Decision Making
in Robots and Autonomous Agents

Model-based Control and Task Encoding

Subramanian Ramamoorthy
School of Informatics

1 February 2019

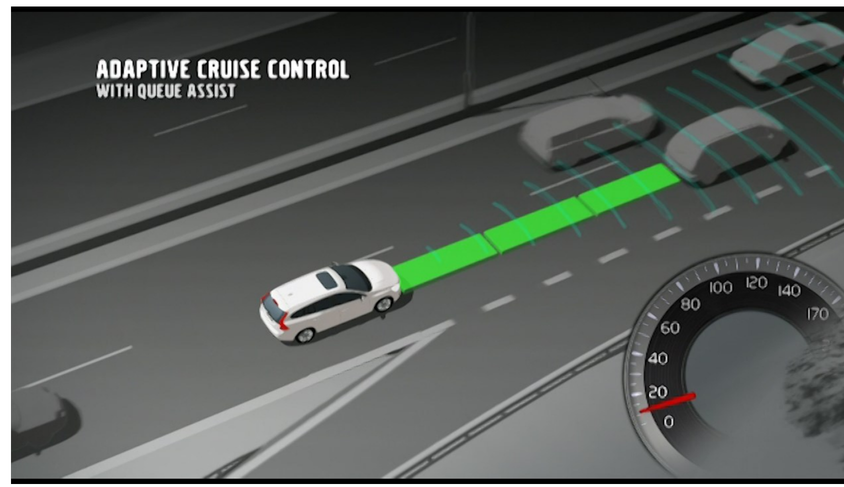
Objectives of this Lecture

- Give a selective recap of key ideas from control theory, as a very first approach to the “synthesis of robot motion”
  - If you have studied control theory before, you should recognize the concepts although the narrative may still be new
  - If you have not studied control before, this might give you useful background that will help contextualize later concepts

- After the first section surveying a few key concepts, we will spend the second half of the lecture on case studies – showing a few worked examples of robot control problems to illustrate design principles

Controller Synthesis as a *Design* Problem

- Consider a practical problem: cruise control (keep your distance with respect to car in front)
- How would you specify what the task entails?
  - Discuss
- How would you synthesize a control strategy that satisfies these specifications?
  - Role of model based design?



[Source: Volvo V60]

An Example of a Design *Specification*

- Consider how to say that the *following* car f should be behind another *leading* car l
- With v representing velocity and b, B representing braking power ($B > b$),

$$(x_f \leq x_l) \wedge (x \neq l) \rightarrow \left(x_f < x_l \wedge x_f + \frac{v_f^2}{2b} < x_l + \frac{v_l^2}{2B} \wedge v_f \geq 0 \wedge v_l \geq 0 \right)$$

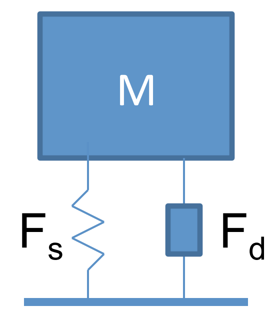
- Such a safe distance formula is an invariant requirement
- So is the fact that physical laws will hold true
- Two questions:
  1. What is the language for such specs in robotics, in general?
  2. What techniques exist to check if spec holds?

Linear Time Invariant (LTI) Systems

- Consider the simple spring-mass-damper system:
- The force applied by the spring is $F_s = -kz(t)$
- Correspondingly, for the damper: $F_d = \gamma \dot{z}(t)$
- The combined equation of motion of the mass becomes:

$$x\ddot{z}(t) = -\gamma \dot{z}(t) - kz(t)$$

- One could also express this in state space form:



$$x(t) = [x_1(t), x_2(t)]' = [z(t), \dot{z}(t)]'$$

$$\dot{x}(t) = \begin{pmatrix} \dot{z}(t) \\ \ddot{z}(t) \end{pmatrix} = \begin{pmatrix} x_2(t) \\ -\frac{1}{m}(\gamma x_2(t)) + kx_1(t) \end{pmatrix}$$

Linear ODE ←
$$\dot{x}(t) = \begin{pmatrix} 0 & 1 \\ -\frac{k}{m} & \frac{\gamma}{m} \end{pmatrix} x(t) = Ax(t)$$

Linear ODE: Solutions and Properties

- Consider the simplest possible linear ODE:

$$\dot{x} = kx, x \in \Re$$

- Given an initial condition, $\phi(0) = x_0$, what is $x(t)$?
- Solution is in terms of exponentials:

$$\phi(t) = e^{kt} x_0$$

- All linear dynamical systems will admit solutions in terms of such exponentials

Solution of a Linear ODE

The multivariate case:

$$x(t) = e^{A(t-t_0)} x_0$$

$$\begin{aligned} \boxed{e^{A(t-t_0)}} &= \sum_{i=0}^{\infty} \frac{A^i (t-t_0)^i}{i!} \\ &= I_{n \times n} + A(t-t_0) + \frac{A^2 (t-t_0)^2}{2!} + \dots \end{aligned}$$

This is state transition matrix $\phi(t)$:
In linear algebra, there are numerous ways to compute this…

Example

Determine the matrix exponential, and hence the state transition matrix, and the homogeneous response to the initial conditions $x_1(0) = 2$, $x_2(0) = 3$ of the system with state equations:

Unforced:
Which symbol
is affected?

$$\begin{aligned} \dot{x}_1 &= -2x_1 + u \\ \dot{x}_2 &= x_1 - x_2. \end{aligned}$$

The system matrix is

$$\mathbf{A} = \begin{bmatrix} -2 & 0 \\ 1 & -1 \end{bmatrix}.$$

Example, contd.

$$
\begin{aligned}
\mathbf{\Phi}(t) &= e^{\mathbf{A}t} \\
&= \left(\mathbf{I} + \mathbf{A}t + \frac{\mathbf{A}^2 t^2}{2!} + \frac{\mathbf{A}^3 t^3}{3!} + \ldots + \frac{\mathbf{A}^k t^k}{k!} + \ldots \right) \\
&= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} -2 & 0 \\ 1 & -1 \end{bmatrix} t + \begin{bmatrix} 4 & 0 \\ -3 & 1 \end{bmatrix} \frac{t^2}{2!} \\
&\qquad + \begin{bmatrix} -8 & 0 \\ 7 & -1 \end{bmatrix} \frac{t^3}{3!} + \ldots \\
&= \begin{bmatrix} 1 - 2t + \frac{4t^2}{2!} - \frac{8t^3}{3!} + \ldots & 0 \\ 0 + t - \frac{3t^2}{2!} + \frac{7t^3}{3!} + \ldots & 1 - t + \frac{t^2}{2!} - \frac{t^3}{3!} + \ldots \end{bmatrix}.
\end{aligned}
$$

$$
\mathbf{\Phi}(t) = \begin{bmatrix} e^{-2t} & 0 \\ e^{-t} - e^{-2t} & e^{-t} \end{bmatrix}
$$

Example, contd.

$$\mathbf{x}_h(t) = \mathbf{\Phi}(t)\mathbf{x}(0)$$

$$\begin{aligned} x_1(t) &= x_1(0)e^{-2t} \\ x_2(t) &= x_1(0)\left(e^{-t} - e^{-2t}\right) + x_2(0)e^{-t}. \end{aligned}$$

$$\begin{aligned} x_1(t) &= 2e^{-2t} \\ x_2(t) &= 2\left(e^{-t} - e^{-2t}\right) + 3e^{-t} \\ &= 5e^{-t} - 2e^{-2t}. \end{aligned}$$

Basic Notion: Stability

- Simple question:

  Given the system, $\dot{x}(t) = Ax(t)$

  where in phase space, $(x, \dot{x})$, will it come to rest *eventually*?

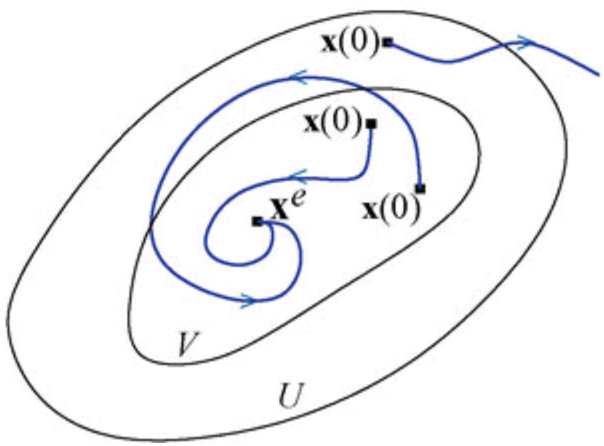
Any guesses?
Think about solution in previous slide…

- This point is called the equilibrium point
  - If initialized there, dynamics will not take it away
  - If perturbed, system will eventually return and stay there

Stability

An equilibrium position $x = 0$ is *stable* (in Lyapunov's sense) if given $\epsilon > 0$, $\exists \delta > 0$ (not dependent on t), s.t. $\forall x_0, |x_0| < \delta$ the solution satisfies $|\phi(t)| < \epsilon$, $\forall t > 0$

Asymptotic stability: Lyapunov stabile and $\lim_{t \to +\infty} \phi(t) = 0$



Some worked examples

- General solution of linear autonomous plane systems
- Classification of orbits of a linear system

Stability for an LTI System, $\dot{x}(t) = Ax(t)$

Unforced (homogeneous) response $x_i(t) = \sum_{j=1}^{n} m_{ij} e^{\lambda_j t}$

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & \dots & m_{1n} \\ m_{21} & m_{22} & \dots & m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & \dots & m_{nn} \end{bmatrix} \begin{bmatrix} e^{\lambda_1 t} \\ e^{\lambda_2 t} \\ \vdots \\ e^{\lambda_n t} \end{bmatrix}$$

$$\mathbf{x_h}(t) = \mathbf{M} \begin{bmatrix} e^{\lambda_1 t} \\ e^{\lambda_2 t} \\ \vdots \\ e^{\lambda_n t} \end{bmatrix}$$

Stability for an LTI System

If you differentiate the homogeneous response, $\frac{dx_i}{dt} = \sum_{j=1}^{n} \lambda_j m_{ij} e^{\lambda_j t}$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} \lambda_1 m_{11} & \lambda_2 m_{12} & \dots & \lambda_n m_{1n} \\ \lambda_1 m_{21} & \lambda_2 m_{22} & \dots & \lambda_2 m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_1 m_{n1} & \lambda_2 m_{n2} & \dots & \lambda_n m_{nn} \end{bmatrix} \begin{bmatrix} e^{\lambda_1 t} \\ e^{\lambda_2 t} \\ \vdots \\ e^{\lambda_n t} \end{bmatrix}.$$

The system being considered is $\dot{x}(t) = Ax(t)$, so:

$$\begin{bmatrix} \lambda_1 m_{11} & \lambda_2 m_{12} & \dots & \lambda_n m_{1n} \\ \lambda_1 m_{21} & \lambda_2 m_{22} & \dots & \lambda_2 m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_1 m_{n1} & \lambda_2 m_{n2} & \dots & \lambda_n m_{nn} \end{bmatrix} \begin{bmatrix} e^{\lambda_1 t} \\ e^{\lambda_2 t} \\ \vdots \\ e^{\lambda_n t} \end{bmatrix} = \mathbf{A} \begin{bmatrix} m_{11} & m_{12} & \dots & m_{1n} \\ m_{21} & m_{22} & \dots & m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & \dots & m_{nn} \end{bmatrix} \begin{bmatrix} e^{\lambda_1 t} \\ e^{\lambda_2 t} \\ \vdots \\ e^{\lambda_n t} \end{bmatrix}$$

