \sum_{i=1}^{W} \phi_i\big(z_{w(i)}(1)\big) + \phi\big(z_{\#\sigma}(1)\big)$$

subject to:

$$\begin{pmatrix} \dot{z}_k(t) \\ \dot{r}_k(t) \\ \dot{\gamma}_k(t) \end{pmatrix} = \begin{pmatrix} r_k(t) f_{\sigma_k}\big(z_k(t), \tilde{u}_k(t)\big) \\ 0 \\ r_k(t) L\big(z_k(t), \tilde{u}_k(t)\big) \end{pmatrix}, \quad \begin{pmatrix} z_k(0) \\ r_k(0) \\ \gamma(0) \end{pmatrix} = \begin{pmatrix} z_{k-1}(1) \\ s_k \\ 0 \end{pmatrix}$$

$$h_j\big(z_k(t)\big) \leq 0, \quad \forall j = 1, \ldots, N_c, \quad \forall k = 1, \ldots, \#\sigma, \quad \forall t \in [0,1]$$

where $z_0(1) = x_0$.

Stage 2: Methodology

Given $\xi \in \mathcal{X}$, employ the variation, ρ, to find a new $\hat{\xi} \in \mathcal{X}$ that either reduces the cost if the initialization is feasible or the constraint if the initialization is infeasible:

1. Find an insertion $\hat{\eta}$ that decreases the cost
2. Find a suitable insertion length denoted as $\hat{\lambda}$ and define a new point $\hat{\xi} = \rho^{(\hat{\eta})}(\hat{\lambda})$.

Intuitively, $\hat{\eta}$ is a "descent direction" in the space $\mathcal{X}$ and $\hat{\lambda}$ is the "step size".

Stage 2: Optimality Function

- Fix $\gamma > 0$ and let $\theta : \mathcal{X} \to (-\infty, 0]$ be:

$$\theta(\xi) = \begin{cases} \min_{\eta \in \mathcal{H}} \ \max\left\{ \left.\frac{dJ(\rho^{(\eta)}(\lambda))}{d\lambda}\right|_{\lambda=0}, \psi(\xi) + \left.\frac{d\psi(\rho^{(\eta)}(\lambda))}{d\lambda}\right|_{\lambda=0} \right\} & \text{if } \psi(\xi) \leq 0 \\ \min_{\eta \in \mathcal{H}} \ \max\left\{ \left.\frac{dJ(\rho^{(\eta)}(\lambda))}{d\lambda}\right|_{\lambda=0} - \ \gamma \cdot \psi(\xi), \left.\frac{d\psi(\rho^{(\eta)}(\lambda))}{d\lambda}\right|_{\lambda=0} \right\} & \text{if } \psi(\xi) > 0 \end{cases}.$$

- If $\theta(\xi) < 0$ and
  1. $\psi(\xi) \leq 0$, then the variation produces a reduction in the cost, while remaining feasible.
  2. $\psi(\xi) > 0$, then the variation produces a reduction in the infeasibility.
- $\theta(\xi)$ is always less than or equal to zero since we can always perform an insertion that leaves the trajectory unaffected.
- θ is called an *optimality function* since θ's zeros encode points local minima of our switched system problem.

Stage 2: Step Size

- Fix $\alpha, \beta \in (0,1)$. Let $\hat{\eta}$ be the argument that minimizes θ and denote $\rho(\lambda) = \rho^{(\hat{\eta})}(\lambda)$. Define:

$$\hat{\lambda} = \begin{cases} \max_{k\in\mathbb{N}}\{\beta^k | \psi(\rho(\beta^k)) \leq 0, J(\rho(\beta^k)) - J(\xi) \leq \alpha\beta^k\theta(\xi)\} & \text{if } \psi(\xi) \leq 0 \\ \max_{k\in\mathbb{N}}\{\beta^k | \psi(\rho(\beta^k)) - \psi(\xi) \leq \alpha\beta^k\theta(\xi)\} & \text{if } \psi(\xi) > 0 \end{cases}.$$

Algorithm for Switched Optimal Control

Data: $\xi_0 = (\sigma_0, s_0, u_0, w_0) \in \mathcal{X}$, $\alpha, \beta, \in (0,1)$, $\gamma > 0$.

