Computational aspects of motor control and motor learning

*Michael I. Jordan**

Mark J. Buller

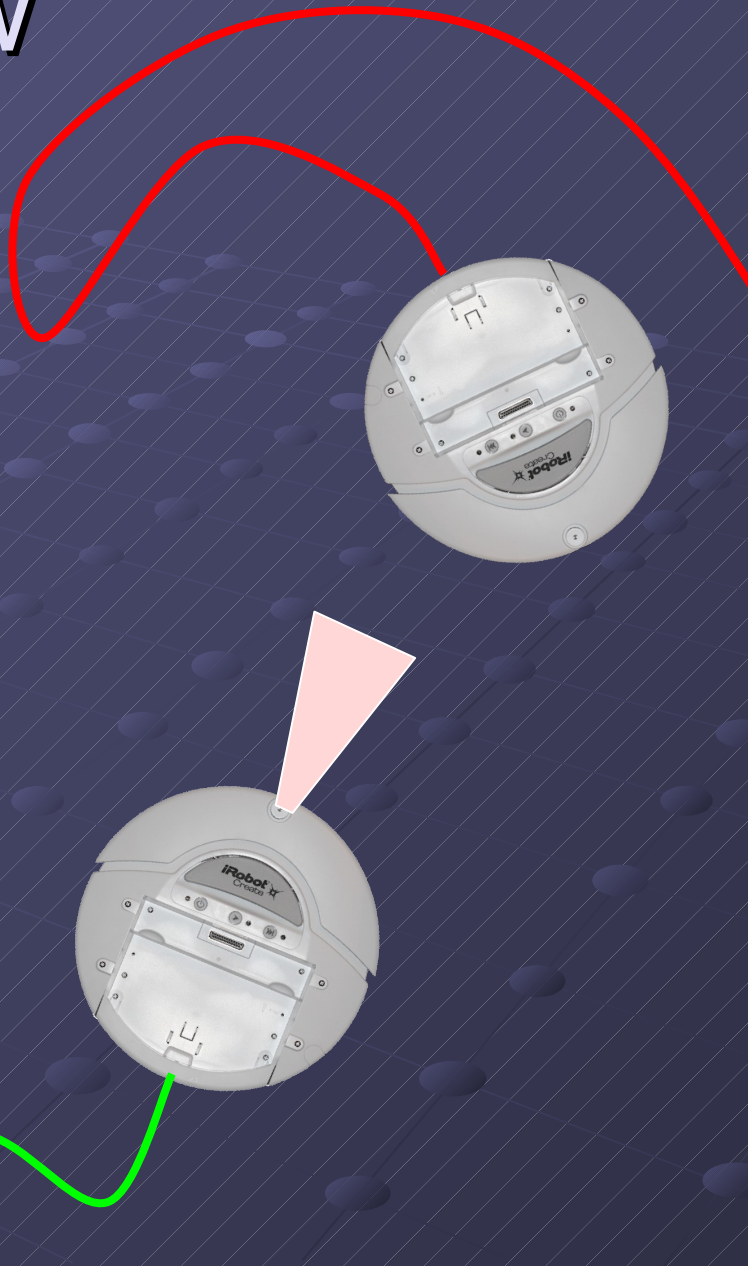
(mbuller)

21 February 2007

*In H. Heuer & S. Keele, (Eds.), Handbook of Perception and Action: Motor Skills. New York: Academic Press, 1996.

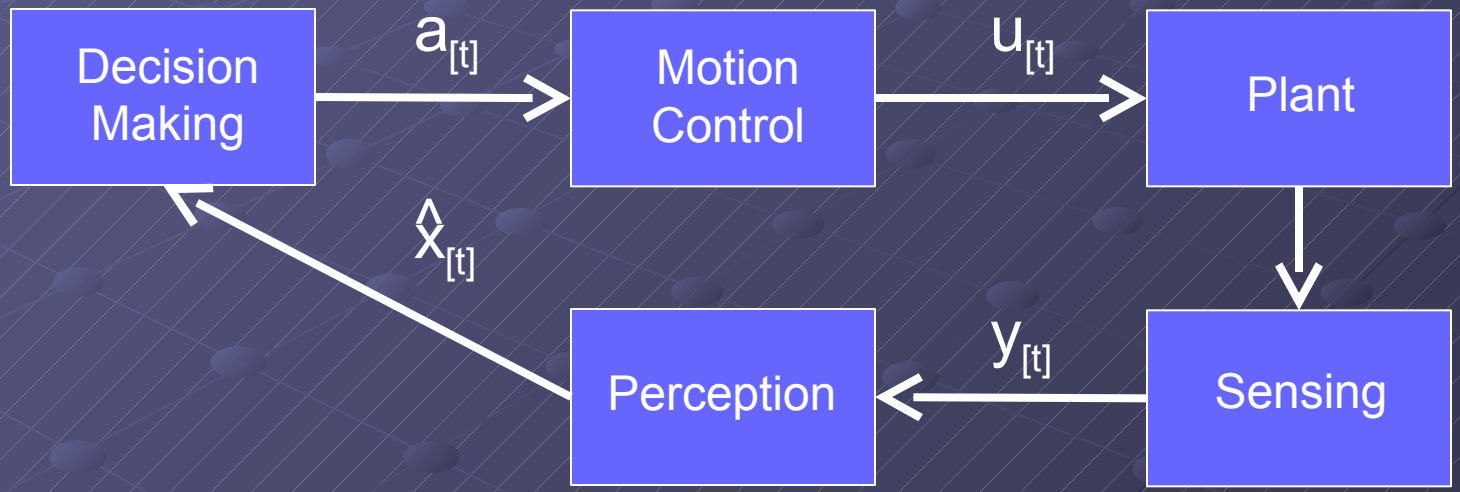
Overview

- Relevance
- Dynamical Systems (DS)
- DS Control Architecture
  - Feedforward
  - Feedback
  - Error Correcting Feedback
  - Composite Control Systems
- State Estimation
- Learning Algorithms
- Plant Controller Learning



