

F.L. Lewis
National Academy of Inventors

Moncrief-O'Donnell Chair, UTA Research Institute (UTARI)
The University of Texas at Arlington, USA
and

Guest Professor, Shanghai Jiao Tong University

New Developments in Integral Reinforcement Learning: Continuous-time Optimal Control and Games




Thanks to
Ning Li



New Research Results

Integral Reinforcement Learning for Online Optimal Control

IRL for Online Solution of Multi-player Games

Multi-Player Games on Communication Graphs

Off-Policy Learning

Experience Replay

Bio-inspired Multi-Actor Critics

Output Synchronization of Heterogeneous MAS

Applications to:

Microgrid

Robotics

Industry Process Control

Optimality and Games

Optimal Control is Effective for:

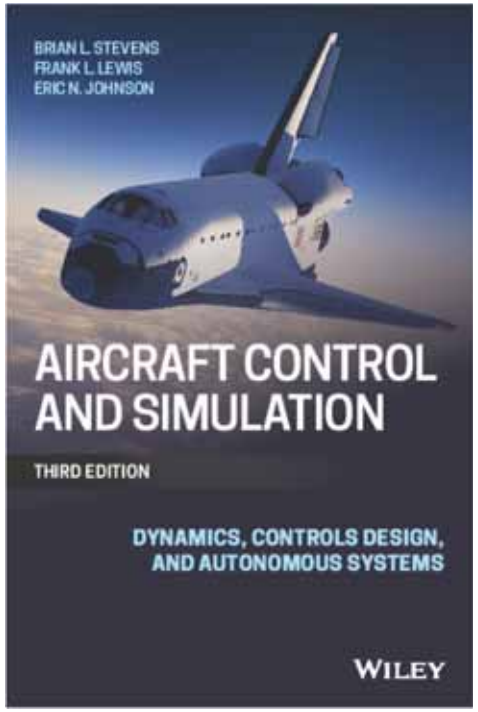
- Aircraft Autopilots
- Vehicle engine control
- Aerospace Vehicles
- Ship Control
- Industrial Process Control

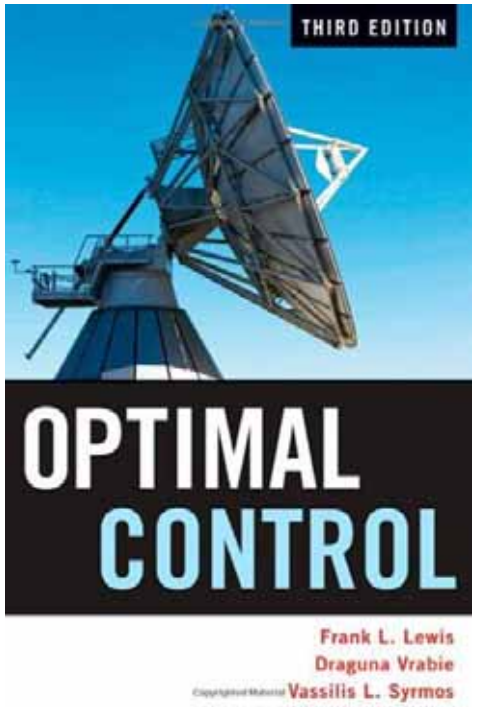
Multi-player Games Occur in:

- Networked Systems Bandwidth Assignment
- Economics
- Control Theory disturbance rejection
- Team games
- International politics
- Sports strategy

But, optimal control and game solutions are found by

- Offline solution of Matrix Design equations
- A full dynamical model of the system is needed





Derivation of Linear Quadratic Regulator

Rudolph Kalman 1960

Full system dynamics must be known
Off-line solution

System $\dot{x} = Ax + Bu$

Cost $V(x(t)) = \int_t^\infty (x^T Qx + u^T Ru)\, d\tau \qquad = x^T(t) P x(t)$

Differentiate using Leibniz' formula

$$\dot{V} = -(x^T Qx + u^T Ru)$$

Differential equivalent is the Bellman equation

$$0 = H(x, \frac{\partial V}{\partial x}, u) = \dot{V} + x^T Qx + u^T Ru = 2\left(\frac{\partial V}{\partial x}\right)^T \dot{x} + x^T Qx + u^T Ru = 2x^T P(Ax + Bu) + x^T Qx + u^T Ru$$

Scalar equation

Given any stabilizing FB policy $u = -Kx$

$$0 = H(x, \frac{\partial V}{\partial x}, u) = x^T \left[(A - BK)^T P + P(A - BK) + Q + K^T RK \right] x$$

Lyapunov equation

Stationarity condition $0 = \frac{\partial H}{\partial u} = 2Ru + 2B^T Px$

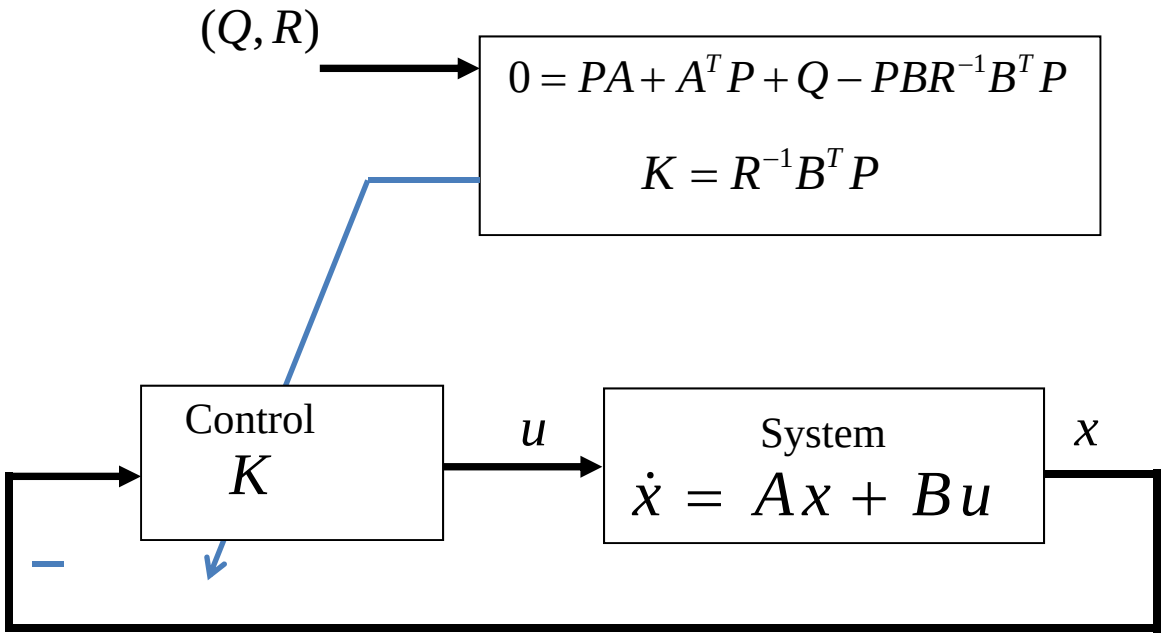
Optimal Control is $u = -R^{-1} B^T Px \equiv -Kx$

HJB equation $0 = 2x^T P(Ax - BR^{-1} B^T Px) + x^T Qx + x^T PBR^{-1} B^T Px$

Riccati equation $0 = x^T PAx + x^T A^T Px + x^T Qx - x^T PBR^{-1} B^T Px$

Optimal Control- The Linear Quadratic Regulator (LQR)

User prescribed optimization criterion $V(x(t)) = \int_t^\infty (x^T Q x + u^T R u)\, d\tau$



Off-line Design Loop
Using ARE

On-line real-time
Control Loop

An Offline Design Procedure
that requires Knowledge of system dynamics model (A,B)

System modeling is expensive, time consuming, and inaccurate

Adaptive Control is online and works for unknown systems.
Generally not Optimal

Optimal Control is off-line,
and needs to know the system dynamics to solve design eqs.

We want to find optimal control solutions
- Online in real-time
- Using adaptive control techniques
- Without knowing the full dynamics

For nonlinear systems and general performance indices

Bring together Optimal Control and Adaptive Control

Reinforcement Learning turns out to be the key to this!

Optimality in Biological Systems

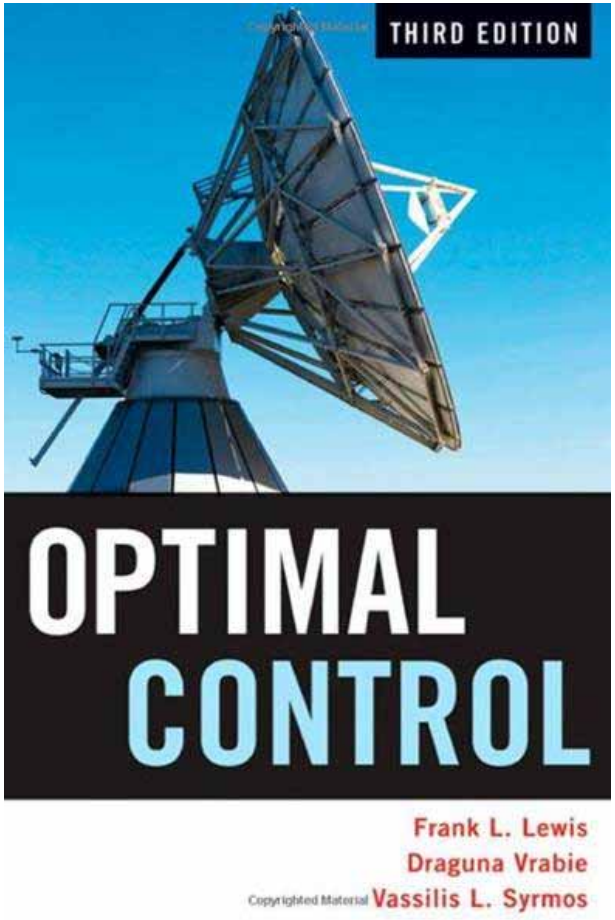
Every living organism improves its control actions based on rewards received from the environment

The resources available to living organisms are usually meager. Nature uses optimal control.

Reinforcement Learning

1. Apply a control. Evaluate the benefit of that control.
2. Improve the control policy.

RL finds optimal policies by evaluating the effects of suboptimal policies



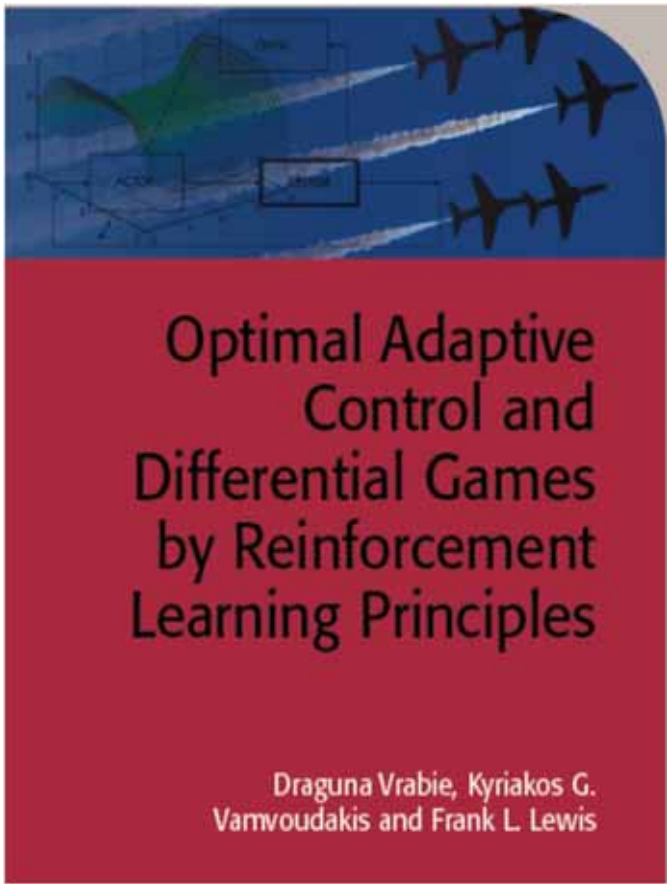
Books

F.L. Lewis, D. Vrabie, and V. Syrmos, *Optimal Control*, third edition, John Wiley and Sons, New York, 2012.

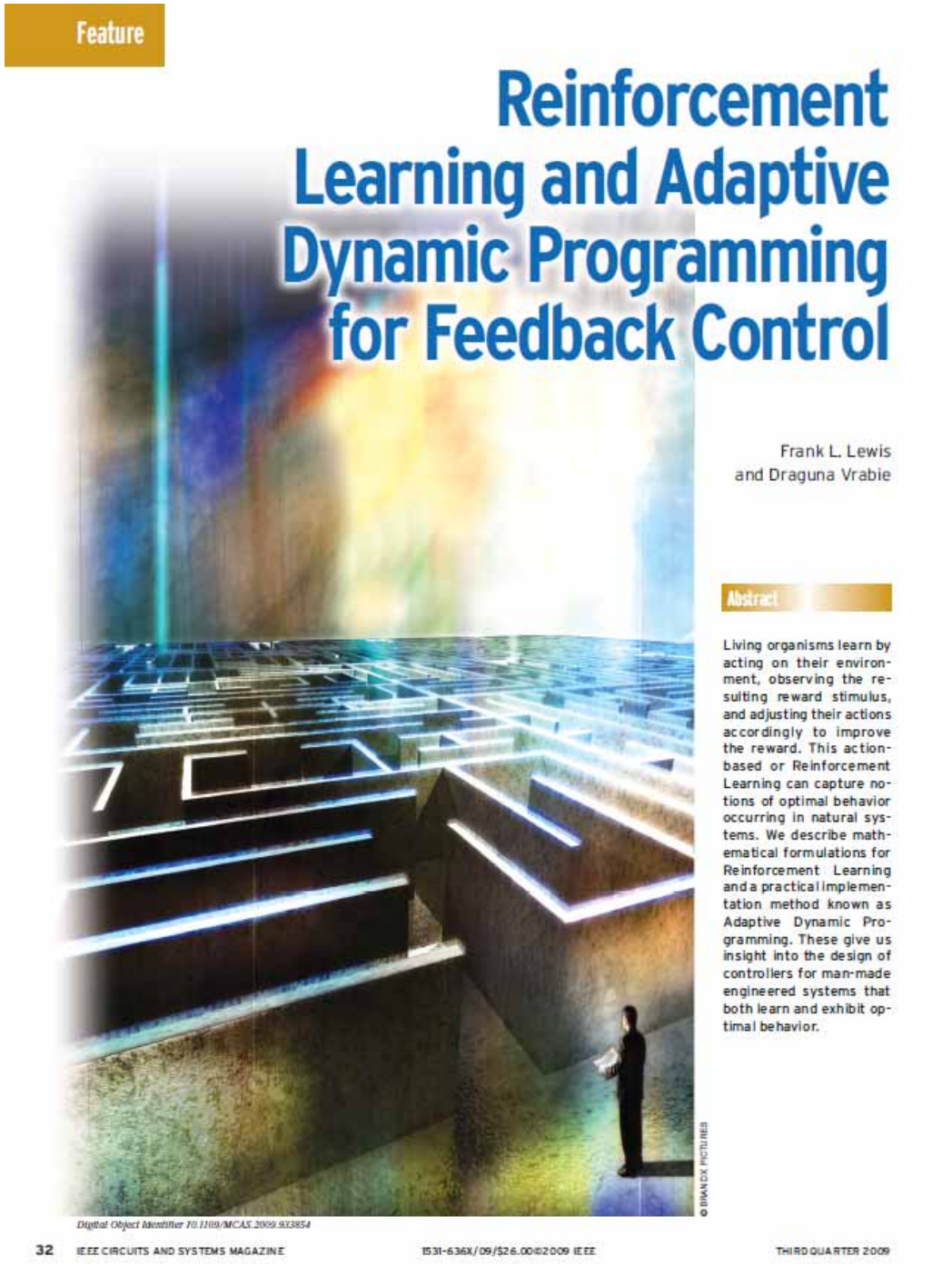
New Chapters on:

Reinforcement Learning

Differential Games



D. Vrabie, K. Vamvoudakis, and F.L. Lewis, *Optimal Adaptive Control and Differential Games by Reinforcement Learning Principles*, IET Press, 2012.



