Intelligent Control

Robert Babuška

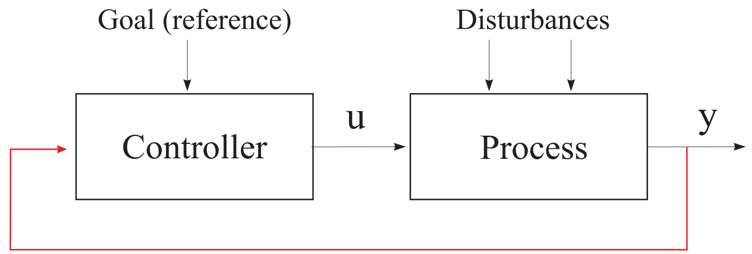
Delft Center for Systems and Control
Delft University of Technology
The Netherlands

r.babuska@dcsc.tudelft.nl
www.dcsc.tudelft.nl/˜babuska

Outline

- Motivation
- Fuzzy Control
  - direct, supervisory
- Model-Based Control
  - use of fuzzy and neural models
- Adaptation and learning
  - indirect, direct methods
- Conclusions

Controller



- Automatic controller (algorithm).
- Human operator.
- Decision support system.

Design Procedure

1) modeling $\longrightarrow$ mathematical model (control-oriented)

2) analysis $\longrightarrow$ properties, (modified) specifications

3) synthesis $\longrightarrow$ control equations and parameters

4) implementation $\longrightarrow$ computer algorithm or hardware

At least, this is the theory ...

Industrial Practice

1. Order the controller
2. Unpack and connect
3. Turn the knobs until it works (or doesn't)

This is of course slightly exaggerated, notable exceptions

aerospace, mechatronics, automotive, . . .

Design Effort / Costs

1) modeling $\approx \mathbf{70}\,\%$

2) analysis $\approx \mathbf{10}\,\%$

3) synthesis $\approx \mathbf{10}\,\%$

4) implementation $\approx \mathbf{10}\,\%$

Main drawback of this standard design:
cannot start anywhere else than at step 1)

Challenges in Control

- modeling of poorly understood systems
- combination of continuous dynamics and logic
- high level of autonomy, coordination
- automatic synthesis, self-learning systems
- complex, distributed systems, networks, agents

'Intelligent' Control Techniques

motivated by human intelligence

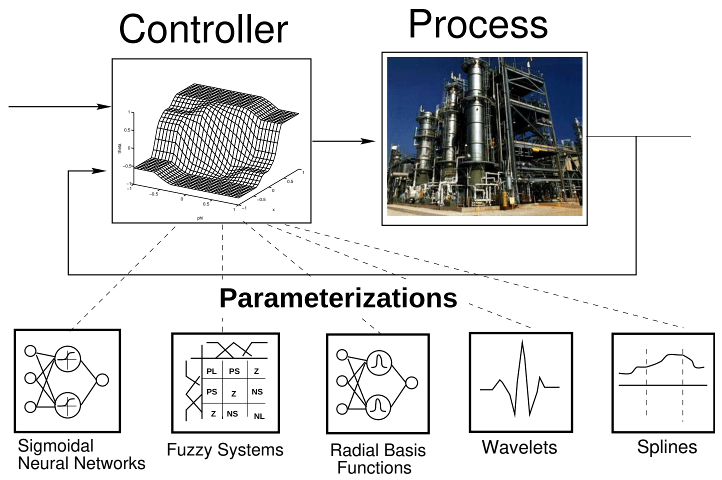
- fuzzy systems – expert knowledge, reasoning
- neural networks – learning, adaptation
- genetic algorithms – optimization

enable the use of extra information in the design,
implement extra functionality

Fuzzy Control: Background

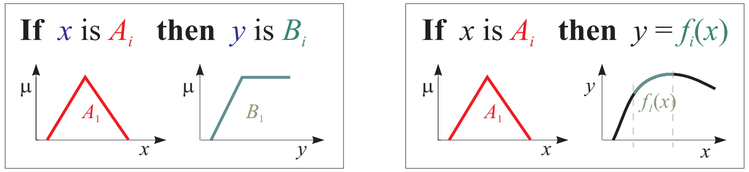
- controller designed by using If–Then rules instead of mathematical formulas (knowledge-based control),
- early motivation: mimic experienced operators,
- fuzzy reasoning: interpolation between discrete outputs,
- currently: also controllers designed on the basis of a fuzzy model (model-based fuzzy control),
- a fuzzy controller represents a *nonlinear* mapping (but completely deterministic!).

Parameterization of Nonlinear Controllers



Two Main Types of Rule-Based Fuzzy Systems

Linguistic (Mamdani) | *Takagi-Sugeno*



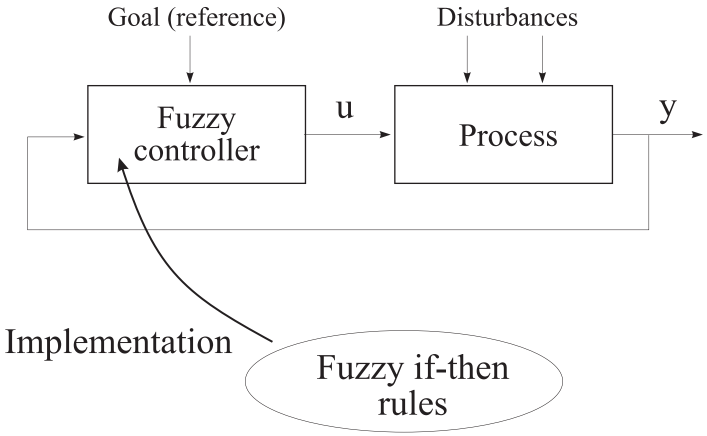
Representing human knowledge:
- decision support
- control and supervision
- fault detection, diagnosis

Function approximation (fitting):
- nonlinear identification
- nonlinear control
- gain scheduling

Basic Fuzzy Control Schemes

- Direct (low-level, Mamdani) fuzzy control
- Fuzzy supervisory (high-level, Takagi–Sugeno) control
- Fuzzy model-based control

Direct Fuzzy Control

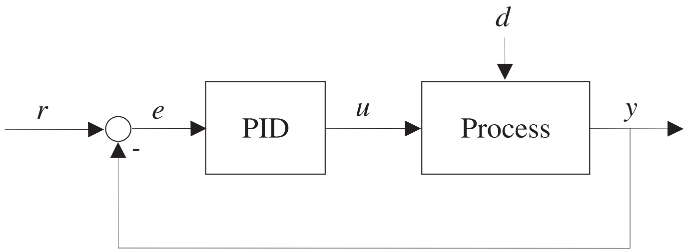


Early Motivation: Manual Process Control

Case	Condition	Action to be taken	Reason
11	BZ OK	a. Decrease fuel rate slightly	To raise percentage of oxygen
	OX low		
	BE OK		
12	BZ OK	a. Reduce fuel rate	To increase percentage of oxygen for action b.
	OX low	b. Reduce fan speed	To lower back-end temperature and
	BE high		maintain burning zone temperature
13	BZ OK	a. Increase fan speed	To raise back-end temperature
	OX OK	b. Increase fuel rate	To maintain burning zone temperature
	BE low		
14	BZ OK	NONE.	
	OX OK		
	BE OK		