LTI Stability, in algebraic equations

- The above equation leads to an eigenvalue problem:

$$\lambda_i \mathbf{m}_i = \mathbf{A}\mathbf{m}_i \qquad\qquad i = 1, 2, \ldots, n.$$

$$[\lambda_i \mathbf{I} - \mathbf{A}]\, \mathbf{m}_i = 0$$

- For this to have nontrivial solutions:

Characteristic eqn.

$$\Delta(\lambda_i) = \det[\lambda_i \mathbf{I} - \mathbf{A}] = 0.$$

$$\lambda^n + a_{n-1}\lambda^{n-1} + a_{n-2}\lambda^{n-2} + \ldots + a_1\lambda + a_0 = 0$$

$$(\lambda - \lambda_1)(\lambda - \lambda_2)\ldots(\lambda - \lambda_n) = 0.$$

Stability: LTI System, $\dot{x}(t) = Ax(t)$

Theorem. *Let* λ_i, $i \in \{1, 2, \ldots, n\}$ *denote the eigenvalues of* A. *Let* $re(\lambda_i)$ *denote the real part of* λ_i. *Then the following holds:*

1. $x_e = 0$ *is stable if and only if* $re(\lambda_i) \leq 0$, $\forall i$
2. $x_e = 0$ *is asymptotically stable if and only if* $re(\lambda_i) < 0$, $\forall i$
3. $x_e = 0$ *is unstable if and only if* $re(\lambda_i) > 0$, *for some* i

For the spring-mass-damper example, the eigenvalues are:

$$\frac{-\gamma \pm \sqrt{\gamma^2 - 4km}}{2m}$$

With positive damping, we get asymptotic stability

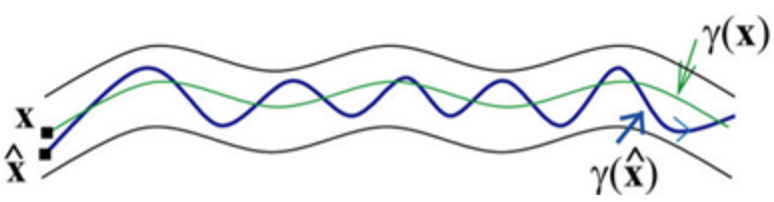
Worked examples on the board

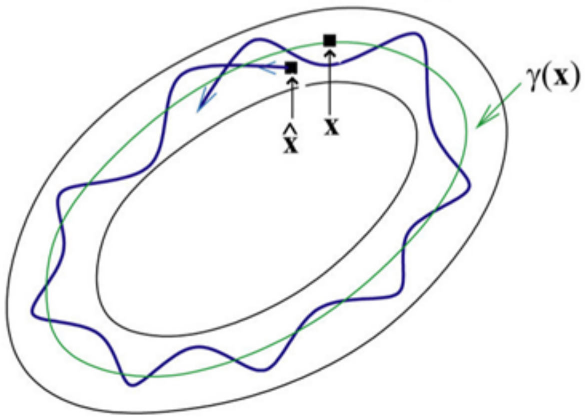
- Simple example of Lyapunov's method for establishing stability

Other Related Notions: Orbital Stability

- Stability doesn't only refer to being at rest at a point
  - could be defined in terms of staying in a subset, e.g., path

Definition. *An orbit $\gamma(x)$ is orbitally stable if for any $\epsilon > 0$, there is a neighbourhood V of x so that for all $\hat{x}$ in V, γx and $\gamma\hat{x}$ are ϵ-close.*
Loosely speaking, $|\gamma(x) - \gamma(\hat{x})| < \epsilon$ at all times.





Stability in the Real World

Some Aircrafts are by *Design* Statically Unstable! Why?

Jump to 2:07 for exciting bit!

[https://www.youtube.com/watch?v=2CUyoi634wc]

Nonlinear Systems

- What do solutions to nonlinear ODEs look like?
- Consider a simple system: $\ddot{x} + \omega^2 \sin x = 0$
- One could apply a transformation to write,

$$\frac{d}{dx}(\frac{1}{2}\dot{x}^2) + \omega^2 \sin x = 0$$

- Integrating this yields,

$$\frac{1}{2}\dot{x}^2 - \omega^2 \cos x = C$$

$$\dot{x} = \pm\sqrt{2}(C + \omega^2 \cos x)^{\frac{1}{2}}$$

Relationship between position and velocity (phase space)

Case Study: Global Control of the Inverted Pendulum

B.J. Kuipers, S. Ramamoorthy, Qualitative modeling and heterogeneous control of global system behavior. In C. J. Tomlin & M. R. Greenstreet (Eds.), Hybrid Systems: Computation and Control, LNCS 2289: 294-307, 2002.

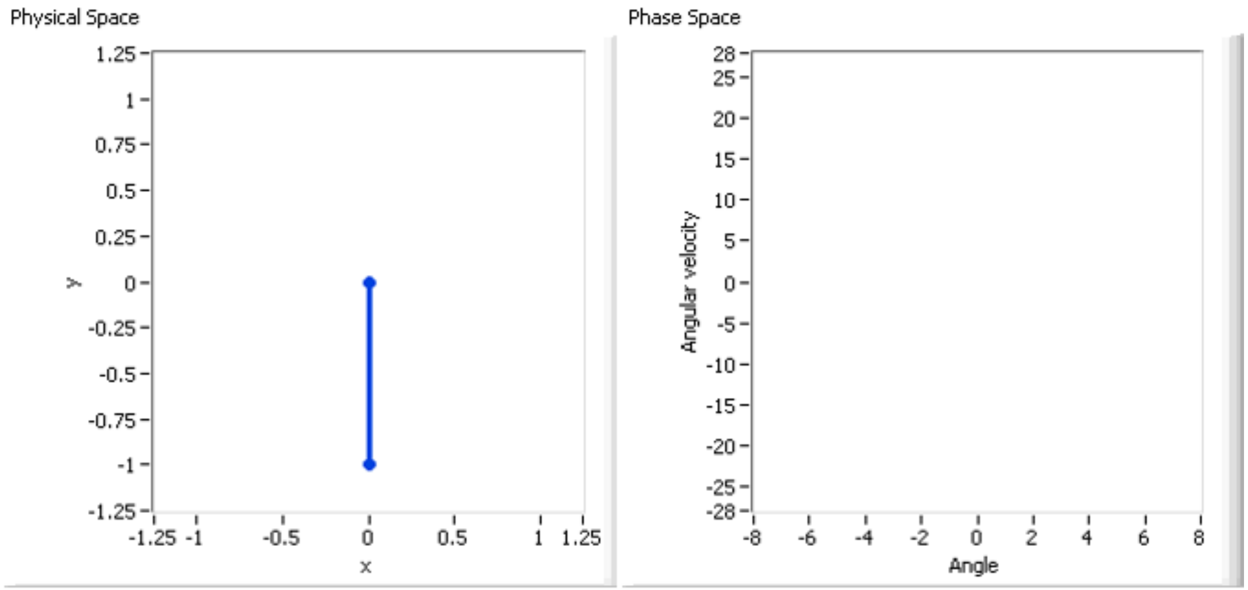
Objectives for Pendulum Case Study

- Show one concrete example – fully worked out – of a non-trivial nonlinear control problem
  - Just complex enough to be non-trivial
  - Amenable to analytical solution to illustrate ideas

- Demonstrate the process of addressing design requirements, and **proving** them to be satisfied by a controller

- In the process, exposing you to the idea of **structure** in the dynamics

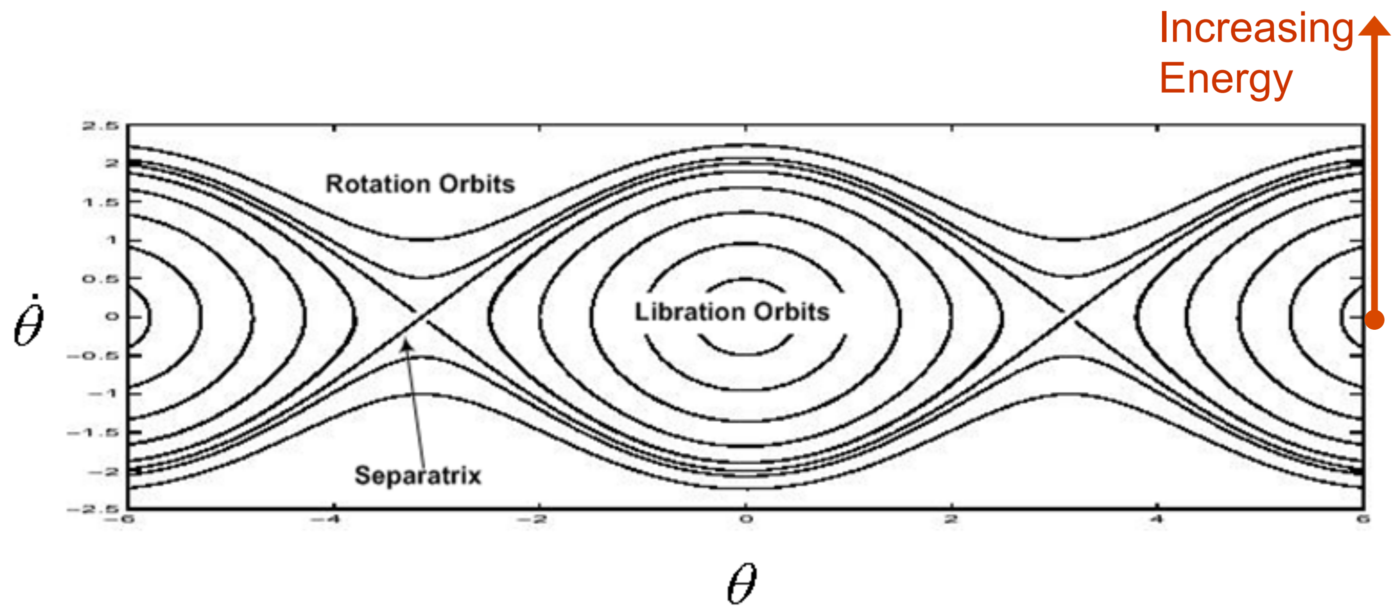
Pendulum Phase Space



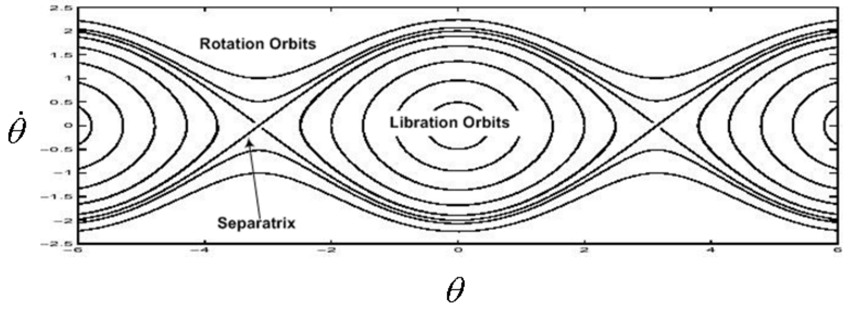
- Phase space is organized into families (open sets) of trajectories
- Trajectories may be parameterized by a single variable: energy

Design Strategy: Use Natural Dynamics

- Passively "ride" orbits ⇔ Energy Efficiency
- Parameterized families of trajectories ⇔ Flexibility
- Topology, structural stability ⇔ Robustness



Using Natural Dynamics for Motion Planning

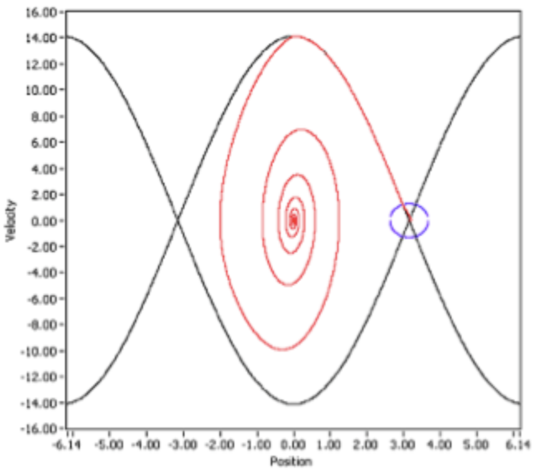


Generate trajectories, on-line,

- From the whole phase space
- To inverted position

Solution:

- Change *E* to move towards separatrix
- Two trajectory classes:
  pump (libration)/ spin (rotation)
- Ride the separatrix, once there



Let us now walk through this construction in some detail!