Step 0. Let $(s_1, u_1) = \hat{a}(s_0, u_0)$, $\sigma_1 = \sigma_0$, $w_1 = w_0$,
define $\xi_1 = (\sigma_1, s_1, u_1, w_1)$.

Step 1. Set $j = 1$.

Step 2. If $\theta(\xi_j) = 0$ then stop and return ξ_j.

Step 3. $\xi_{j+1} = a(\xi_j)$, where a is defined as follows:

1. Let $\hat{\eta} = (\hat{\alpha}, \hat{t}, \hat{u})$ be the argument that minimizes $\theta(\xi_j)$, and let $\rho^{(\hat{\eta})}(\hat{\lambda}) = (\tilde{\sigma}_j, \tilde{s}_j, \tilde{u}_j, \tilde{w}_j)$.
2. Given $\tilde{\sigma}_j$, let $(s_{j+1}, u_{j+1}) = \hat{a}(\tilde{s}_j, \tilde{u}_j)$.
3. Set $\sigma_{j+1} = \tilde{\sigma}_j$, $w_{j+1} = \tilde{w}_j$, and define
$\xi_{j+1} = a(\xi_j) = (\sigma_{j+1}, s_{j+1}, u_{j+1}, w_{j+1})$.

Step 5. Replace j by $j + 1$ and go to Step 2.

Algorithm Analysis

Bevel-Tip Flexible Needle

- Asymmetric needles that move along curved trajectories when a forward pushing force is applied [Cowan et al., 2004].
- Due to the stiffness of the needle, naturally thought of as a switched system.
- Optimal control has been considered using heuristics [Duindam et al., 2008] and RRTs [Xu et al., 2008].

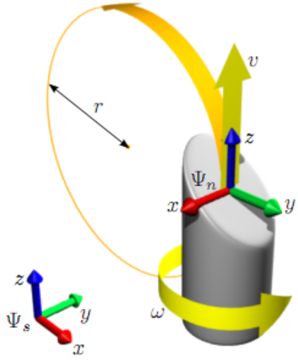


Figure: [Duindam et al., 2008]

Forward/Turn Needle Kinematics

Forward Mode: $q = F$

$$
\begin{aligned}
\dot{x}(t) &= u_1(t)\sin\left(\beta_p(t)\right)\\
\dot{y}(t) &= -u_1(t)\cos\left(\beta_p(t)\right)\sin\left(\beta_y(t)\right)\\
\dot{z}(t) &= u_1(t)\cos\left(\beta_y(t)\right)\cos\left(\beta_p(t)\right)\\
\dot{\beta}_y(t) &= \frac{u_1(t)}{r}\cos\left(\beta_r(t)\right)\sec\left(\beta_p(t)\right)\\
\dot{\beta}_p(t) &= \frac{u_1(t)}{r}\sin\left(\beta_r(t)\right)\\
\dot{\beta}_r(t) &= -\frac{u_1(t)}{r}\cos\left(\beta_r(t)\right)\tan\left(\beta_p(t)\right)
\end{aligned}
$$