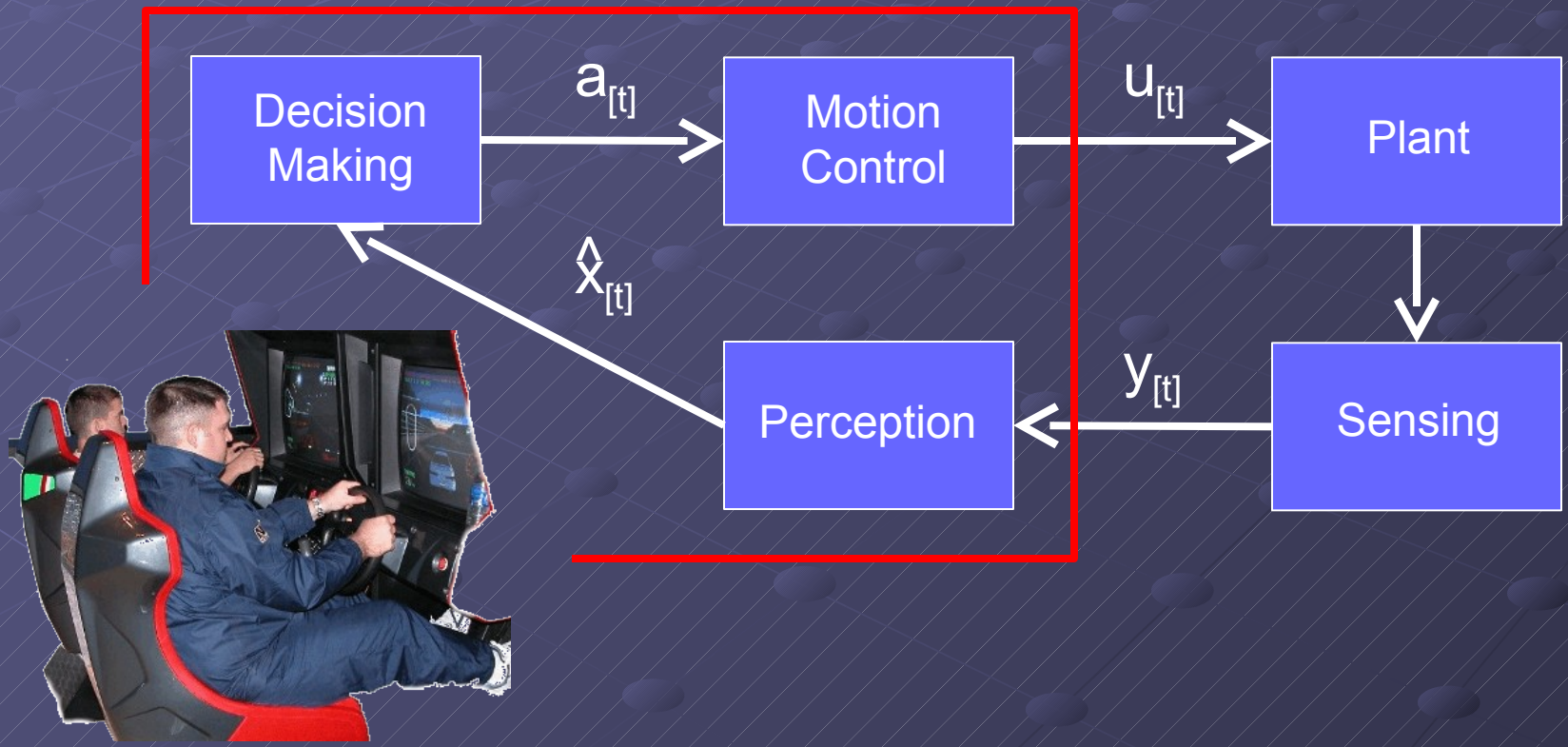
Relevance

- Jordan Provides us the Architectural “Nuts and Bolts” for the Robot Control Loop