Feature

Reinforcement Learning and Adaptive Dynamic Programming for Feedback Control

Frank L. Lewis
and Draguna Vrabie

Abstract

Living organisms learn by acting on their environment, observing the resulting reward stimulus, and adjusting their actions accordingly to improve the reward. This action-based or Reinforcement Learning can capture notions of optimal behavior occurring in natural systems. We describe mathematical formulations for Reinforcement Learning and a practical implementation method known as Adaptive Dynamic Programming. These give us insight into the design of controllers for man-made engineered systems that both learn and exhibit optimal behavior.

© BRAND X PICTURES

Digital Object Identifier 10.1109/MCAS.2009.933854

 1531-636X/09/$26.00©2009 IEEE

F.L. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," IEEE Circuits & Systems Magazine, Invited Feature Article, pp. 32-50, Third Quarter 2009.

IEEE Control Systems Magazine, F. Lewis, D. Vrabie, and K. Vamvoudakis, "Reinforcement learning and feedback Control," Dec. 2012

Game Theory-Based Control System Algorithms with Real-Time Reinforcement Learning



HOW TO SOLVE MULTIPLAYER GAMES ONLINE

KYRIAKOS G. VAMVOUDAKIS, HAMIDREZA MODARES, BAHARE KIUMARSI, and FRANK L. LEWIS

Complex human-engineered systems involve an interconnection of multiple decision makers (or agents) whose collective behavior depends on a compilation of local decisions that are based on partial information about each other and the state of the environment [1]–[4]. Strategic interactions among agents in these systems can be modeled as a multiplayer simultaneous-move game [5]–[8]. The agents involved can have conflicting objectives, and it is natural to make decisions based upon optimizing individual payoffs or costs.

Game theory has been mostly pioneered in the field of economics; [9] considered a finite win-loss game with perfect information between two players, and this classic example of computable economics stands in the long and distinguished tradition of game theory that goes back to [10] and [11]. Reference [12] discusses game theory in algorithmic modes but not in what is today referred to as *algorithmic game theory* after realizing the futility of

Digital Object Identifier 10.1109/MCS.2016.2621461
Date of publication: 19 January 2017

Multi-player Game Solutions
IEEE Control Systems Magazine, Dec 2017

RL ADP has been developed for Discrete-Time Systems

Discrete-Time System Hamiltonian Function

$$x_{k+1} = f(x_k, u_k)$$

$$H(x_k, \nabla V(x_k), h) = r(x_k, h(x_k)) + \gamma V_h(x_{k+1}) - V_h(x_k)$$

- Directly leads to temporal difference techniques
- System dynamics does not occur
- Two occurrences of value allow APPROXIMATE DYNAMIC PROGRAMMING methods

Continuous-Time System Hamiltonian Function

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

$$H(x, \frac{\partial V}{\partial x}, u) = \dot{V} + r(x,u) = \left(\frac{\partial V}{\partial x}\right)^T \dot{x} + r(x,u) = \left(\frac{\partial V}{\partial x}\right)^T f(x,u) + r(x,u)$$

Leads to off-line solutions if system dynamics is known
Hard to do on-line learning

- How to define temporal difference?
- System dynamics DOES occur
- Only ONE occurrence of value gradient

How can one do Policy Iteration for Unknown Continuous-Time Systems?
What is Value Iteration for Continuous-Time systems?
How can one do ADP for CT Systems?

Discrete-Time Systems
Adaptive (Approximate) Dynamic Programming

Four ADP Methods proposed by Paul Werbos

Critic NN to approximate:

Heuristic dynamic programming

Value Iteration

Value $V(x_k)$

AD Heuristic dynamic programming
(Watkins Q Learning)

Q function $Q(x_k, u_k)$

Dual heuristic programming

Gradient $\frac{\partial V}{\partial x}$

AD Dual heuristic programming

Gradients $\frac{\partial Q}{\partial x}, \frac{\partial Q}{\partial u}$

Action NN to approximate the Control

Bertsekas- Neurodynamic Programming

Barto & Bradtke- Q-learning proof (Imposed a settling time)

CT Systems- Derivation of Nonlinear Optimal Regulator

To find online methods for optimal control

Focus on these two equations

Nonlinear System dynamics $\dot{x} = f(x,u) = f(x) + g(x)u$

Cost/value $V(x(t)) = \int_t^\infty r(x,u)\,dt = \int_t^\infty (Q(x) + u^T R u)\,dt$

Leibniz gives Differential equivalent

Bellman Equation, in terms of the Hamiltonian function

$$H(x, \frac{\partial V}{\partial x}, u) = \dot{V} + r(x,u) = \left(\frac{\partial V}{\partial x}\right)^T \dot{x} + r(x,u) = \left(\frac{\partial V}{\partial x}\right)^T \left(f(x) + g(x)u\right) + r(x,u) = 0$$

Stationarity condition $\frac{\partial H}{\partial u} = 0$

Problem- System dynamics shows up in Hamiltonian

Stationary Control Policy $u = h(x) = -\frac{1}{2} R^{-1} g^T(x) \frac{\partial V}{\partial x}$

HJB equation $0 = \left(\frac{dV^*}{dx}\right)^T f + Q(x) - \frac{1}{4}\left(\frac{dV^*}{dx}\right)^T g R^{-1} g^T \frac{dV^*}{dx}$, $V(0) = 0$

Off-line solution

HJB hard to solve. May not have smooth solution.

Dynamics must be known

CT Policy Iteration – a Reinforcement Learning Technique

Given any admissible *policy* $u(x) = h(x)$

The cost is given by solving the CT Bellman equation

$$0 = \left(\frac{\partial V}{\partial x}\right)^T f(x,u) + r(x,u) \equiv H(x, \frac{\partial V}{\partial x}, u)$$

Scalar equation

Utility $r(x,u) = Q(x) + u^T R u$

Policy Iteration Solution

Pick stabilizing initial control policy $h_0(x)$

Policy Evaluation - Find cost, Bellman eq.

$$0 = \left(\frac{\partial V_j}{\partial x}\right)^T f(x, h_j(x)) + r(x, h_j(x)), \quad V_j(0) = 0$$

Policy improvement - Update control

$$h_{j+1}(x) = -\tfrac{1}{2} R^{-1} g^T(x) \frac{\partial V_j}{\partial x}$$

- Convergence proved by Leake and Liu 1967,
  Saridis 1979 if Lyapunov eq. solved exactly
- Beard & Saridis used Galerkin Integrals to solve Lyapunov eq.
- Abu Khalaf & Lewis used NN to approx. V for nonlinear systems and proved convergence

Full system dynamics must be known
Off-line solution

Converges to solution of HJB

$$0 = \left(\frac{dV^*}{dx}\right)^T f + Q(x) - \tfrac{1}{4}\left(\frac{dV^*}{dx}\right)^T g R^{-1} g^T \frac{dV^*}{dx}$$

M. Abu-Khalaf, F.L. Lewis, and J. Huang, "Policy iterations on the Hamilton-Jacobi-Isaacs equation for H-infinity state feedback control with input saturation," IEEE Trans. Automatic Control, vol. 51, no. 12, pp. 1989-1995, Dec. 2006.

Policy Iterations for the Linear Quadratic Regulator

System $\dot{x} = Ax + Bu$

Cost $V(x(t)) = \int_t^\infty (x^T Q x + u^T R u)\, d\tau \qquad = x^T(t) P x(t)$

Differential equivalent is the Bellman equation

$$0 = H(x, \frac{\partial V}{\partial x}, u) = \dot{V} + x^T Q x + u^T R u = 2\left(\frac{\partial V}{\partial x}\right)^T \dot{x} + x^T Q x + u^T R u = 2x^T P(Ax + Bu) + x^T Q x + u^T R u$$

Given any stabilizing FB policy $u = -Kx$

The cost value is found by solving **Lyapunov equation = Bellman equation**

$$0 = (A - BK)^T P + P(A - BK) + Q + K^T R K$$

Optimal Control is

$$u = -R^{-1} B^T P x = -Kx$$

Algebraic Riccati equation

$$0 = PA + A^T P + Q - PBR^{-1} B^T P$$

Full system dynamics must be known
Off-line solution

LQR Policy iteration = Kleinman algorithm

1. For a given control policy $u = -K_j x$ solve for the cost:

$$0 = A_j^T P_j + P_j A_j + Q + K_j^T R K_j$$

Bellman eq. = Lyapunov eq.

Matrix equation

$$A_j = A - BK_j$$

2. Improve policy:

$$K_{j+1} = R^{-1} B^T P_j$$

- If started with a stabilizing control policy K_0 the matrix P_j monotonically converges to the unique positive definite solution of the Riccati equation.
- Every iteration step will return a stabilizing controller.
- The system has to be known.

OFF-LINE DESIGN

MUST SOLVE LYAPUNOV EQUATION AT EACH STEP.

Kleinman 1968

Integral Reinforcement Learning

Work of Draguna Vrabie

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

Can Avoid knowledge of drift term *f(x)*

Policy iteration requires repeated solution of the CT Bellman equation

$$0 = \dot{V} + r(x,u(x)) = \left(\frac{\partial V}{\partial x}\right)^T \dot{x} + r(x,u(x)) = \left(\frac{\partial V}{\partial x}\right)^T f(x,u(x)) + Q(x) + u^T R u \equiv H(x, \frac{\partial V}{\partial x}, u(x))$$

This can be done online without knowing *f(x)*
using measurements of *x(t), u(t)* along the system trajectories

D. Vrabie, O. Pastravanu, M. Abu-Khalaf, and F. L. Lewis, "Adaptive optimal control for continuous-time linear systems based on policy iteration," Automatica, vol. 45, pp. 477-484, 2009.

Integral Reinforcement Learning

Work of Draguna Vrabie 2009

value $$V(x(t)) = \int_t^{\infty} r(x,u)\, d\tau = \int_t^{t+T} r(x,u)\, d\tau + \int_{t+T}^{\infty} r(x,u)\, d\tau$$

Key Idea= US Patent

Lemma 1 – Draguna Vrabie

$$0 = \left(\frac{\partial V}{\partial x}\right)^T f(x,u) + r(x,u) \equiv H(x, \frac{\partial V}{\partial x}, u), \quad V(0) = 0$$

Bad Bellman Equation

Is equivalent to Integral reinf. form (IRL) for the CT Bellman eq.

$$V(x(t)) = \int_t^{t+T} r(x,u)\, d\tau \quad + \quad V(x(t+T)), \quad V(0) = 0$$

Good Bellman Equation

Solves Bellman equation without knowing *f(x,u)*

Allows definition of temporal difference error for CT systems

$$e(t) = -V(x(t)) + \int_t^{t+T} r(x,u)\, d\tau \quad + \quad V(x(t+T))$$

LQR Case

IRL Bellman equation $V(x(t)) = \int_t^{t+T} r(x,u)\, d\tau \quad + \quad V(x(t+T)), \quad V(0) = 0$

Value function $V(x(t)) = x^T(t) P x(t)$

Lemma 1 - D. Vrabie- LQR case

Lyapunov equation

$$A_c{}^T P + P A_c + L^T R L + Q = 0 \qquad , \quad A_c = A - BL$$

is equivalent to Integral Reinforcement Learning form

$$x^T(t) P x(t) = \int_t^{t+T} x^T(\tau)(Q + L^T R L) x(\tau) d\tau \quad + \quad x^T(t+T) P x(t+T)$$

Solves Lyapunov equation without knowing A or B

Proof:

$$\frac{d(x^T P x)}{dt} = x^T (A_c{}^T P + P A_c) x = -x^T (L^T R L + Q) x$$

$$\int_t^{t+T} x^T (Q + L^T R L) x d\tau = -\int_t^{t+T} d(x^T P x) = x^T(t) P x(t) - x^T(t+T) P x(t+T)$$

Integral Reinforcement Learning (IRL)- Draguna Vrabie

IRL Policy iteration

Policy evaluation- IRL Bellman Equation

Cost update $$V_k(x(t)) = \int_t^{t+T} r(x,u_k)\,dt \quad + \quad V_k(x(t+T))$$

CT Bellman eq.

f(x) and *g(x)* do not appear

Equivalent to $$0 = \left(\frac{\partial V}{\partial x}\right)^T f(x,u) + r(x,u) \equiv H(x,\frac{\partial V}{\partial x},u)$$

Solves Bellman eq. (nonlinear Lyapunov eq.) without knowing system dynamics

Policy improvement

Control gain update $$u_{k+1} = h_{k+1}(x) = -\tfrac{1}{2} R^{-1} g^T(x) \frac{\partial V_k}{\partial x}$$

g(x) needed for control update

Initial stabilizing control is needed

Converges to solution to HJB eq. $$0 = \left(\frac{dV^*}{dx}\right)^T f + Q(x) - \tfrac{1}{4}\left(\frac{dV^*}{dx}\right)^T g R^{-1} g^T \frac{dV^*}{dx}$$

D. Vrabie proved convergence to the optimal value and control
Automatica 2009, Neural Networks 2009

CT Policy Iteration – How to implement online?
Linear Systems Quadratic Cost- LQR

Value function is quadratic $V(x(t)) = x^T(t) P x(t)$

Policy evaluation- solve IRL Bellman Equation

$$x^T(t) P_k x(t) = \int_t^{t+T} x^T(\tau)(Q + K_k{}^T R K_k) x(\tau)\, d\tau \quad + \quad x^T(t+T) P_k x(t+T)$$

$$x^T(t) P_k x(t) - x^T(t+T) P_k x(t+T) = \int_t^{t+T} x^T(\tau)(Q + K_k{}^T R K_k) x(\tau)\, d\tau$$

$$\begin{bmatrix} x^1(t) & x^2(t) \end{bmatrix} \begin{bmatrix} p_{11} & p_{12} \\ p_{12} & p_{22} \end{bmatrix} \begin{bmatrix} x^1(t) \\ x^2(t) \end{bmatrix} - \begin{bmatrix} x^1(t+T) & x^2(t+T) \end{bmatrix} \begin{bmatrix} p_{11} & p_{12} \\ p_{12} & p_{22} \end{bmatrix} \begin{bmatrix} x^1(t+T) \\ x^2(t+T) \end{bmatrix}$$

$$= \begin{bmatrix} p_{11} & p_{12} & p_{22} \end{bmatrix} \begin{bmatrix} (x^1)^2 \\ 2x^1x^2 \\ (x^2)^2 \end{bmatrix}_{(t)} - \begin{bmatrix} p_{11} & p_{12} & p_{22} \end{bmatrix} \begin{bmatrix} (x^1)^2 \\ 2x^1x^2 \\ (x^2)^2 \end{bmatrix}_{(t+T)}$$

Quadratic basis set

$$= \bar{p}_k^T \left[\bar{x}(t) - \bar{x}(t+T) \right]$$

$$\bar{p}_k{}^T \phi(t) \equiv \bar{p}_k{}^T \left[\bar{x}(t) - \bar{x}(t+T) \right] = \int_t^{t+T} x(\tau)^T (Q + L_k{}^T R L_k) x(\tau) d\tau \equiv \rho(t, t+T)$$

Same form as standard System ID problems

Nonlinear Case- Approximate Dynamic Programming

Value Function Approximation (VFA) to Solve Bellman Equation
– Paul Werbos (ADP), Dimitri Bertsekas (NDP)

Optimal Control and Adaptive Control come together On this slide. Because of RL

$$V_k(x(t)) = \int_t^{t+T} \left(Q(x) + u_k{}^T R u_k \right) dt + V_k(x(t+T))$$

Approximate value by Weierstrass Approximator Network $V = W^T \phi(x)$

$$W_k^T \phi(x(t)) = \int_t^{t+T} \left(Q(x) + u_k{}^T R u_k \right) dt + W_k^T \phi(x(t+T))$$

$$W_k^T \underbrace{\left[\phi(x(t)) - \phi(x(t+T)) \right]}_{\text{regression vector}} = \underbrace{\int_t^{t+T} \left(Q(x) + u_k{}^T R u_k \right) dt}_{\text{Reinforcement on time interval } [t,\ t+T]}$$

Scalar equation with vector unknowns

Same form as standard System ID problems in Adaptive Control

Now use RLS along the trajectory to get new weights W_k

Then find updated FB

$$u_{k+1} = h_{k+1}(x) = -\tfrac{1}{2} R^{-1} g^T(x) \frac{\partial V_k}{\partial x} = -\tfrac{1}{2} R^{-1} g^T(x) \left[\frac{\partial \phi(x(t))}{\partial x(t)} \right]^T W_k$$