Extract from Peray's textbook for kiln operators (Oestergaard, 1999)

Replacement of PID Control

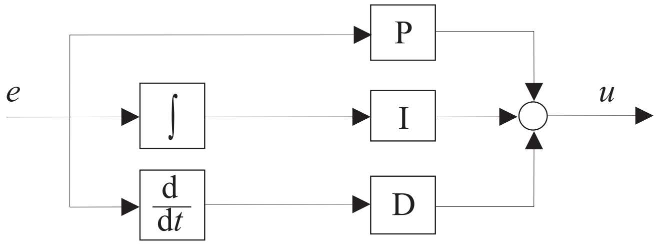


Controller:

- dynamic: $u(t) = Pe(t) + I\int_0^t e(\tau)d\tau + D\frac{de(t)}{dt}$
- P, I and D are the proportional, integral and derivative gains, respectively

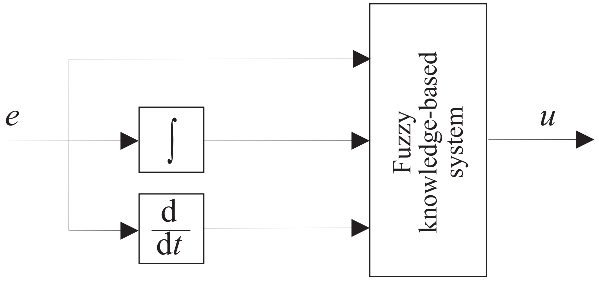
PID Control: Internal View

$$u(t) = Pe(t) + I\int_0^t e(\tau)d\tau + D\frac{de(t)}{dt}$$

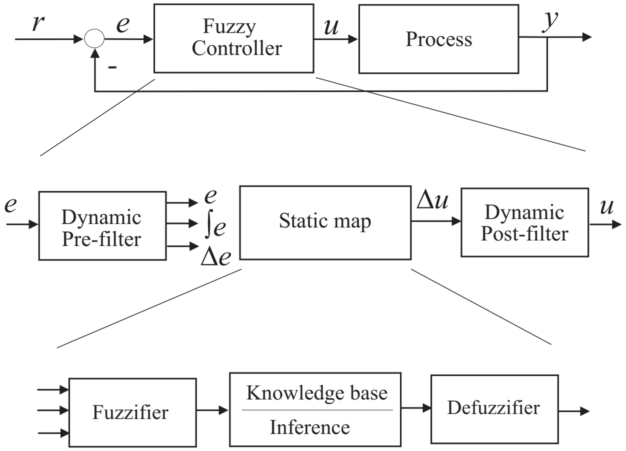


Fuzzy PID Control

$$u(t) = f\left(e(t), \int_0^t e(\tau)d\tau, \frac{de(t)}{dt}\right)$$



Fuzzy PID Control



Fuzzy PD Controller: Rule Table

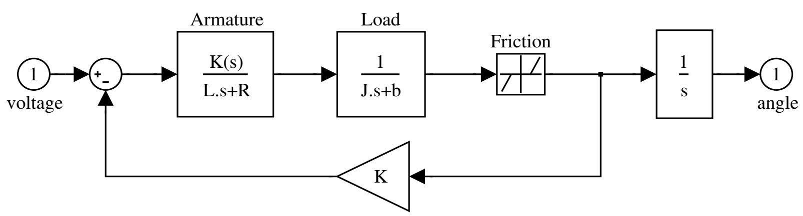
error \ *error rate*	NB	ZE	PB
NB	NB	NB	ZE
ZE	NB	ZE	PB
PB	ZE	PB	PB

R_{12}: If *error* is NB and *error rate* is ZE then *control* is NB

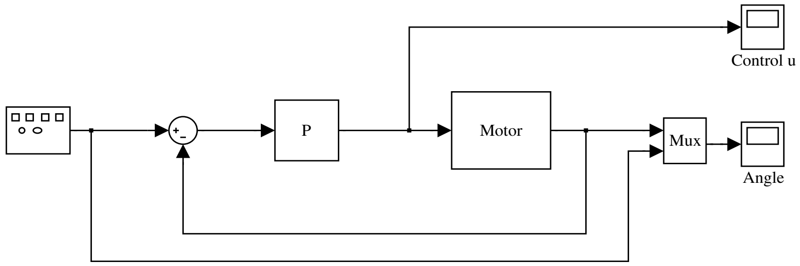
Example: Friction Compensation

1. DC motor with static friction.
2. Fuzzy rules to represent "normal" proportional control.
3. Additional rules to prevent undesirable states.

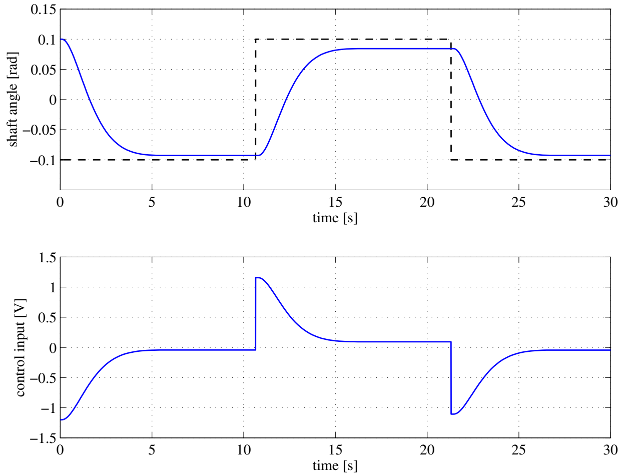
DC Motor: Model



Proportional Controller



Linear Control



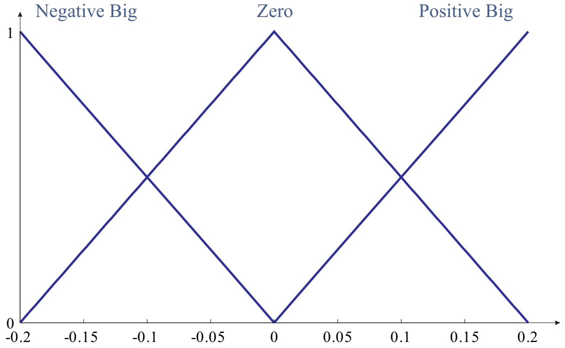
Fuzzy Control Rule Base

If error is Positive Big then control input is Positive Big;

If error is Negative Big then control input is Negative Big;

If error is Zero then control input is Zero;

Membership Functions for Error



Additional Rules

If error is Positive Big then control input is Positive Big;

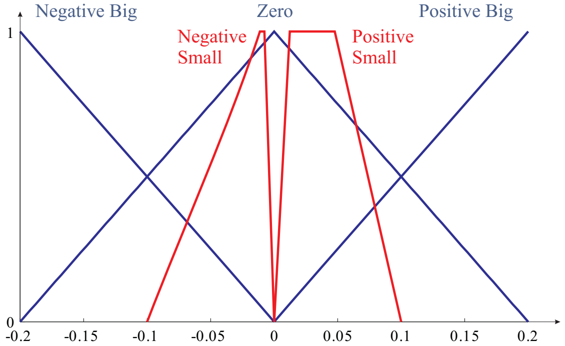
If error is Negative Big then control input is Negative Big;