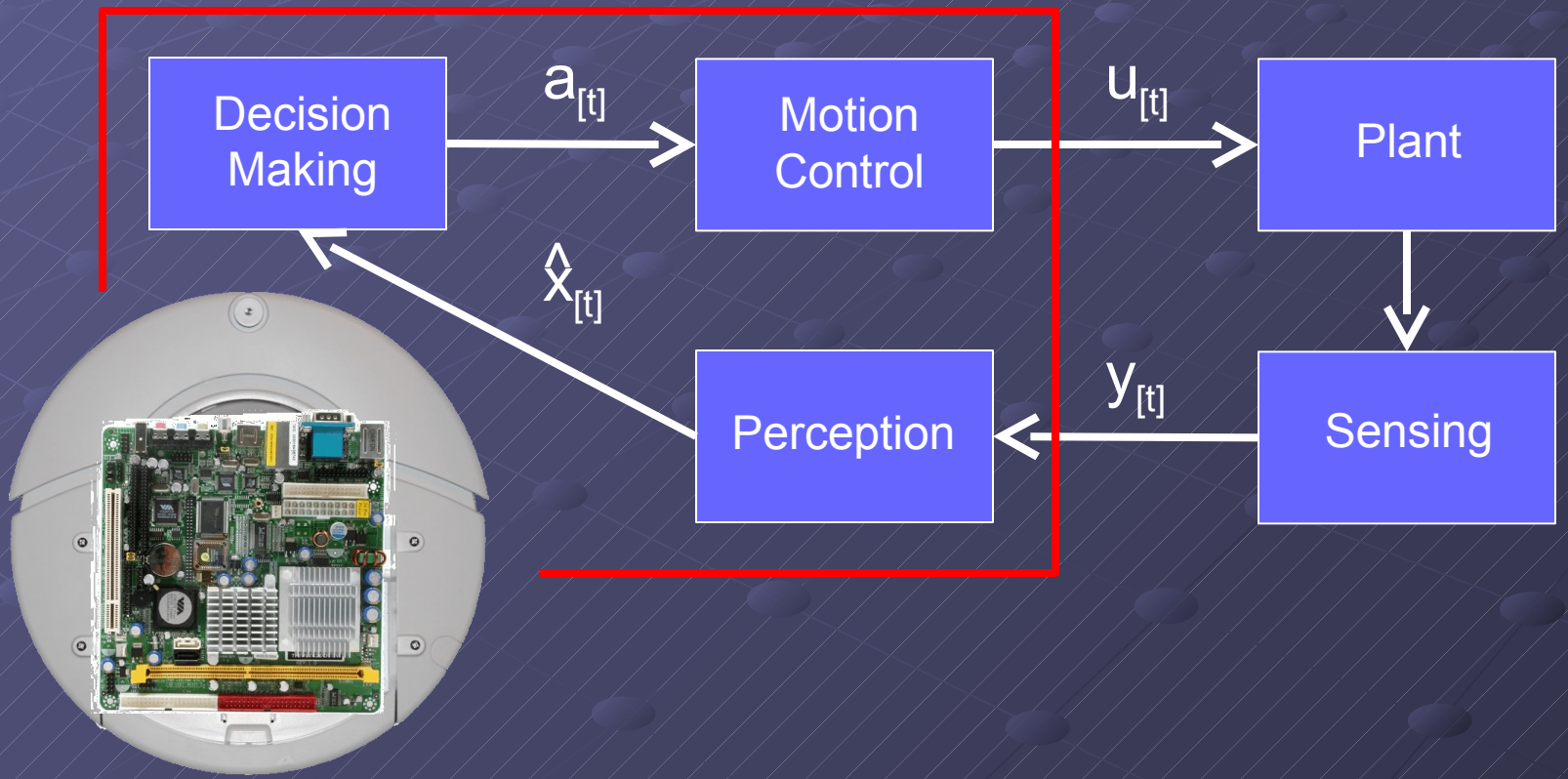
Relevance

- Jordan Provides us the Architectural “Nuts and Bolts” for the Robot Control Loop



Relevance

- Jordan Provides us the Architectural “Nuts and Bolts” for the Robot Control Loop



Dynamical Systems

- An entity with a state time dependence e.g.

- “Many useful dynamical systems models are simply descriptive models of the temporal evolution of an interrelated set of variables.” *(Jordan p7)*

Dynamical Systems

- An entity with a state time dependence e.g.

Ball

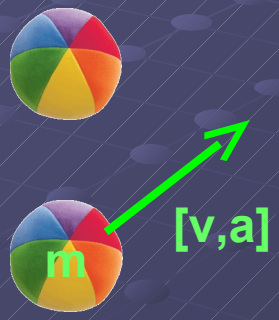
- “Many useful dynamical systems models are simply descriptive models of the temporal evolution of an interrelated set of variables.” *(Jordan p7)*

Dynamical Systems

- An entity with a state time dependence e.g.

Ball

Ball
[Mass, Velocity, Acceleration]



- “Many useful dynamical systems models are simply descriptive models of the temporal evolution of an interrelated set of variables.” *(Jordan p7)*

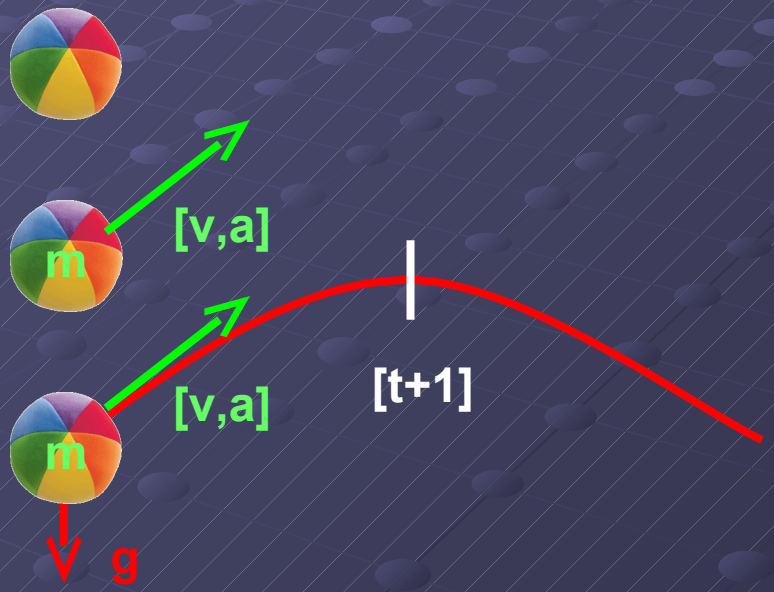
Dynamical Systems

- An entity with a state time dependence e.g.

Ball

Ball
[Mass, Velocity, Acceleration]

Newtonian Mechanics
allow us to predict location
of ball at time [t+1]



- "Many useful dynamical systems models are simply descriptive models of the temporal evolution of an interrelated set of variables." *(Jordan p7)*

Dynamical System Control

- Given a “Dynamical System” what inputs are required to produce a given output. E.g.
  - What force needs to be applied and in what direction to get the ball to the friend
  - Next State Equation:

    $\mathbf{x}_{n+1} = f(\mathbf{x}_n, \mathbf{u}_n)$

  - Output Function

    $\mathbf{y}_n = g(\mathbf{x}_n)$

  - Input Output Mapping Equation

    $\mathbf{y}_{n+1} = h(\mathbf{x}_n, \mathbf{u}_n)$

$[y_n]$

?