Remarks: Use of *Qualitative* Models

- A qualitative differential equation (QDE) expresses partial knowledge of a dynamical system.
  - One QDE describes a set of ODEs,
  - non-linear as well as linear systems.

- A QDE can express partial knowledge of a plant or a controller design.

- QSIM can predict all possible behaviors of all ODEs described by the given QDE.

Qualitative Design of a Heterogeneous Controller

- Design local models with the desired behavior.
- Identify qualitative constraints to guarantee the right transitions.
- Provide weak conditions sufficient to guarantee desired behavior.
  - Remaining degrees of freedom are available for optimization by any other criterion.
- Demonstrate with a global pendulum controller.
  - Local models: Pump, Balance, Spin.

Some Remarks about QSIM notation

- Each variable is a *reasonable* function.
  - Continuously differentiable, etc.
  - Range described by landmark values and intervals.
- Constraints link variables.
  - ADD, MULT, MINUS, D/DT
  - Monotonic functions: $y=f(x)$ for f in M_0^+
  - $[x]_0=\text{sign}(x)$

There are QSIM software tools to for:

- Computing semi-quantitative bounds
- Predicting all possible behaviors over time
- Temporal logic model-checking

The Monotonic Damped Spring

- The spring is defined by Hooke's Law:

$$F = ma = m\ddot{x} = -k_1 x$$

- Include damping friction

$$m\ddot{x} = -k_1 x - k_2 \dot{x}$$

- Rearrange and redefine constants

$$\ddot{x} + b\dot{x} + cx = 0$$

- Generalize to a Qualitative Differential Equation (QDE) with monotonic functions f and g

$$\ddot{x} + f(\dot{x}) + g(x) = 0$$

Lemma 1:
The Monotonic Damped Spring

Let a system be described by

$$\ddot{x} + f(\dot{x}) + g(x) = 0$$

where

$$f \in M_0^+ \text{ and } [g(x)]_0 = [x]_0$$

Then it is asymptotically stable at (0,0), with a Lyapunov function:

$$V(x, \dot{x}) = \tfrac{1}{2}\dot{x}^2 + \int_0^x g(x)\,dx$$

What does knowing such a V enable?

- Proof in the paper (link on course web page)

Lemma 2: The Spring with Anti-Damping

Suppose a system is described by

$$\ddot{x} - f(\dot{x}) + g(x) = 0$$

where

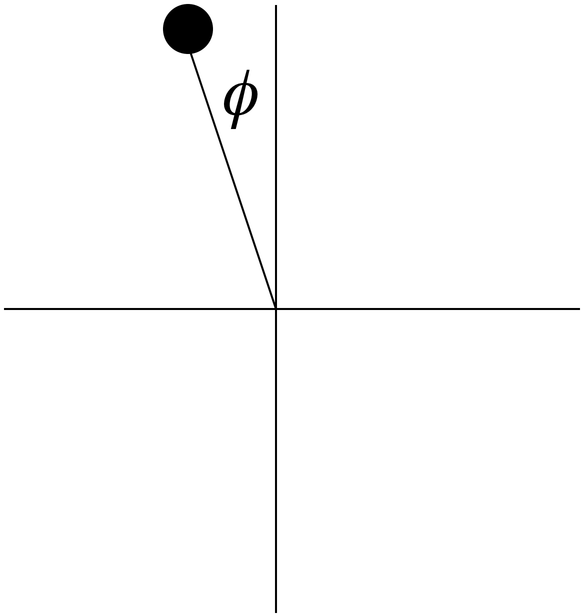
$$f \in M_0^+ \text{ and } [g(x)]_0 = [x]_0$$

Then the system has an unstable fixed-point at (0,0), and no limit cycle (i.e., stable periodic orbit).

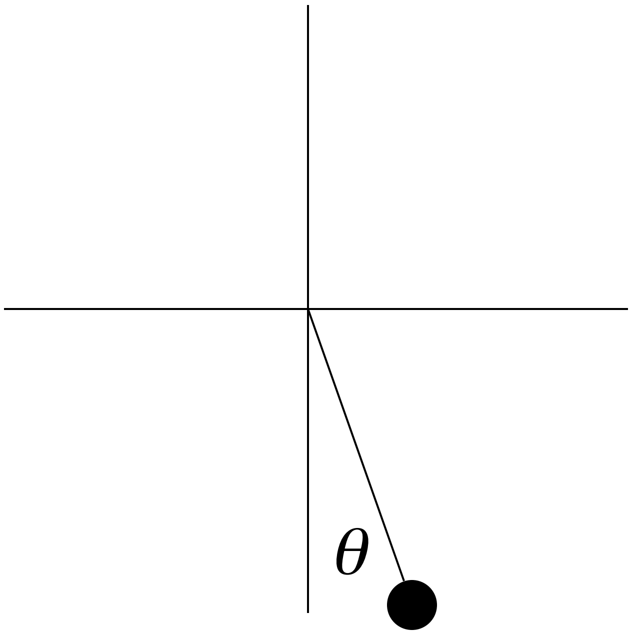
- Proof in the paper.

Pendulum Models: Equations of Motion

- Near the top.
- Near the bottom.



$$\ddot{\phi} + f(\dot{\phi}) - k\sin\phi = 0$$



$$\ddot{\theta} + f(\dot{\theta}) + k\sin\theta = 0$$

Balance the Pendulum

- Design the control input *u* to make the pendulum into a damped spring.

$$\ddot{\phi} + f(\dot{\phi}) - k\sin\phi + u(\phi,\dot{\phi}) = 0$$

- Define the **Balance** controller:

$$u(\phi,\dot{\phi}) = g(\phi)$$

such that

$$[g(\phi) - k\sin\phi]_0 = [\phi]_0$$

- Lemma 1 shows that it converges to (0,0).

$$\ddot{\phi} + f(\dot{\phi}) + g(\phi) - k\sin\phi = 0$$

The Balance Region: *Underactuation*

- If the control action has upper bound u_{max} then gravity defines the limiting angle:

$$u_{\max} = k \sin \phi_{\max}$$

- Energy defines maximum velocity at top:

$$\tfrac{1}{2}\dot{\phi}_{\max}^2 = \int_0^{\phi_{\max}} g(\phi) - k \sin \phi \, d\phi$$

- Define the **Balance** region:

$$\frac{\phi^2}{\phi_{\max}^2} + \frac{\dot{\phi}^2}{\dot{\phi}_{\max}^2} \leq 1$$

Pump the Hanging Pendulum

- Define the control action *u* to make the pendulum into a spring with negative damping.
- Define the **Pump** controller

$$u(\theta,\dot{\theta}) = -h(\dot{\theta})$$

such that

$$h - f \in M_0^+$$

gives

$$\ddot{\theta} - (h - f)(\dot{\theta}) + k\sin\theta = 0$$

- Lemma 2 proves it pumps without a limit cycle.

Slow the Spinning Pendulum

If the pendulum is spinning rapidly, define the **Spin** control law to augment natural friction:

$$u(\theta, \dot{\theta}) = f_2(\dot{\theta})$$

such that $f_2 \in M_0^+$

The Pump-Spin Boundary

- Prevent a limit-cycle behavior that cycles between **Pump** and **Spin** regions, overshooting **Balance**.
- Define the Pump-Spin boundary to be the *separatrix* of the undamped pendulum.
- **Pump** and **Spin** create what is known as a *sliding mode* controller
  - Special type of switching based control strategy
- The separatrix leads straight to the heart of **Balance.**

The Separatrix as Boundary

- A separatrix is a trajectory that begins and ends at the unstable saddle point of the undamped, uncontrolled pendulum:

$$\ddot{\theta} + k\sin\theta = 0$$

- Points on the separatrix have the same energy as the balanced pendulum:

$$KE + PE = \tfrac{1}{2}\dot{\theta}^2 + \int_0^{\theta} k\sin\theta\, d\theta = 2k$$

- Simplify to define the separatrix:

$$s(\theta,\dot{\theta}) = \tfrac{1}{2}\dot{\theta}^2 - k(1+\cos\theta) = 0$$

The Sliding Mode Controller

- Differentiate to see how *s* changes with time:

$$\dot{s} = -\dot{\theta} f(\dot{\theta}) - \dot{\theta} u(\theta, \dot{\theta})$$

- In the **Pump** region:

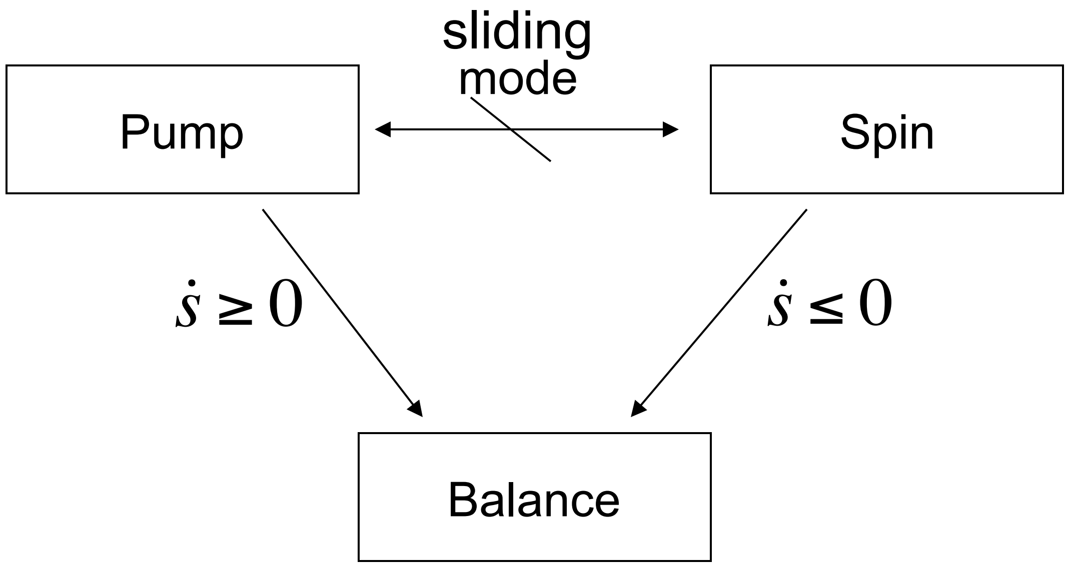
$$s < 0 \quad \text{and} \quad \dot{s} = \dot{\theta}(h - f)(\dot{\theta}) \geq 0$$

- In the **Spin** region:

$$s > 0 \quad \text{and} \quad \dot{s} = -\dot{\theta}(f + f_2)(\dot{\theta}) \leq 0$$

- Therefore, both regions approach $s = 0$

The Global Pendulum Controller



The Global Controller

The control law:

if **Balance**

$$u(\phi,\dot{\phi}) = g(\phi)$$

else if **Pump**

$$u(\theta,\dot{\theta}) = -h(\dot{\theta})$$

else **Spin**

$$u(\theta,\dot{\theta}) = f_2(\dot{\theta})$$

Constraints:

$$[g(\phi) - k\sin\phi]_0 = [\phi]_0$$

$$h - f \in M_0^+$$

$$f_2 \in M_0^+$$

Pendulum Controller Example

System:

$$\ddot{\theta} + c\dot{\theta} + k\sin\theta + u(\theta,\dot{\theta}) = 0$$

$$c = 0.01, k = 10, u_{\max} = 4$$

Balance:

$$c_{11} = 0.4, c_{12} = 0.3$$

$$u = (c_{11} + k)(\theta - \pi) + c_{12}\dot{\theta}$$

$$\phi_{\max} = 0.4, \dot{\phi}_{\max} = 0.3$$

Spin:

$$u = c_2\dot{\theta}$$

$$c_2 = 0.5$$

Pump:

$$u = -(c + c_3)\dot{\theta}$$

$$c_3 = 0.5$$

Pendulum Example, cont.

