HUMAN MOTOR CONTROL

Emmanuel Guigon

Institut des Systèmes Intelligents et de Robotique
Sorbonne Université
CNRS / UMR 7222
Paris, France

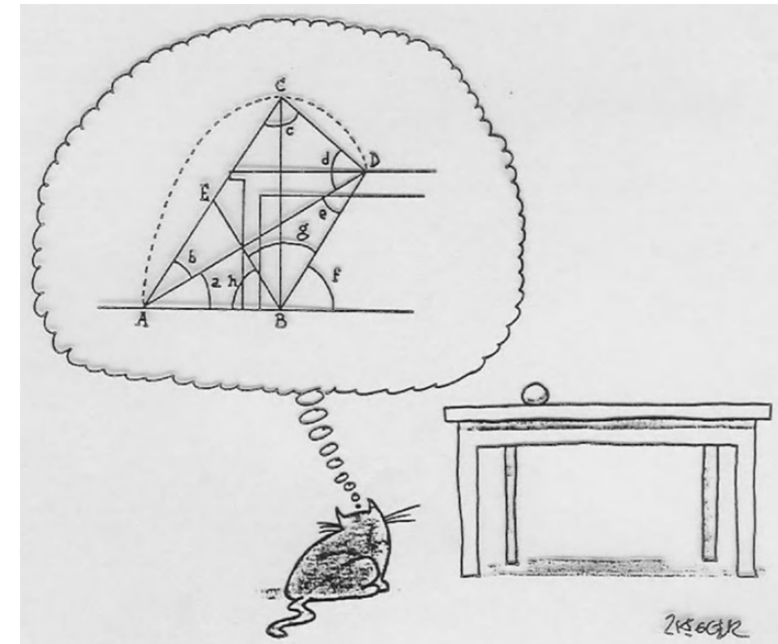
emmanuel.guigon@sorbonne-universite.fr

2

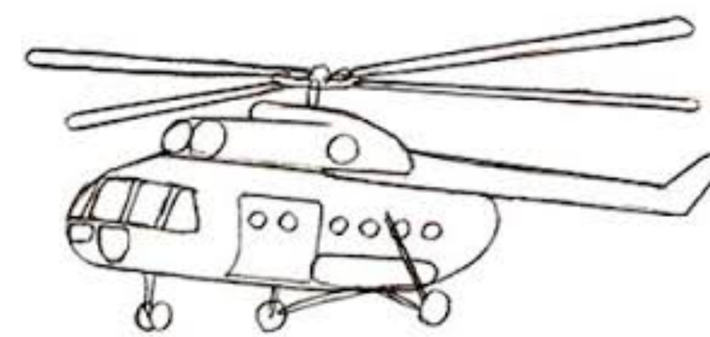
2. Computational motor control

LEVELS OF ANALYSIS

- **Computational**
description (mathematical) of a function that a system is supposed to achieve
explicit vs *implicit*



- **Algorithmic (procedural)**
how the computational problem can be solve



- **Implementation**
the physical substrate or mechanism, and its organisation, in which computation is performed

— Marr, 1982, *Vision*, Freeman
— Rosenbaum, 2009, *Human Motor Control*, Academic Press

DESCRIPTIVE VS NORMATIVE

Descriptive (mechanistic) vs normative models

- Descriptive statements present an account of how the world is

Action characteristics result from properties of synapses, neurons, neural networks, muscles, …

- Normative statements present an evaluative account, or an account of how the world should be

Action characteristics result from principles, overarching goals, …

THEORETICAL BASES

• Dynamical systems theory

Describes the behavior in space and time of complex, coupled systems

$x[n]$ *state* $y[n]$ *output (observation)*

$u[n]$ *input (control)*

$x[n+1] = f(x[n], u[n])$ *state equation*

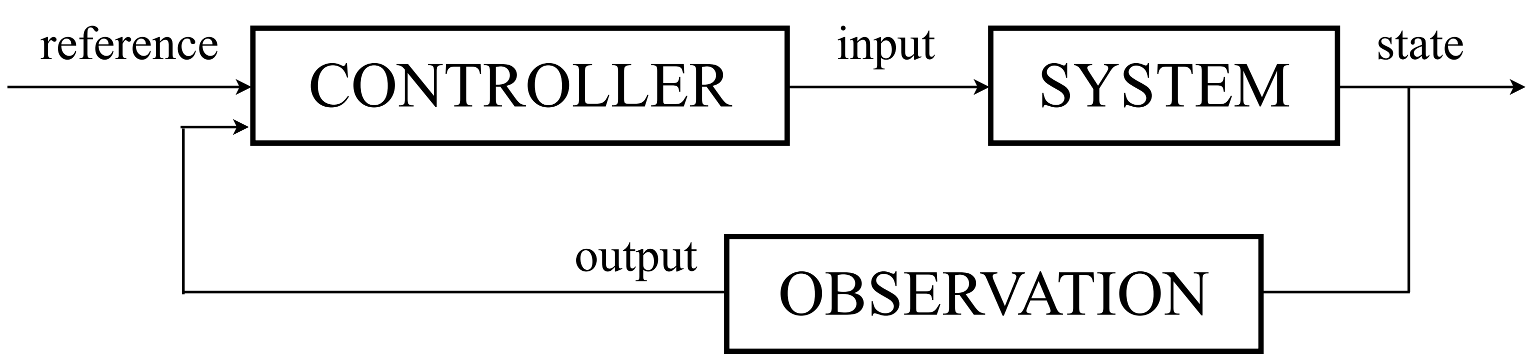
$y[n] = g(x[n])$ *output equation*

$y[n+1] = h(x[n], u[n])$

state: « *the smallest possible subset of system variables that can represent the entire state of the system at any given time* »

• Control theory

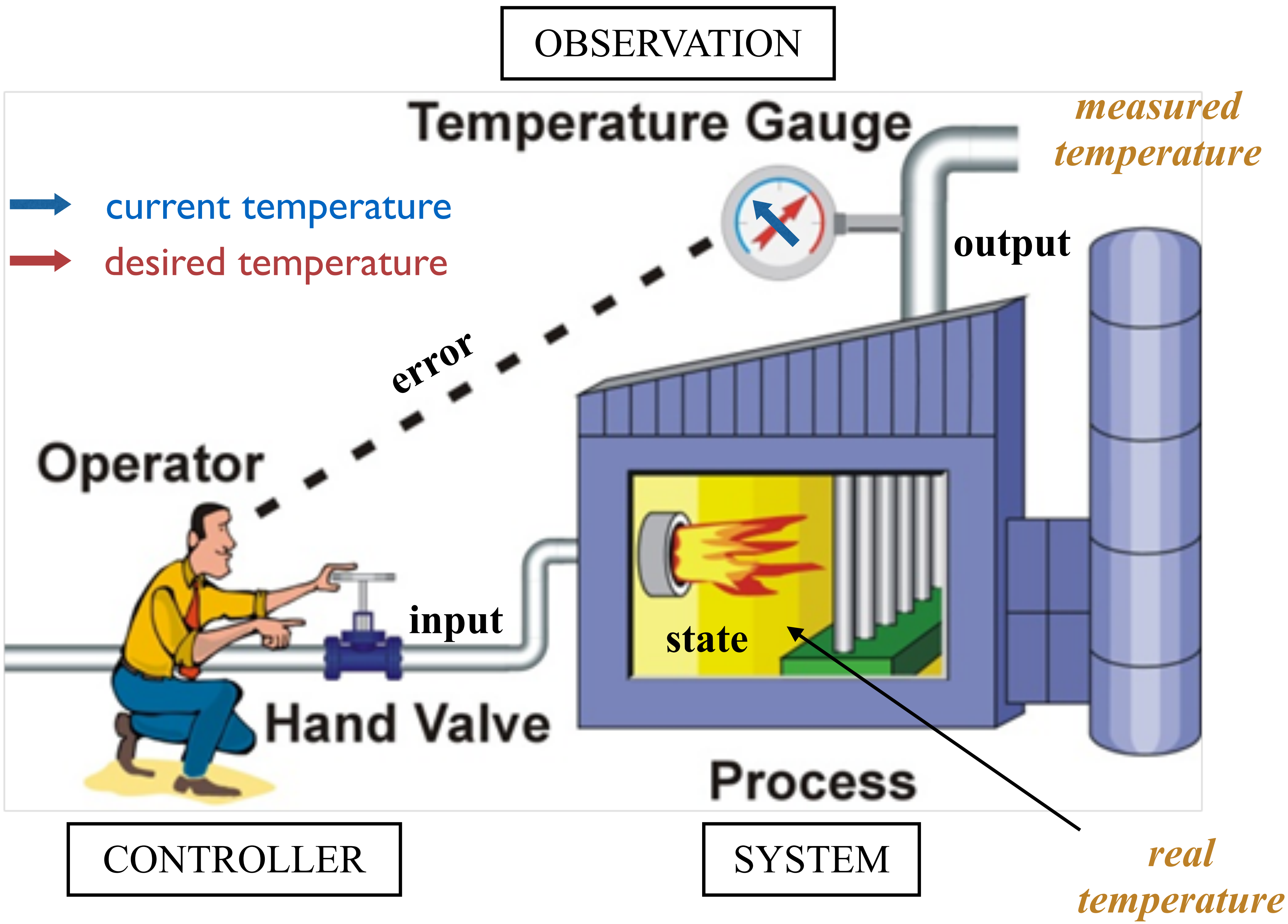
Deals with the behavior of dynamical systems with inputs, and how their behavior is modified by feedback