Direct Optimal Adaptive Control for Partially Unknown CT Systems

Solving the IRL Bellman Equation

Solve for value function parameters $\begin{bmatrix} p_{11} & p_{12} \\ p_{12} & p_{22} \end{bmatrix}$ $\qquad W^T = \begin{bmatrix} p_{11} & p_{12} & p_{22} \end{bmatrix}$

Need data from 3 time intervals to get 3 equations to solve for 3 unknowns

$$W_k^T \left[\Delta\phi(x(t))\right] \equiv W_k^T \left[\phi(x(t)) - \phi(x(t+T))\right] = \int_t^{t+T} \left(Q(x) + u_k{}^T R u_k\right) dt \equiv \rho(t)$$

$$W_k^T \left[\Delta\phi(x(t+T))\right] \equiv W_k^T \left[\phi(x(t+T)) - \phi(x(t+2T))\right] = \int_{t+T}^{t+2T} \left(Q(x) + u_k{}^T R u_k\right) dt \equiv \rho(t+T)$$

$$W_k^T \left[\Delta\phi(x(t+2T))\right] \equiv W_k^T \left[\phi(x(t+2T)) - \phi(x(t+3T))\right] = \int_{t+2T}^{t+3T} \left(Q(x) + u_k{}^T R u_k\right) dt \equiv \rho(t+2T)$$

Put together

$$W_k^T \left[\Delta\phi(x(t)) \;\; \Delta\phi(x(t+T)) \;\; \Delta\phi(x(t+2T))\right] = \left[\rho(t) \;\; \rho(t+T) \;\; \rho(t+2T)\right]$$

Now solve by Batch least-squares

Or can use Recursive Least-Squares (RLS)

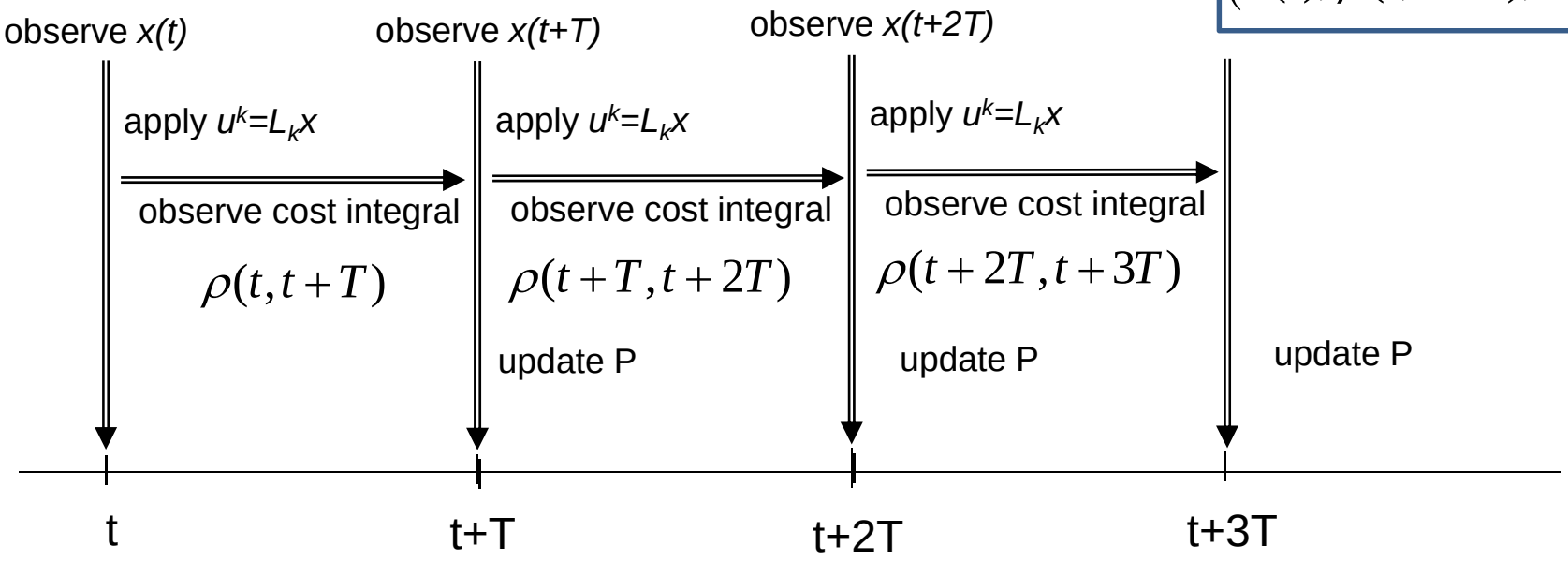
Integral Reinforcement Learning (IRL)

Solve Bellman Equation - Solves Lyapunov eq. without knowing dynamics

$$W_k^T\left[\phi(x(t))-\phi(x(t+T))\right]=\int_t^{t+T} x(\tau)^T\left(Q+K_k^T R K_k\right)x(\tau)d\tau=\rho(t,t+T)$$

Data set at time *[t,t+T)*

$$\left(x(t),\rho(t,t+T),x(t+T)\right)$$



Do RLS until convergence to P_k

Or use batch least-squares

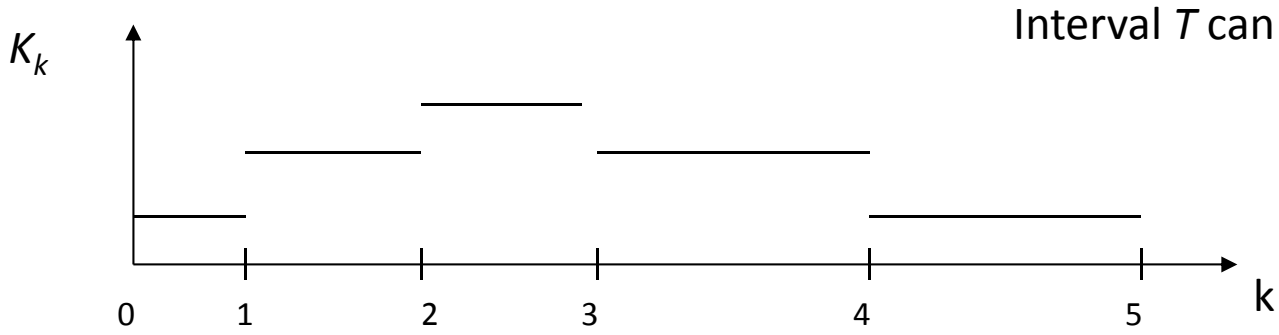
A is not needed anywhere

This is a data-based approach that uses measurements of *x(t), u(t)*
Instead of the plant dynamical model.

update control gain

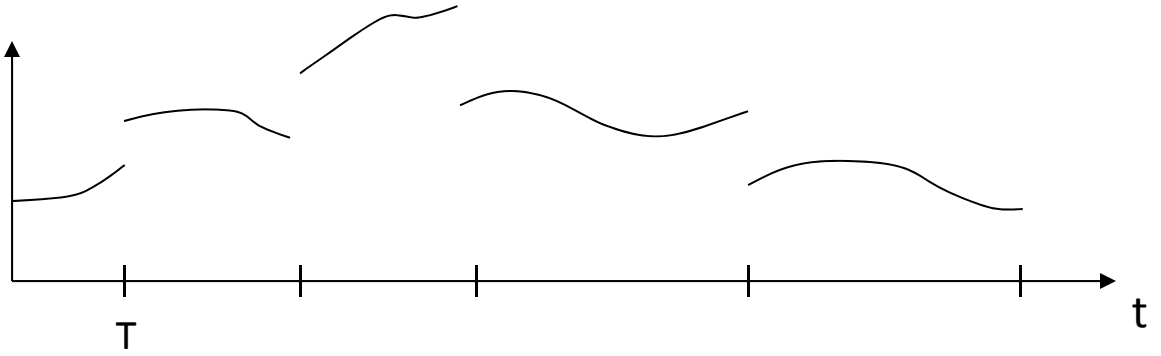
$$K_{k+1}=R^{-1}B^T P_k$$

Gain update (Policy)



Control

$$u_k(t) = -K_k x(t)$$



Reinforcement Intervals T need not be the same
They can be selected on-line in real time

Continuous-time control with discrete gain updates

Persistence of Excitation

$$W_k^T \left[\phi(x(t)) - \phi(x(t+T))\right] = \int_t^{t+T} \left(Q(x) + u_k{}^T R u_k\right) dt$$

Regression vector must be PE

Relates to choice of reinforcement interval T

Implementation

Policy evaluation
Need to solve online

$$W_k^T\left[\phi(\mathrm{x}(t))-\phi(x(t+T))\right]=\int_t^{t+T} x(\tau)^T(Q+K_k^T RK_k)x(\tau)d\tau=\rho(t,t+T)$$

Add a new state= Integral Reinforcement

$$\dot{\rho} = x^T Q x + u^T R u$$

This is the controller dynamics or memory

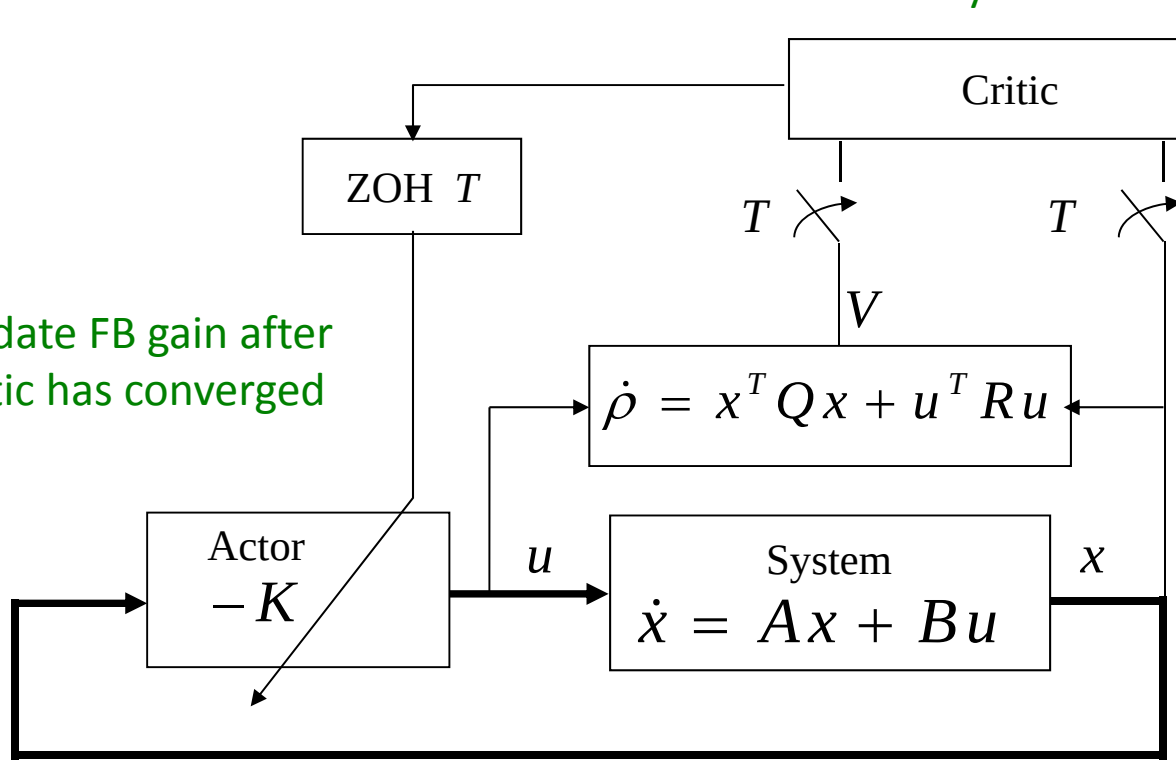
Draguna Vrabie

Direct Optimal Adaptive Controller

Solves Riccati Equation Online without knowing A matrix

CT time Actor-Critic Structure

Run RLS or use batch L.S.
To identify value of current control



Dynamic Control System w/ MEMORY

Update FB gain after Critic has converged

A hybrid continuous/discrete dynamic controller
whose internal state is the observed cost over the interval

Reinforcement interval T can be selected on line on the fly – can change

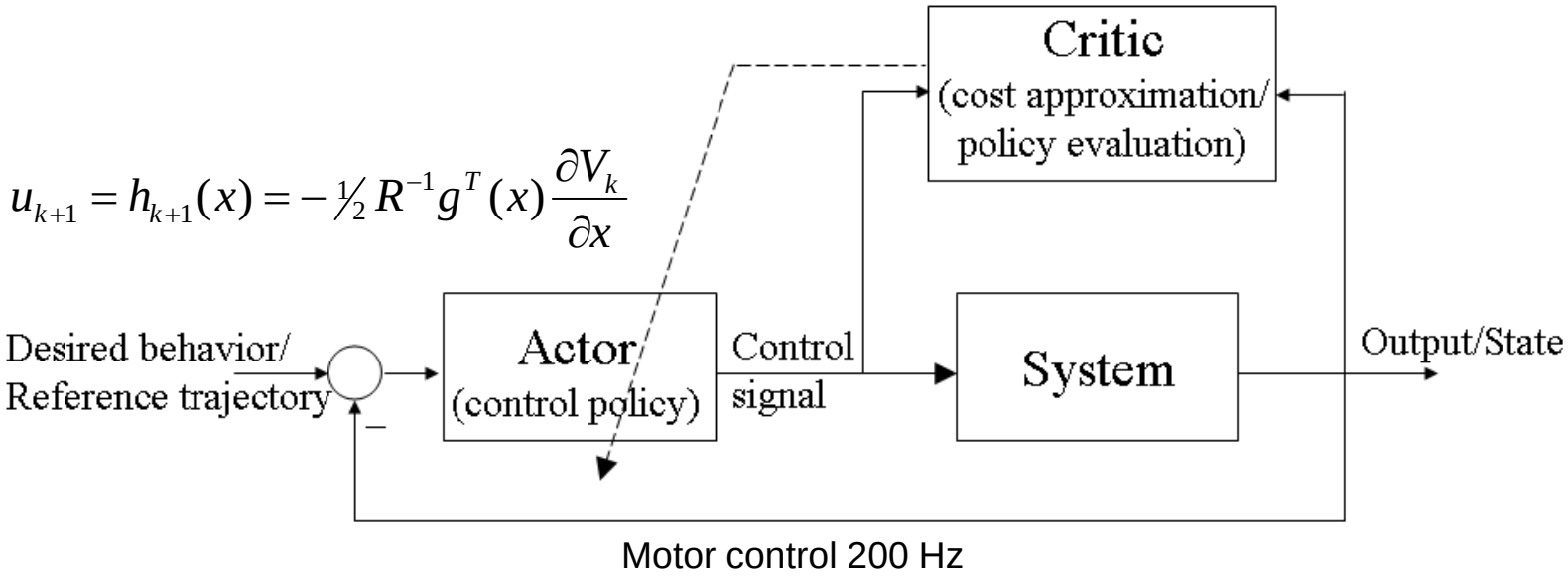
Optimal Adaptive IRL for CT systems

D. Vrabie, 2009

Actor / Critic structure for CT Systems

Reinforcement learning $V_k(x(t)) = \int_t^{t+T} r(x,u_k)\,dt \quad + \quad V_k(x(t+T))$