The switching strategy:

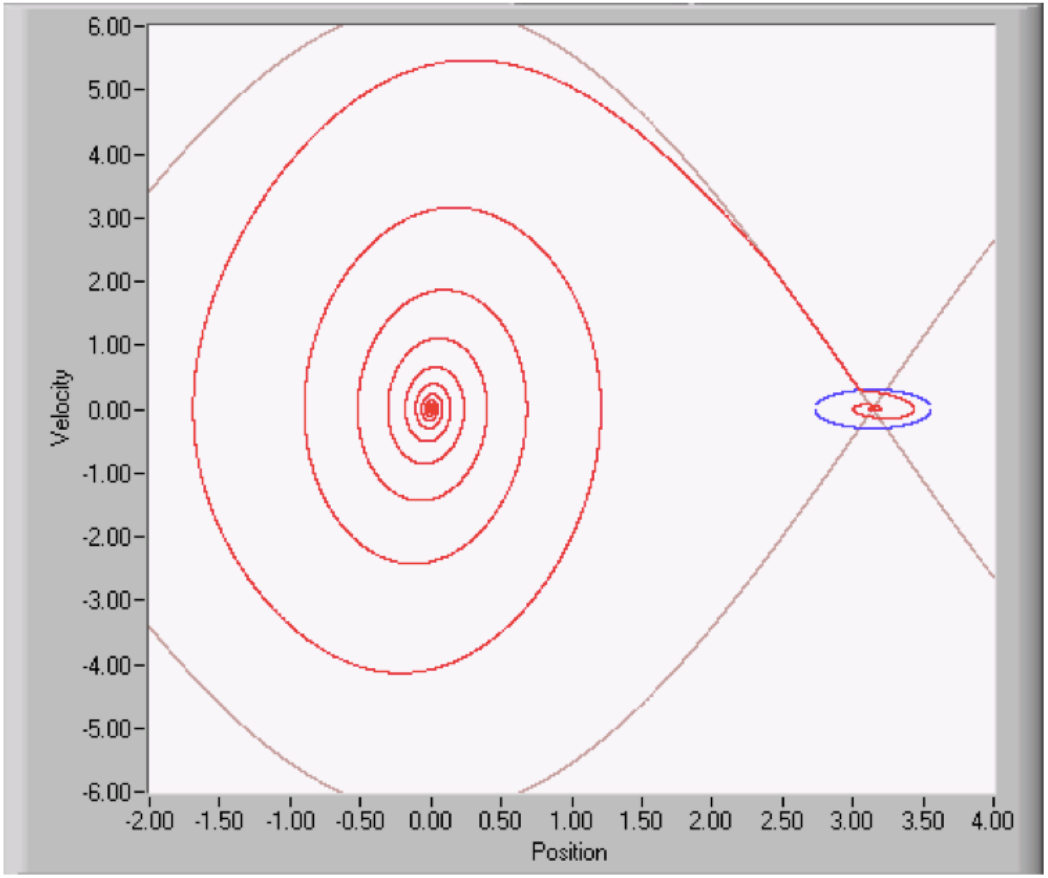
If $\alpha \leq 1$ then ***Balance***

else if $s < 0$ then ***Pump***

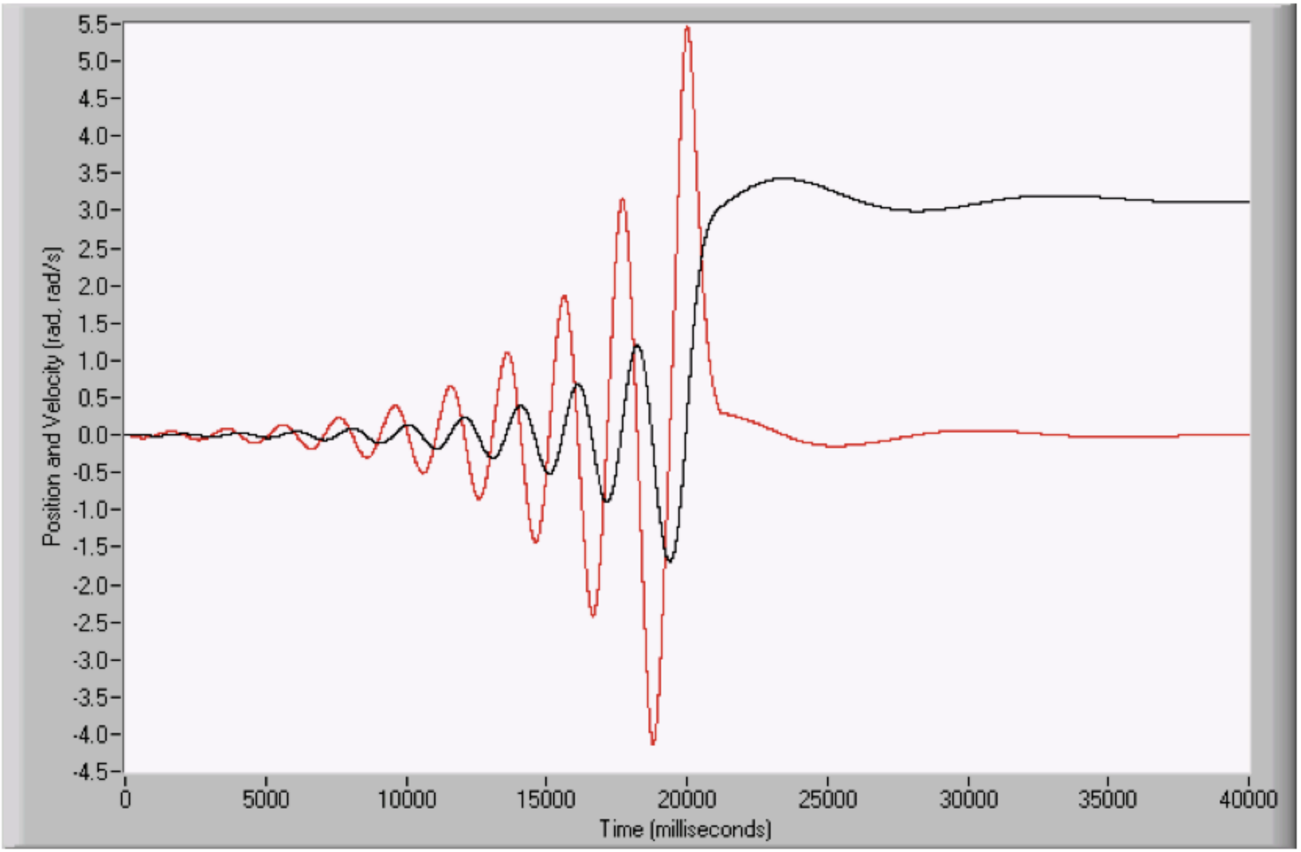
else ***Spin***

$$\alpha = \frac{\phi^2}{\phi_{\max}^2} + \frac{\dot{\phi}^2}{\dot{\phi}_{\max}^2} \qquad s = \frac{1}{2}\dot{\theta}^2 - k(1 + \cos\theta)$$

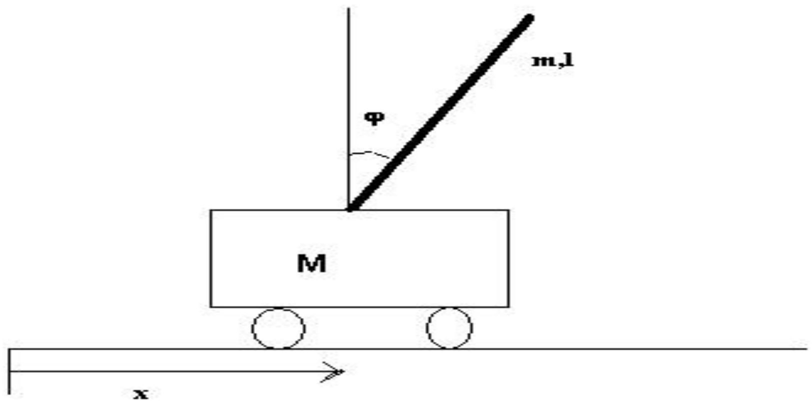
The Controlled Pendulum



The Controlled Pendulum



Now quite the full system, yet: The Cart-Pole System



$\ddot{x}$: Control Action

$\phi = \theta + \pi$

Cart Pole System:

$$\ddot{\phi} + f(\dot{\phi}) - k\sin\phi - \ddot{x}\cos\phi = 0$$

$$\ddot{\theta} + f(\dot{\theta}) + k\sin\theta + \ddot{x}\cos\theta = 0$$

Example:
Heterogeneous Cart-Pole Controller

The Cart-Pole System

$$\ddot{\theta} + f(\dot{\theta}) + k\sin\theta + \ddot{x}\cos\theta = 0$$

Compare to Pivot-Torque Pendulum System:

$$\ddot{\theta} + f(\dot{\theta}) + k\sin\theta + u(\theta,\dot{\theta}) = 0$$

Heterogeneous Cart-Pole Controller:

$$\ddot{x} = sat\left\{-\frac{f_1(\dot{x})}{\cos\theta} - \frac{g_1(x)}{\cos\theta} + \frac{u(\theta,\dot{\theta})}{\cos\theta}\right\}$$