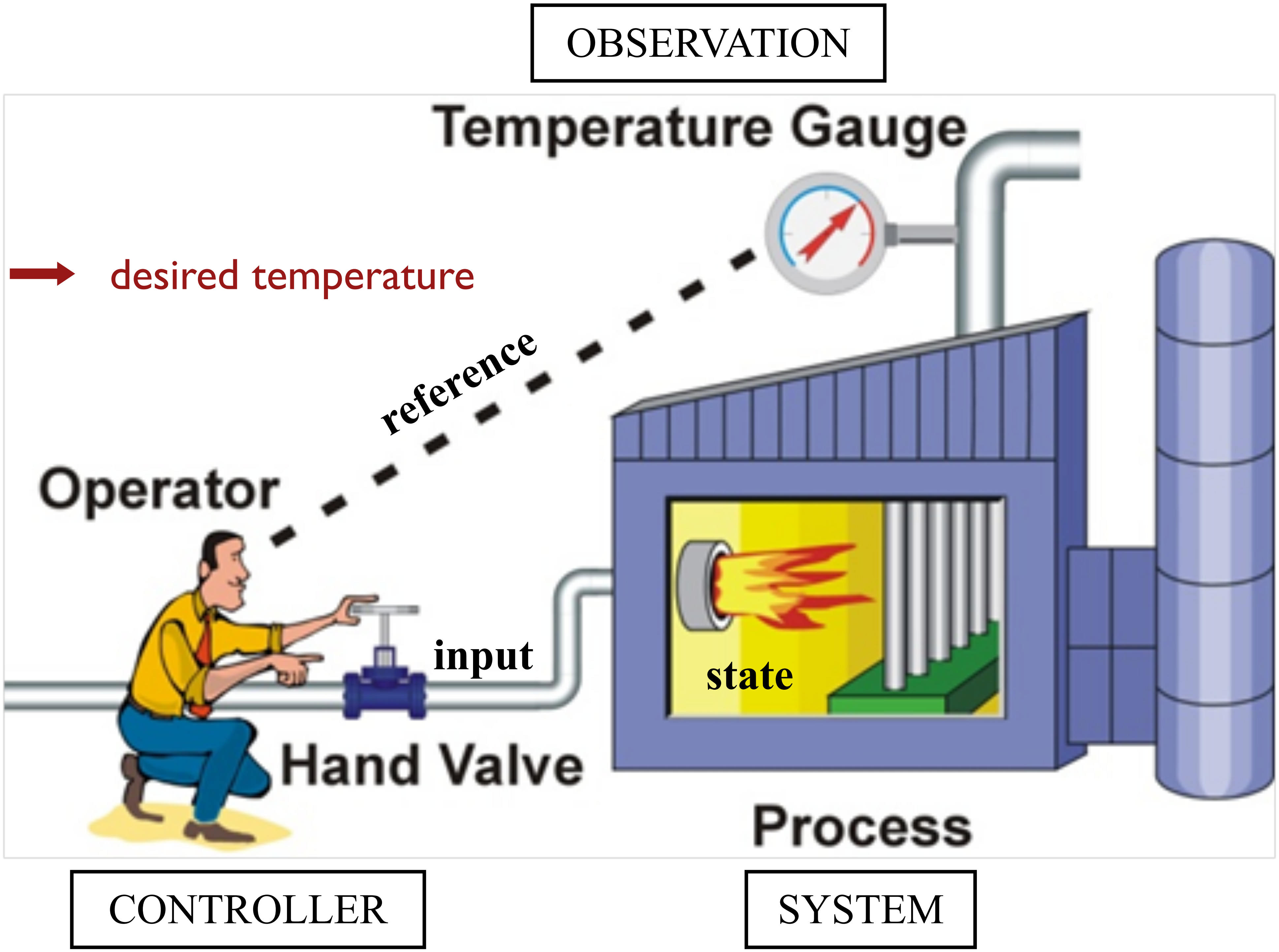
reference

- desired trajectory
- fixed point

TWO CONTROL PRINCIPLES
CLOSED LOOP



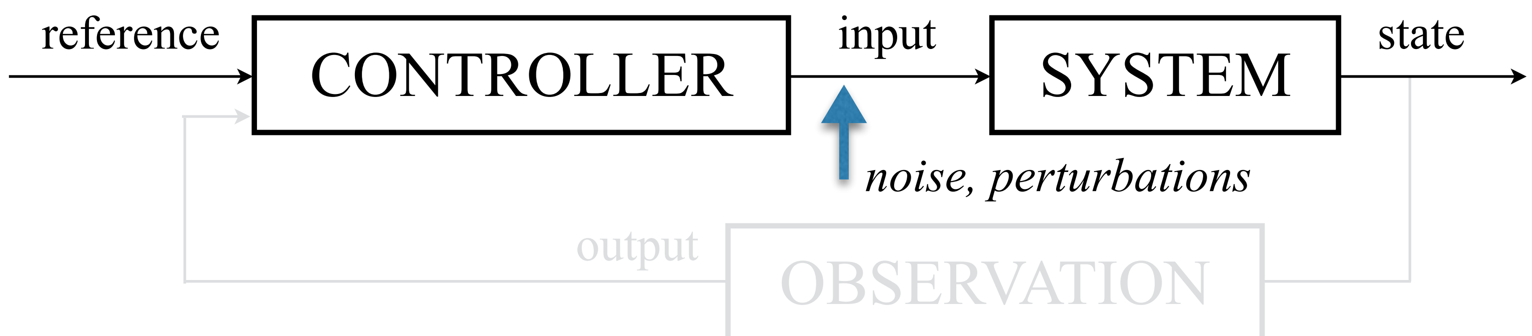
TWO CONTROL PRINCIPLES OPEN LOOP



TWO CONTROL PRINCIPLES

- **Open-loop (feedforward)**

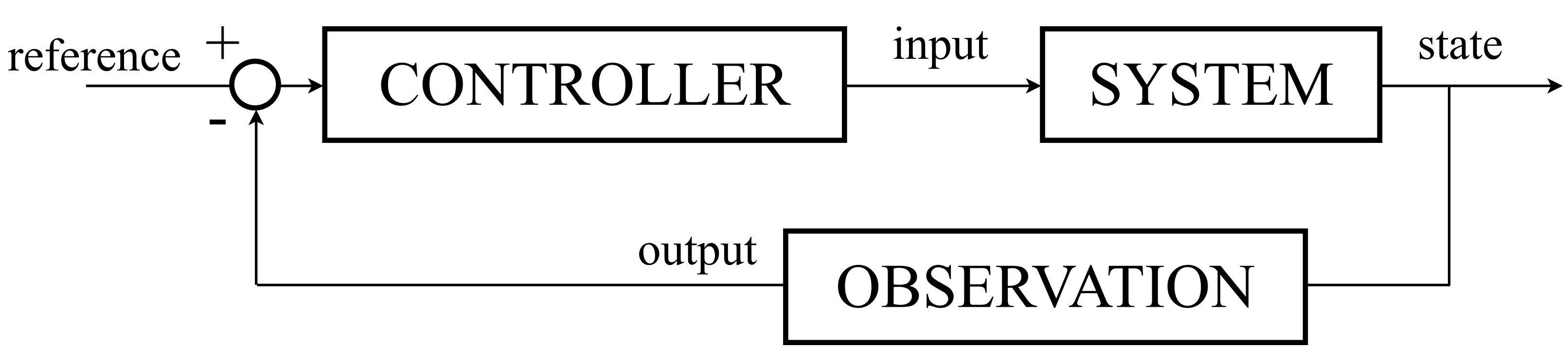
The controller is an ***inverse*** model of the system



- Predictive control
- Model-based
- Sensitive to modeling uncertainty
- Sensitive to unexpected, unmodeled perturbations

- **Closed-loop (feedback)**

The controller is a function of an error signal



- Error correction
- No model
- Not sensitive to modeling uncertainty
- Robust to perturbations

EQUATIONS

- **Open-loop (feedforward)**

The controller is an ***inverse*** model of the system

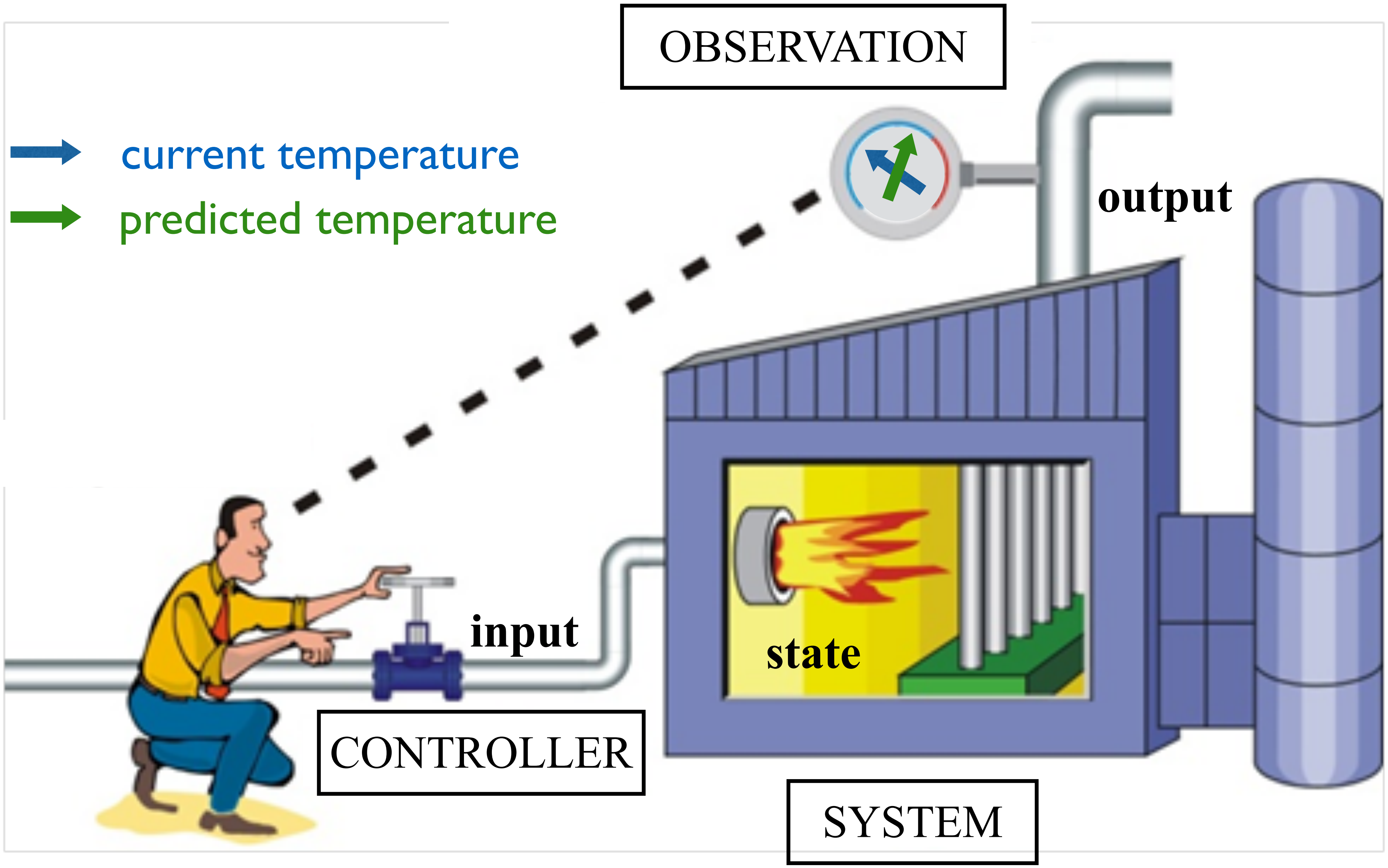
$$y[n+1] = h(x[n], u[n])$$
$$u_{ff}[n] = \phi(x[0], y^*[n+1])$$
$$\phi \approx h^{-1}$$
$$y^* \quad \text{reference}$$

- **Closed-loop (feedback)**

The controller is a function of an error signal

$$u_{fb}[n] = K(y^*[n] - y[n])$$
$$K \quad \text{constant gain}$$

FORWARD MODEL

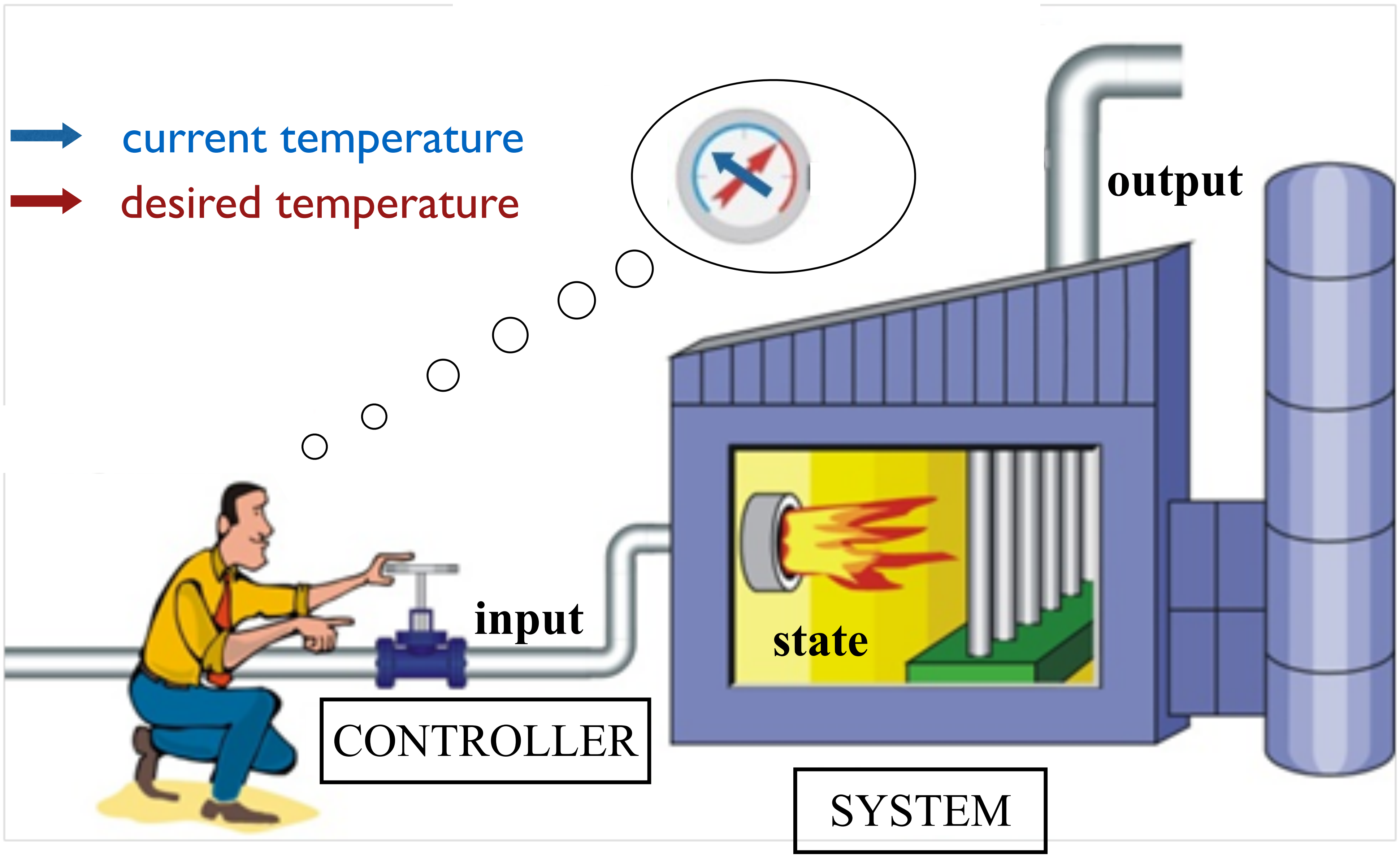


Model of the causal relationship between inputs and their consequences (states, outputs)