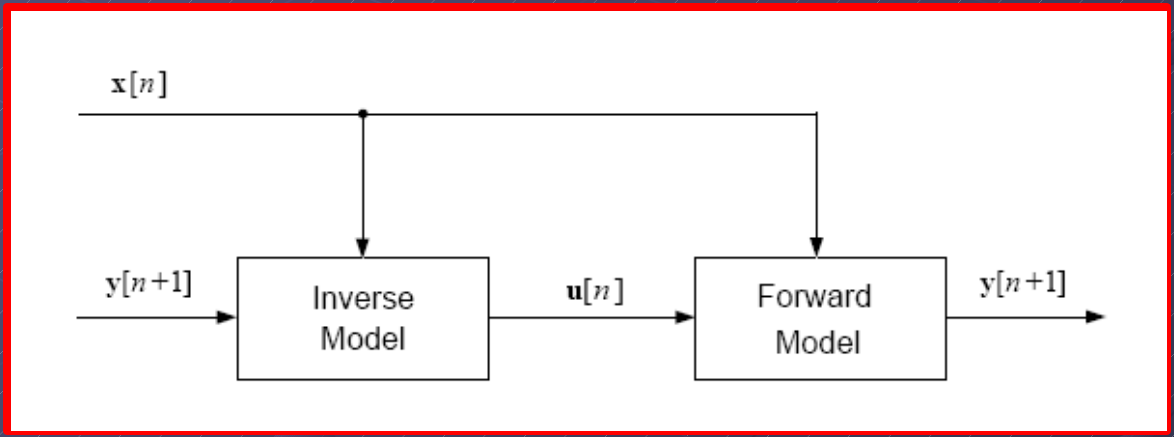
$[y^*_{n+1}]$

Models

- A Dynamical System Model is at the heart of our ability to produce control inputs

- Forward Model
  - Causal Model or Forward Transformation Model
  - Maps inputs to an output
  - Many to One Mapping
  - E.g. Ball and Newtonian Physics

- Inverse Model
  - Directional Flow Model
  - One to Many Mapping
    - e.g. joint angles & spatial position In an articulated arm. A new position can be achieved in multiple ways



Control

- Problem of computing an input to the system that will achieve some desired behavior at its output.

- Seems to involve the notion of computing the inverse (explicitly or implicitly) of the control model
  - Jordan uses a simple first order plant model as an example:

    $\mathbf{x}_{n+1} = 0.5\mathbf{x}_n + 0.4\mathbf{u}_n$

    $\mathbf{y}_n = \mathbf{x}_n$

    $\mathbf{y}_{n+1} = 0.5\mathbf{x}_n + 0.4\mathbf{u}_n$

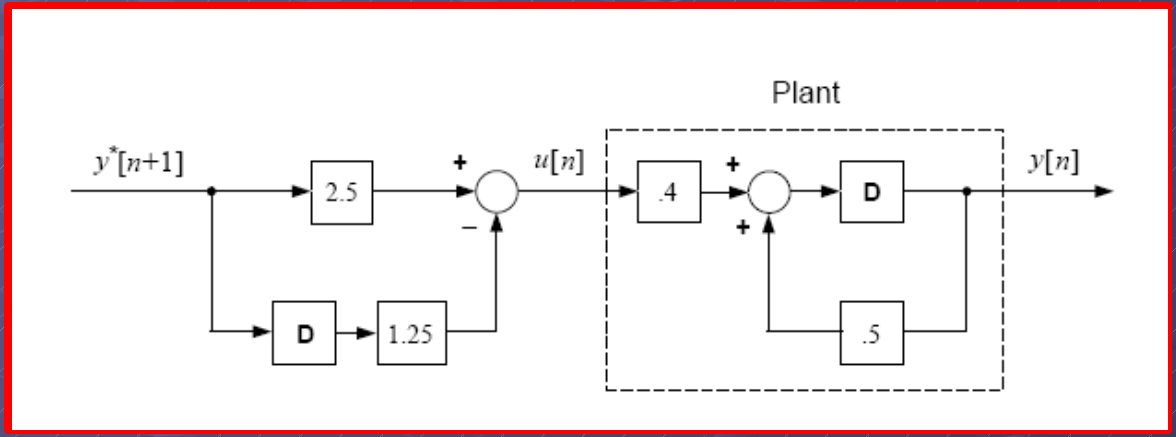
    Solving for u_n:

    $\mathbf{u}_n = -1.25\hat{\mathbf{x}}_n + 2.5\mathbf{y}^*_{n+1}$

    Where: $\hat{\mathbf{x}}_n$ is estimated state and $\mathbf{y}^*_{n+1}$

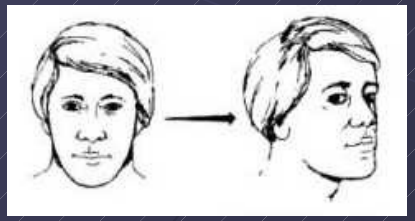
Open Loop Feedforward Controller

- $\hat{x}_n$ is estimated from y^*_n (desired output)



- Pros
  - Simple controller.
  - If model is good then y* and y will be close.
- Cons
  - Large assumption that model is correct
  - Errors can grow and compound