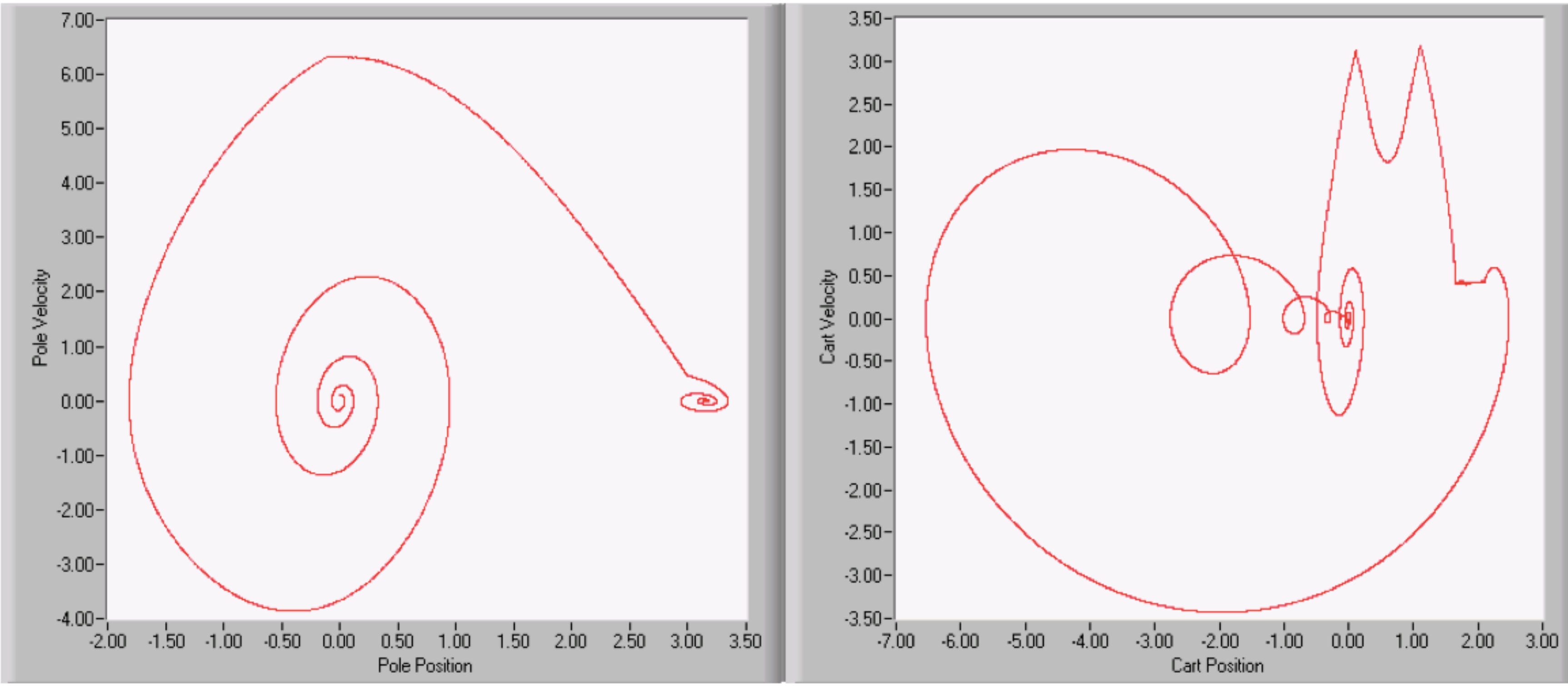
Heterogeneous Cart-Pole Controller

- Pivot torque controller stabilizes the pole (heterogeneous, 3 regions)
- Negative feedback stabilizes the cart, Lemma 1
- Combination of the two should preserve sliding mode for the heterogeneous pole controller
- We can derive the desired constraints:

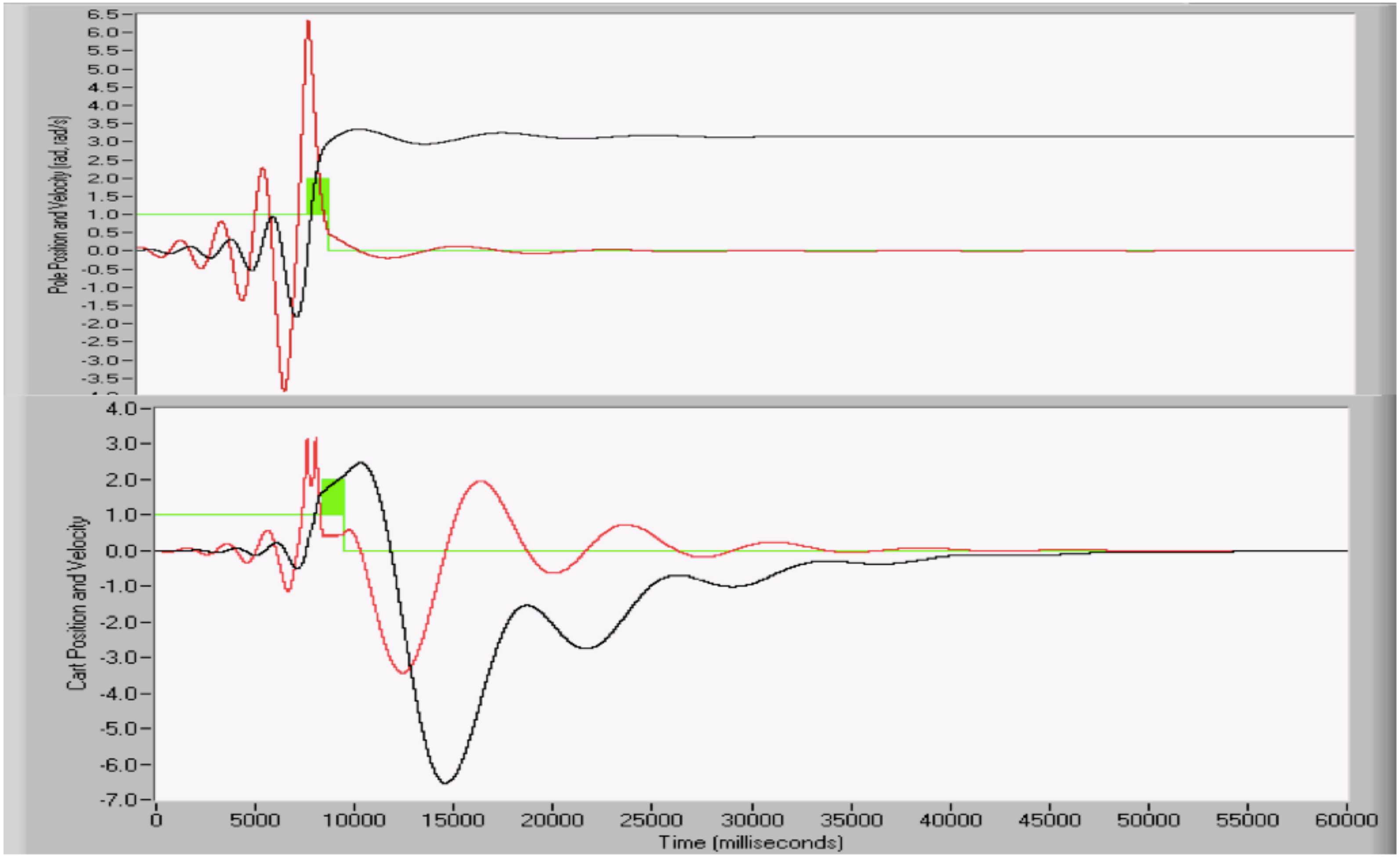
$$[(h-f)(\dot{\theta})+f_1(\dot{x})+g_1(x)]_0=[\dot{\theta}]_0$$

$$[(f+f_d)(\dot{\theta})-f_1(\dot{x})-g_1(x)]_0=[\dot{\theta}]_0$$

The Controlled Cart-Pole System



The Controlled Cart-Pole System



Take Home Messages from Pendulum

- Control can be thought of as “shaping of dynamics”
- Specifications for realistic systems can be non-trivial
  - Can be stated in the language of dynamical systems
  - Represented in terms of linear algebra and constraints
- Hybrid systems (switching between different dynamical system models/regimes) address global issues better
- Reasoning qualitatively with dynamical systems models provides a useful approach to specifying non-linear controller:
  - identifies weak sufficient conditions
  - any instance of QDE will achieve the behavior. So, separation of concerns between specification and optimization.

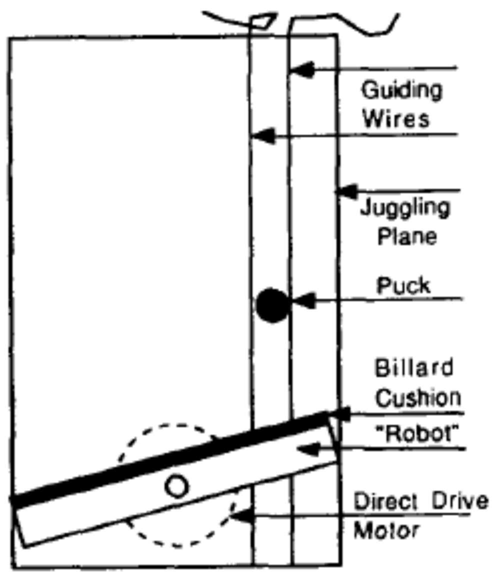
Case Study: Juggling

Objectives for Juggling Case Study

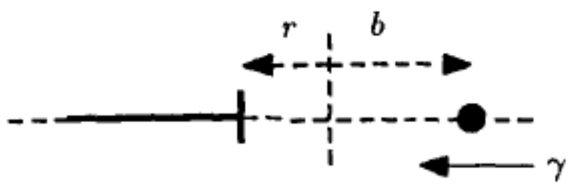
- Show an example that is clearly beyond the realm of traditional state feedback control
  - Still admits a solution that is based only on relatively simple local control laws
- Solution strategy that has genuinely been implemented very successfully on real robots
- Give a concrete another concrete example of a hybrid system, where a planner decides on controller choice

1-*dim* Mirror Law

Start with a line juggler



Task:



An open loop way would be to enforce post-contact vel.

$$\dot{b}' = \alpha \dot{b} + (1 + \alpha)\dot{r}$$

The free dynamics of the ball is simply ($\dot{r}, \dot{b}$ are robot/ball vel., γ is accel.):

$$b(t) = b' + \dot{b}'t - \frac{1}{2}\gamma t^2$$

$$\dot{b}(t) = \dot{b}' - \gamma t$$

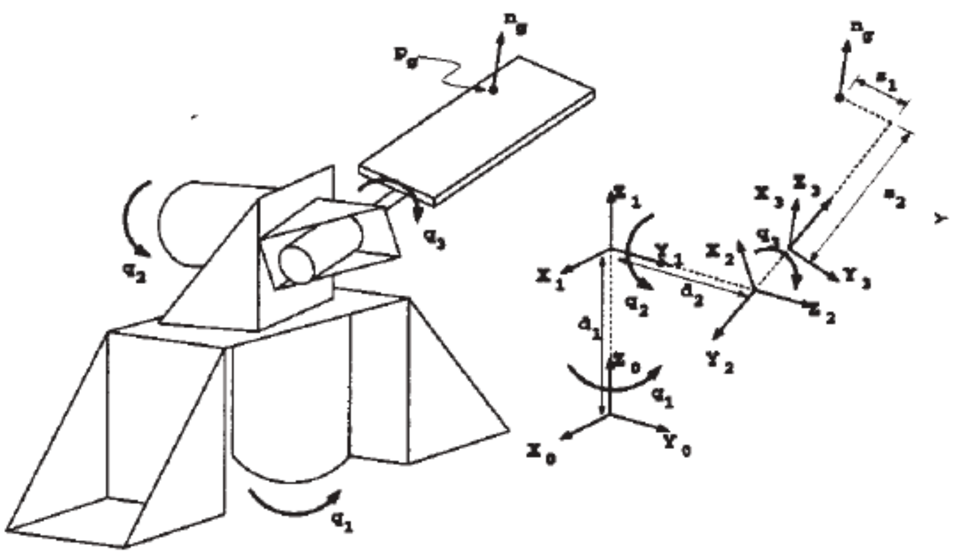
Remark: These two sets of equations apply in turn, between hits. This loop works but can be sensitive to noise.