input → predicted output

input → predicted state

INVERSE MODEL



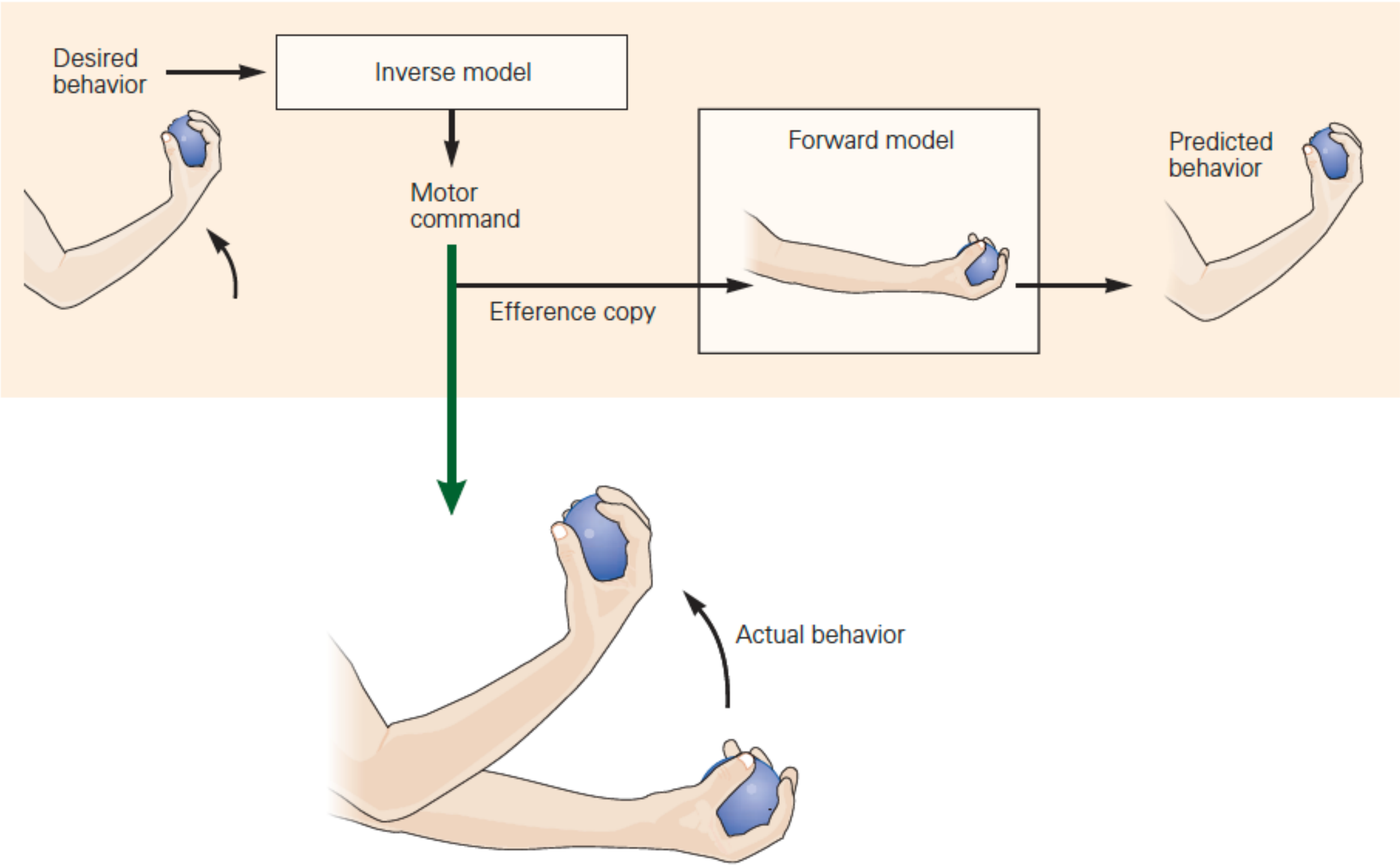
Model of the relationship between desired consequences (outputs, states) and corresponding inputs

desired state → input

desired output → input

FORWARD AND INVERSE MODEL

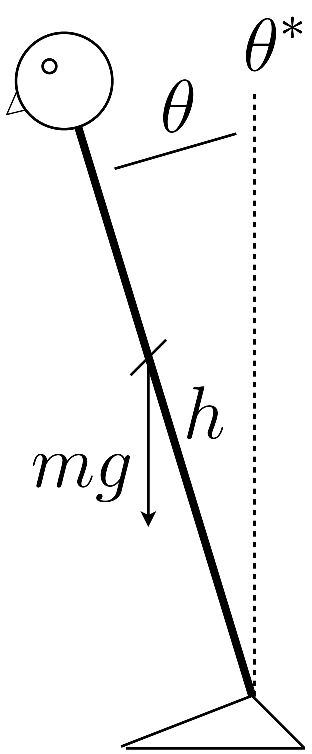
For motor control



EXAMPLE I

Inverted pendulum

maintain the pendulum to a reference position
classical feedback control (PID controller)



control policy

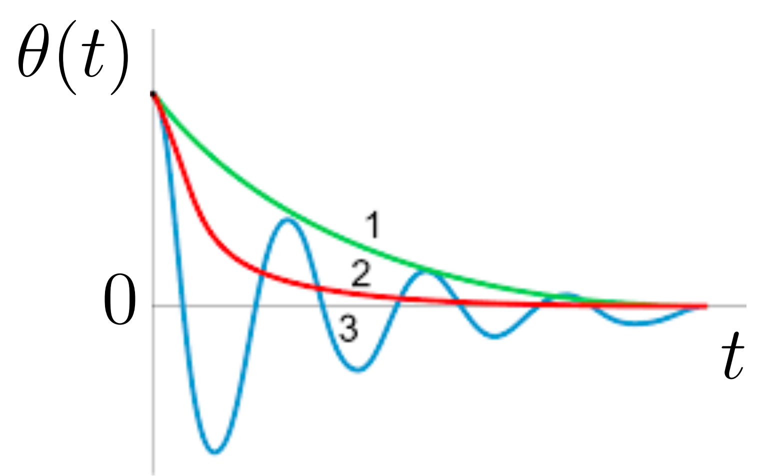
$$I\ddot{\theta}(t) = mgh\theta(t) + u(t)$$

$$u(t) = K_P(\theta^* - \theta(t))$$ proportional

$$- K_D\dot{\theta}(t)$$ derivative

$$+ K_I \int_{t_0}^{t} (\theta^*(\tau) - \theta(\tau))\, d\tau$$ integral

$$K_P > mgh$$



note: the controller has no knowledge of the system to be controlled (e.g. mass, height) — the policy of the PD controller depends only on state and not explicitly on time

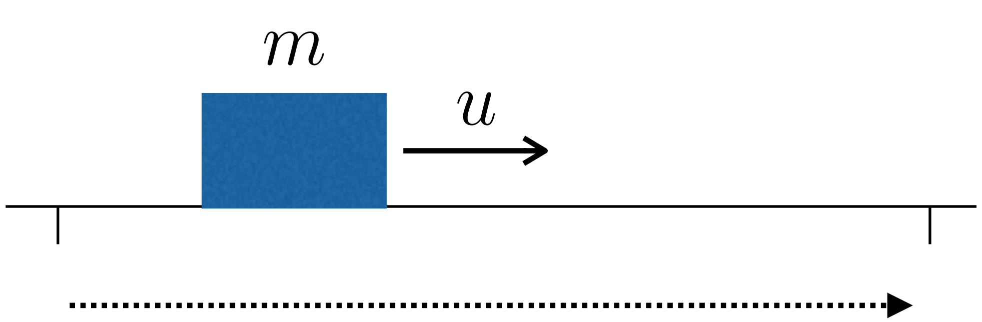
EXAMPLE II

Mass point

— displace the mass along a given trajectory

— inverse controller

$$m\ddot{x}(t) = u(t)$$



desired trajectory $x^*(t)$

control policy

$$u(t) = \hat{m}\ddot{x}^*(t)$$

$\hat{m}$ estimated mass

note: the controller has a (approximate) knowledge of the system to be controlled (mass) – the policy of the inverse controller depends explicitly on time

INTERNAL MODELS AND CAUSALITY

Forward (direct) model

- model of the causal relationship between inputs (actions) and outputs (consequences)
- choice of input and output variables

e.g. input = muscular activation - output = joint torque

e.g. input = joint torque - output = displacement

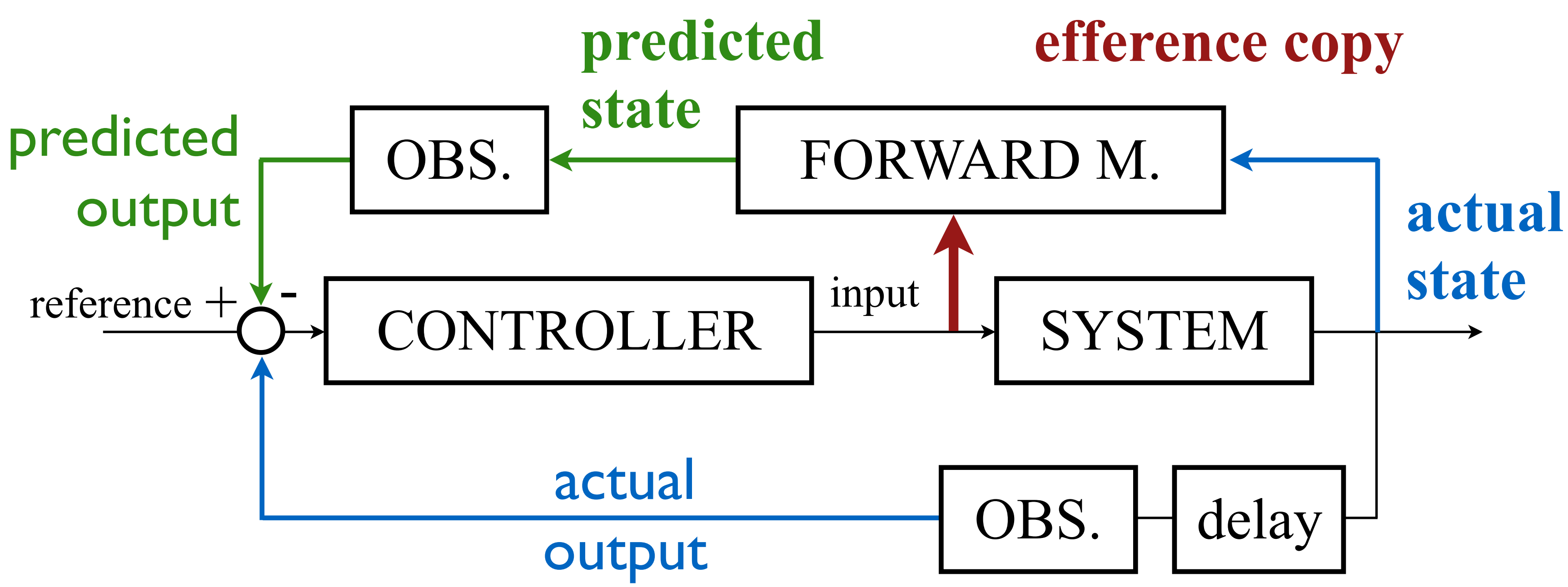
Inverse model

- model of the relationship between outputs (desired consequences) and inputs (actions)
- causality is extended to functional relationships between variables
- in general, not a function (redundancy)