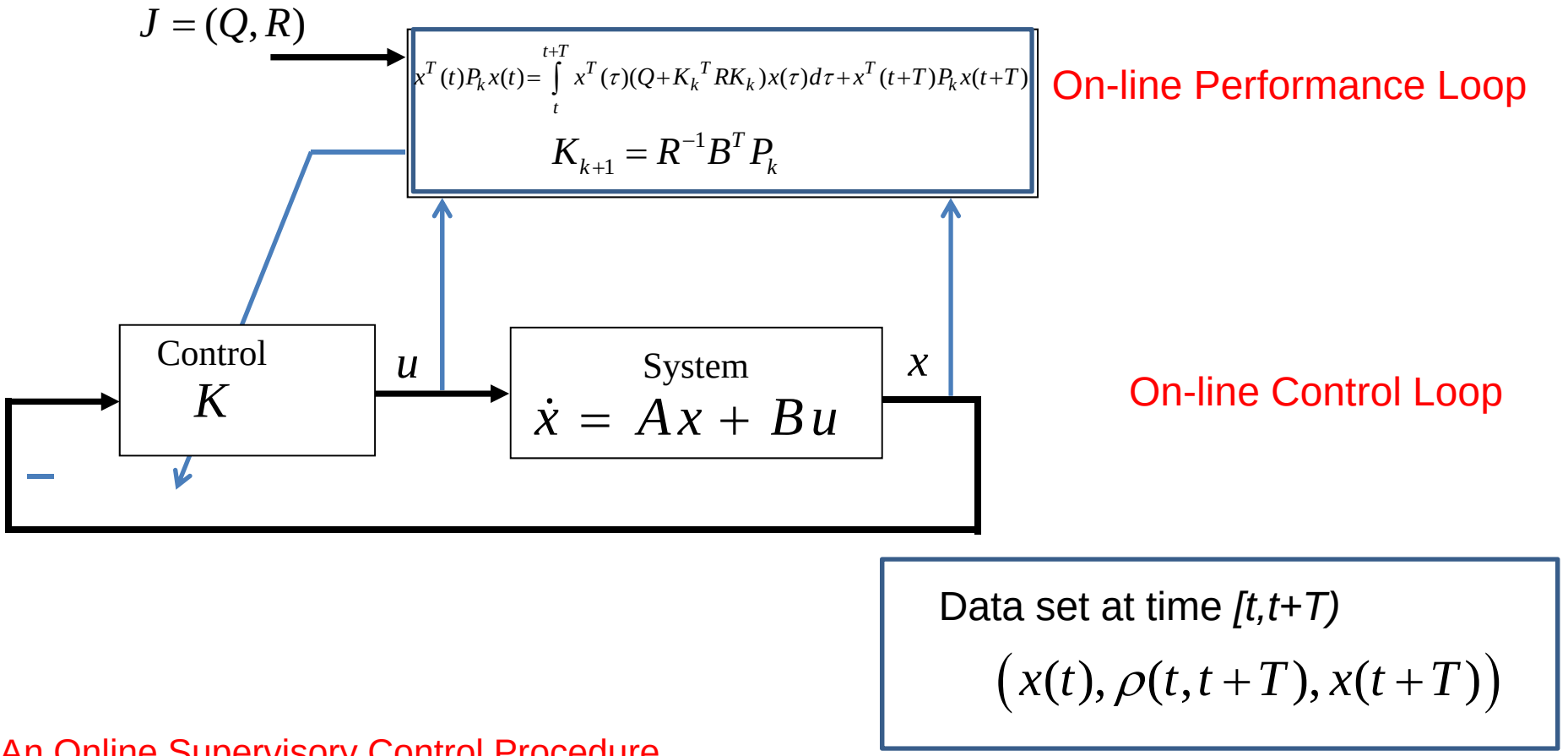
Theta waves 4-8 Hz



A new structure of adaptive controllers

Data-driven Online Adaptive Optimal Control DDO

User prescribed optimization criterion



An Online Supervisory Control Procedure
that requires no Knowledge of system dynamics model A

Automatically tunes the control gains in real time to optimize a user given cost function
Uses measured data *(u(t),x(t))* along system trajectories

Simulation 1- F-16 aircraft pitch rate controller

$$\dot{x} = \begin{bmatrix} -1.01887 & 0.90506 & -0.00215 \\ 0.82225 & -1.07741 & -0.17555 \\ 0 & 0 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u$$

Stevens and Lewis 2003

$x = [\alpha \quad q \quad \delta_e]$

$Q = I, \quad R = I$

ARE $\quad 0 = PA + A^T P + Q - PBR^{-1}B^T P$

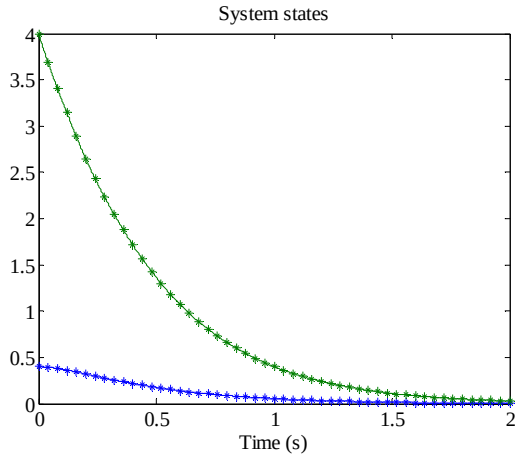
Select quadratic NN basis set for VFA

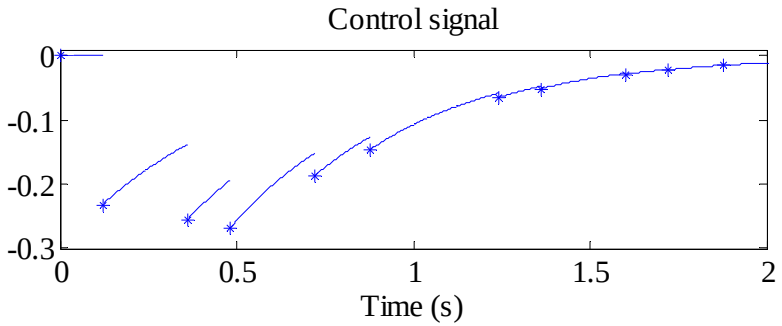
Exact solution $\quad W_1^* = [p_{11} \quad 2p_{12} \quad 2p_{13} \quad p_{22} \quad 2p_{23} \quad p_{33}]^T$

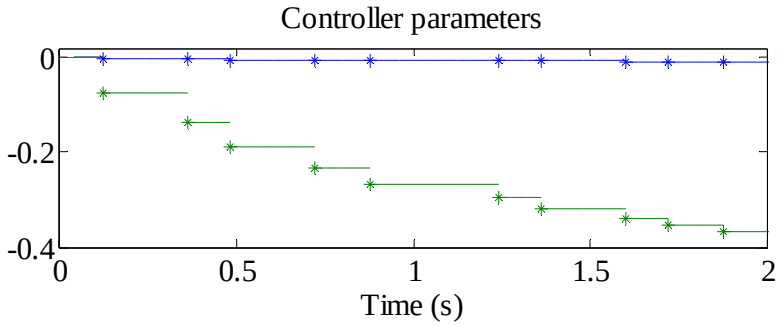
$= [1.4245 \quad 1.1682 \quad -0.1352 \quad 1.4349 \quad -0.1501 \quad 0.4329]^T$

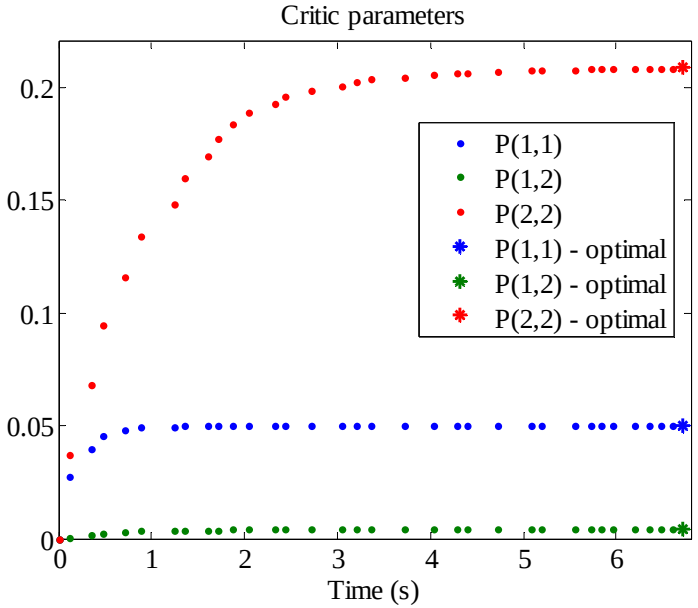
Simulations on: F-16 autopilot

A matrix not needed









Converge to SS Riccati equation soln

Solves ARE online without knowing A

$$0 = PA + A^T P + Q - PBR^{-1}B^T P$$

A. Al-Tamimi, D. Vrabie, Youyi Wang

Simulation 2: Load Frequency Control of Electric Power system

$$\dot{x} = Ax + Bu$$

$$x(t) = [\Delta f(t) \quad \Delta P_g(t) \quad \Delta X_g(t) \quad \Delta E(t)]^T$$

Frequency
Generator output
Governor position
Integral control

$$A = \begin{bmatrix} -1/T_p & K_p/T_p & 0 & 0 \\ 0 & -1/T_T & 1/T_T & 0 \\ -1/RT_G & 0 & -1/T_G & -1/T_G \\ K_E & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ 1/T_G \\ 0 \end{bmatrix}$$

ARE

$$0 = PA + A^T P + Q - PBR^{-1}B^T P$$

ARE solution using full dynamics model (A,B)

$$P_{ARE} = \begin{bmatrix} 0.4750 & 0.4766 & 0.0601 & 0.4751 \\ 0.4766 & 0.7831 & 0.1237 & 0.3829 \\ 0.0601 & 0.1237 & 0.0513 & 0.0298 \\ 0.4751 & 0.3829 & 0.0298 & 2.3370 \end{bmatrix}.$$

$$0 = PA + A^T P + Q - PBR^{-1}B^T P$$

Solves ARE online without knowing A

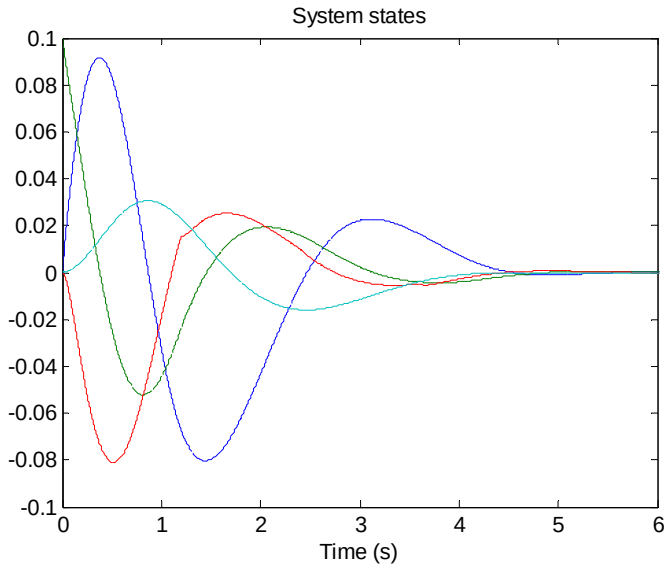
$$P_{ARE} = \begin{bmatrix} 0.4750 & 0.4766 & 0.0601 & 0.4751 \\ 0.4766 & 0.7831 & 0.1237 & 0.3829 \\ 0.0601 & 0.1237 & 0.0513 & 0.0298 \\ 0.4751 & 0.3829 & 0.0298 & 2.3370 \end{bmatrix}.$$

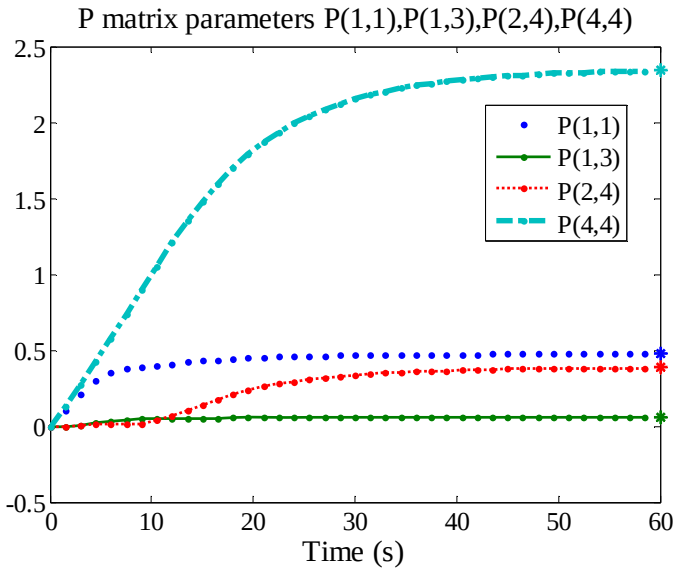
$$P_{critic\ NN} = \begin{bmatrix} 0.4802 & 0.4768 & 0.0603 & 0.4754 \\ 0.4768 & 0.7887 & 0.1239 & 0.3834 \\ 0.0603 & 0.1239 & 0.0567 & 0.0300 \\ 0.4754 & 0.3843 & 0.0300 & 2.3433 \end{bmatrix}.$$

IRL period of T= 0.1s.

Fifteen data points $\left(x(t), x(t+T), \rho(t:t+T)\right)$

Hence, the value estimate was updated every 1.5s.





Optimal Control Design Allows a Lot of Design Freedom

The Power of Optimal Design

Once you can do optimal design that minimizes a performance index, many sorts of designs are immediately possible.

Minimum energy

$$J = \frac{1}{2}\int_0^\infty x^T Q x + u^T R u \, dt$$

Minimum fuel

$$J = \frac{1}{2}\int_0^\infty x^T Q x + \rho |u| \, dt$$

Minimum time

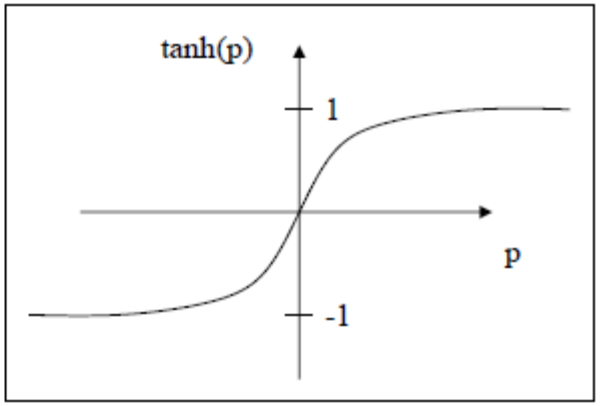
$$J = \int_0^T 1 \, dt = T$$

Constrained control inputs

$$J = \frac{1}{2}\int_0^\infty \left(Q(x) + \int_0^u \sigma^{-1}(v) dv \right) dt$$

Approximate minimum time with smooth control inputs

$$J = \frac{1}{2}\int_0^\infty \left(\tanh(x^T Q x) + \rho \int_0^u \sigma^{-1}(v) dv \right) dt$$



IRL Value Iteration - Draguna Vrabie

IRL Policy iteration Initial stabilizing control is needed

Policy evaluation- IRL Bellman Equation

Cost update $$V_k(x(t)) = \int_t^{t+T} r(x,u_k)\,dt \quad + \quad V_k(x(t+T))$$

CT PI Bellman eq. = Lyapunov eq.

Policy improvement

Control gain update $$u_{k+1} = h_{k+1}(x) = -\tfrac{1}{2} R^{-1} g^T(x) \frac{\partial V_k}{\partial x}$$

Converges to solution to HJB eq. $$0 = \left(\frac{dV^*}{dx}\right)^T f + Q(x) - \tfrac{1}{4}\left(\frac{dV^*}{dx}\right)^T g R^{-1} g^T \frac{dV^*}{dx}$$

IRL Value iteration Initial stabilizing control is NOT needed

Value evaluation- IRL Bellman Equation

Cost update $$V_{k+1}(x(t)) = \int_t^{t+T} r(x,u_k)\,dt \quad + \quad V_k(x(t+T))$$

CT VI Bellman eq.