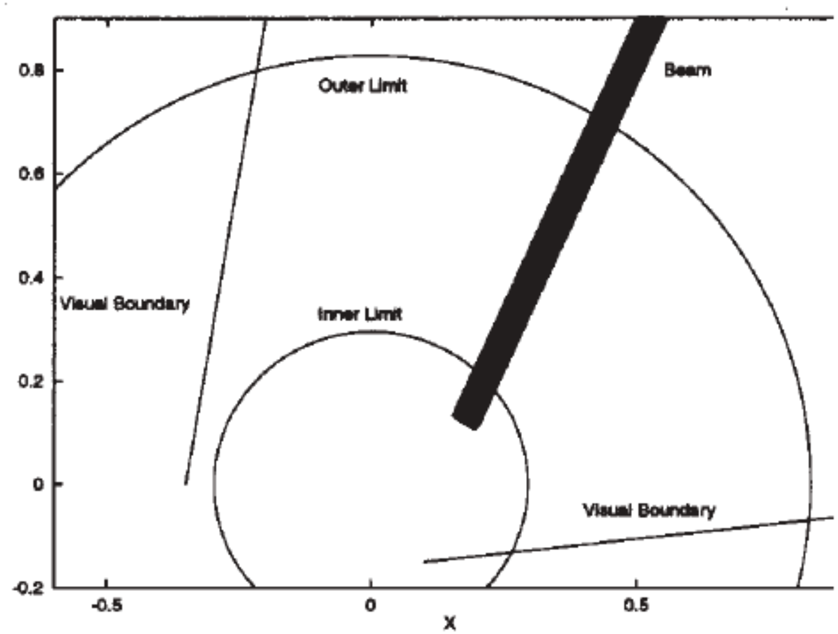
Q: What would happen if we removed the cylinder?

A Slightly More Complex Juggling Task: Dynamically Dexterous Manipulation

- Robot with flat paddle
  - required to strike repeatedly at thrown ball
  - until ball is brought to rest on the paddle at specified location
- Reachable workspace is disconnected
  - Ball and paddle can't remain in contact and approach goal location
  - Forces machine to *let go* for a time to bring the ball to desired state

The Buhgler Arm

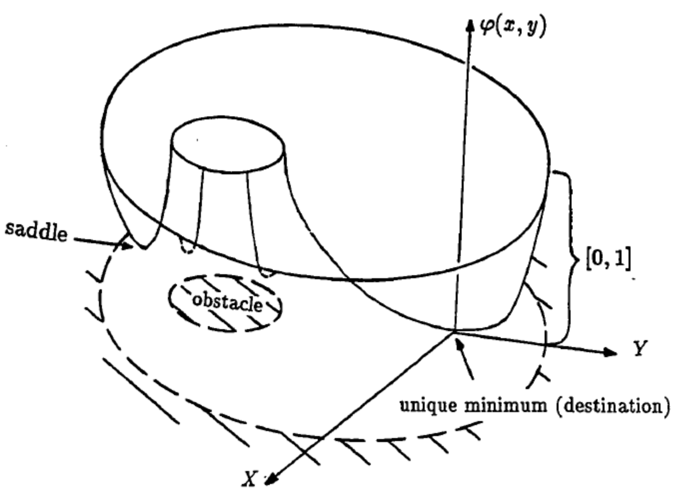




[Burridge, R. R., Rizzi, A. A., & Koditschek, D. E. (1999). Sequential composition of dynamically dexterous robot behaviors. International Journal of Robotics Research, 18(6), 534-555.]

Technical Questions

Potential functions were designed as a simple way to handle two concerns: (a) path planning w.r.t. obstacles, (b) actuator-level control, locally



Can we go further with this style of reasoning?

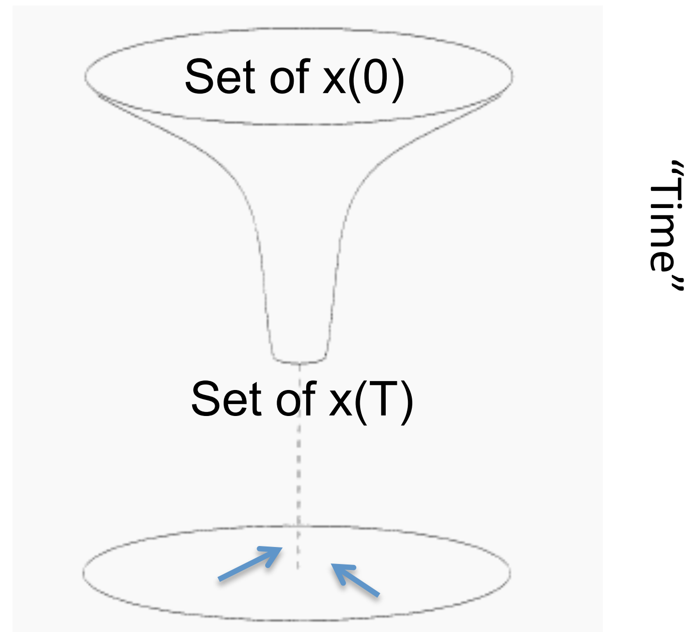
(How) can we encode a complex dynamically dexterous behaviour involving:

- Large unforeseen disturbances
- requiring some understanding of global dynamic behaviour
- With natural limits on sensing and actuation

Feedback Strategies (Controllers) as Funnels

- For our purposes, feedback strategies result in invariant regions

- These invariant regions are characterized by monotonically decreasing "energy functions" (e.g., as in *Lyapunov* stability)



State space

Sequential Controller Composition

- Controller compositions guarantee that a ball introduced in the "safe workspace" remains there and is ultimately brought to the goal
- Partition of state space induced by a palette of pre-existing feedback controllers
- Each cell associated with a unique controller, chosen such that entry into a cell guarantees passage to successively "lower" cells until the "lowest" goal cell is reached

Behaviours = Effect of "Local Controllers"

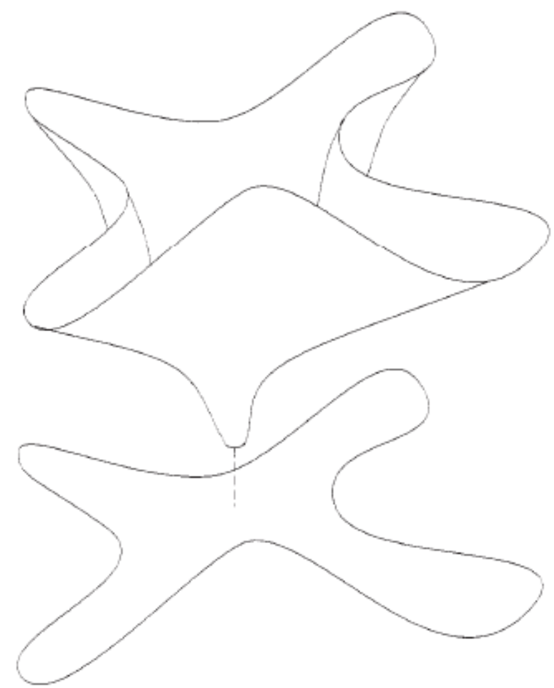
- Robotic implementations of user specified tasks
  - Might need different local controller based on specs
- Amenable to representation as state regulation (via feedback) to specified goal set, in the presence of obstacles

- **Closed loop dynamics of a plant operating under feedback**

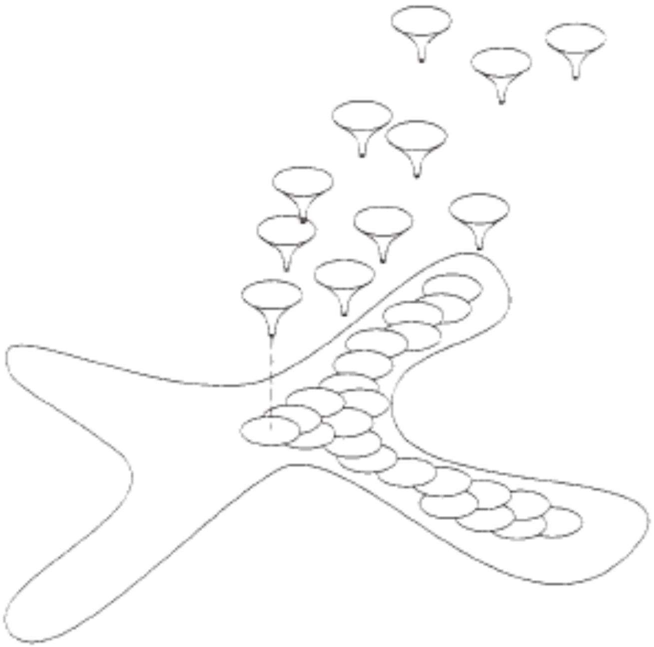
- No single feedback algorithm will successfully stabilize the large range of initial conditions
  - We already saw this with the pendulum case study

Feedback Strategies with Obstacles

- Most meaningful tasks include obstacles of one kind or another
- Obstacles tend to 'warp' the shape of the funnels
- In severe cases, obstacles can result in "disjoint functions" in state/ configuration space



Sequential Composition
- Visual Depiction of the Core Idea



Note: Will need a planner (back-chaining) to pick sequence

Physical Setting for Paddle Experiment

Hardware

- 3 DOF direct drive machine
- 2 cameras detect ball at 60 Hz
- Obstacle is a beam just above the paddle's height
- State space (T refers to tangent space, as in phase plane analysis)

$$Tb = (b, \dot{b}) \in TB$$

$$q = (\phi, \theta, \psi)$$

$$Tq = (q, \dot{q}) \in TQ$$

Physical Setting

Software

- Ball states, *Tb*, interpreted at 60 Hz by vision
- Vision data used by observer to estimate true *Tb*, interpolated at 330 Hz
- A memory-less transformation (mirror law) produces reference robot positions
- The reference robot positions fed to an inverse dynamics, joint-space controller

Discuss: Why do you **need** the "mirror law"?

The control system

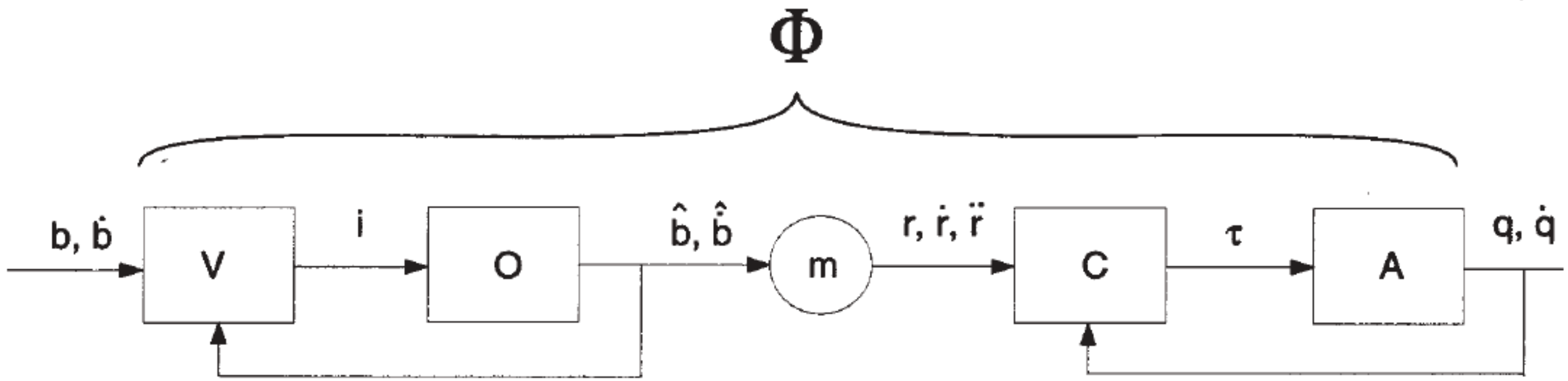


Fig. 6. Flow chart showing the various functional blocks of the system: vision, V; observer, O; mirror law, m; control, C; and actuation, A. The parameters of interest to this paper all reside in m.