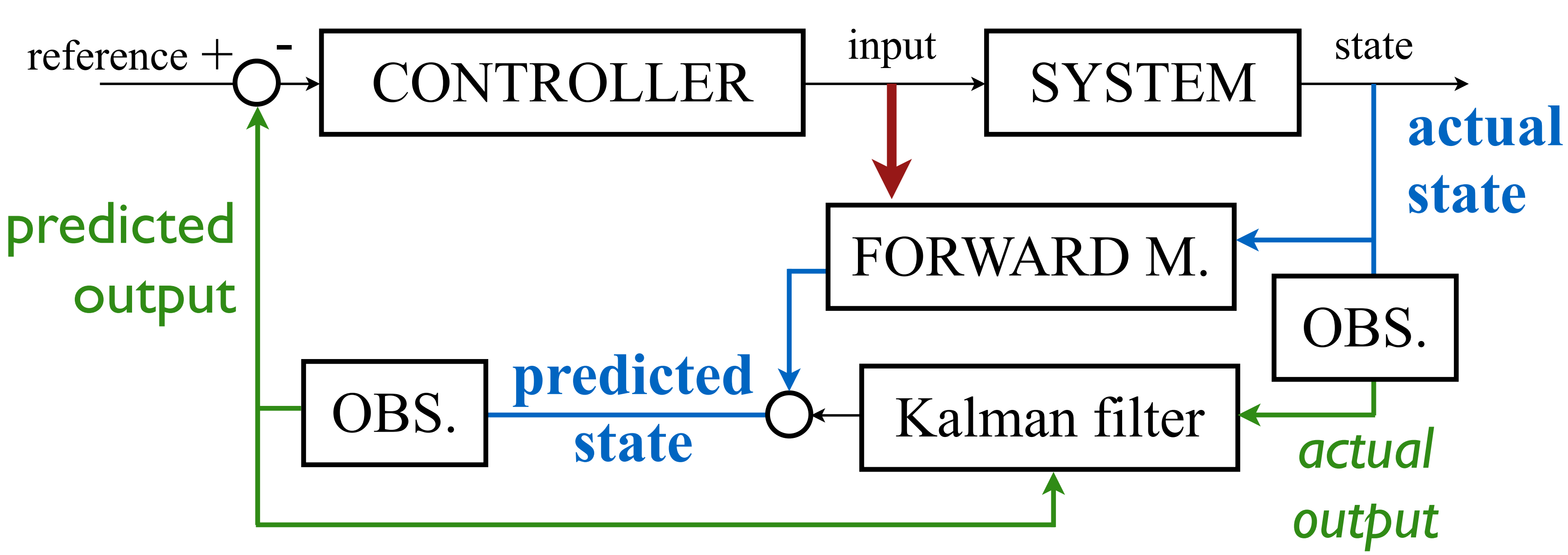
e.g. inverse kinematics (spatial coordinates to joint coordinates)

ROLE OF FORWARD MODELS

Fast compensation for delay



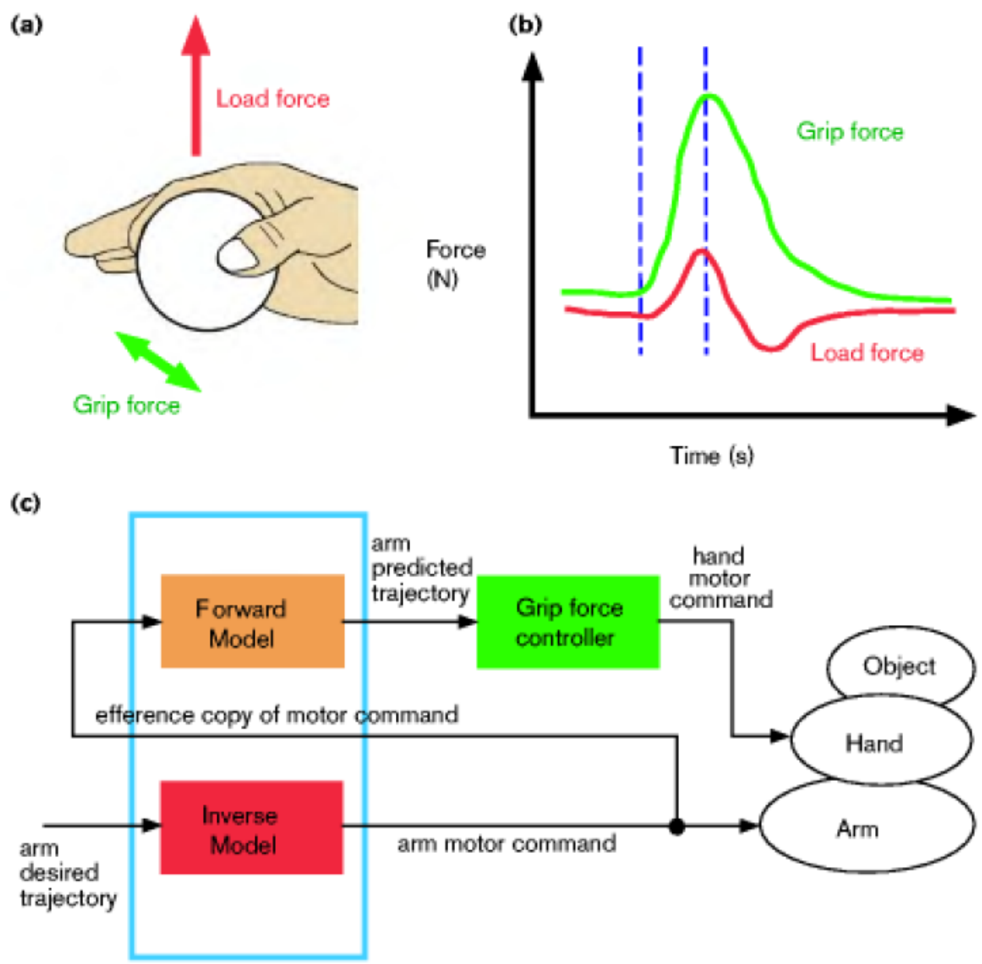
Compensation for uncertainty: state estimator

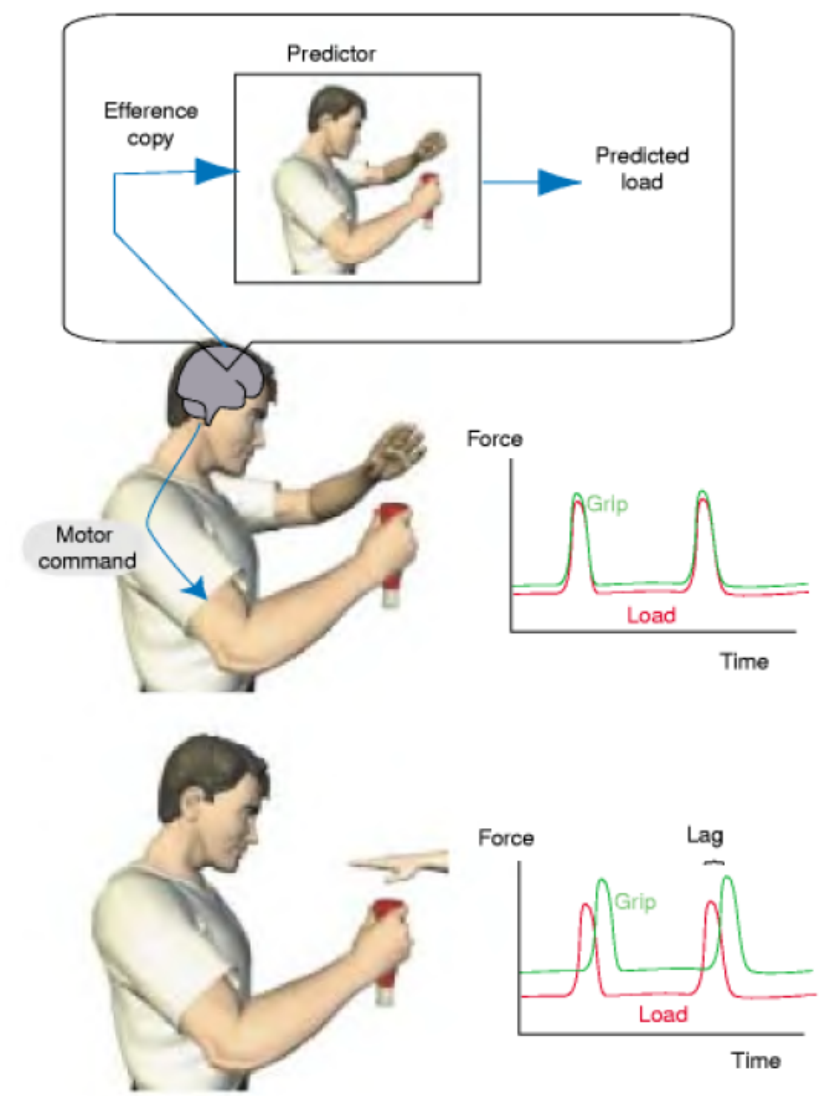


EXISTENCE OF FORWARD MODELS

Grip/load force

to prevent a manipulated object to slip during movement, a grip force must be exerted to compensate for the load force





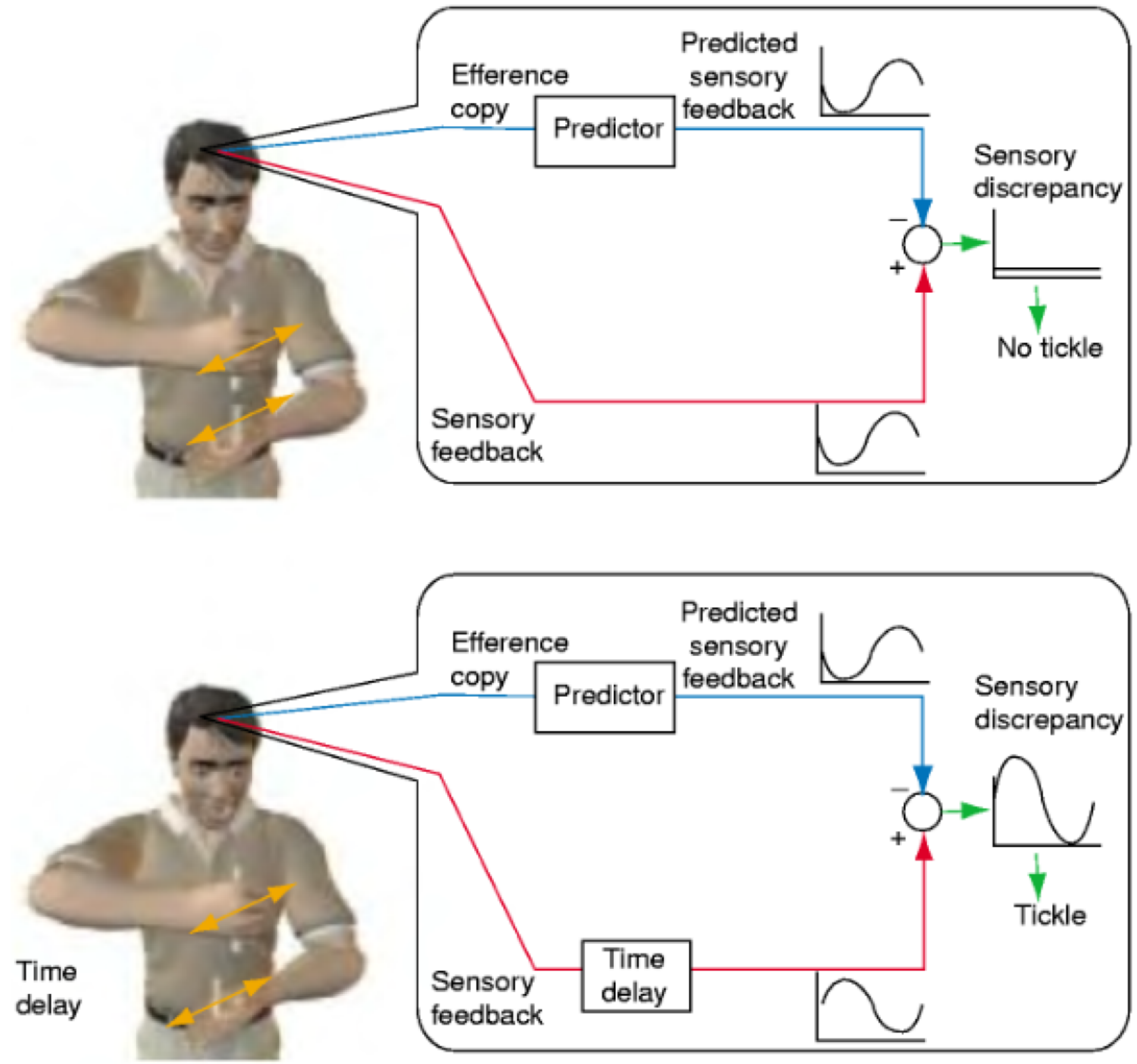
— Kawato, 1999, *Curr Opin Neurobiol* 9:718

— Wolpert & Flanagan, 2001, *Curr Biol* 11:R729

EXISTENCE OF FORWARD MODELS

Tickling

a subject creates a tactile stimulation on one hand through a robotic device actuated by the other hand. When the transmission is direct, the subject can subtract the predicted sensory effect from the actual sensory effect due to the tactile stimulation. The subject perceives no tickling.