Policy improvement

Control gain update $$u_{k+1} = h_{k+1}(x) = -\tfrac{1}{2} R^{-1} g^T(x) \frac{\partial V_{k+1}}{\partial x}$$

Converges if T is small enough

Kung Tz 500 BC

Confucius

孔子

Tian xia da tong
Harmony under heaven

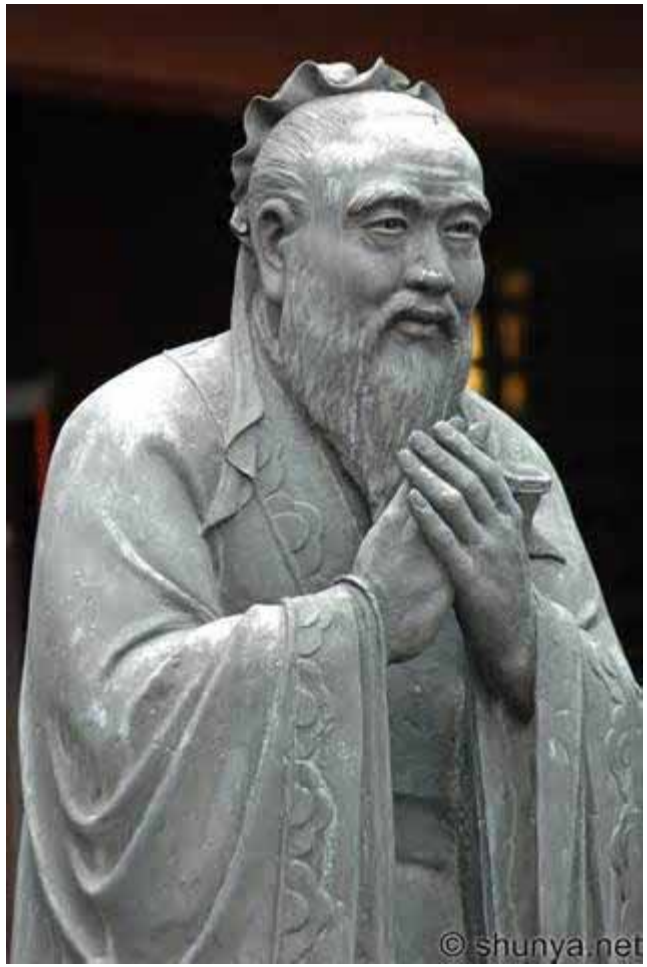
Archery
Chariot driving

Music
Rites and Rituals

Poetry
Mathematics



Man's relations to
- Family
- Friends
- Society
- Nation
- Emperor
- Ancestors





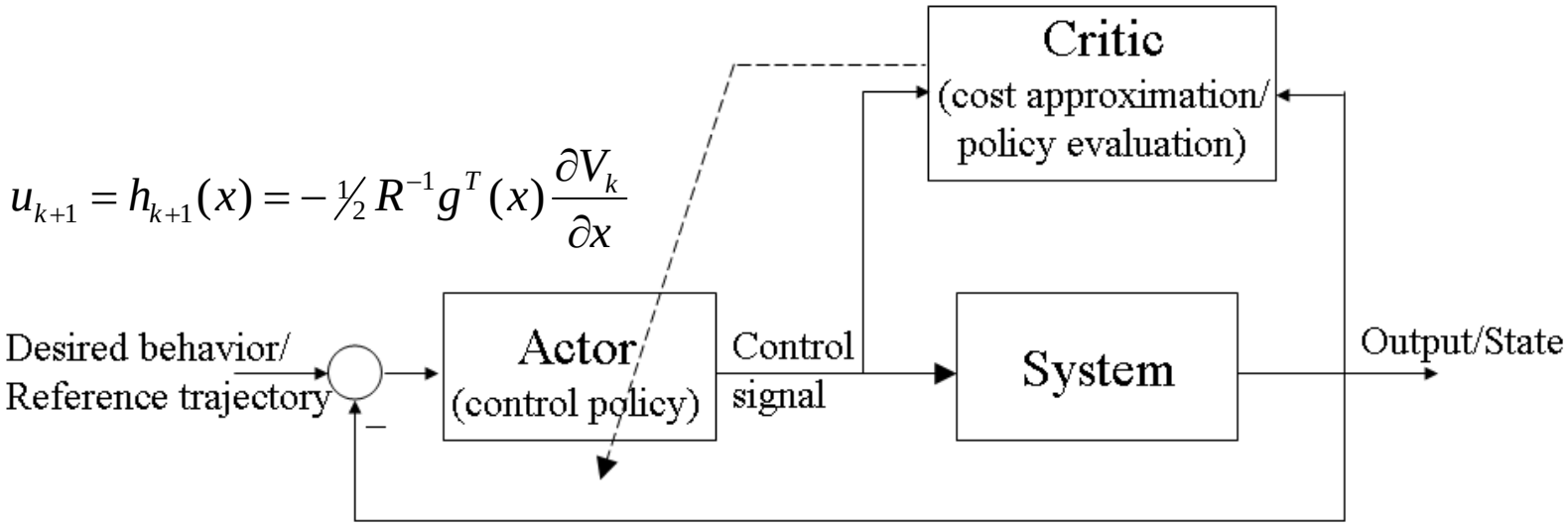
Optimal Adaptive IRL for CT systems

D. Vrabie, 2009

Actor / Critic structure for CT Systems

Reinforcement learning $V_k(x(t)) = \int_t^{t+T} r(x,u_k)\,dt \quad + \quad V_k(x(t+T))$

Theta waves 4-8 Hz



A new structure of adaptive controllers

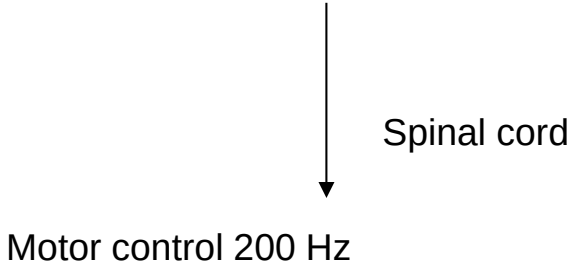
Oscillation is a fundamental property of neural tissue

Brain has multiple adaptive clocks with different timescales

gamma rhythms 30-100 Hz, hippocampus and neocortex
high cognitive activity.
- consolidation of memory
- spatial mapping of the environment – place cells

The high frequency processing is due to the large amounts of sensorial data to be processed

theta rhythm, Hippocampus, Thalamus, 4-10 Hz
sensory processing, memory and voluntary control of movement.



D. Vrabie, F.L. Lewis, D. Levine, “Neural Network-Based Adaptive Optimal Controller- A Continuous-Time Formulation -,” Proc. Int. Conf. Intelligent Control, Shanghai, Sept. 2008.

D. Vrabie and F.L. Lewis, “Neural network approach to continuous-time direct adaptive optimal control for partially-unknown nonlinear systems,” Neural Networks, vol. 22, no. 3, pp. 237-246, Apr. 2009.

Deliberative evaluation

Limbic system

control

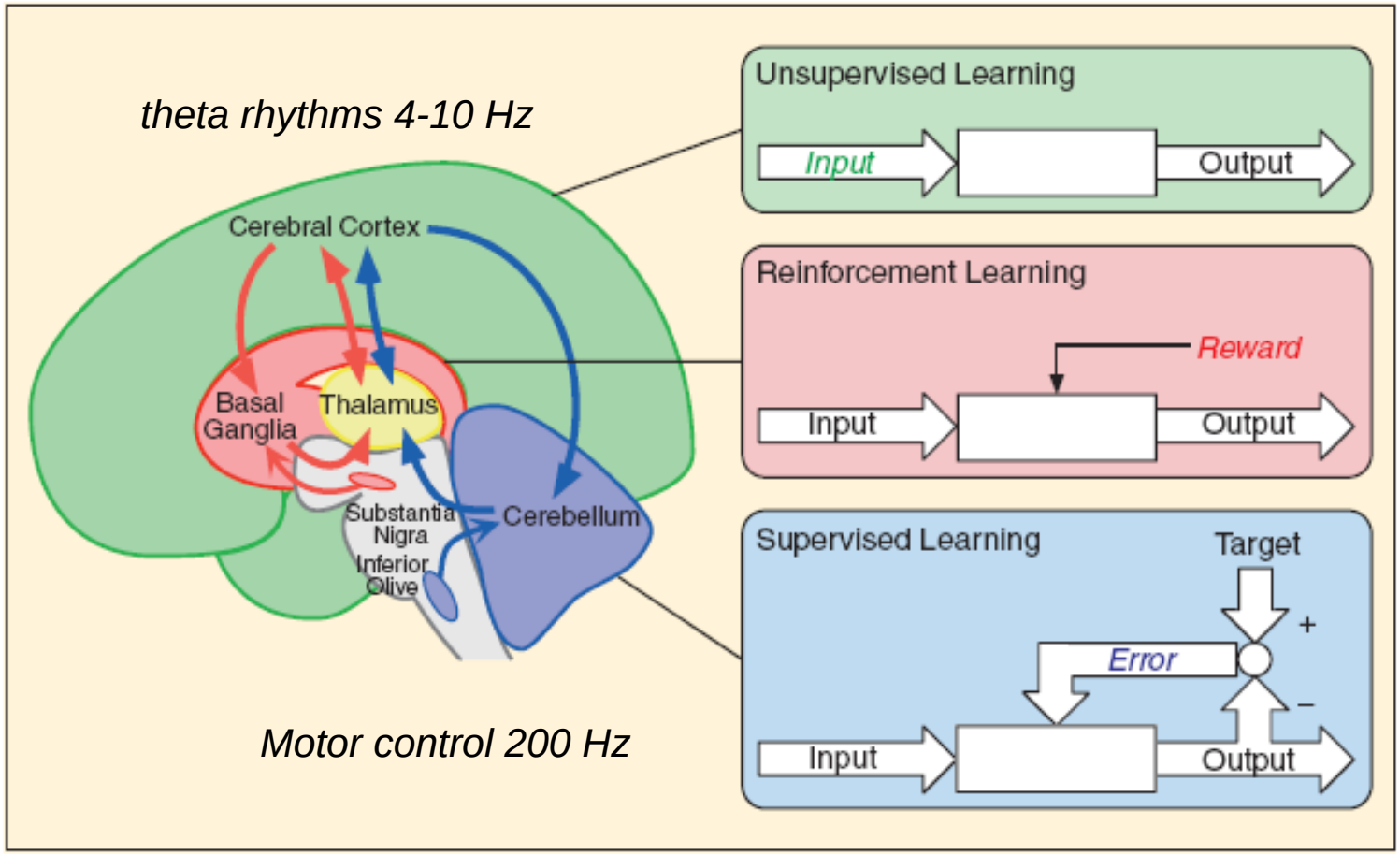
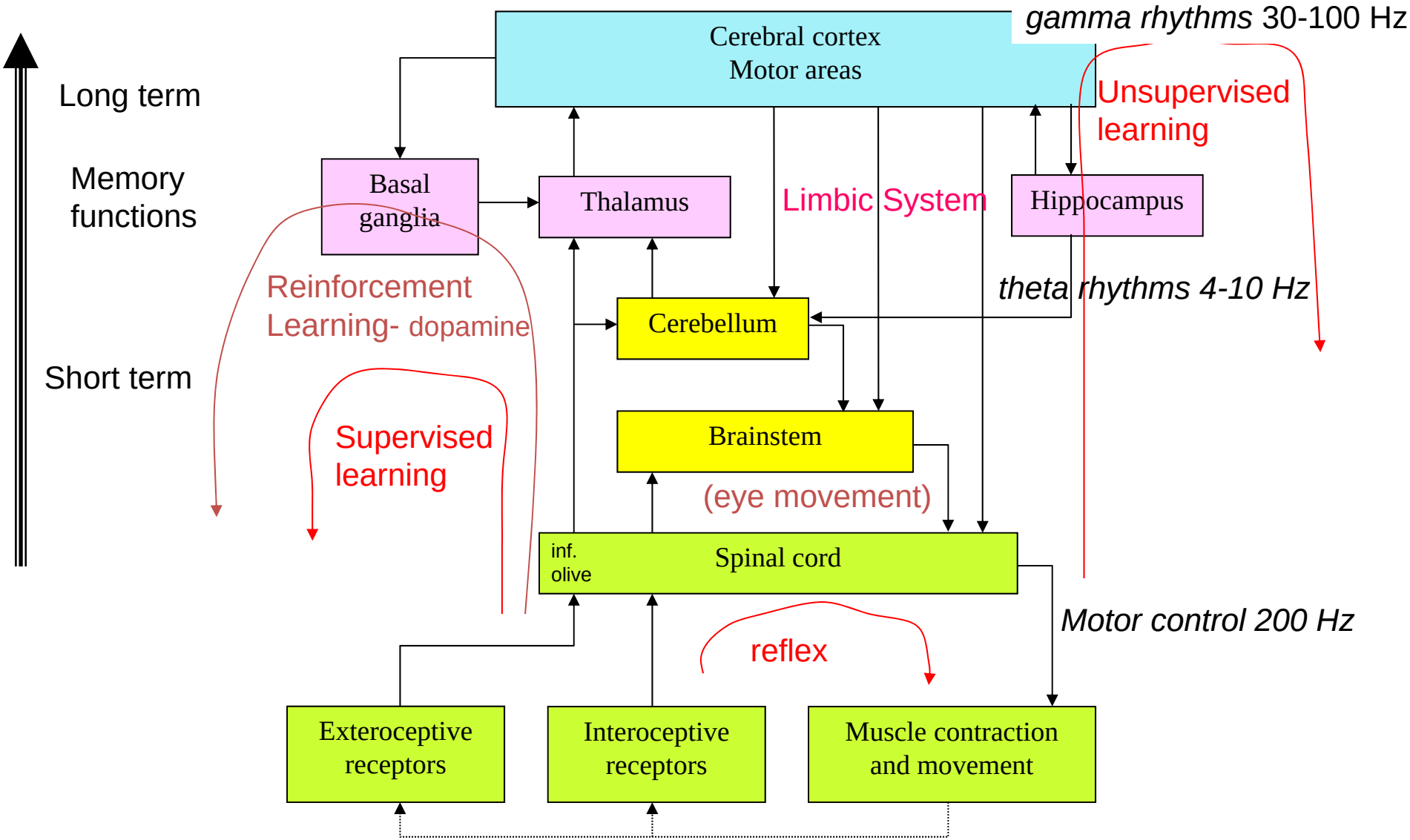


Figure 1. *Learning-oriented specialization of the cerebellum, the basal ganglia, and the cerebral cortex [1], [2]. The cerebellum is specialized for supervised learning based on the error signal encoded in the climbing fibers from the inferior olive. The basal ganglia are specialized for reinforcement learning based on the reward signal encoded in the dopaminergic fibers from the substantia nigra. The cerebral cortex is specialized for unsupervised learning based on the statistical properties of the input signal.*

Doya, Kimura, Kawato 2001

Summary of Motor Control in the Human Nervous System

picture by E. Stingu
D. Vrabie



Hierarchy of multiple parallel loops




Synchronous Real-time Data-driven Optimal Control

Optimal Adaptive
Integral Reinforcement Learning for CT systems

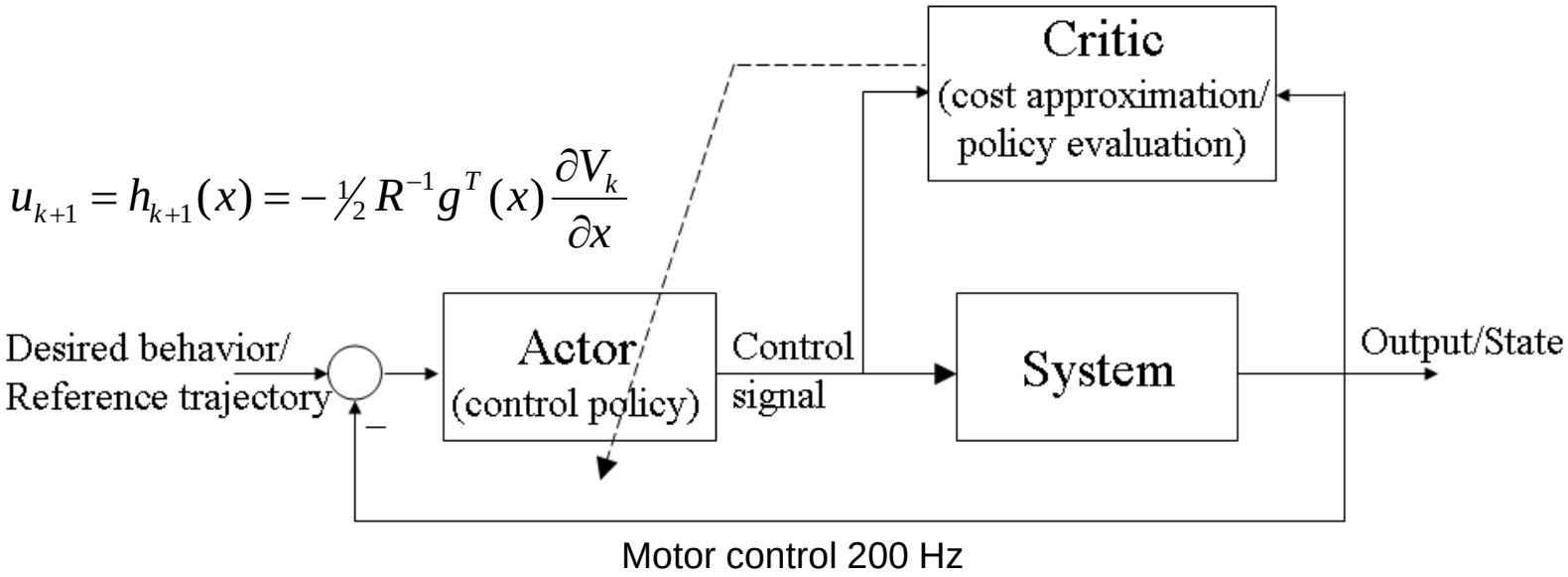
D. Vrabie, 2009

Policy Iteration gives the structure needed for online optimal solution

Actor / Critic structure for CT Systems

$$V_k(x(t)) = \int_t^{t+T} r(x,u_k)\,dt \quad + \quad V_k(x(t+T))$$

Theta waves 4-8 Hz



A new structure of adaptive controllers

Critic Network

Take VFA as $V(x) = \hat{W}_1^T \phi_1(x) + \varepsilon(x)$, $\nabla V(x) = \nabla \phi_1^T \hat{W}_1$

