Modeling & Control of a Longboard-Riding Robot

Matt Keeter
mkeeter@mit.edu

6.832 Final Project
Spring 2012
Inspiration
State Variables
Control Inputs
System Summary

- \( q = [x, y, \alpha]' \)

Gliding and pushing modes

- \( y_{toe} \leq 0 \to \text{pushing} \)
- \( y_{toe} > 0 \to \text{gliding} \)
System Summary

- \( q = [x, y, \alpha]' \)
- \( u = [F, \tau]' \)
System Summary

- \( q = [x, y, \alpha]' \)
- \( u = [F, \tau]' \)
- Gliding and pushing modes
  - \( y_{toe} \leq 0 \rightarrow \) pushing
  - \( y_{toe} > 0 \rightarrow \) gliding
Wrote tools to automatically solve dynamics
• Wrote tools to automatically solve dynamics
• Written in Python using Sage
• Wrote tools to automatically solve dynamics
• Written in Python using Sage
• Solves for:
  • Second derivatives $\ddot{q}$
  • Co-located PFL equations
Wrote tools to automatically solve dynamics
Written in Python using Sage
Solves for:
  - Second derivatives $\ddot{q}$
  - Co-located PFL equations
Exports as MATLAB scripts
Feedback Linearization

- Solve dynamics equations for $\ddot{x}$, $F$, $\tau$

- Plug in $\ddot{y}$, $\ddot{\alpha}$ for feedback linearization

- $\ddot{y} = \ddot{y}_{\text{desired}}$

- $\ddot{\alpha} = \ddot{\alpha}_{\text{desired}}$
- Solve dynamics equations for $\ddot{x}$, $F$, $\tau$
- Solutions are in terms of $q$, $\dot{q}$, $\ddot{y}$, $\dddot{\alpha}$
  (as well as constants)
• Solve dynamics equations for $\ddot{x}$, $F$, $\tau$
• Solutions are in terms of $q$, $\dot{q}$, $\ddot{y}$, $\dddot{\alpha}$ (as well as constants)
• Plug in $\ddot{y}$, $\dddot{\alpha}$ for feedback linearization
  • $\ddot{y} = \ddot{y}_{\text{desired}}$
  • $\dddot{\alpha} = \dddot{\alpha}_{\text{desired}}$
Feedback Linearization

- Solve dynamics equations for $\ddot{x}$, $F$, $\tau$
- Solutions are in terms of $q$, $\dot{q}$, $\ddot{y}$, $\ddot{\alpha}$ (as well as constants)
- Plug in $\ddot{y}$, $\ddot{\alpha}$ for feedback linearization
  - $\ddot{y} = \ddot{y}_{\text{desired}}$
  - $\ddot{\alpha} = \ddot{\alpha}_{\text{desired}}$
- Double integrator control
Controller Strategy

High-level strategy breaks motion into stages

- Push off
- Stand up
- Lower self
- Swing leg forward
- Switch to pushing state
- Switch to gliding state

Push off
Stand up
Lower self
Swing leg forward
Controller Strategy

- Push off
- Stand up
- Lower self
- Switch to pushing state
- Swing leg forward
- Switch to gliding state
Controller Strategy

- Push off
- Stand up
- Lower self
- Switch to pushing state
- Swing leg forward
- Switch to gliding state
Controller Strategy

Push off
Stand up
Lower self
Switch to pushing state
Swing leg forward
Switch to gliding state
Controller Strategy

1. Stand up
2. Push off
3. Lower self
4. Swing leg forward
5. Switch to gliding state
6. Switch to pushing state

Diagram:
- Stand up
- Push off
- Lower self
- Swing leg forward
- Switch to gliding state
- Switch to pushing state
Controller Strategy

Stand up -> Lower self

Switch to pushing state

Push off

Swing leg forward

Switch to gliding state
Controller Parameters

Controller is parameterized by three terms:

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha_{\text{hit}}$</td>
<td>Angle of collision</td>
</tr>
<tr>
<td>$\alpha_{\text{stand}}$</td>
<td>Angle ending push</td>
</tr>
<tr>
<td>$\ddot{\alpha}_{\text{swing}}$</td>
<td>Swinging acceleration</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>$\alpha_{\text{hit}}$</td>
<td>Angle of collision</td>
</tr>
<tr>
<td>$\alpha_{\text{stand}}$</td>
<td>Angle ending push</td>
</tr>
<tr>
<td>$\ddot{\alpha}_{\text{swing}}$</td>
<td>Swinging acceleration</td>
</tr>
</tbody>
</table>

![Diagram](image-url)
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha_{hit}$</td>
<td>Angle of collision</td>
</tr>
<tr>
<td>$\alpha_{stand}$</td>
<td>Angle ending push</td>
</tr>
<tr>
<td>$\ddot{\alpha}_{swing}$</td>
<td>Swinging acceleration</td>
</tr>
</tbody>
</table>

**Diagram:**

- A pendulum system with points indicating collision and swinging phases.
- The pendulum is shown in two positions, one for collision and one for swinging.
- Arrows pointing to the points indicate the direction of motion.
Controller Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha_{hit}$</td>
<td>Angle of collision</td>
</tr>
<tr>
<td>$\alpha_{stand}$</td>
<td>Angle ending push</td>
</tr>
<tr>
<td>$\ddot{\alpha}_{swing}$</td>
<td>Swinging acceleration</td>
</tr>
</tbody>
</table>

Diagram:

- A diagram showing a pendulum system with angles and vectors indicating the parameters described.
Stochastic Gradient Descent

Optimized for distance travelled in fixed time
Stochastic Gradient Descent

Optimized for distance travelled in fixed time
Optimized Demo

Figure 1

File  Edit  View  Insert  Tools  Desktop  Window  Help

[Graph with data points and lines]
20% improvement!
• Developed simplified system model
• Wrote dynamics-solving tools
• Designed high-level controller behavior
• Used gradient descent to optimize parameters