[Figure from the Burridge et al. paper]

The Closed Loop System

- Repetitive continuous trajectories represented as "return map"
  - A discrete system from hit-to-hit, with a dynamics equation at that level
- Discrete event sampled mapping of the periodic orbit
  - This is also known as a *"Poincare section"* in dynamical systems

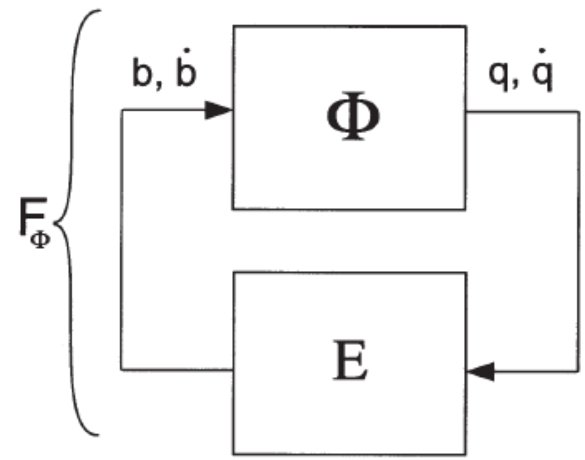


Fig. 7. The closed-loop dynamics, F_Φ, induced by Φ and the environment, E.

Discuss on board
(related to orbital stability)

Mirror Law Control

Define a mapping from the phase space of ball to configuration space of robot arm (as in slide on 1-dim case)

Mirror law is based on getting the effector to $q(b) = (\phi_b, \theta_b, 0)$

1. $\phi_r = \phi_b$ causes the paddle to track under the ball at all times.
2. θ_r mirrors the vertical motion of the ball (as it evolves in θ_b): as the ball goes up, the paddle goes down, and vice-versa, meeting at zero height. Differences between the desired and actual total ball energy lead to changes in paddle velocity at impact.
3. Radial motion or offset of the ball causes changes in θ_r, resulting in a slight adjustment of the normal at impact, tending to push the ball back toward the set point.
4. Angular motion or offset of the ball causes changes in ψ_r, again adjusting the normal so as to correct for lateral position errors.

FYI: Appendix on Mirror Law

- Define vertical energy and radial distance as: $\eta = \gamma b_z + \frac{1}{2}\dot{b}_z^2$

$$\rho_b = sin(\theta_b)s_b$$

- The "mirror law" has the following form for different components:

$$m(w) \triangleq \left[\begin{array}{c} \overbrace{\phi_b}^{(i)} \\ \underbrace{-\frac{\pi}{2} - (\kappa_0 + \kappa_1(\eta - \bar{\eta}))\left(\theta_b + \frac{\pi}{2}\right)}_{(ii)} \quad + \\ \underbrace{\kappa_{00}(\rho_b - \bar{\rho}_b) + \kappa_{01}\dot{\rho}_b}_{(iii)} \\ \underbrace{\kappa_{10}(\phi_b - \bar{\phi}_b) + \kappa_{11}\dot{\phi}_b}_{(iv)} \end{array}\right]$$

Note: Sophistication of these expressions is minimal… (PD, really) The controller itself is of low complexity!

Domain of the "mirror law" m_j

- **No closed form expression of return map** (hard to write down explicitly)
- Therefore, difficult to ascertain the shape of the boundaries of domain of attraction
- Use experimental data to formulate an approximation of *safe domain*
- To speed up deployment, create numerical simulation of the juggler and use it to determine domain of attraction

Domain of m_j : Experiments

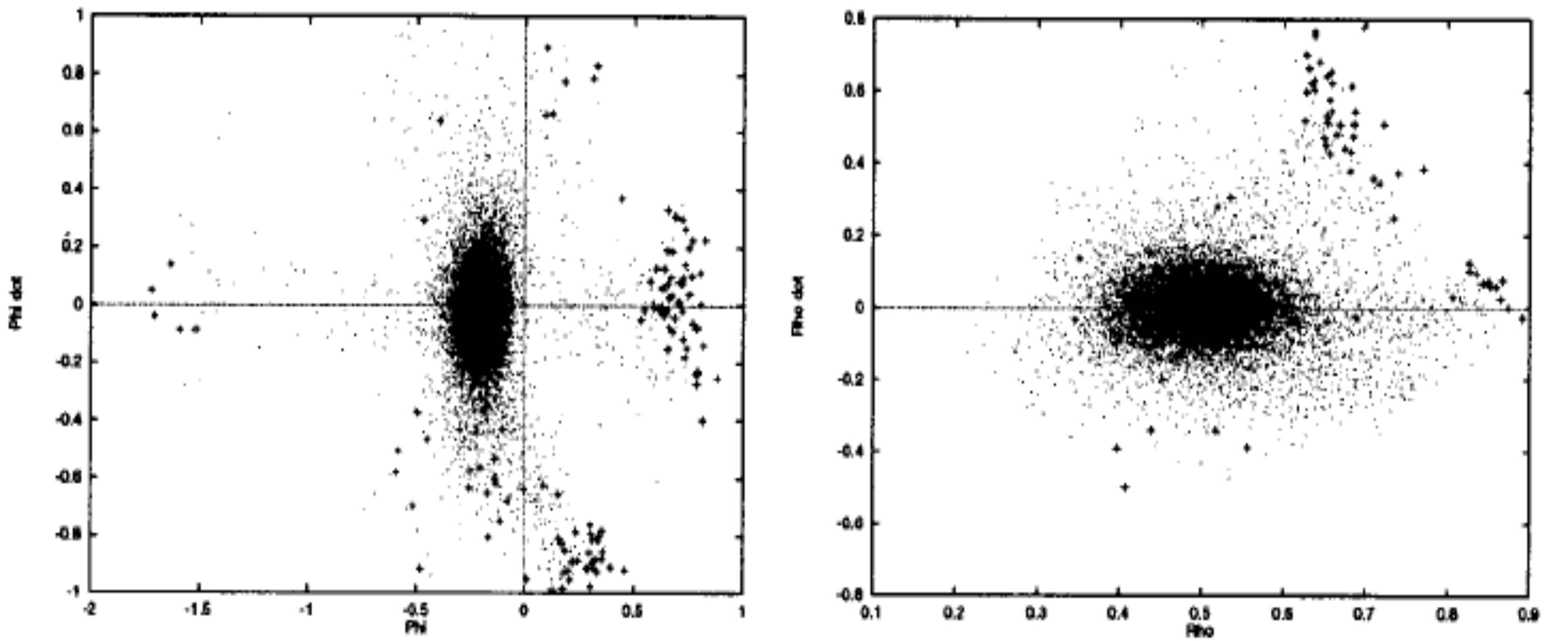


Fig. 8. Empirical data used to estimate the juggling domain, $\mathcal{D}(\Phi_J)$. Each dot (+ sign) represents in apex coordinates a ball trajectory that was successfully (unsuccessfully) handled under the action of Φ_J. Because of the properties of the vertical subsystem, most of these points are at nearly the same height, so only the horizontal coordinates are plotted.

Complete Control Strategy

- A set of controllers $\mathcal{U} = \{\Phi_1, ..., \Phi_N\}$ is designed to handle various scenarios
- Scenarios include:
  - Juggle (mirror law)
  - Palming
  - Catching

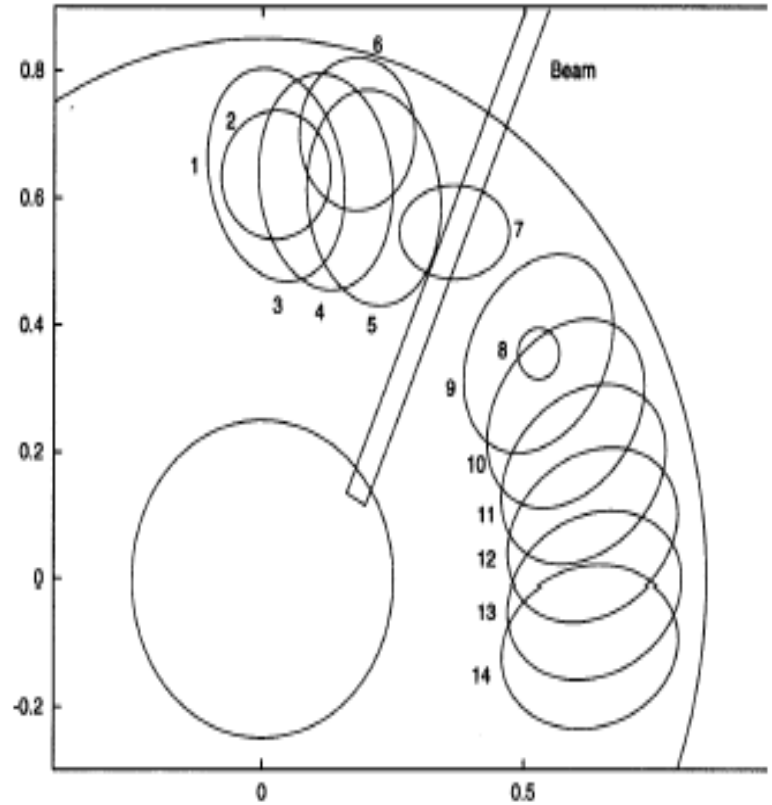
1. Let the queue contain Φ_1. Let $\mathcal{C}(\Phi_1) = \mathcal{D}(\Phi_1)$, $N = 1$, $\mathcal{U}'(1) = \{\Phi_1\}$, and $\mathcal{D}_1(\mathcal{U}') = \mathcal{D}(\Phi_1)$.
2. Remove the first element of the queue, and append the list of all controllers which *prepare* it to the back of the list.
3. While the first element of the queue has a previously defined cell, $\mathcal{C}$, remove the first element without further processing.
4. For Φ_j, the first unprocessed element on the queue, let $\mathcal{C}(\Phi_j) = \mathcal{D}(\Phi_j) - \mathcal{D}_N(\mathcal{U}')$. Let $\mathcal{U}'(N+1) = \mathcal{U}' \cup \{\Phi_j\}$, and $\mathcal{D}_{N+1}(\mathcal{U}') = \mathcal{D}_N(\mathcal{U}') \cup \mathcal{D}(\Phi_j)$. Increment N.
5. Repeat steps 2, 3, and 4 until the queue is empty.