efference copy − sensory feedback = corollary discharge

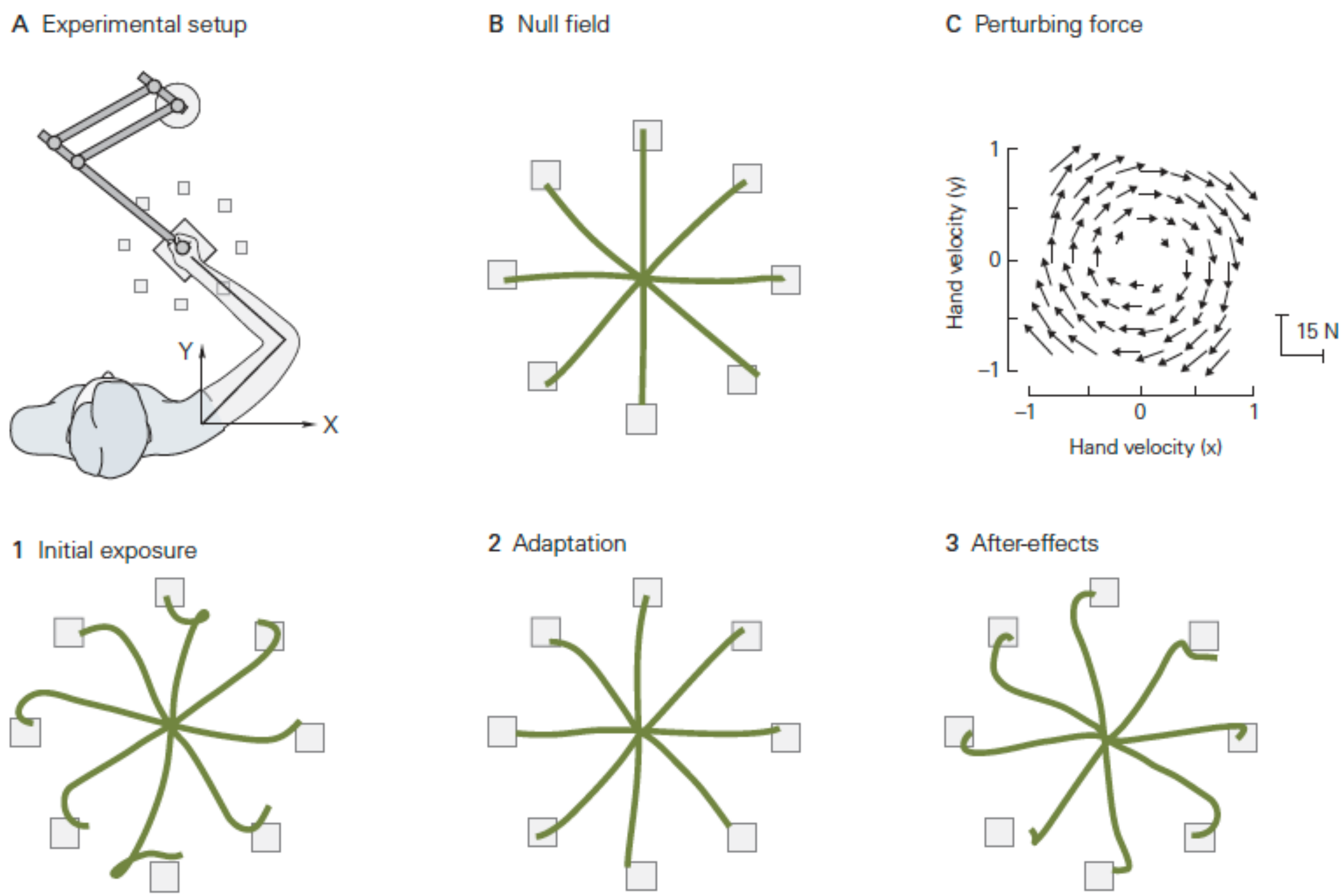
— Blakemore et al., 2000, *NeuroReport* 11:R11

— Wolpert & Flanagan, 2001, *Curr Biol* 11:R729

EXISTENCE OF INVERSE MODELS

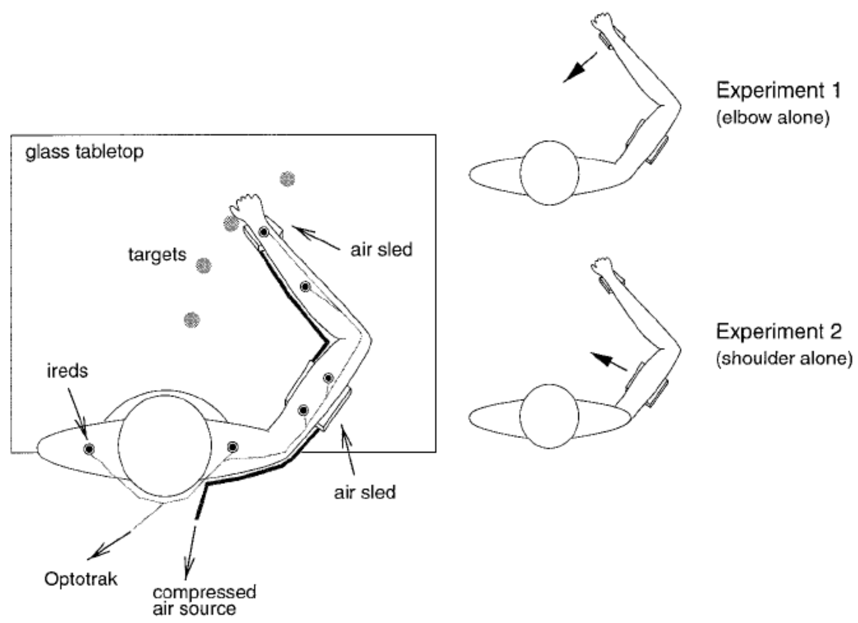
Learning state-dependent dynamic perturbations