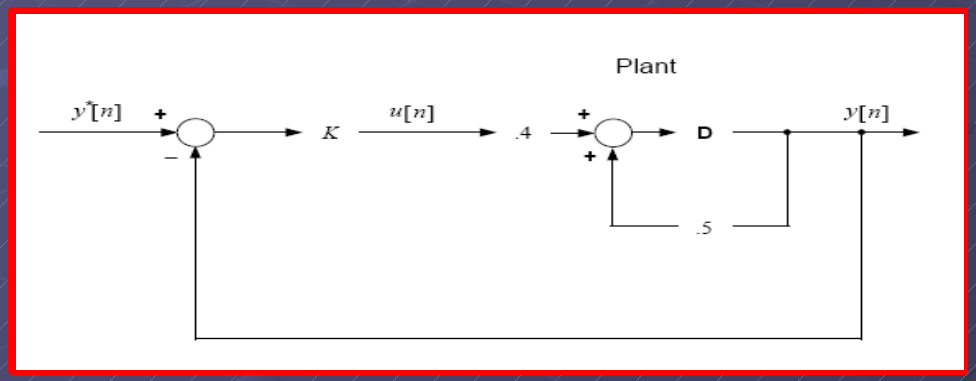
- Example: Vestibulo-ocular Reflex (VOR)
  - Couple movement of eyes to motion of head. Transform head velocity to eye velocity



Error Correcting Feedback Controller

- Does not rely on an explicit inverse of the plant model
- Works directly to correct the error at the current time step between the desired plant output $\mathbf{y}^*_n$ and actual plant output $\mathbf{y}_n$.

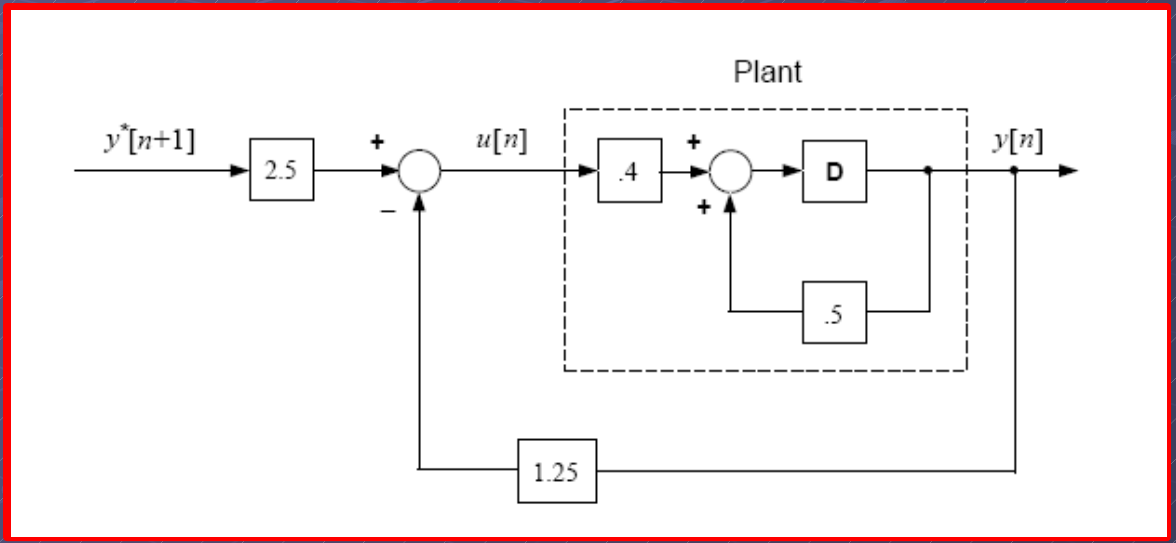
$\mathbf{u}_n = K(\mathbf{y}^*_n - \mathbf{y}_n)$ where K = gain (scalar)



- Pros
  - Does not depend on a explicit inverse of the plant model
  - More robust on unanticipated disturbances
- Cons
  - Corrects error after it has occurred
  - Still has error under ideal situations
  - Can be unstable

Feedback Controller

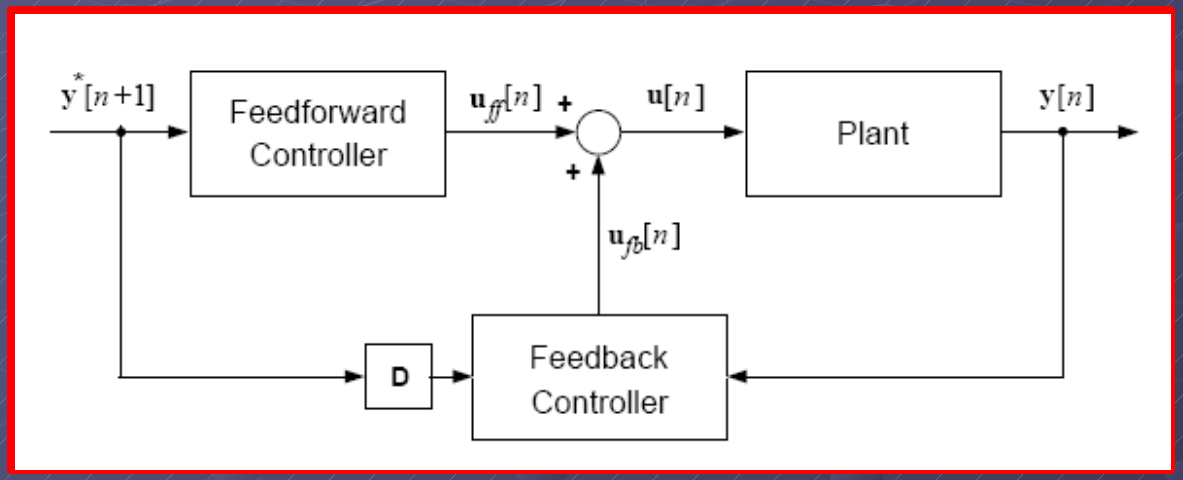
- $\hat{x}_n$ is estimated from y_n (model output)



- Pros
  - Very simple controller
  - More robust with unanticipated disturbances
  - Can avoid compounding of errors
- Cons
  - What if the model is not good or has inaccuracies
  - Feedback can introduce instability

Composite Control Systems

- Combine complimentary strengths of feedforward controller and feedback controller.