If error is Zero then control input is Zero;

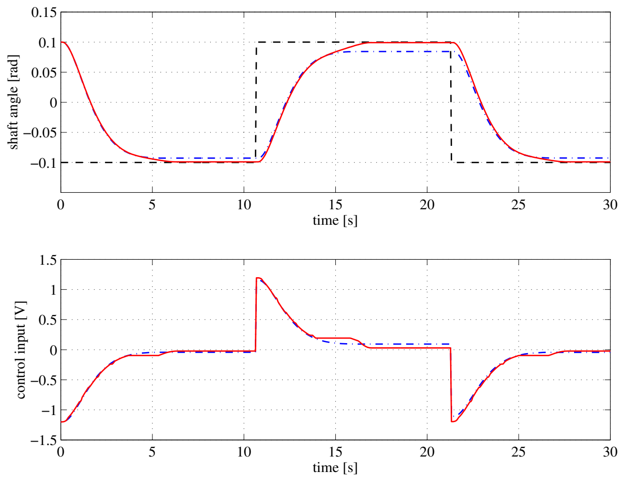
If error is Negative Small
then control input is not Negative Small;

If error is Positive Small
then control input is not Positive Small;

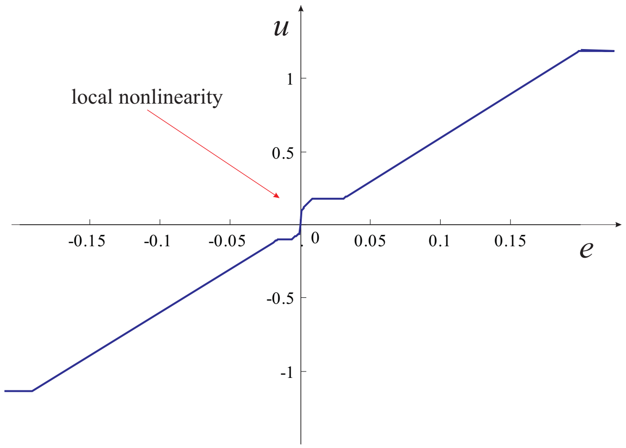
Membership Functions for Error



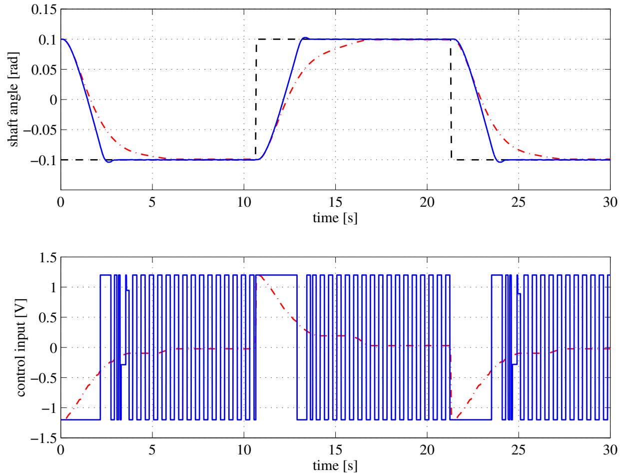
Fuzzy Control



Input–Output Mapping of the Controller

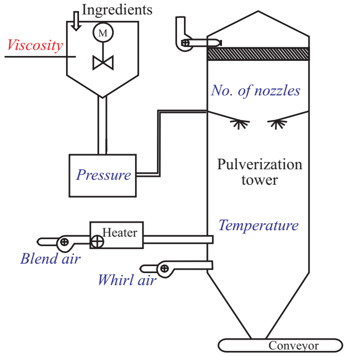


Sliding Mode Control



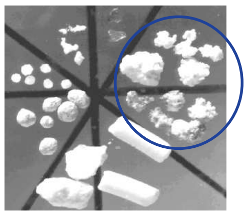
Decision Support in Process Control

Production of detergents

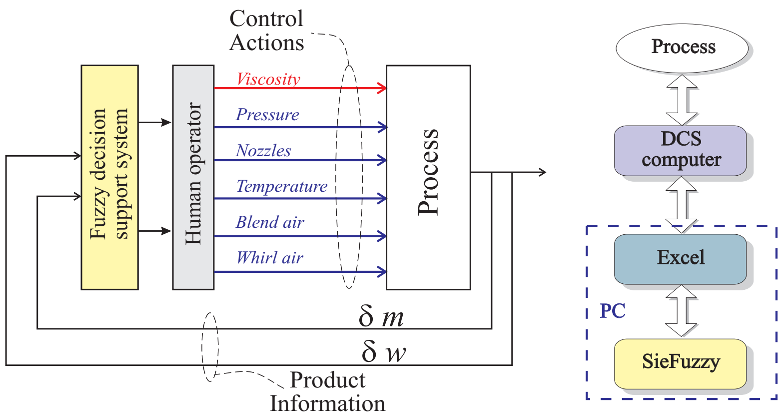


Quality:

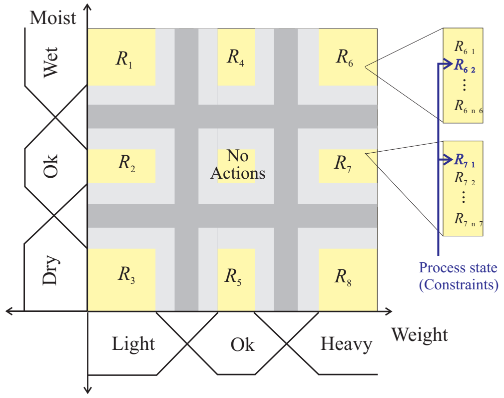
- liter weight
- moist content



Fuzzy Decision Support System

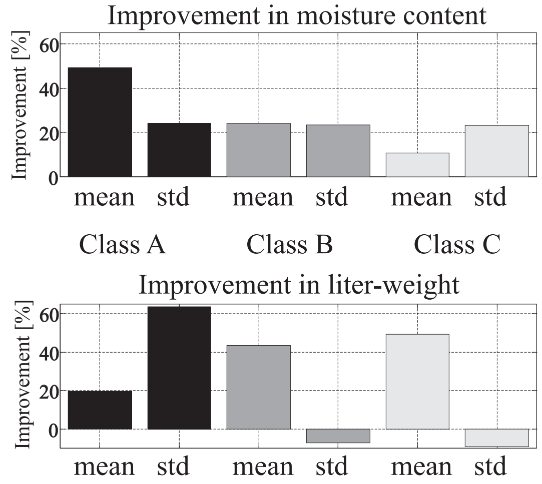


Fuzzy Rule Base

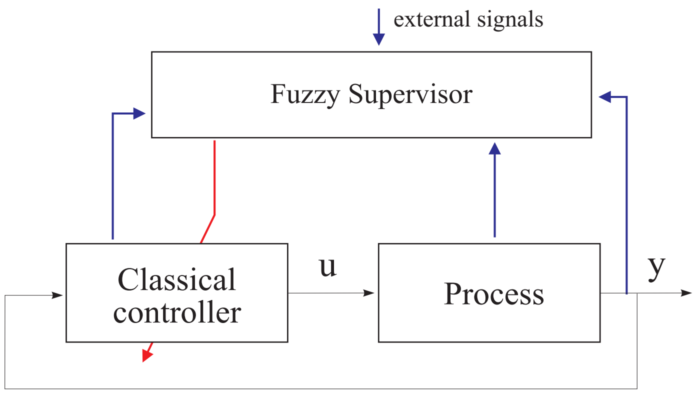


- Partitioning into error regions
- Each region has an ordered set of control rules