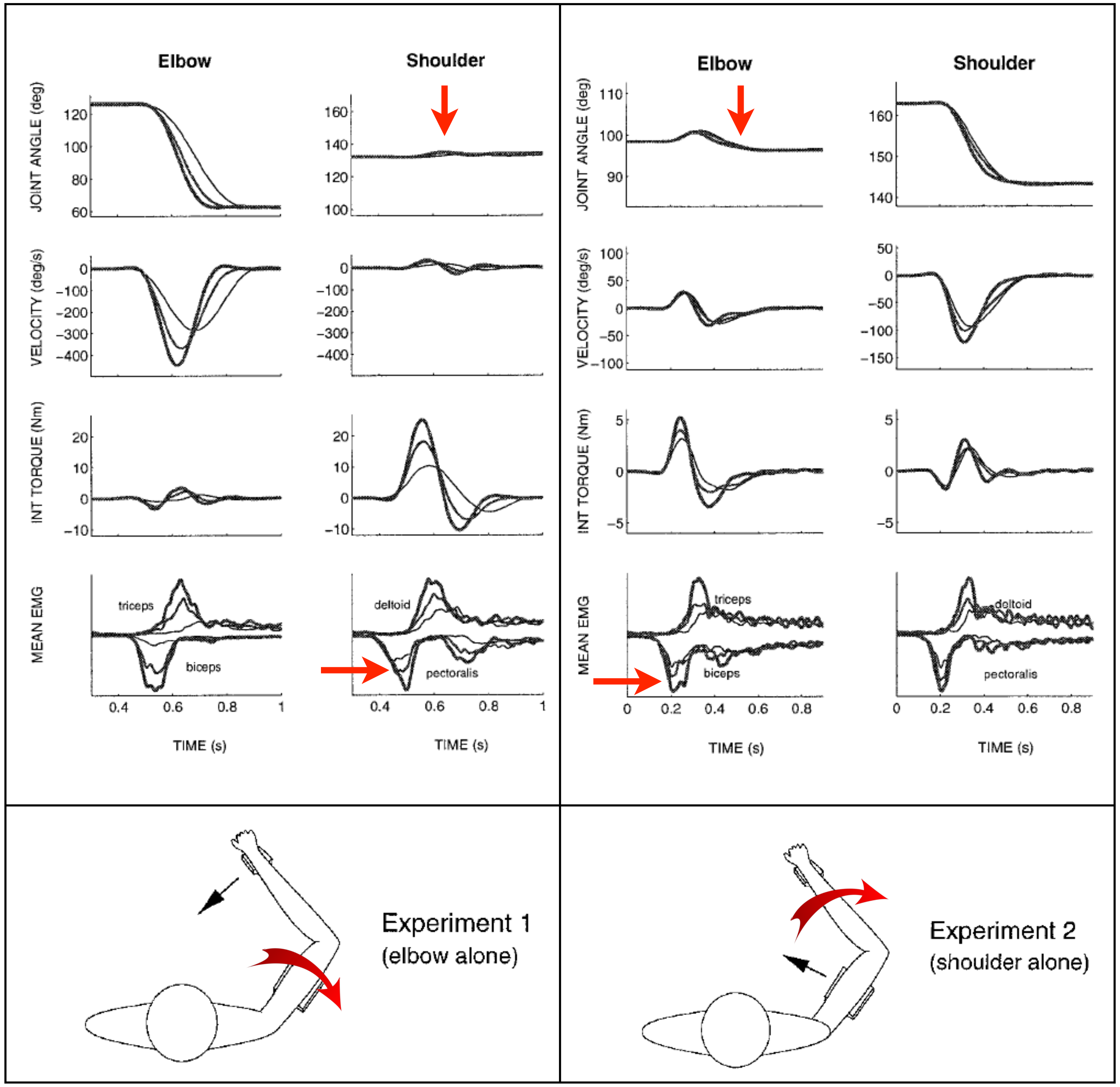
velocity-dependent force field



— Shadmehr & Mussa-Ivaldi, 1994, *J Neurosci* 14:3208

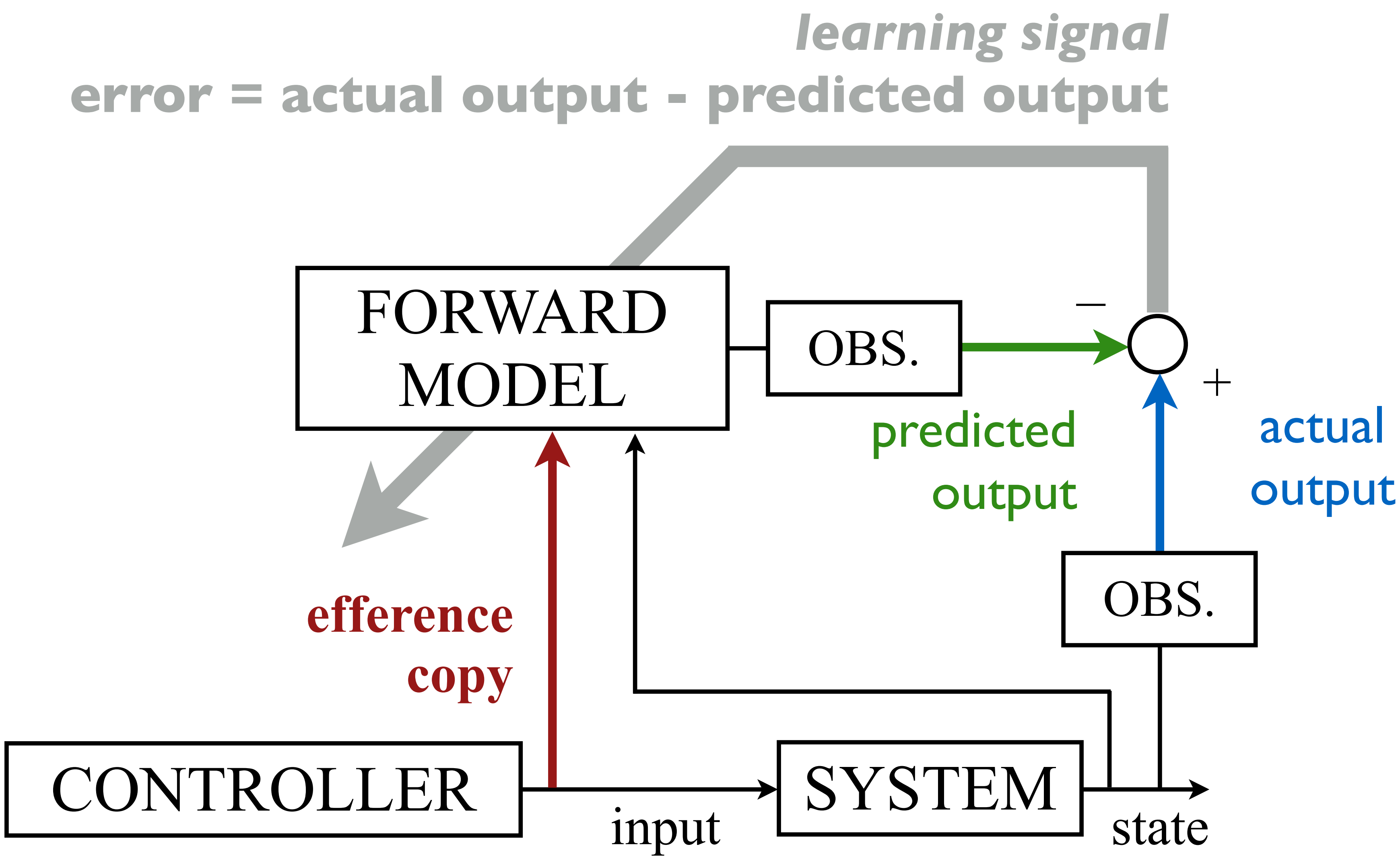
EXISTENCE OF INVERSE MODELS





— Gribble & Ostry, 1999, *J Neurophysiol* 82:2310

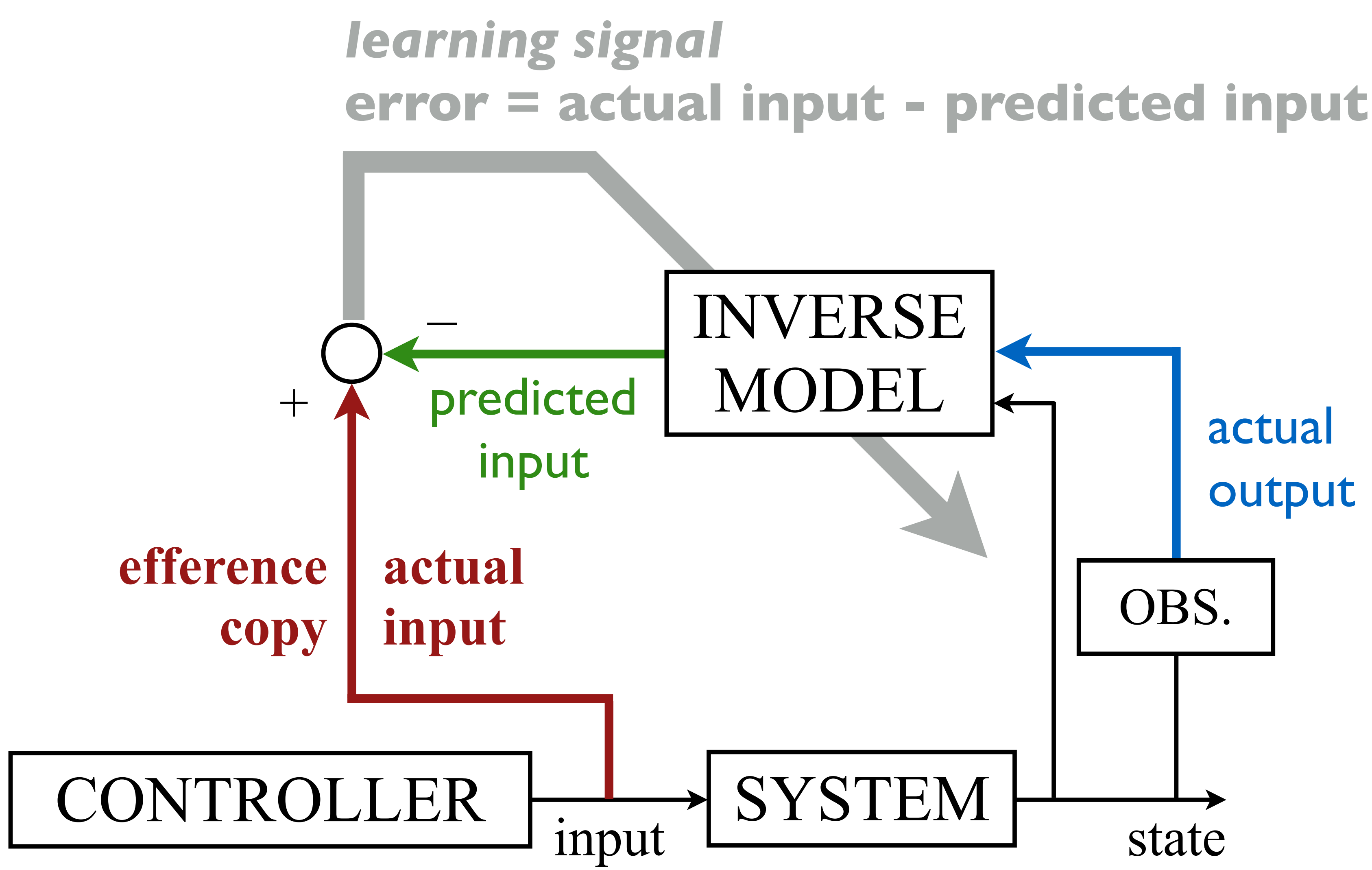
BUILDING A FORWARD MODEL



BUILDING AN INVERSE MODEL (I)

Direct inverse learning

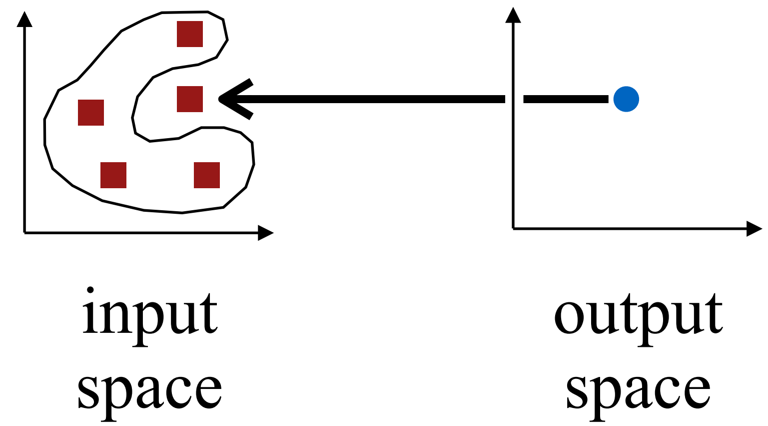
a transformation is learned by sampling the inverse transformation

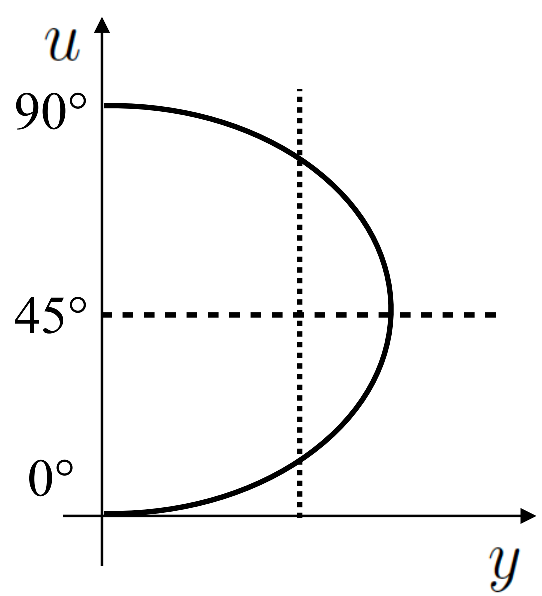


BUILDING AN INVERSE MODEL (II)

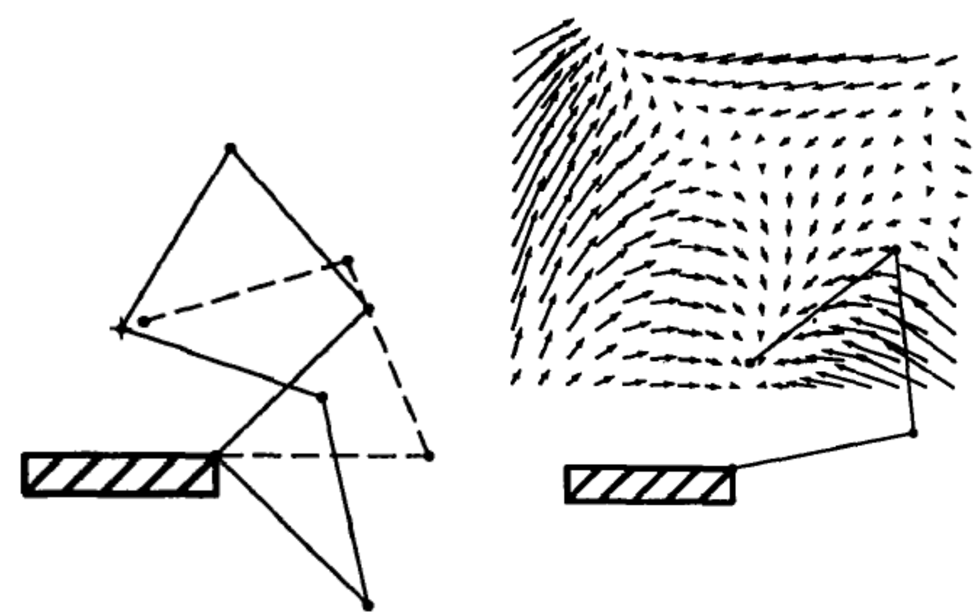
Direct inverse learning

counterexample (convexity problem)






```
the system
converges to an
incorrect
controller that
maps each
target distance
to the same 45°
control signal
```



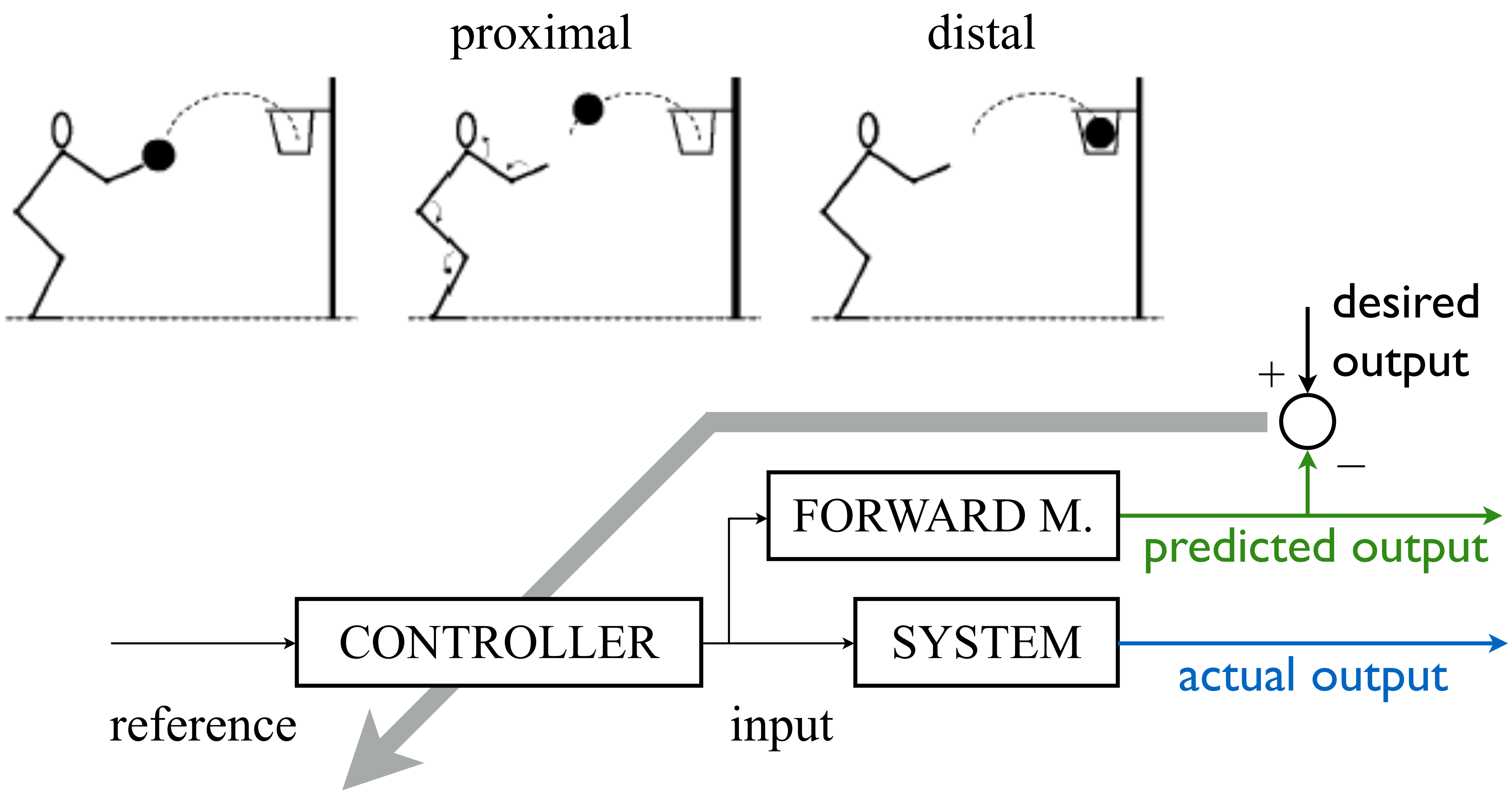
— Jordan, 1995, in *The Cognitive Neurosciences*, MIT Press

— Jordan & Rumelhart, 1992, *Cogn Sci* 16:307

BUILDING AN INVERSE MODEL (III)