State Estimation

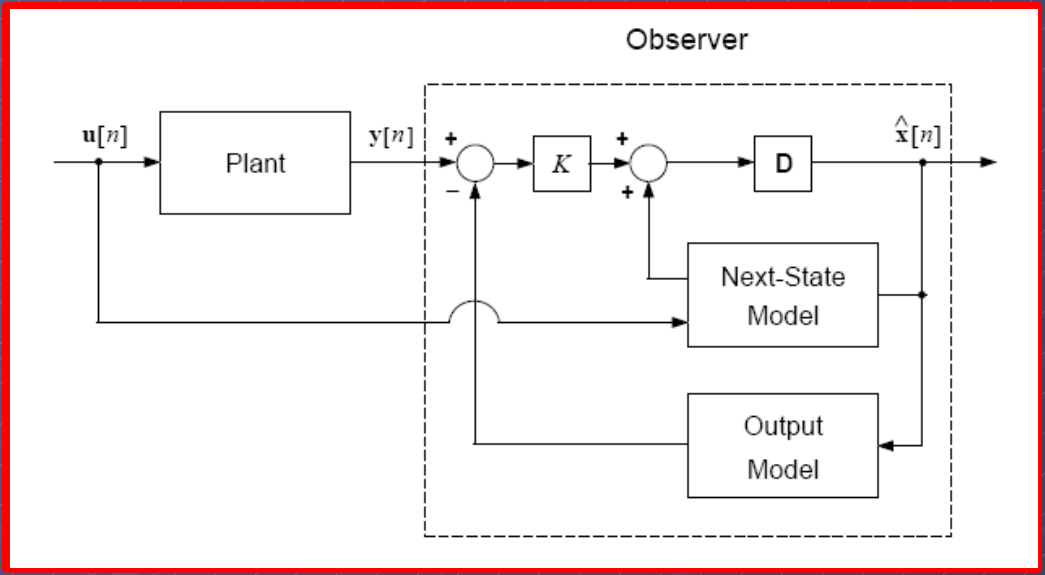
- Previous examples assume that state can either be determined from output of the system or assumed to be the desired output. This estimated state is then used to estimate the input variables for the next iteration.

- Often the system output is a more complex function of state:
  - Inverting the output function will often not work:
    1) More state variables than output variables and thus the function is not uniquely invertible.
    2) There is uncertainty about the about the dynamics of the system as seen through the output function.

- "State estimation is a dynamic process"

- "Robust estimation of the state of a system requires observing the output of the system over an extended period of time"

State Estimation - Observers



- Observer is an internal simulation of the plant running in parallel
- Actual Plant output is compared to observer predicted output
  - Errors in output are used to correct the state estimate:

$$\hat{\mathbf{x}}[n+1] = \hat{f}(\hat{\mathbf{x}}[n], \mathbf{u}[n]) + K(\mathbf{y}[n] - \hat{g}(\hat{\mathbf{x}}[n]))$$

  - K is set based upon relative noise levels in NEXT STATE and OUTPUT measurement processes.
    - If OUTPUT noise > NEXT STATE noise K is low
    - If NEXT STATE noise > OUTPUT K is high

Learning Algorithms

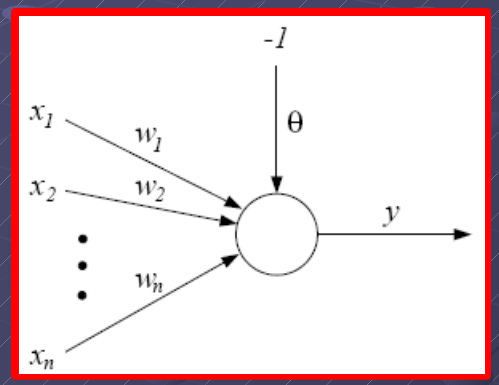
- Previous examples have dealt with systems and plants in relatively benign finite settings. Systems that need to interact with the real world will encounter situations or objects etc. that do not conform the system's model. An adaptive process would allow the system to update its control mechanisms.

- Learning algorithms can be taught in two ways:
  1) Present whole gamut of available data prior to the deployment of the system or periodically update the learning algorithm
  2) Dynamically update control models after the presentation of each new piece of learning data. a.k.a On-Line Learning.

Machine Learning Tools

- Jordan presents two main classes of Learning Algorithms:
  - Classifiers
    - Map inputs into a set of discrete outputs e.g. The Perceptron

Perceptron updates weights based upon performance with the training examples. (On-line technique)

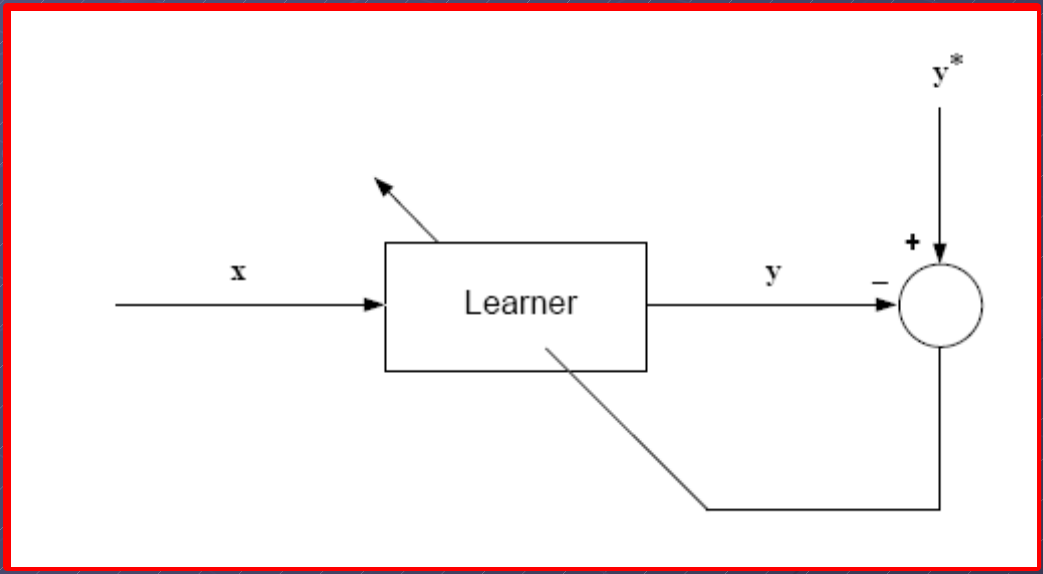


  - Regression
    - Maps inputs into a continuous output variable e.g. Least Squares Regression (Linear or Polynomial)