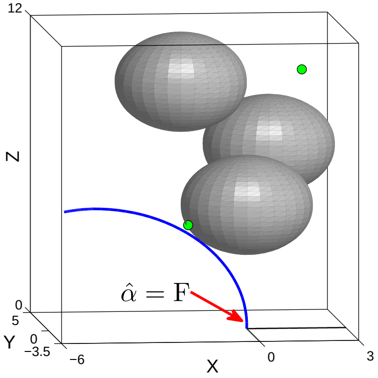
Turn Mode: $q = T$

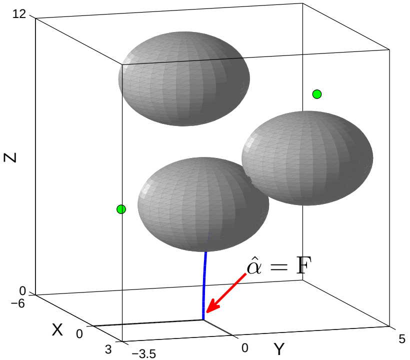
$$
\begin{aligned}
\dot{x}(t) &= 0\\
\dot{y}(t) &= 0\\
\dot{z}(t) &= 0\\
\dot{\beta}_y(t) &= 0\\
\dot{\beta}_p(t) &= 0\\
\dot{\beta}_r(t) &= u_2(t)
\end{aligned}
$$

Cost Function

$$
J(\xi) = \int_0^{\mu_f}\left(0.05\cdot u_1^2(t) + 0.005\cdot u_2^2(t) + 1\right)dt + 100\cdot\left\|\begin{bmatrix} x(\mu_{w_1})\\ y(\mu_{w_1})\\ z(\mu_{w_1})\end{bmatrix} - \hat{w}_1\right\|^2 + 30\cdot\left\|\begin{bmatrix} x(\mu_f)\\ y(\mu_f)\\ z(\mu_f)\end{bmatrix} - \hat{w}_f\right\|^2
$$