Evaluation: Results



Supervisory Fuzzy Control



Supervisory Control Rules: Example

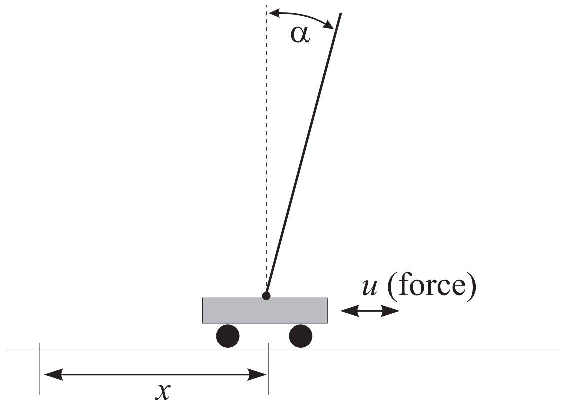
If process output is *High*

then reduce proportional gain *Slightly* ***and***

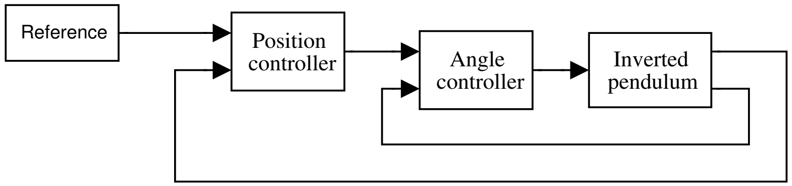
increase derivative gain *Moderately*.

(Supervised PD controller)

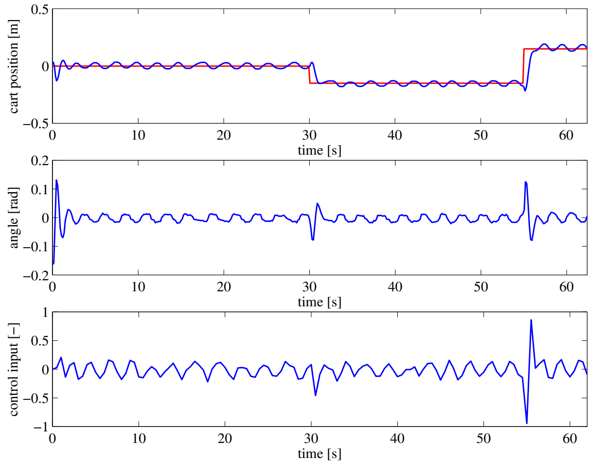
Example: Inverted Pendulum



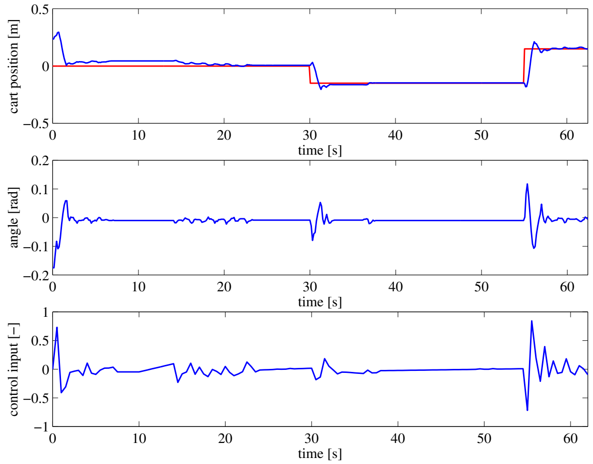
Cascade Control Scheme



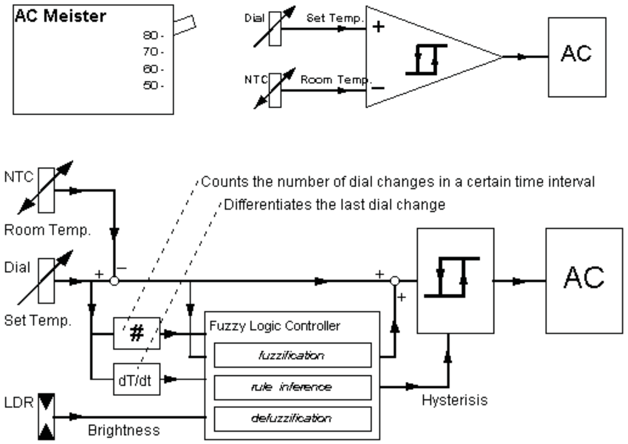
Conventional PD controller



Fuzzy Supervised PD controller



Intelligent Thermostat

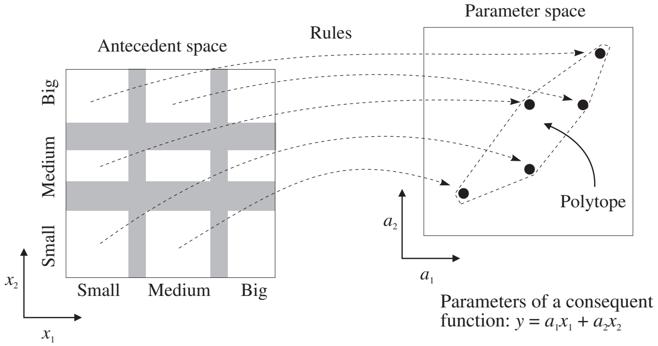


Models in Systems and Control

- Identification
- Stability analysis
- State estimation
- Control design
- Fault detection and isolation
- Control reconfiguration
- Adaptive systems
- etc.

TS Model as a Polytopic (LPV) System

$$y = \sum_{i=1}^{l} w_i(x)\mathbf{a}_i^T\mathbf{x} = \mathbf{a}(x)^T\mathbf{x} \quad \text{with} \quad \mathbf{a}(x) = \sum_{i=1}^{l} w_i(x)\mathbf{a}_i,$$



Continuous-Time Dynamic TS Model

$$R_i: \quad \textbf{If } x_1 \textbf{ is } F_1^i \textbf{ and } x_2 \textbf{ is } F_2^i \textbf{ and } \ldots \textbf{ and } x_q \textbf{ is } F_q^i$$

$$\textbf{then} \begin{cases} \dot{\mathbf{x}} = A_i x + B_i u + \mathbf{a}_i \\ \mathbf{y} = C_i x + \mathbf{c}_i \end{cases}$$

$$\dot{\mathbf{x}} = A(x)x + B(x)u + \mathbf{a}(x)$$
$$\mathbf{y} = C(x)x + \mathbf{c}(x)$$