Composition of Domains

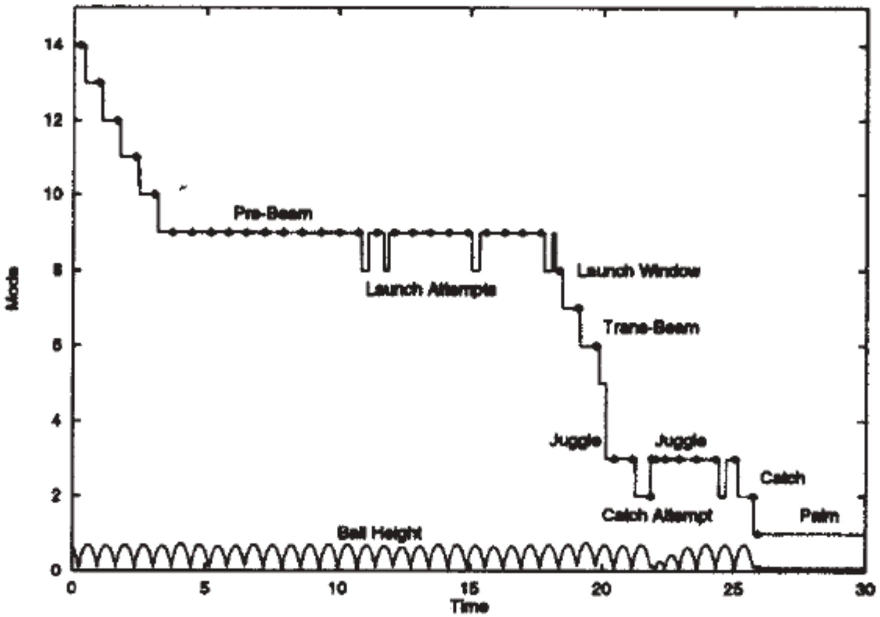
Table 1. The Full Deployment, with Controller Types, Goal Points, and Domain Types[a]

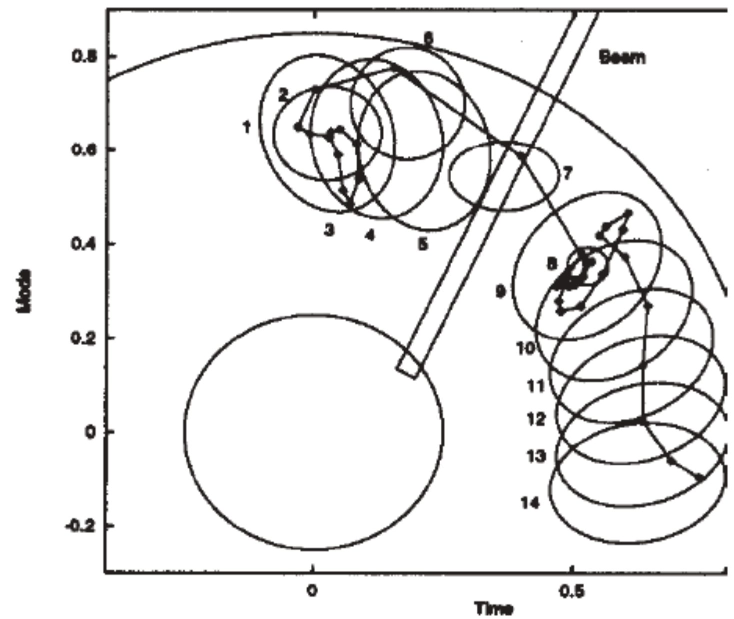
Ellipses	Type	Goal: $\bar{\phi}$	Domain Type
1	Φ_P	0.3	$\mathcal{D}_P$
2	Φ_C	0.3	$\mathcal{D}_C$
3	Φ_J	0.3	$\mathcal{D}_0$
4	Φ_J	0.15	$\mathcal{D}_0$
5–8	Φ_J	0.0	$\mathcal{D}_1$
9	Φ_J	−0.64	$\mathcal{D}_0$
10	Φ_J	−0.81	$\mathcal{D}_0$
11	Φ_J	−0.97	$\mathcal{D}_0$
12	Φ_J	−1.12	$\mathcal{D}_0$
13	Φ_J	−1.26	$\mathcal{D}_0$
14	Φ_J	−1.4	$\mathcal{D}_0$

a. All goal points have the same radial component, $\bar{\rho}_0 = 0.6$, so we list here only the angular component, $\bar{\phi}$. Ellipse numbers correspond to those in Figure 16.



A Typical Run





What do the Results Look Like?

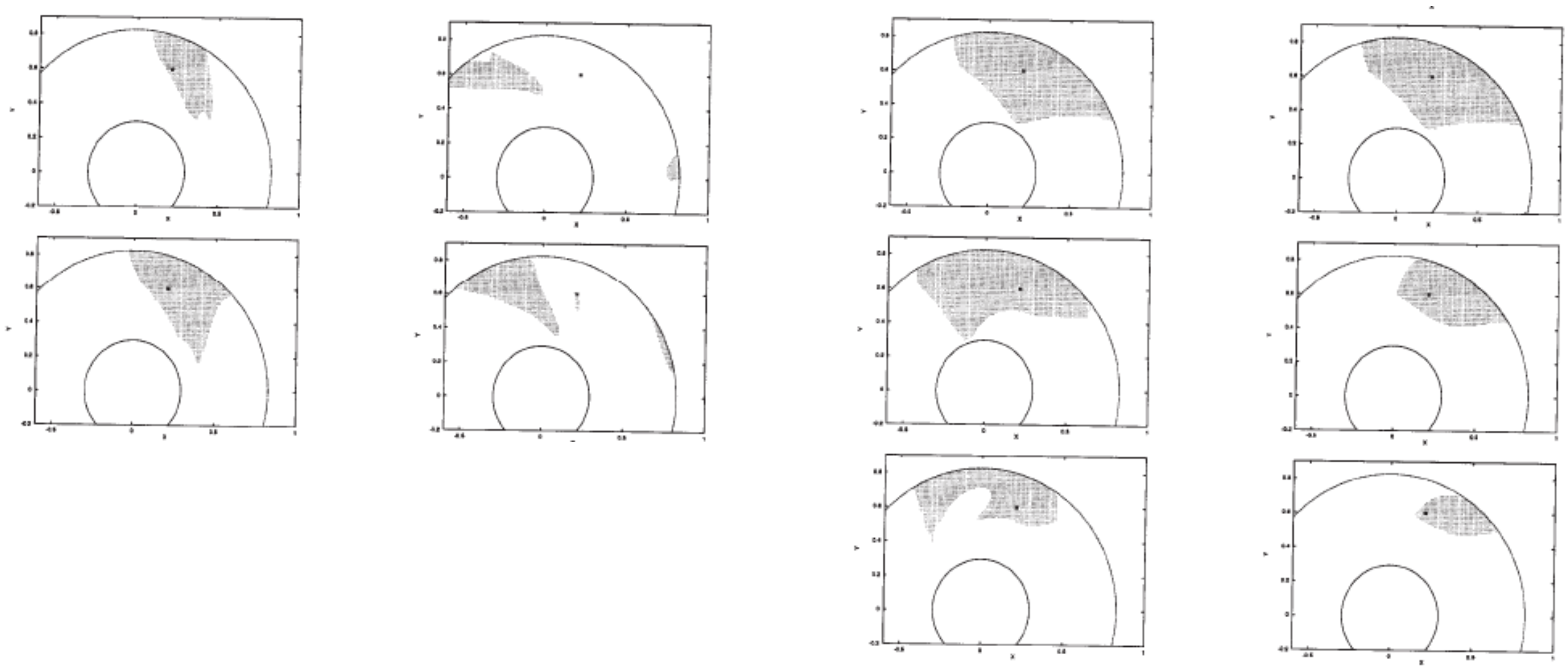


Fig. 11. Shaded regions denote initial conditions that were successfully contained in the workspace while being brought to the goal via iteration of f_{Φ_J}: varying initial $\dot{x}_i$ from negative (top) to positive (bottom) with $\dot{y}_i = 0$ (left column); varying initial $\dot{y}_i$ from negative to positive with $\dot{x}_i = 0$ (right column). The scale for all plots is meters.

Results: Effect of Obstacle

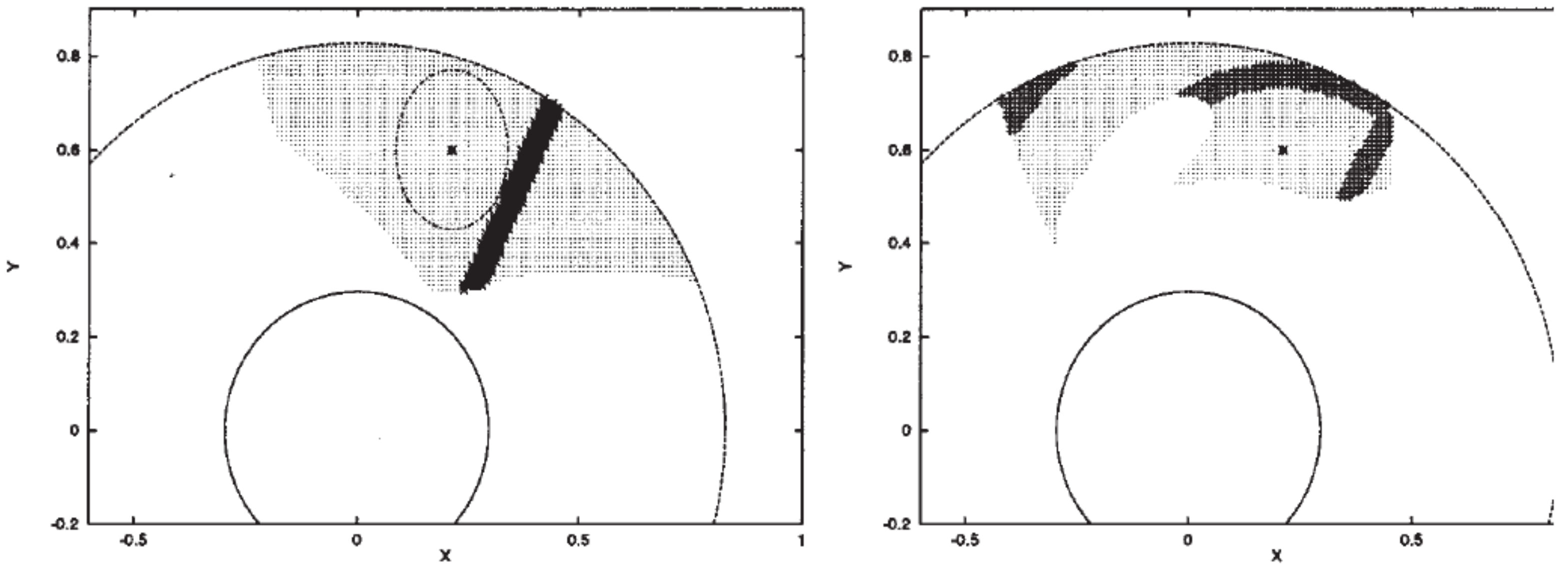
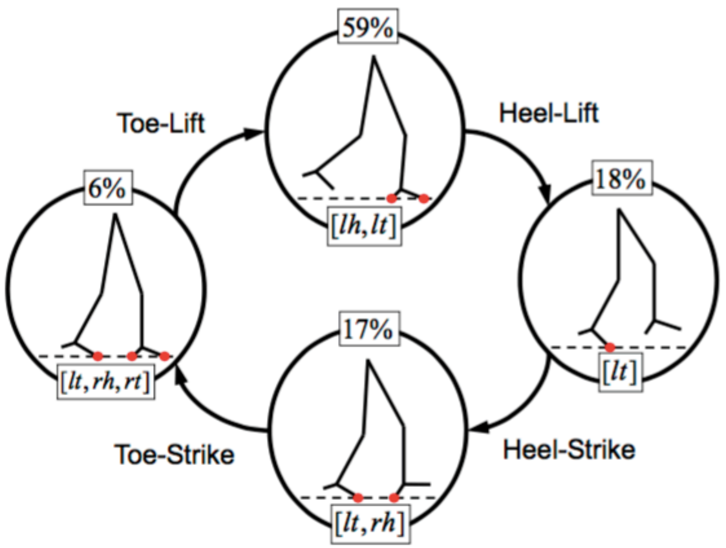


Fig. 13. The safe domain for Φ_{J_0} with the beam inserted. Light-gray areas represent successful initial states, while darker areas show states that eventually hit the beam. Zero initial velocity is shown (a), and the appropriate slice of $\mathcal{D}_0$ is added for comparison. For $\dot{x}_i = 1.5\frac{m}{s}$ (b), preimages of the beam are evident.

Funnels in the “Real” World



[Grizzle et al. Automatica 50.8 (2014)]
Also see: Pratt’s VMC (link on course web site)