Then IRL Bellman eq $V(x(t)) = \int_{t-T}^{t} \left(Q(x) + u_k{}^T R u_k \right) dt + V(x(t+T))$

becomes $\hat{W}_1^T \phi(x(t-T)) = \int_{t-T}^{t} \left(Q(x) + u_k{}^T R u_k \right) dt + \hat{W}_1^T \phi(x(t))$

Action Network for Control Approximation

$$u(x) = -\frac{1}{2} R^{-1} g^T(x) \nabla \phi_1{}^T \hat{W}_2,$$

Define $\Delta\phi(x(t)) \equiv \phi(x(t)) - \phi(x(t-T))$

Bellman eq becomes $\Delta\phi(x(t))^T \hat{W}_1 + \int_{t-T}^{t} \left(Q(x) + \frac{1}{4} \hat{W}_2{}^T \overline{D}_1 \hat{W}_2 \right) = 0$

Data-driven Online Synchronous Policy Iteration using IRL

Does not need to know *f(x)*

Vamvoudakis & Vrabie

Theorem (Vamvoudakis & Vrabie)- Online Learning of Nonlinear Optimal Control

Let $\Delta\phi(x(t)) \equiv \phi(x(t)) - \phi(x(t-T))$ be PE. Tune critic NN weights as

$$\dot{\hat{W}}_1 = -a_1 \frac{\Delta\phi(x(t))}{\left(1+\Delta\phi(x(t))^T \Delta\phi(x(t))\right)^2}\left(\Delta\phi(x(t))^T \hat{W}_1 + \int_{t-T}^{t}\left(Q(x) + \frac{1}{4}\hat{W}_2{}^T \overline{D}_1 \hat{W}_2\right)d\tau\right)$$

Learning the Value

Tune actor NN weights as

$$\dot{\hat{W}}_2 = -a_2\left(F_2\hat{W}_2 - F_1\Delta\phi(x(t))^T \hat{W}_1\right) - \tfrac{1}{4}a_2 \overline{D}_1(x)\hat{W}_2 \frac{\Delta\phi(x(t))^T}{\left(1+\Delta\phi(x(t))^T \Delta\phi(x(t))\right)^2}\hat{W}_1$$

Learning the control policy

Then there exists an N_0 such that, for the number of hidden layer units $N > N_0$

the closed-loop system state, the critic NN error $\tilde{W}_1 = W_1 - \hat{W}_1$

and the actor NN error $\tilde{W}_2 = W_1 - \hat{W}_2$ are UUB bounded.

Data set at time *[t,t+T)*

$$\left(x(t), \rho(t-T,t), x(t-T)\right)$$

Lyapunov energy-based Proof:

$$L(t) = V(x) + \frac{1}{2} tr(\tilde{W}_1^T a_1^{-1} \tilde{W}_1) + \frac{1}{2} tr(\tilde{W}_2^T a_2^{-1} \tilde{W}_2).$$

V(x)= Unknown solution to HJB eq.

$$0 = \left(\frac{dV}{dx}\right)^T f + Q(x) - \tfrac{1}{4}\left(\frac{dV}{dx}\right)^T g R^{-1} g^T \frac{dV}{dx}$$

Guarantees stability

$$\tilde{W}_1 = W_1 - \hat{W}_1$$

$$\tilde{W}_2 = W_1 - \hat{W}_2$$

W_1= Unknown LS solution to Bellman equation for given N

$$H(x, W_1, u) = W_1^T \nabla \phi_1 (f + gu) + Q(x) + u^T R u = \varepsilon_H$$

Synchronous Online Solution of Optimal Control for Nonlinear Systems

K.G. Vamvoudakis and F.L. Lewis, "Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem," Automatica, vol. 46, no. 5, pp. 878-888, May 2010.

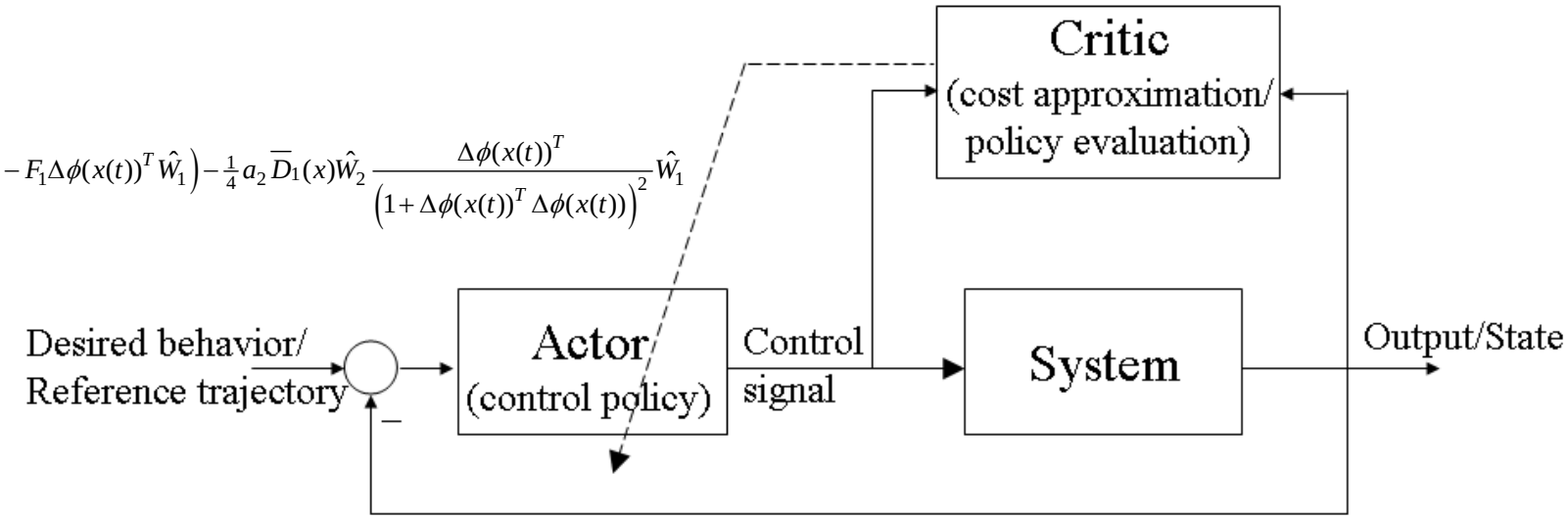
A new form of Adaptive Control with TWO tunable networks

Adaptive Critic structure

Reinforcement learning

$$\dot{\hat{W}}_1 = -a_1 \frac{\Delta\phi(x(t))}{\left(1+\Delta\phi(x(t))^T \Delta\phi(x(t))\right)^2}\left(\Delta\phi(x(t))^T \hat{W}_1 + \int_{t-T}^{t}\left(Q(x)+\frac{1}{4}\hat{W}_2{}^T \overline{D}_1 \hat{W}_2\right)d\tau\right)$$

$$\dot{\hat{W}}_2 = -a_2\left(F_2\hat{W}_2 - F_1\Delta\phi(x(t))^T \hat{W}_1\right) - \tfrac{1}{4}a_2 \overline{D}_1(x)\hat{W}_2 \frac{\Delta\phi(x(t))^T}{\left(1+\Delta\phi(x(t))^T \Delta\phi(x(t))\right)^2}\hat{W}_1$$

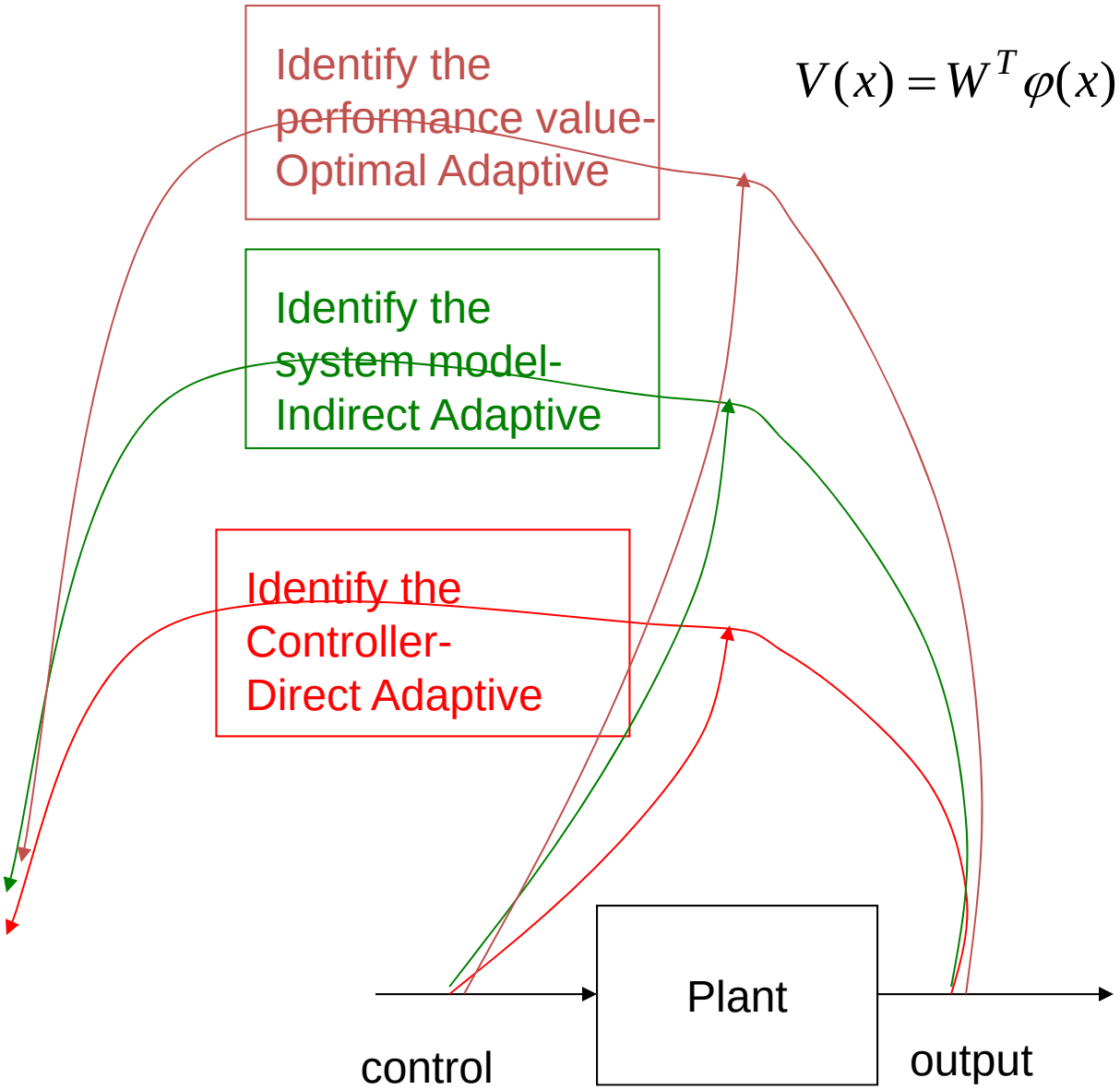


Two Learning Networks

Tune them Simultaneously

A new structure of adaptive controllers

A New Class of Adaptive Control



Simulation 1- F-16 aircraft pitch rate controller

$$\dot{x} = \begin{bmatrix} -1.01887 & 0.90506 & -0.00215 \\ 0.82225 & -1.07741 & -0.17555 \\ 0 & 0 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u$$

Stevens and Lewis 2003

$$x = [\alpha \quad q \quad \delta_e]$$

$$Q = I, \quad R = I$$

Solves ARE online

$$0 = PA + A^T P + Q - PBR^{-1}B^T P$$

Select quadratic NN basis set for VFA

Exact solution $W_1^* = [p_{11} \quad 2p_{12} \quad 2p_{13} \quad p_{22} \quad 2p_{23} \quad p_{33}]^T$

$= [1.4245 \quad 1.1682 \quad -0.1352 \quad 1.4349 \quad -0.1501 \quad 0.4329]^T$

Must add probing noise to get PE

$$u(x) = -\frac{1}{2} R^{-1} g^T(x) \nabla \phi_1^T \hat{W}_2 + n(t)$$

(exponentially decay *n(t)*)

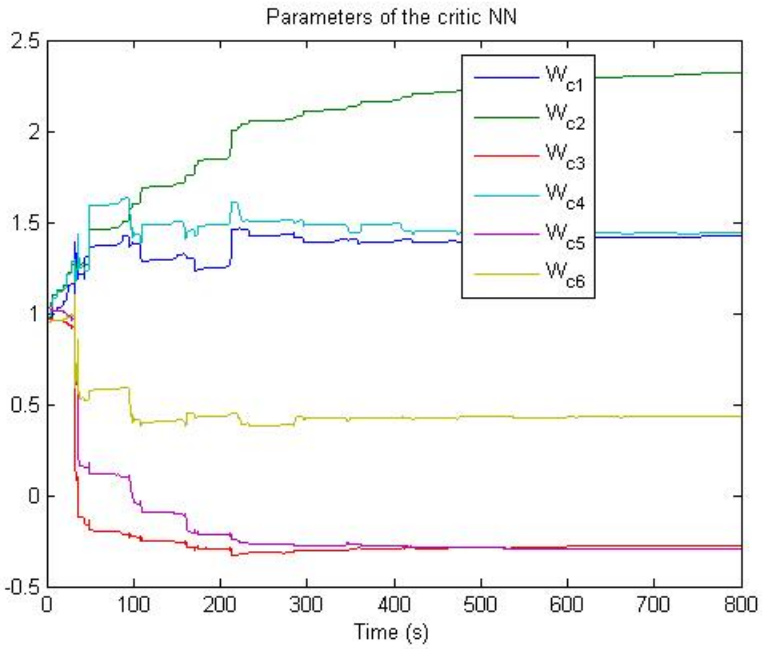
Algorithm converges to

$\hat{W}_1(t_f) = [1.4279 \quad 1.1612 \quad -0.1366 \quad 1.4462 \quad -0.1480 \quad 0.4317]^T$.