LPV system with

$$A(x) = \sum_{i=1}^{l} w_i(x)A_i, \quad B(x) = \sum_{i=1}^{l} w_i(x)B_i, \quad C(x) = \sum_{i=1}^{l} w_i(x)C_i, \quad \ldots$$

Discrete-Time Dynamic TS Model

Input–output form:

If $y(k)$ is F_1^i **and** $u(k)$ is F_2^i

then $y_i(k+1) = \sum_{j=1}^{n_y} a_{ij} y(k-j+1) + \sum_{j=1}^{n_u} b_{ij} u(k-j+1) + c_i$

State-space form:

If $x(k)$ is F_1^i **and** $u(k)$ is F_2^i

$$\textbf{then} \begin{cases} x_i(k+1) = \boldsymbol{\Phi}_i x(k) + \boldsymbol{\Gamma}_i u(k) + a_i \\ y_i(k) = \boldsymbol{C}_i x(k) + \boldsymbol{D}_i u(k) + c_i \end{cases}$$

Stability of TS Systems

$$\dot{\mathbf{x}} = \sum_{i=1}^{l} w_i(x) A_i x$$

Sufficient condition for asymptotic stability: $\exists P = P^T$ s.t.

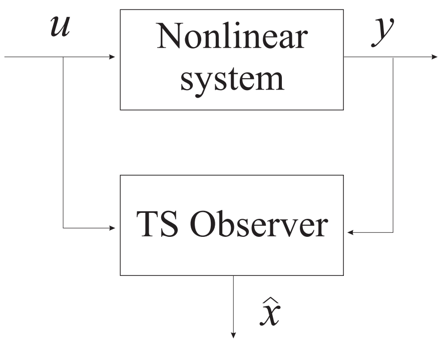
$$P > 0$$
$$A_i^T P + P A_i < 0 \quad \forall i = 1, 2, \dots, l$$

Proof: Lyapunov function $V(x) = x^T P x$

$$\dot{V}(x) = \dot{x}^T P x + x^T P \dot{x} = \sum_{i=1}^{l} w_i(x) x^T (A_i^T P + P A_i) x$$

Use linear matrix inequalities (LMI) to find P.

Fuzzy Observers



Fuzzy TS Observer

$$\dot{\hat{x}} = \sum_{i=1}^{l} w_i(x)(A_i \hat{x} + B_i u + a_i + L_i(y - \hat{y}))$$
$$\hat{y} = \sum_{i=1}^{l} w_i(x)(C_i \hat{x} + c_i)$$

Design straightforward if x does not depend on $\hat{x}$

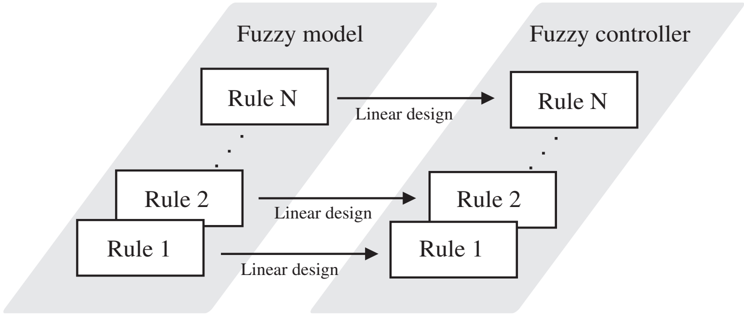
Error Dynamics

Error dynamics ($e = x - \hat{x}$):

$$\begin{aligned} \dot{e} &= \sum_{i=1}^{l} w_i(x)(A_i x + B_i u + a_i) - \sum_{i=1}^{l} w_i(x)(A_i \hat{x} + B_i u + a_i + L_i(y - \hat{y})) \\ &= \sum_{i=1}^{l} w_i(x) A_i e - \sum_{i=1}^{l} w_i(x) L_i \sum_{j=1}^{l} w_j(x) C_j e \\ &= \sum_{i=1}^{l} \sum_{j=1}^{l} w_i(x) w_j(x)(A_i - L_i C_j) e \end{aligned}$$

Lyapunov function: $V(e) = e^T P e$

Control Design Based on a TS Model



Apply classical synthesis and analysis methods locally, analyze global stability.

Closed Loop

Model:

If x is F_i **then** $\dot{\mathbf{x}}_i = \mathbf{A}_i \mathbf{x} + \mathbf{B}_i \mathbf{u}$

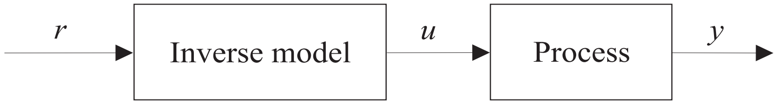
Controller:

If x is F_i **then** $\mathbf{u}_i = -\mathbf{K}_i \mathbf{x}$

Closed loop:

$$\dot{\mathbf{x}} = \sum_{i=1}^{l} \sum_{j=1}^{l} w_i(\mathbf{x}) w_j(\mathbf{x})(\mathbf{A}_i - \mathbf{B}_i \mathbf{K}_j)\mathbf{x}$$

Inverse Control (Feedforward)



Process model: $y(k+1) = f(\mathbf{x}(k), u(k))$, where

$$\mathbf{x}(k) = [y(k), \ldots, y(k - n_y + 1), u(k-1), \ldots, u(k - n_u + 1)]^T$$

Controller: $u(k) = f^{-1}(\mathbf{x}(k), r(k+1))$

When is Inverse-Model Control Applicable?

1. Process (model) is stable and invertible
2. The inverse model is stable
3. Process model is accurate (enough)
4. Little influence of disturbances
5. In combination with feedback techniques

How to invert $f(\cdot)$?

1. Numerically (general solution, but slow):

$$J(u(k)) = [r(k+1) - f(\mathbf{x}(k), u(k))]^2$$

minimize w.r.t. u(k)

2. Analytically (for some special forms of $f(\cdot)$ only):
   - affine in $u(k)$
   - singleton fuzzy model
3. Construct inverse model directly from data

Inverse of an Affine Model

affine model:

$$y(k+1) = g(\mathbf{x}(k)) + h(\mathbf{x}(k)) \cdot u(k)$$

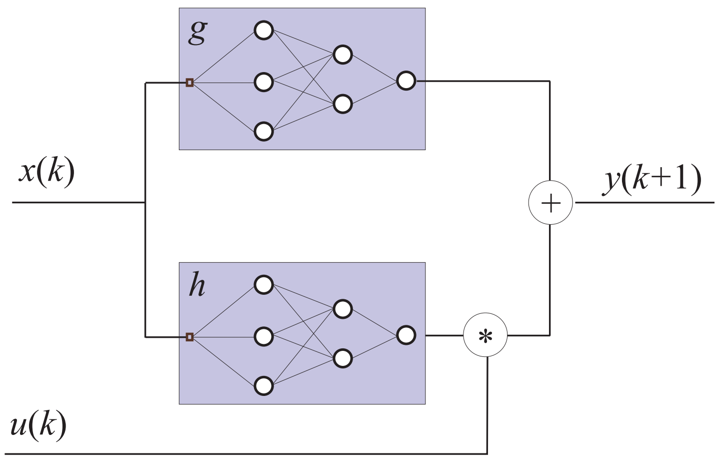
express $u(k)$:

$$u(k) = \frac{y(k+1) - g(\mathbf{x}(k))}{h(\mathbf{x}(k))}$$

substitute $r(k+1)$ for $y(k+1)$

necessary condition $h(\mathbf{x}) \neq 0$ for all $\mathbf{x}$ of interest

Example: Affine Neural Network



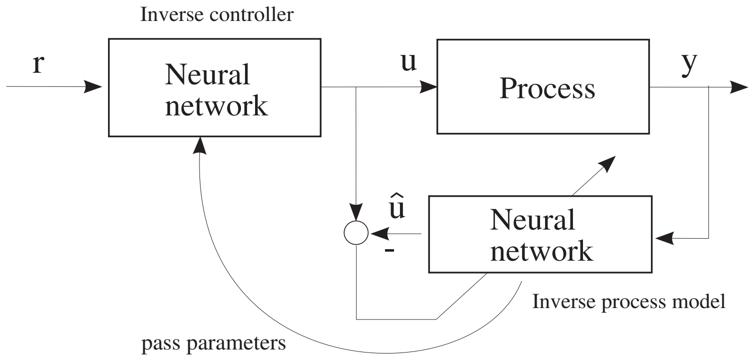
Example: Affine TS Fuzzy Model

R_i : **If** $y(k)$ **is** A_{i1} **and** ... **and** $y(k-n_y+1)$ **is** A_{in_y} **and** $u(k-1)$ **is** B_{i2} **and** ... **and** $u(k-n_u+1)$ **is** B_{in_u} **then**

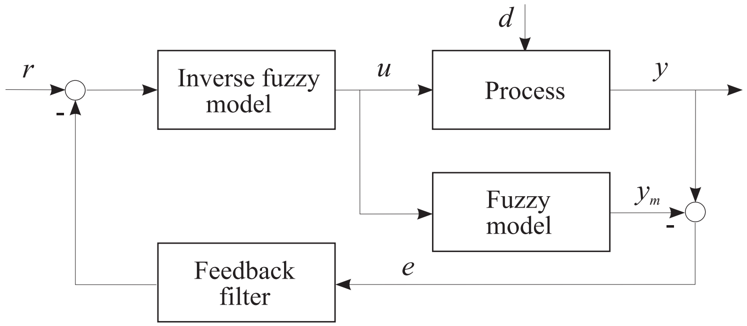
$$y_i(k+1) = \sum_{j=1}^{n_y} a_{ij} y(k-j+1) + \sum_{j=1}^{n_u} b_{ij} u(k-j+1) + c_i,$$

$$y(k+1) = \sum_{i=1}^{K} \gamma_i(\mathbf{x}(k)) \left[\sum_{j=1}^{n_y} a_{ij} y(k-j+1) + \sum_{j=2}^{n_u} b_{ij} u(k-j+1) + c_i \right] + \sum_{i=1}^{K} \gamma_i(\mathbf{x}(k)) b_{i1} u(k)$$

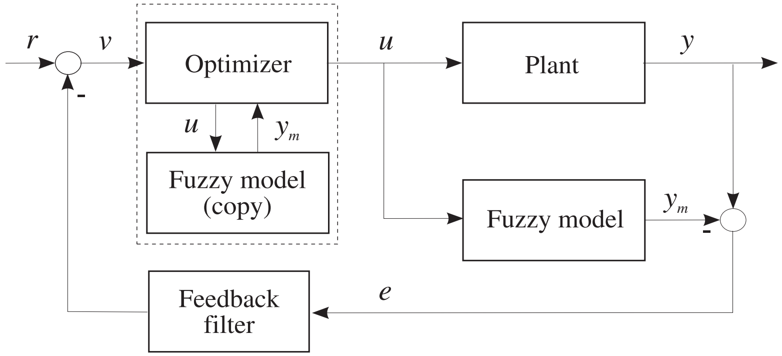
Learning Inverse (Neural) Model



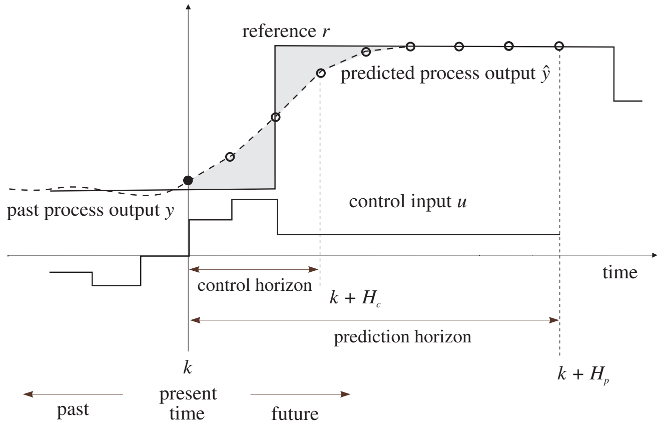
Internal Model Control



Model-Based Predictive Control



Model-Based Predictive Control



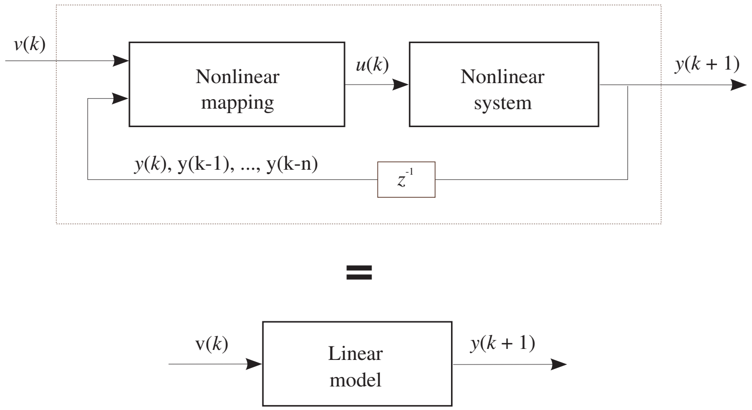
Objective Function and Constraints

$$J = \sum_{i=1}^{H_p} \|(\mathbf{r}(k+i) - \hat{\mathbf{y}}(k+i))\|^2_{P_i} + \sum_{i=1}^{Hc} \|(\mathbf{u}(k+i-1))\|^2_{Q_i}$$

$$\hat{y}(k+1) = f(\hat{\mathbf{x}}(k), u(k))$$

$$\begin{aligned}
\mathbf{u}^{\min} &\leq \mathbf{u} \leq \mathbf{u}^{\max} \\
\Delta\mathbf{u}^{\min} &\leq \Delta\mathbf{u} \leq \Delta\mathbf{u}^{\max} \\
\mathbf{y}^{\min} &\leq \mathbf{y} \leq \mathbf{y}^{\max} \\
\Delta\mathbf{y}^{\min} &\leq \Delta\mathbf{y} \leq \Delta\mathbf{y}^{\max}
\end{aligned}$$