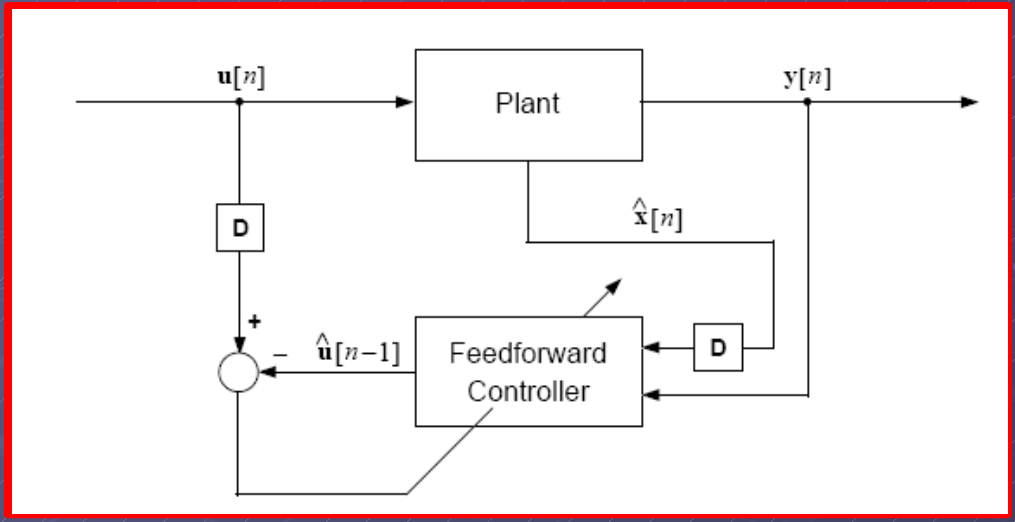
  - Many other Machine Learning techniques are applicable see:
    - *Bishop CM. (2006). Pattern Recognition and Machine Learning. Springer, NY*

Bringing it All Together

- Motor Learning or Plant Controller Learning
  - Problem of learning an inverse model of the plant
    - Direct Inverse Modeling
    - Distal Supervised Learning
    - Feedback Error Learning



Direct Inverse Learning

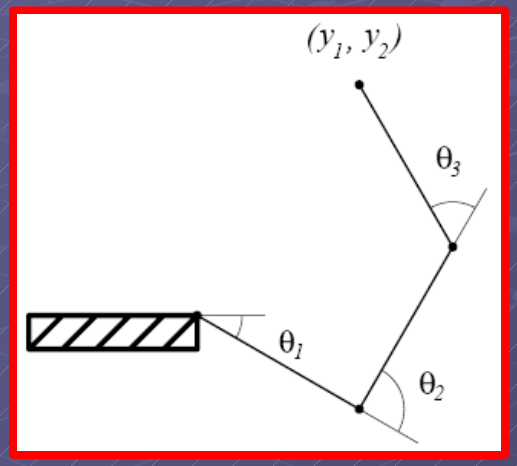


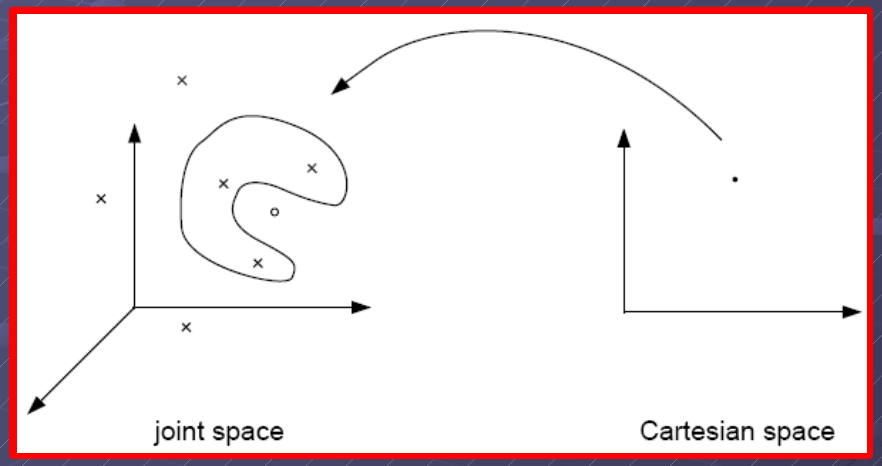
- Present input output pairs to the supervised learning algorithm. (*offline technique*)
  - The supervised learning algorithm will minimize:

$$J = \frac{1}{2}\|\mathbf{u}[n-1] - \hat{\mathbf{u}}[n-1]\|^2$$

  - Given the plant input at time [t-1] and the plant output and estimated state the learning algorithm attempts to minimize the error between its estimate of control inputs and the actual control inputs at [t-1]
  - Approach works well for linear systems but can yield controller inputs for non-linear systems