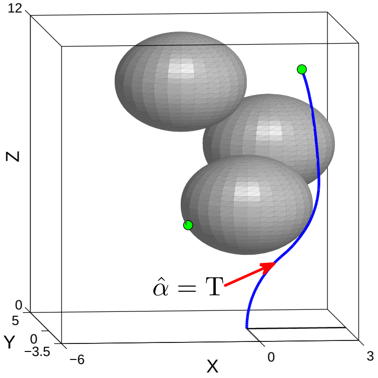
First Iteration

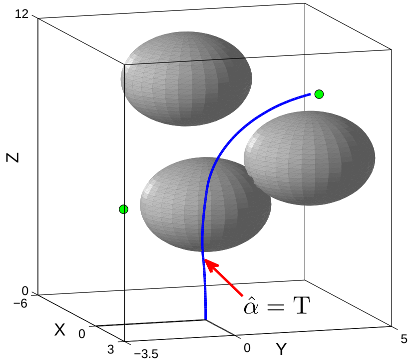




- $\sigma = (T, F, T, F)$
- $J = 1564.5$

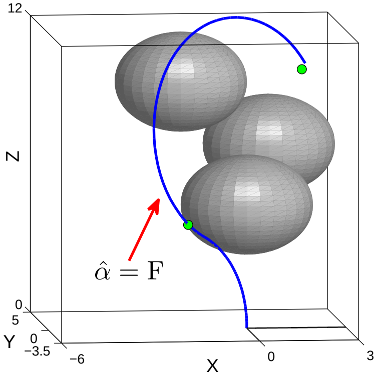
Second Iteration

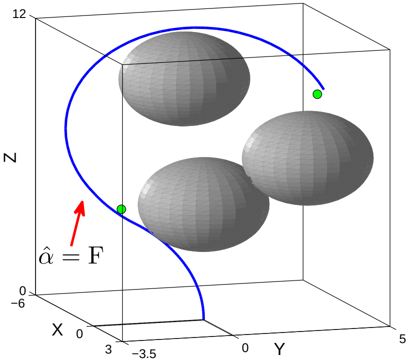




- $\sigma = (T, F, T, F, T, F)$
- $J = 1103.5$

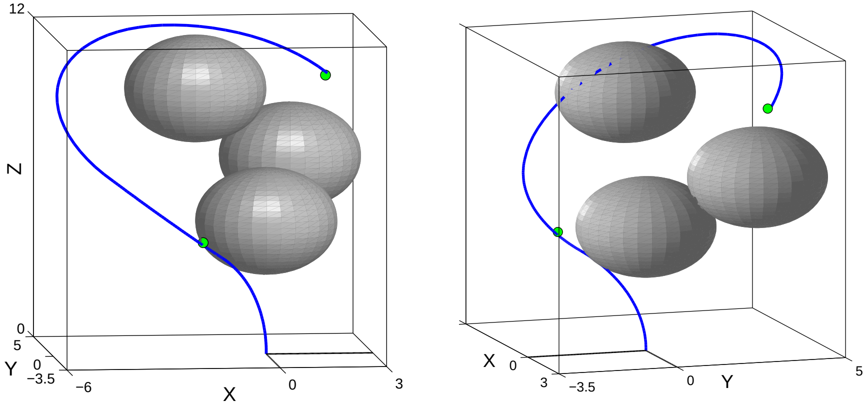
Third Iteration





- $\sigma = (T, F, T, T, F, T, F)$
- $J = 68.532$

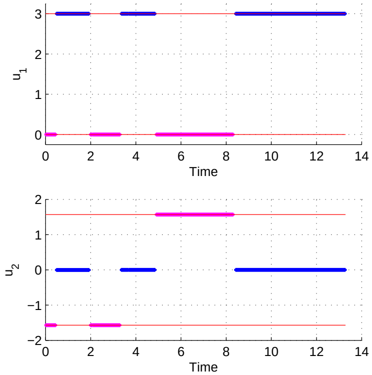
Fourth Iteration



- $\sigma = (T, F, T, T, F, F, T, F)$
- $J = 15.819$

Discussion

- AMD Opteron, 8 cores, 2.2 GHz, 16 GB RAM.
- Total time to solve Stage 1: 156.49[s]
- Total time to solve Stage 2: 200.86[s]



i	1	2	3	4	5	6	7	8
σ	T	F	T	T	F	F	T	F
s	0.47[s]	1.51[s]	0.10[s]	1.26[s]	0.35[s]	1.20[s]	3.52[s]	5.00[s]

Conclusion and Future Work

Acknowledgements

Questions?

Sufficient Descent

Definition (Sufficient Descent)

An algorithm $a : \mathcal{X} \to \mathcal{X}$ is said to have the *sufficient descent* property with respect to an optimality function, θ, if for all ξ in $\mathcal{X}$ with $\theta(\xi) < 0$, there exists a $\delta_\xi > 0$ and a neighborhood of ξ, $U_\xi \subset \mathcal{X}$, such that given a cost function J and feasible set $\mathcal{F}$ the following inequality is satisfied:

$$J\big(a(\xi')\big) - J(\xi') \leq -\delta_\xi, \quad \forall \xi' \in U_\xi \cap \mathcal{F}.$$

Theorem (Theorem 1, Polak 1997)

If the cost and constraint functions are continuous, and an algorithm satisfies the sufficient descent property with respect to an optimality function, then the sequence of points generated by the algorithm converges to the zeros of the optimality function.

Sufficient Descent

Definition (Sufficient Descent)

An algorithm $a : \mathcal{X} \to \mathcal{X}$ is said to have the *sufficient descent* property with respect to an optimality function, θ, if for all ξ in $\mathcal{X}$ with $\theta(\xi) < 0$, there exists a $\delta_\xi > 0$ and a neighborhood of ξ, $U_\xi \subset \mathcal{X}$, such that given a cost function J and feasible set $\mathcal{F}$ the following inequality is satisfied:

$$J\big(a(\xi')\big) - J(\xi') \leq -\delta_\xi, \quad \forall \xi' \in U_\xi \cap \mathcal{F}.$$

Theorem (Theorem 1, Polak 1997)

If the cost and constraint functions are continuous, and an algorithm satisfies the sufficient descent property with respect to an optimality function, then the sequence of points generated by the algorithm converges to the zeros of the optimality function.

Outline of Convergence

- Show that the standard cost is continuous (Proposition 1).
- Show that the constraint function is continuous (Proposition 2).
- Compute expressions for the variation of the cost and constraint function (Propositions 5 and 6).
- Prove that the algorithm has the sufficient descent property and has the Phase I/Phase II property (Theorem 1).