Feedback linearization



Feedback Linearization (continued)

given affine system: $y(k+1) = g(\mathbf{x}(k)) + h(\mathbf{x}(k)) \cdot u(k)$

express $u(k)$:

$$u(k) = \frac{y(k+1) - g(\mathbf{x}(k))}{h(\mathbf{x}(k))}$$

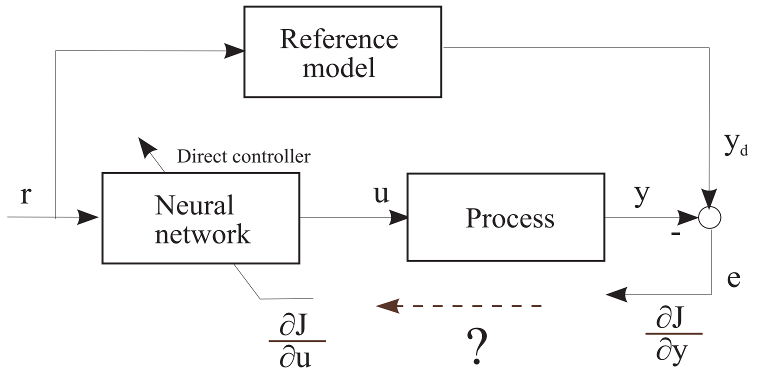
substitute $A(q)y(k) + B(q)v(k)$ **for** $y(k+1)$**:**

$$u(k) = \frac{A(q)y(k) + B(q)v(k) - g(\mathbf{x}(k))}{h(\mathbf{x}(k))}$$

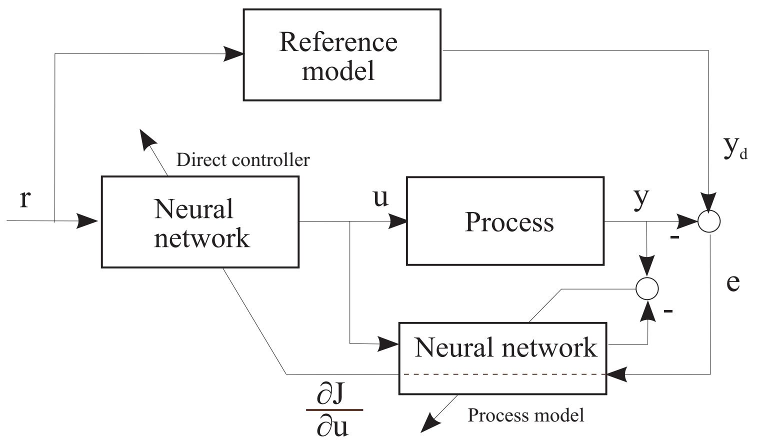
Adaptive Control

- Model-based techniques (use explicit process model):
  - model reference control through backpropagation
  - indirect adaptive control
- Model-free techniques (no explicit model used)
  - reinforcement learning

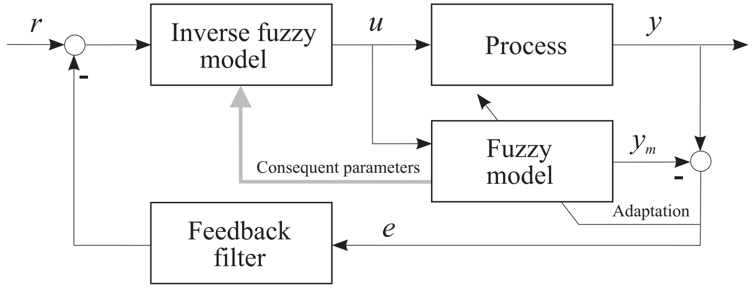
Model Reference Adaptive Neurocontrol



Model Reference Adaptive Neurocontrol

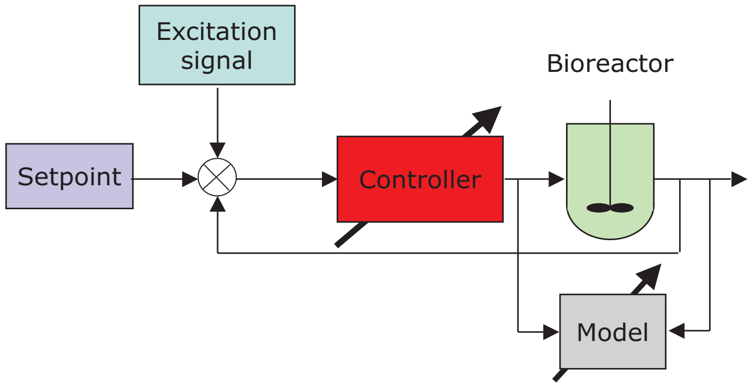


Indirect Adaptive Control

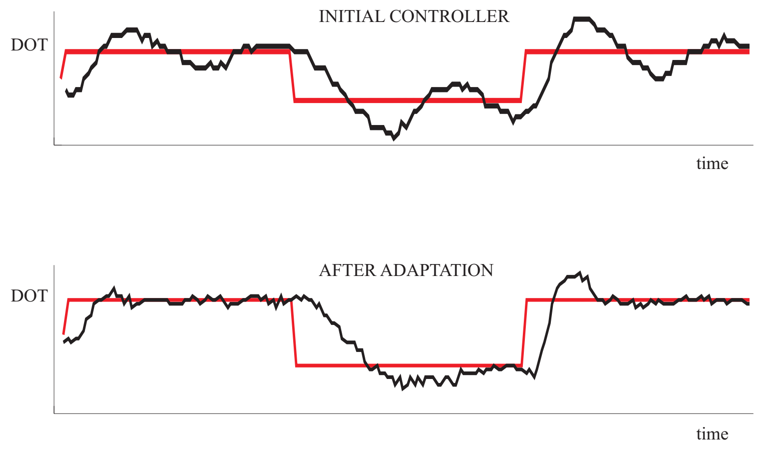


not only for fuzzy models, but also for affine NNs, etc.

Application Example



Typical result



Reinforcement Learning

- Inspired by principles of human and animal learning.
- No explicit model of the process used.
- No detailed feedback, only reward (or punishment).
- A control strategy can be learnt from scratch.

Problem Formulation

- After each control action, reinforcement is received:

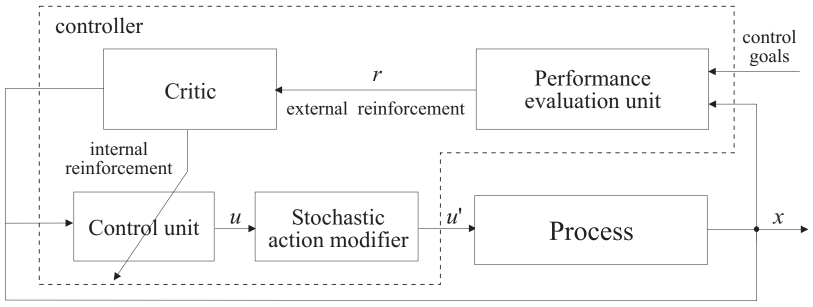
$$r(k) = \begin{cases} 0, & \text{control goal satisfied (reward)} \\ -1, & \text{control goal not satisfied (punishment)} \end{cases}$$

- Learn a control policy that maximizes the reinforcement received:

$$V(k) = \sum_{i=k}^{\infty} \gamma^{i-k} r(i) \quad \text{with} \quad 0 < \gamma < 1$$

- The control policy is adapted by means of exploration, i.e., deliberate modification of the control actions computed by the controller.