Distal supervised learning

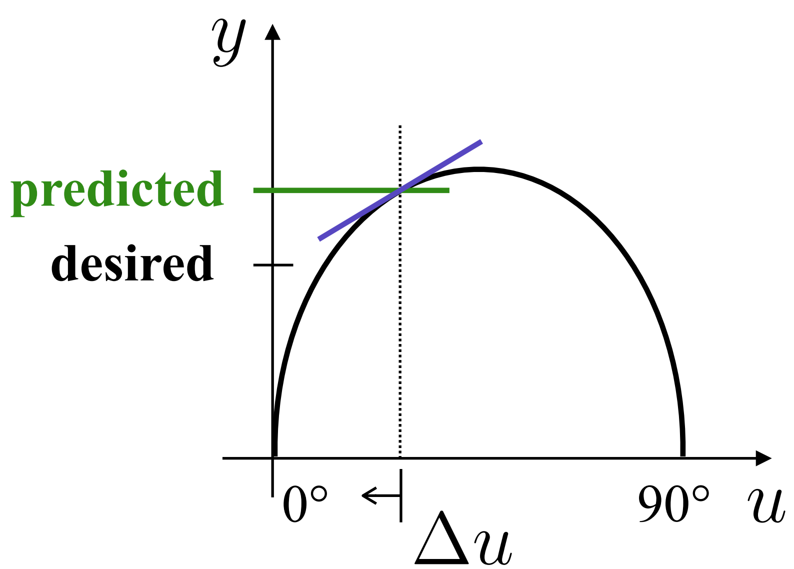
translation of performance error in distal space (difference between desired and predicted output) into an error in proximal space

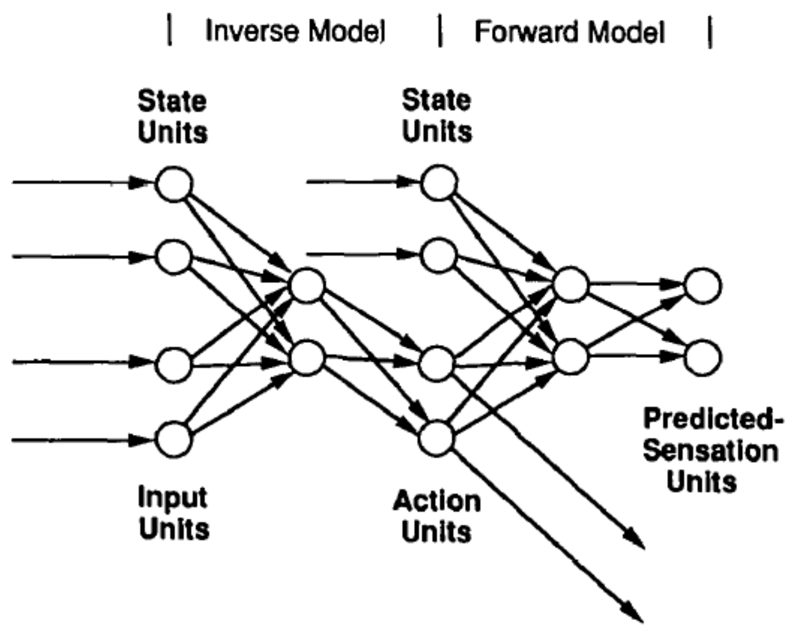


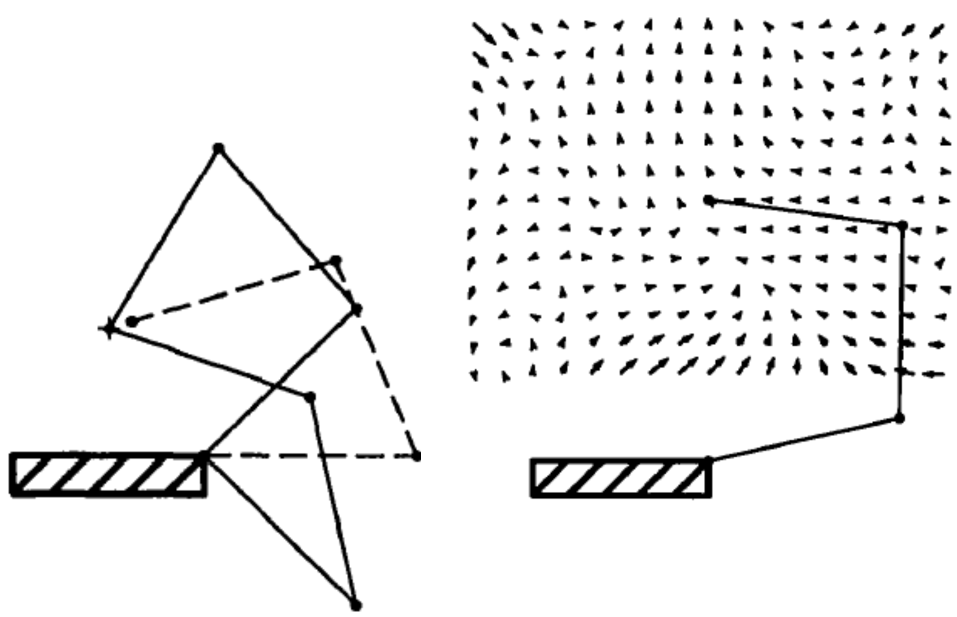
BUILDING AN INVERSE MODEL (IV)

Distal supervised learning

multilayer neural network optimization





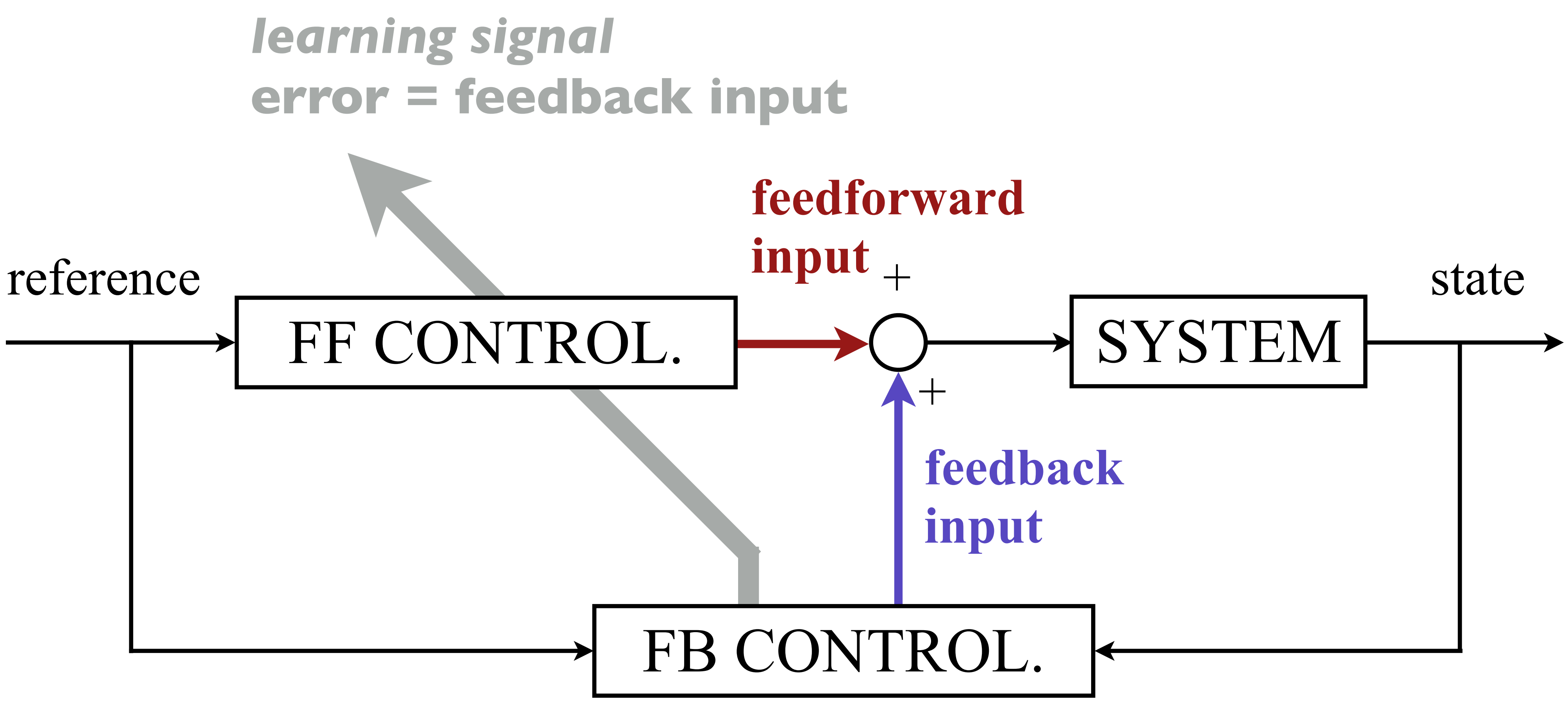


```
the nonconvexity of the problem
does not prevent the system from
converging to a unique solution;
the system simply heads downhill
two one solution or the other
```

BUILDING AN INVERSE MODEL (V)

Feedback-error learning

the feedback input becomes null when there is no more error (perfect feedforward controller)



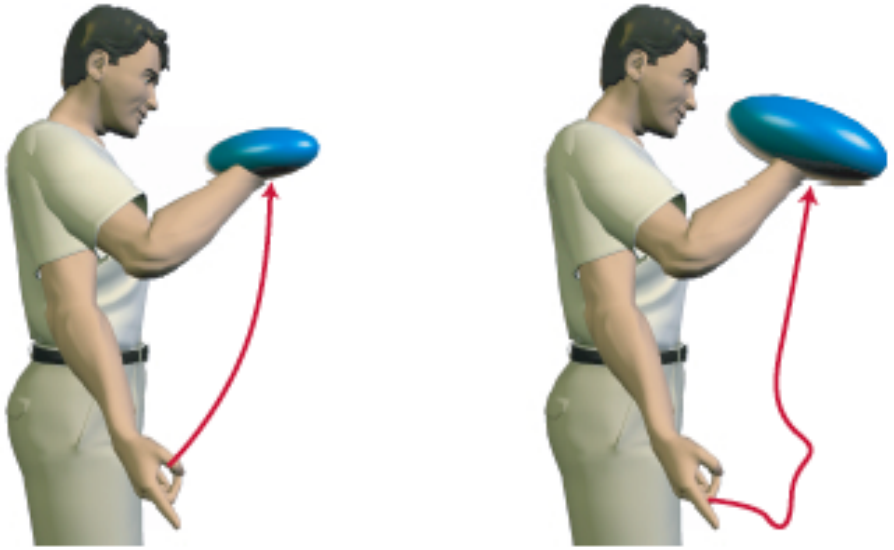
OPTIMALITY PRINCIPLE

Principle

- the interaction between the behavior and the environment leads a better adaptation of the former to the latter. The tendency could lead to an optimal behavior, i.e. the best behavior corresponding to a goal, according to a given criterion.

- the idea is to describe a movement not in terms of its apparent characteristics (kinematics, dynamics), but in an abstract way, using a global value to be maximized or minimized.

e.g. smoothness, energy, variability, ...