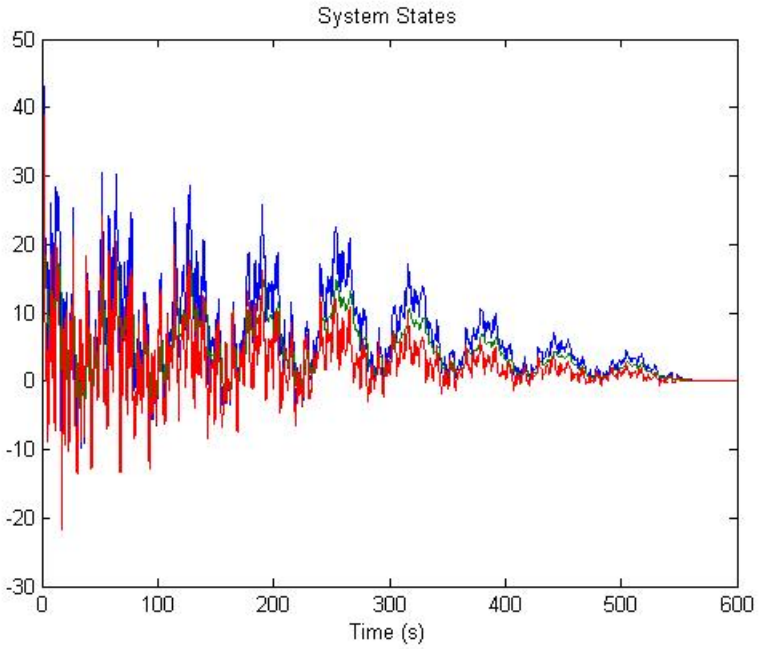
$\hat{W}_2(t_f) = [1.4279 \quad 1.1612 \quad -0.1366 \quad 1.4462 \quad -0.1480 \quad 0.4317]^T$

$$\hat{u}_2(x) = -\frac{1}{2} R^{-1} B^T P x = -\frac{1}{2} R^{-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}^T \begin{bmatrix} 2x_1 & 0 & 0 \\ x_2 & x_1 & 0 \\ x_3 & 0 & x_1 \\ 0 & 2x_2 & 0 \\ 0 & x_3 & x_2 \\ 0 & 0 & 2x_3 \end{bmatrix}^T \begin{bmatrix} 1.4279 \\ 1.1612 \\ -0.1366 \\ 1.4462 \\ -0.1480 \\ 0.4317 \end{bmatrix}$$

Critic NN parameters-
Converge to ARE solution



System states



Nevistic V. and Primbs J. A. (1996)
Converse optimal

Simulation 2. – Nonlinear System

$$\dot{x} = f(x) + g(x)u, \qquad x \in R^2$$

$$f(x) = \begin{bmatrix} -x_1 + x_2 \\ -0.5x_1 - 0.5x_2(1-(\cos(2x_1)+2)^2) \end{bmatrix}$$

$$g(x) = \begin{bmatrix} 0 \\ \cos(2x_1)+2 \end{bmatrix}.$$

$Q = I, \quad R = I$

Solves HJB equation online

$$0 = \left(\frac{dV^*}{dx}\right)^T f + Q(x) - \tfrac{1}{4}\left(\frac{dV^*}{dx}\right)^T g R^{-1} g^T \frac{dV^*}{dx}$$

Optimal Value $\quad V^*(x) = \frac{1}{2}x_1^2 + x_2^2$

Optimal control $\quad u^*(x) = -(\cos(2x_1)+2)x_2.$

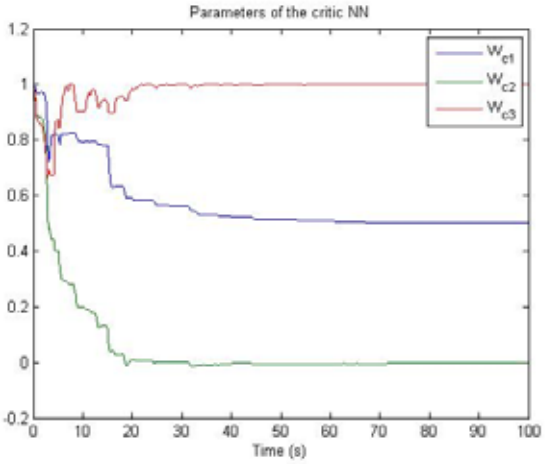
Select VFA basis set $\quad \phi_1(x) = [\; x_1^2 \quad x_1x_2 \quad x_2^2\;]^T,$

Algorithm converges to

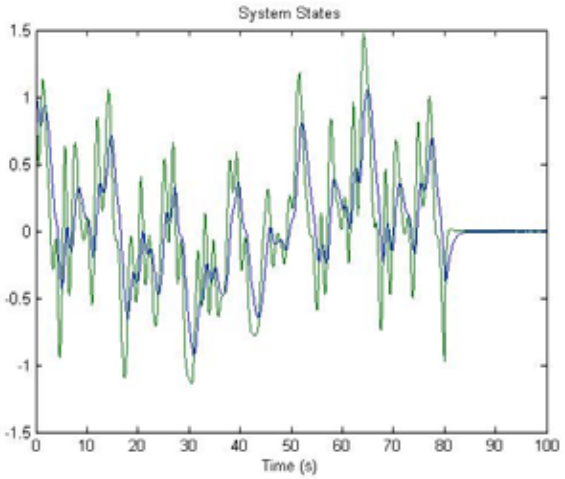
$\hat{W}_1(t_f) = [0.5017 \quad -0.0020 \quad 1.0008]^T.$

$\hat{W}_2(t_f) = [0.5017 \quad -0.0020 \quad 1.0008]^T.$

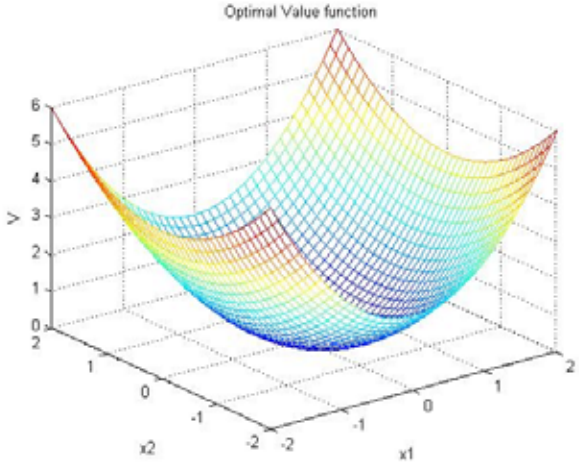
$$\hat{u}_2(x) = -\tfrac{1}{2}R^{-1}\begin{bmatrix} 0 \\ \cos(2x_1)+2 \end{bmatrix}^T \begin{bmatrix} 2x_1 & 0 \\ x_2 & x_1 \\ 0 & 2x_2 \end{bmatrix}^T \begin{bmatrix} 0.5017 \\ -0.0020 \\ 1.0008 \end{bmatrix}$$



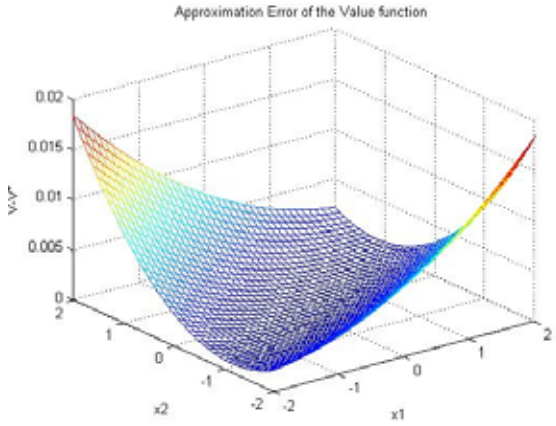
Critic NN parameters



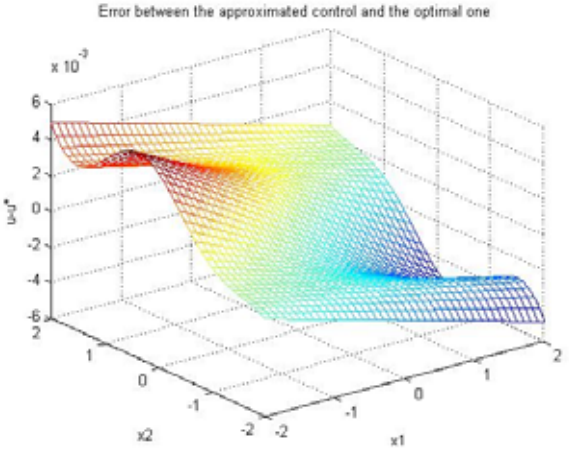
states



Optimal value fn.



Value fn. approx. error

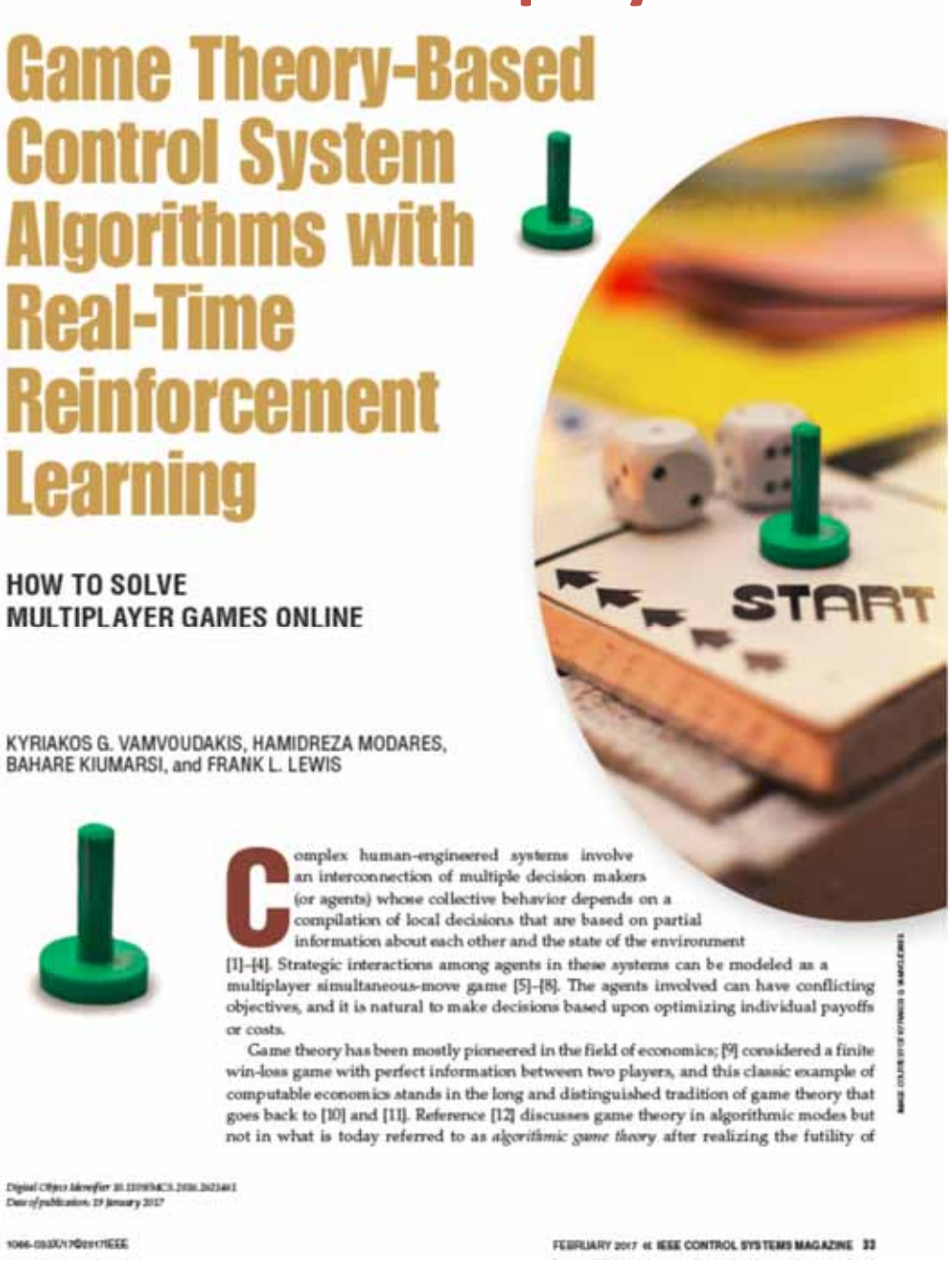


Control approx error

Data-driven Online Solution of Differential Games
Synchronous Solution of Multi-player Non Zero-sum Games



Multi-player Differential Games



Game Theory-Based Control System Algorithms with Real-Time Reinforcement Learning

HOW TO SOLVE MULTIPLAYER GAMES ONLINE

KYRIAKOS G. VAMVOUDAKIS, HAMIDREZA MODARES, BAHARE KIUMARSI, and FRANK L. LEWIS

Complex human-engineered systems involve an interconnection of multiple decision makers (or agents) whose collective behavior depends on a compilation of local decisions that are based on partial information about each other and the state of the environment [1]–[4]. Strategic interactions among agents in these systems can be modeled as a multiplayer simultaneous-move game [5]–[8]. The agents involved can have conflicting objectives, and it is natural to make decisions based upon optimizing individual payoffs or costs.

Game theory has been mostly pioneered in the field of economics; [9] considered a finite win-loss game with perfect information between two players, and this classic example of computable economics stands in the long and distinguished tradition of game theory that goes back to [10] and [11]. Reference [12] discusses game theory in algorithmic modes but not in what is today referred to as *algorithmic game theory* after realizing the futility of

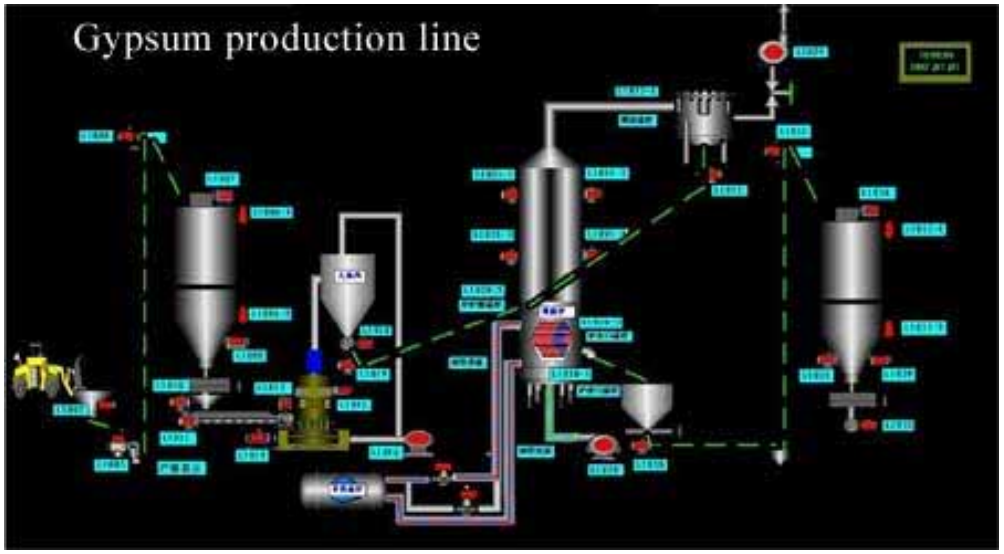
Digital Object Identifier 10.1109/MCS.2016.2621461
Date of publication: 19 January 2017

1066-033X/17©2017IEEE

Multi-player Game Solutions
IEEE Control Systems Magazine,
Dec 2017

Manufacturing as the Interactions of Multiple Agents

Each machine has it own dynamics and cost function
Neighboring machines influence each other most strongly
There are local optimization requirements as well as global necessities



Each process has its own dynamics $\dot{\delta}_i = A\delta_i + (d_i + g_i)B_i u_i - \sum_{j \in N_i} e_{ij} B_j u_j$

And cost function $J_i(\delta_i(0), u_i, u_{-i}) = \frac{1}{2}\int_0^\infty (\delta_i^T Q_{ii} \delta_i + u_i^T R_{ii} u_i + \sum_{j \in N_i} u_j^T R_{ij} u_j)\, dt$

Each process helps other processes achieve optimality and efficiency

Real-Time Solution of Multi-Player NZS Games

Kyriakos Vamvoudakis, Automatica 2011

Multi-Player Nonlinear Systems $\dot{x} = f(x) + \sum_{j=1}^{N} g_j(x) u_j$ Continuous-time, *N* players

Optimal control $V_i^*(x(0), \mu_1, \mu_2, \ldots \mu_N) = \min_{\mu_i} \int_0^\infty (Q_i(x) + \sum_{j=1}^{N} \mu_i^T R_{ij} \mu_i)\, dt; \quad i \in N$

Nash equilibrium $V_i^* \triangleq V_i\ (\mu_1^*, \mu_2^*, \mu_i^*, ..., \mu_N^*) \le V_1\ (\mu_1^*, \mu_2^*, \mu_i, ..., \mu_N^*), i \in N$

Requires Offline solution of coupled Hamilton-Jacobi –Bellman eqs.

$$0 = (\nabla V_i)^T \left(f(x) - \tfrac{1}{2} \sum_{j=1}^{N} g_j(x) R_{jj}^{-1} g_j^T(x) \nabla V_j \right) + Q_i(x) + \tfrac{1}{4} \sum_{j=1}^{N} \nabla V_j{}^T g_j(x) R_{jj}^{-T} R_{ij} R_{jj}^{-1} g_j^T(x) \nabla V_j, \quad V_i(0) = 0$$

Control policies

$$\mu_i(x) = -\tfrac{1}{2} R_{ii}^{-1} g_i^T(x) \nabla V_i \quad , \ i \in N$$

Linear Quadratic Regulator Case- coupled AREs

$$0 = P_i A_c + A_c{}^T P_i + Q_i + \sum_{j=1}^{N} P_j B_j R_{jj}^{-T} R_{ij} R_{jj}^{-1} B_j^T P_j, \quad i \in N$$

These are hard to solve

In the nonlinear case, HJB generally cannot be solved

Team Interest vs. Self Interest

The objective functions of each player can be written as a *team average* term plus a *conflict of interest* term:

$$J_1 = \tfrac{1}{3}(J_1 + J_2 + J_3) + \tfrac{1}{3}(J_1 - J_2) + \tfrac{1}{3}(J_1 - J_3) \equiv J_{team} + J_1^{coi}$$

$$J_2 = \tfrac{1}{3}(J_1 + J_2 + J_3) + \tfrac{1}{3}(J_2 - J_1) + \tfrac{1}{3}(J_2 - J_3) \equiv J_{team} + J_2^{coi}$$

$$J_3 = \tfrac{1}{3}(J_1 + J_2 + J_3) + \tfrac{1}{3}(J_3 - J_1) + \tfrac{1}{3}(J_3 - J_2) \equiv J_{team} + J_3^{coi}$$

For N-players

$$J_i = \tfrac{1}{N}\sum_{j=1}^{N} J_j + \tfrac{1}{N}\sum_{j=1}^{N}(J_i - J_j) \equiv J_{team} + J_i^{coi}, \quad i = 1, N$$

For *N-player zero-sum games*, the first term is zero, i.e. the players have no goals in common.