Reinforcement Learning Scheme



Process: $\mathbf{x}(k+1) = f(\mathbf{x}(k), u(k))$

Critic and Its Adaptation

Learns to predict the value function $V(k+1)$ for $\mathbf{x}(k)$ and $u(k)$:

$$\hat{V}(k+1) = h(\mathbf{x}(k), u(k); \theta(k))$$

Update law for θ:

$$\theta(k+1) = \theta(k) + a_h \frac{\partial h}{\partial \theta}(k)\Delta(k)$$

with the error: $\Delta(k) = V(k) - \hat{V}(k)$

Computing $\triangle$ (Temporal Difference)

$\Delta(k) = V(k) - \hat{V}(k)$, $\quad V(k)$ unknown! $\rightarrow$ approximate it

Value function:

$$V(k) = \sum_{i=k}^{\infty} \gamma^{i-k} r(i) = r(k) + \gamma V(k+1)$$

$$\Delta(k) = V(k) - \hat{V}(k) = r(k) + \gamma \hat{V}(k+1) - \hat{V}(k)$$

$\Delta(k)$ is called temporal difference (internal reinforcement)

Controller and Its Adaptation

Nonlinear state feedback:

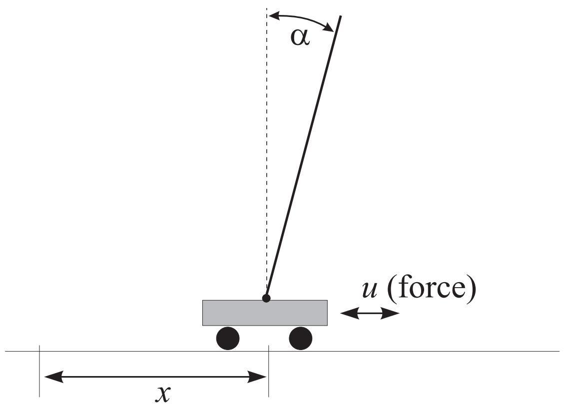
$$u(k) = g(\mathbf{x}(k); \varphi(k))$$

Update law for φ:

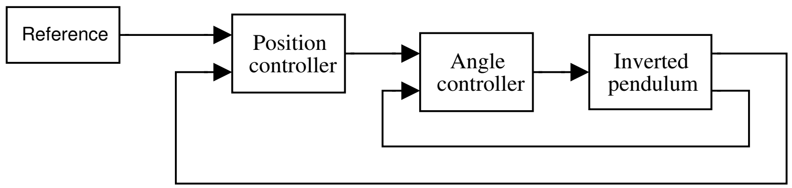
$$\varphi(k+1) = \varphi(k) + a_g \frac{\partial g}{\partial \varphi}(k)[u'(k) - u(k)]\Delta(k)$$

where u' is the (randomly) modified control action

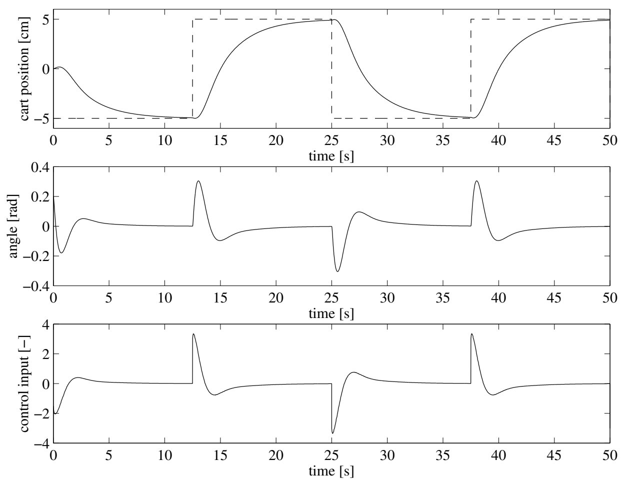
Example: Inverted Pendulum



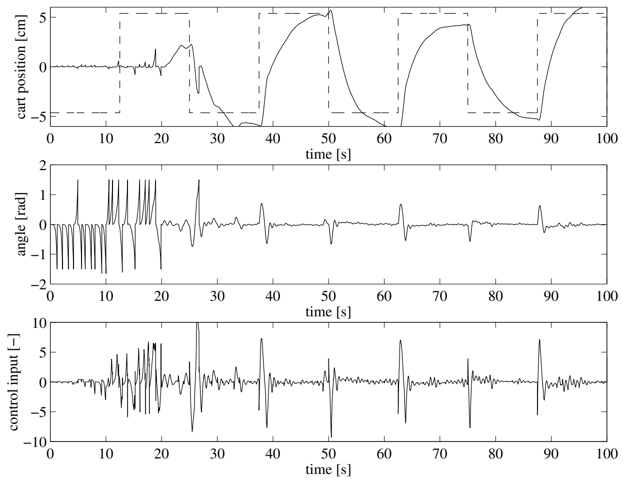
Cascade Control Scheme



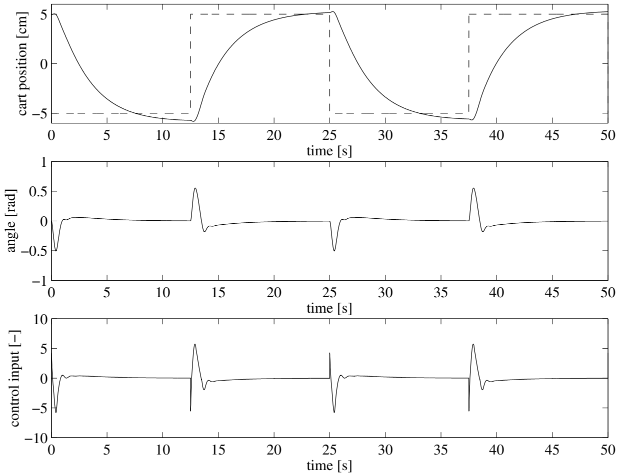
PD Control



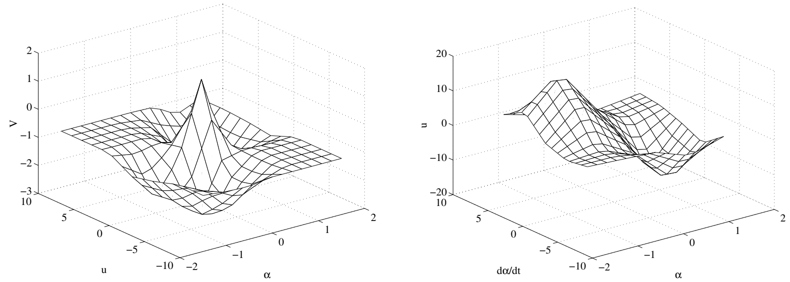
Reinforcement Learning



Reinforcement Learning: Final Performance



Critic and Controller Surfaces



(a) Critic

(b) Controller

Concluding Remarks

- Intelligent control – umbrella for various techniques
  - knowledge based fuzzy control
  - control based on fuzzy and neural models
  - use of genetic optimization in control
- Approaching true intelligence?
  - well, not really, but
  - useful tools that allow for the use of extra information
  - new concepts vs. mathematical rigor