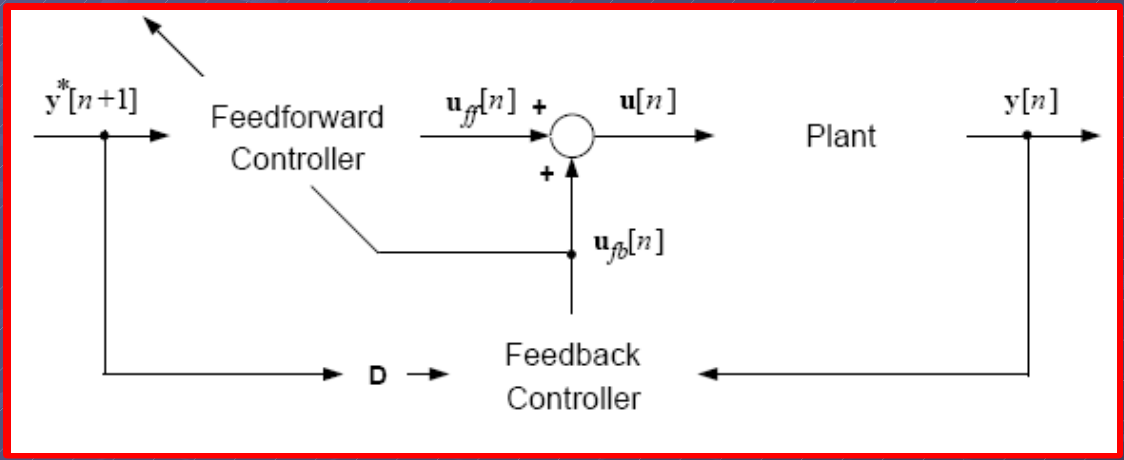
Direct Inverse Learning - Problems





- Nonconvexity Problem:
  - If learning data is presented to the learning algorithm where one output exists for the location of the arm in Cartesian space and three different sets of input variables map to this output space then many learning algorithms will provide a learned solution that is an impossibility for the arm.

Feedback Error Learning

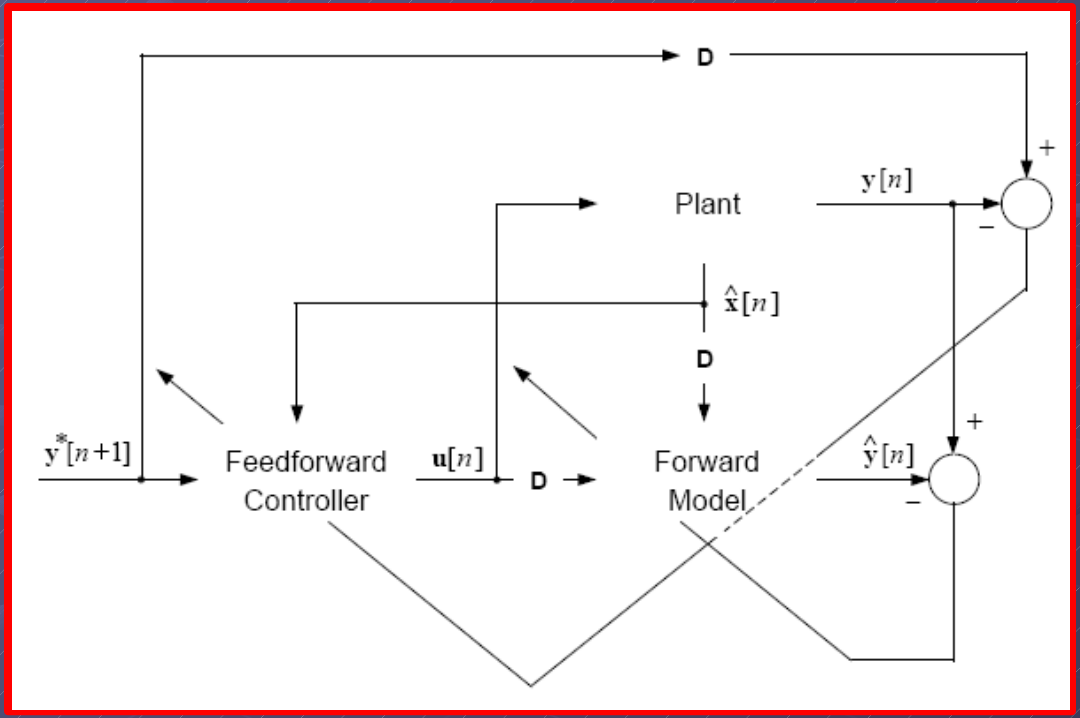


- Desired plant output is used for both control and learning
- Learning can be conducted online
- Is goal oriented:
  - In the sense tries to minimize error between actual plant output and desired plant output.
- "Guides" learning of the feedforward controller

Distal Supervised Learning

- Approach aims to solve the nonlinear model inverse problem as a composite system of forward plant model and feedforward controller model
- Two interactive processes used in learning the system
  - Forward model is learned
  - Forward model is used in the learning of the feedforward controller
- This approach avoids nonconvexity problem as the feedforward controller learns to minimize error.

Distal Supervised Learning II



- The Forward Model is trained using the prediction error: (y[n] - y^[n]).
- The composite learning system (Forward Model & Feedforward Controller) is trained using the performance error (y*[n] - y[n]). Where the Forward model is held fixed.

Conclusions

- Jordan presents a series of control architectures and control policy learning techniques
- Inverse and Forward models play complimentary roles
  - Inverse models are the basis for predictive control
  - Forward models can be used to anticipate and cancel delayed feedback
  - Basic blocks for dynamical state estimation
- When the models are learned using machine learning algorithms or techniques they provide capabilities for prediction, control and error correction that allow the system to cope with difficult non-linear control problems

"General rule…partial knowledge is better than no knowledge, if used appropriately"

Applications to Roomba Tag

- What Control Architecture/s
- What Learning algorithm/s
- Holistic vs. Set of Desired Behaviors
- Single control architecture or multiple control architectures for different functions
  - Navigate
  - Find Roomba
  - Stalk Roomba
  - Find Hiding Spot
  - Navigate to Hiding Spot