EXAMPLE

Minimum-jerk trajectory

finding among all one-dimensional trajectories of given amplitude and duration the one that minimizes the overall derivative of acceleration (jerk)

$$\text{Find } x(t), t \in [t_0, t_f]$$

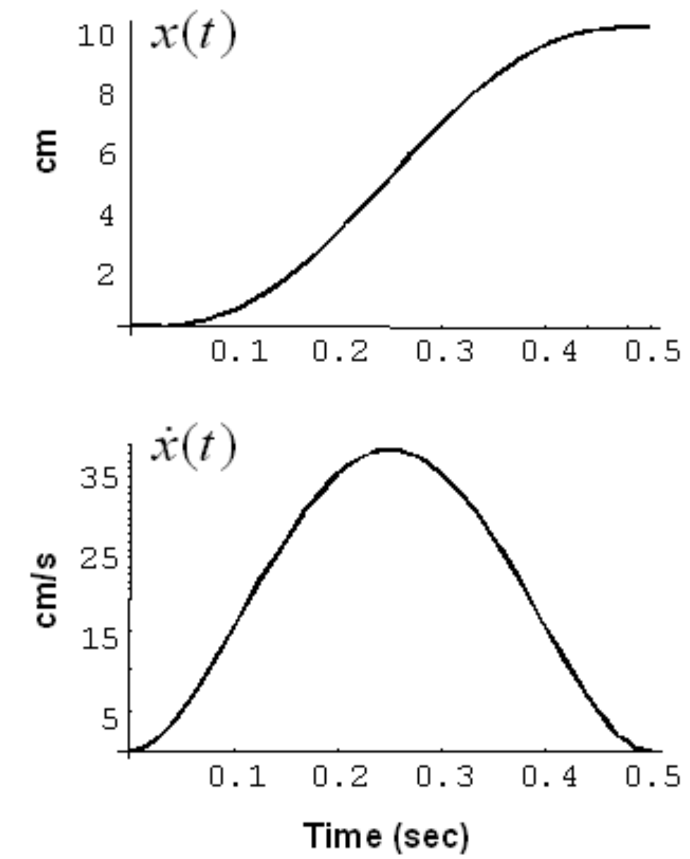
such that

$$\int_{t_0}^{t_f} \dddot{x}^2(t)\, dt \quad \text{is minimum}$$

$$x(t_0) = x_0, x(t_f) = x_f$$

$$\dot{x}(t_0) = v_0, \dot{x}(t_f) = v_f$$

$$\ddot{x}(t_0) = a_0, \ddot{x}(t_f) = a_f$$



$$\Rightarrow \quad x(t) = \alpha_0 + \alpha_1 t + \alpha_2 t^2 + \alpha_3 t^3 + \alpha_4 t^4 + \alpha_5 t^5$$

OPTIMAL CONTROL

- **Minimum-cost trajectory**

Find $\mathbf{u}(t), t \in [t_0, t_f]$

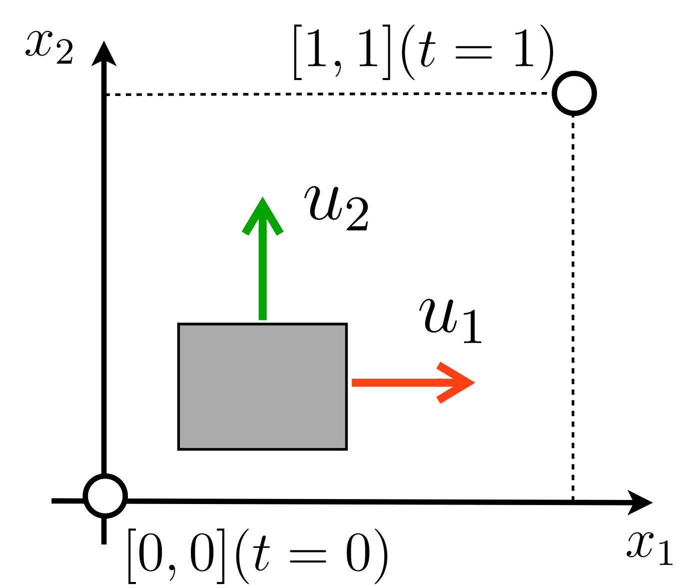
such that

$$\int_{t_0}^{t_f} C\left(\mathbf{x}(t), \dot{\mathbf{x}}(t), \mathbf{u}(t)\right)\, dt \text{ is minimum}$$

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t))$$

$$\mathbf{x}(t_0) = \mathbf{x}_0, \mathbf{x}(t_f) = \mathbf{x}_f$$

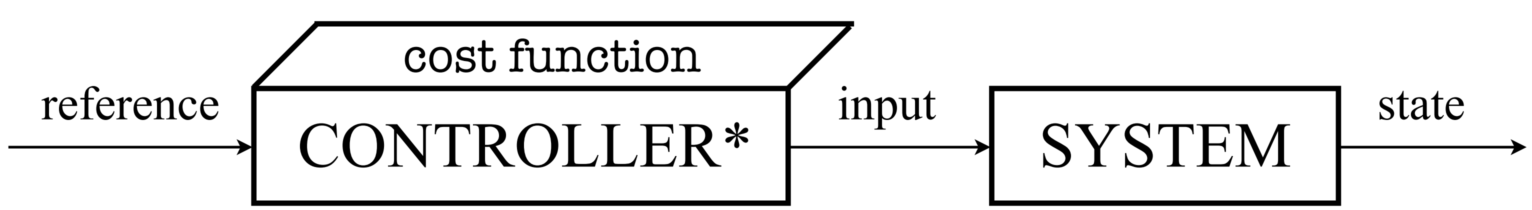
$\implies \mathbf{u}^*, \mathbf{x}^*$ *optimal* control and state



$$\begin{cases} m_1 \ddot{x}_1 = u_1 \\ m_2 \ddot{x}_2 = u_2 \end{cases}$$

$$\int_0^1 \left(u_1^2(t) + u_2^2(t)\right)\, dt$$

- **Optimal controller(*) as an inverse model**



OPTIMAL FEEDBACK CONTROL

Recalculate optimal control at each time step

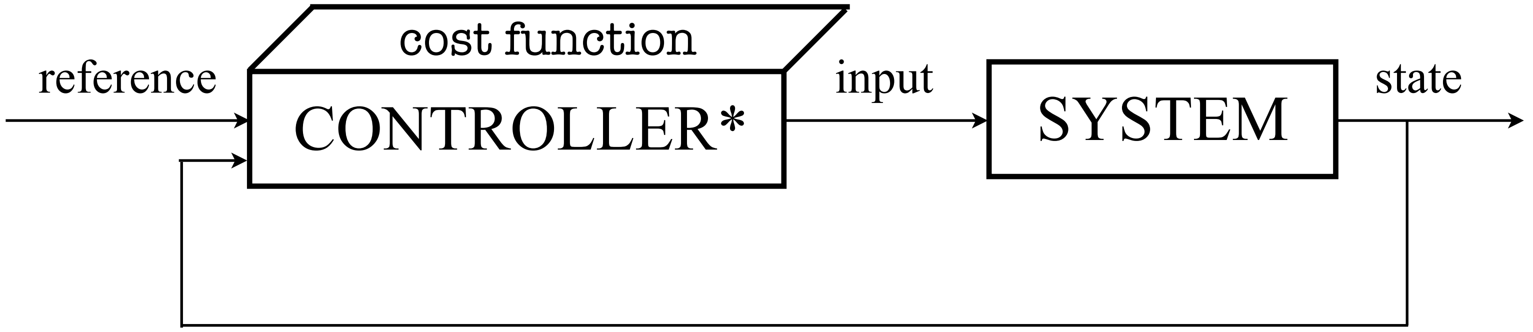
At each τ find $\mathbf{u}(t), t \in [\tau, t_f]$

such that

$$\int_{\tau}^{t_f} C\left(\mathbf{x}(t), \dot{\mathbf{x}}(t), \mathbf{u}(t)\right)\, dt \quad \text{is minimum}$$

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t))$$

$$\mathbf{x}(\tau), \mathbf{x}(t_f) = \mathbf{x}_f$$




```
note: neither feedforward, nor feedback
– both feedforward and feedback
```

OPTIMAL STATE ESTIMATION

Optimal linear estimation

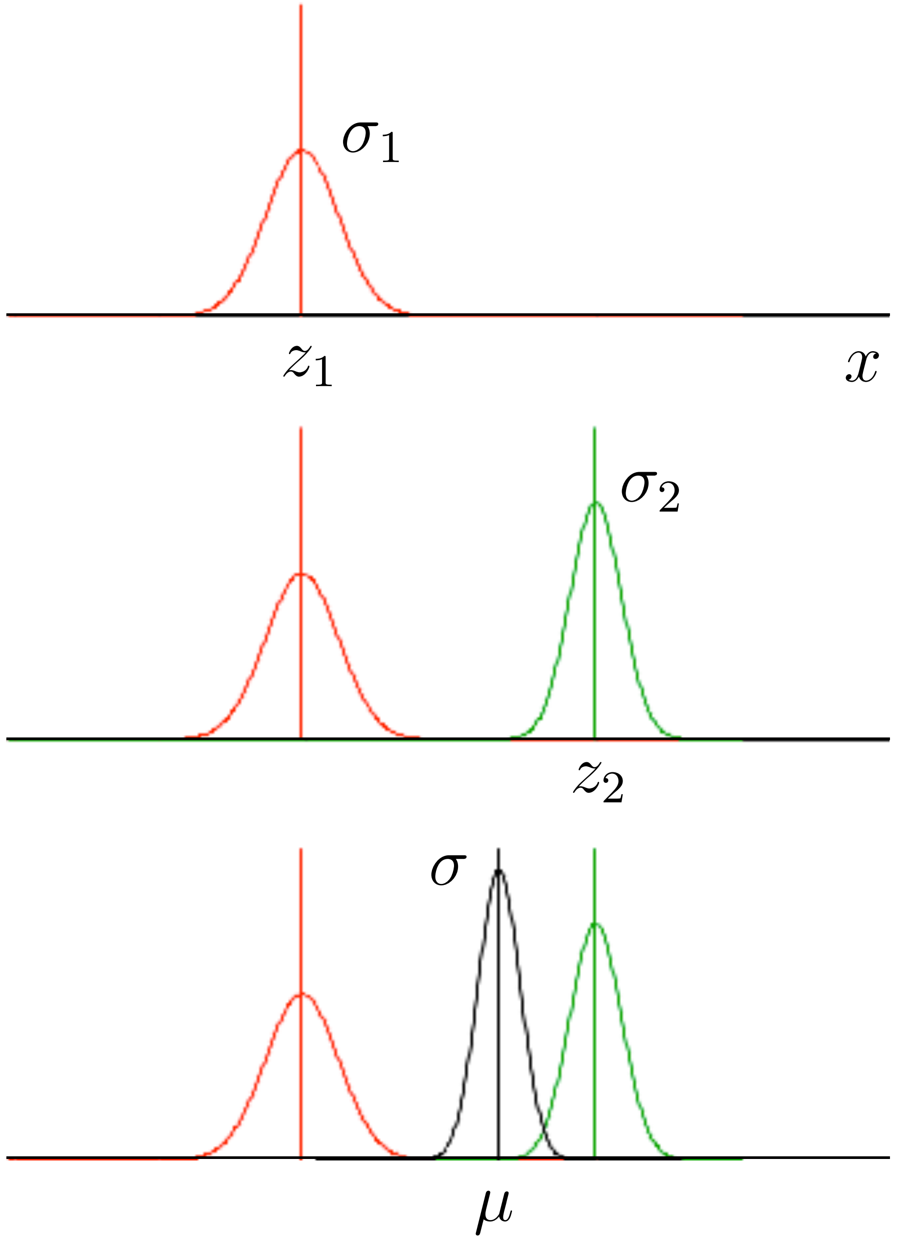
$$\hat{x}(t_1) = z_1 \qquad \sigma_x^2(t_1) = \sigma_1$$

$$\hat{x}(t_2) = \mu \qquad \sigma_x^2(t_2) = \sigma$$

$$\mu = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2} z_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} z_2$$

$$\frac{1}{\sigma^2} = \frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2}$$

$$\boxed{\begin{aligned} \hat{x}(t_2) &= \hat{x}(t_1) + K(t_2)[z_2 - \hat{x}(t_1)] \\ K(t_2) &= \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \end{aligned}}$$



— Maybeck, 1979, *Stochastic Models, Estimation, and Control,* Academic Press

SOLUTIONS TO OPTIMAL CONTROL

- **Linear system, quadratic cost, deterministic**

linear quadratic regulator (LQR): analytic solution

$$\text{Find } \mathbf{u}(t), t \in [t_0, t_f]$$

$$\text{such that}$$

$$\int_{t_0}^{t_f} \left(\mathbf{x}^T(t)\mathbf{Q}\mathbf{x}(t) + \mathbf{u}^T(t)\mathbf{R}\mathbf{u}(t)\right)\ dt \ \text{ is minimum}$$

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$

$$\mathbf{x}(t_0) = \mathbf{x}_0, \mathbf{x}(t_f) = \mathbf{x}_f$$

- **Linear system, quadratic cost, Gaussian noise**

linear quadratic Gaussian (LQG): analytic solution

- **Nonlinear systems, ...**

numerical solutions: nonlinear programming

LINEAR CASE EXPLAINED

actual state | input | state noise

next state

$$x_{k+1} = Ax_k + B(u_k + w_k)$$

actual observation

$$y_k = Hx_k + v_k$$

observation matrix
observation noise

cost to minimize

$$J = \sum_{k=0}^{p-1} \Big(\boxed{y_{k+1}^T Q y_{k+1}} + \boxed{u_k^T R u_k} \Big)$$

tracking cost | control cost

feedback control policy

$$u_k = -L_k \hat{x}_k$$

next estimated state

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + K_k(y_k - H\hat{x}_k)$$

actual observation | predicted observation

THE ENGINEER VS THE BRAIN

At each τ find $\mathbf{u}(t), t \in [\tau, t_f]$
such that

$$\int_{\tau}^{t_f} C\left(\mathbf{x}(t), \dot{\mathbf{x}}(t), \mathbf{u}(t)\right)\, dt \quad \text{is minimum}$$

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t))$$

$$\mathbf{x}(\tau), \mathbf{x}(t_f) = \mathbf{x}_f$$