Real-Time Solution of Multi-Player Games

Non-Zero Sum Games – Synchronous Policy Iteration

Kyriakos Vamvoudakis

Value functions $$V_i(x(0),\mu_1,\mu_2,\dots\mu_N)=\int_0^\infty (Q_i(x)+\sum_{j=1}^N \mu_i^T R_{ij}\mu_i)\,dt; \qquad i\in N$$

Leibniz gives Differential equivalent

Differential equivalent gives coupled Bellman eqs.

$$0=Q_i(x)+\sum_{j=1}^N u_j^T R_{ij}u_j+(\nabla V_i)^T(f(x)+\sum_{j=1}^N g_j(x)u_j) \equiv H_i(x,\nabla V_i,u_1,\dots,u_N),\ i\in N$$

Policy Iteration Solution:

Solve Bellman eq. $$0=r(x,\mu^k{}_1,\dots,\mu^k{}_N)+(\nabla V_i^k)^T\left(f(x)+\sum_{j=1}^N g_j(x)\mu_j^i\right),\quad V^k{}_i(0)=0 \qquad i\in N$$

Policy Update $$\mu_i^{k+1}(x)=-\tfrac{1}{2}R_{ii}^{-1}g_i^T(x)\nabla V_i^k \quad ,\ i\in N$$

Convergence has not been proven
Hard to solve Hamiltonian equation
But this gives the structure we need for online Synchronous PI Solution

Real-Time Solution of Multi-Player Games

Kyriakos Vamvoudakis

Online Synchronous PI Solution for Multi-Player Games

Each player needs 2 NN – a Critic and an Actor

2-player case

N Critic Neural Networks for VFA

N Actor Neural Networks

Player 1

$$\hat{V}_1(x) = \hat{W}_1^T \phi_1(x),$$

$$u_1(x) = -\tfrac{1}{2} R_{11}^{-1} g_1^T(x) \nabla \phi_1^T \hat{W}_3,$$

Player 2

$$\hat{V}_2(x) = \hat{W}_2^T \phi_2(x)$$

$$u_2(x) = -\tfrac{1}{2} R_{22}^{-1} g_2^T(x) \nabla \phi_2^T \hat{W}_4$$

On-Line Learning – for Player 1:

$$\dot{\hat{W}}_1 = -a_1 \frac{\sigma_1}{(\sigma_1^T \sigma_1 + 1)^2} [\sigma_1^T \hat{W}_1 + Q_1(x) + u_1^T R_{11} u_1 + u_2^T R_{12} u_2]$$

Learns Bellman eq. solution

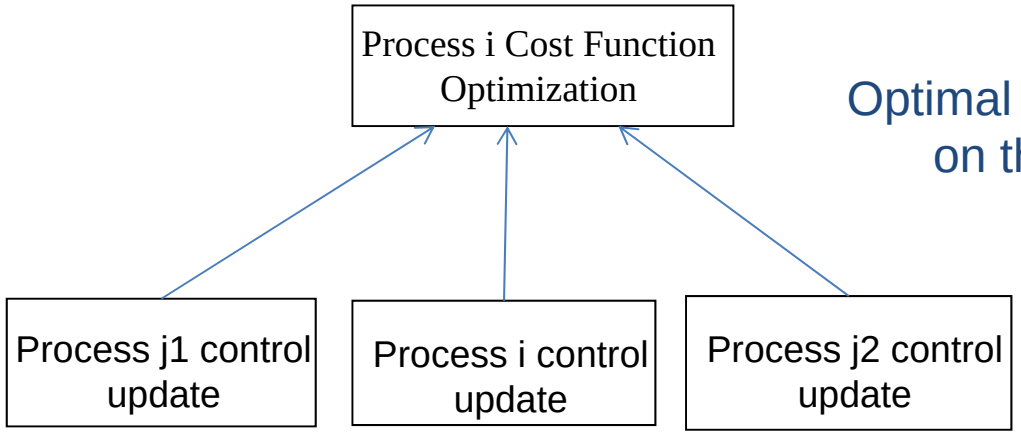
$$\dot{\hat{W}}_3 = -\alpha_3 \{(F_2 \hat{W}_3 - F_1 \bar{\sigma}_3^T \hat{W}_1) - \frac{1}{4} \nabla \phi_1 g(x) R_{11}^{-T} R_{21} R_{11}^{-1} g^T(x) \nabla \phi_1^T \hat{W}_3 m_2^T \hat{W}_2 - \frac{1}{4} \bar{D}_1(x) \hat{W}_3 m_1^T \hat{W}_1\}$$

Learns control policy

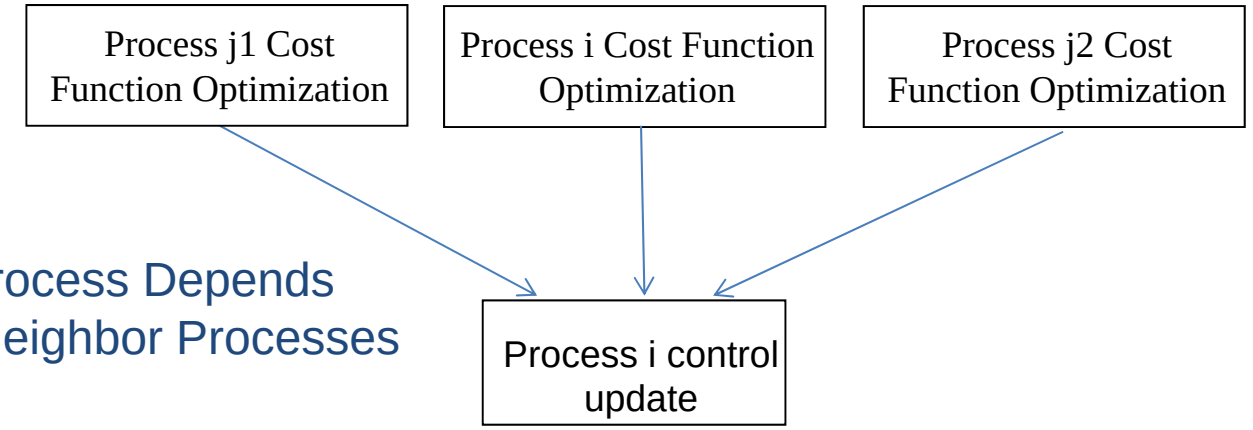
Convergence is proven using Lyapunov functions

K.G. Vamvoudakis and F.L. Lewis, "Multi-Player Non-Zero Sum Games: Online Adaptive Learning Solution of Coupled Hamilton-Jacobi Equations," Automatica, vol. 47, pp. 1556-1569, 2011.

Multi-player Games for Multi-Process Optimal Control
Data-Driven Optimization (DDO)



Optimal Performance of Each Process Depends on the Control of its Neighbor Processes



Control Policy of Each Process Depends on the Performance of its Neighbor Processes

Simulation. – Nonlinear System – 2-player game

$$\dot{x} = f(x) + g(x)u + k(x)d, \qquad x \in \mathbb{R}^2$$

$$f(x) = \begin{bmatrix} x_2 \\ -x_2 - \frac{1}{2}x_1 + \frac{1}{4}x_2(\cos(2x_1)+2)^2 + \frac{1}{4}x_2(\sin(4x_1^2)+2)^2 \end{bmatrix}$$

$$g(x) = \begin{bmatrix} 0 \\ \cos(2x_1)+2 \end{bmatrix}, \quad k(x) = \begin{bmatrix} 0 \\ (\sin(4x_1)+2) \end{bmatrix}.$$

$$Q_1 = 2Q_2 = 2I, \quad R_{11} = 2R_{22} = 2I, \quad R_{12} = 2R_{21} = 2I$$

Optimal Value s $\qquad V_1^*(x) = \frac{1}{2}x_1^2 + x_2^2 \qquad V_2^*(x) = \frac{1}{4}x_1^2 + \frac{1}{2}x_2^2$

Optimal Policies $\qquad u^*(x) = -2(\cos(2x_1)+2)x_2 \qquad d^*(x) = -(\sin(4x_1^2)+2)x_2$

Solves HJB equations online

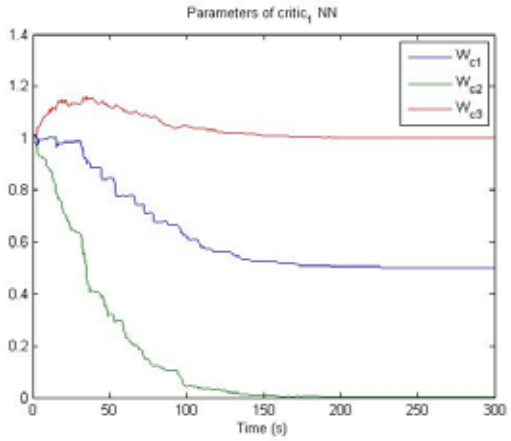
$$0 = (\nabla V_i)^T \left(f(x) - \frac{1}{2}\sum_{j=1}^{N} g_j(x) R_{jj}^{-1} g_j^T(x) \nabla V_j \right) + Q_i(x) + \frac{1}{4}\sum_{j=1}^{N} \nabla V_j{}^T g_j(x) R_{jj}^{-T} R_{ij} R_{jj}^{-1} g_j^T(x) \nabla V_j, \quad V_i(0) = 0$$

Select VFA basis set $\qquad \varphi_1(x) = \varphi_2(x) \equiv [x_1^2 \;\; x_1x_2 \;\; x_2^2]$

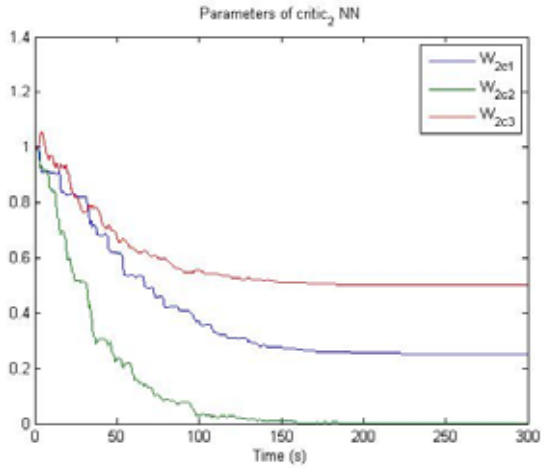
Algorithm converges to $\qquad \hat{W}_1(t_f) = [0.5015 \quad 0.0007 \quad 1.0001]^T = \hat{W}_3(t_f)$

$$\hat{W}_2(t_f) = [0.2514 \quad 0.0006 \quad 0.5001]^T = \hat{W}_4(t_f)$$

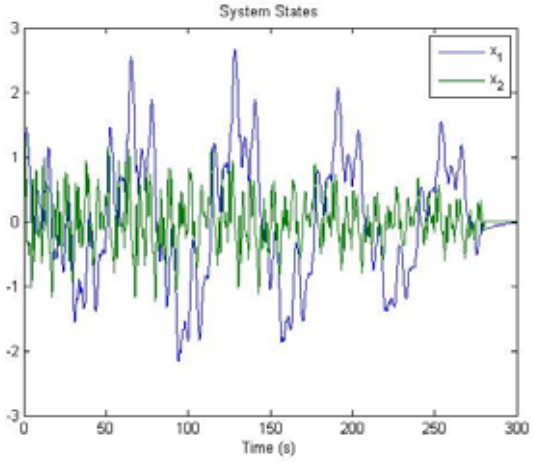
$$\hat{u}(x) = -\frac{1}{2}R_{11}^{-1}\begin{bmatrix} 0 \\ \cos(2x_1)+2 \end{bmatrix}^T \begin{bmatrix} 2x_1 & 0 \\ x_2 & x_1 \\ 0 & 2x_2 \end{bmatrix}^T \begin{bmatrix} 0.5015 \\ 0.0007 \\ 1.0001 \end{bmatrix} \quad \hat{d}(x) = -\frac{1}{2}R_{22}^{-1}\begin{bmatrix} 0 \\ \sin(4x_1^2)+2 \end{bmatrix}^T \begin{bmatrix} 2x_1 & 0 \\ x_2 & x_1 \\ 0 & 2x_2 \end{bmatrix}^T \begin{bmatrix} 0.2514 \\ 0.0006 \\ 0.5001 \end{bmatrix}$$



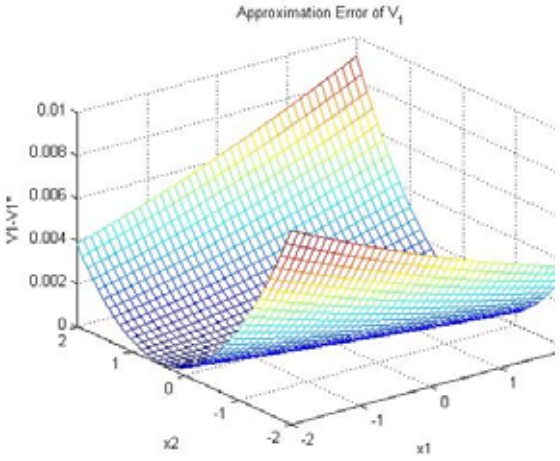
Critic 1 NN parameters



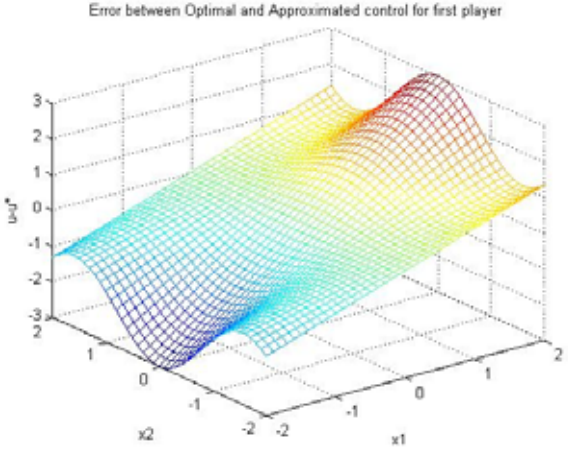
Critic 2 NN parameters



Evolution of the States

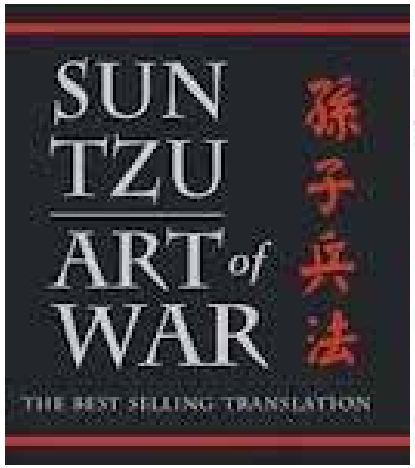


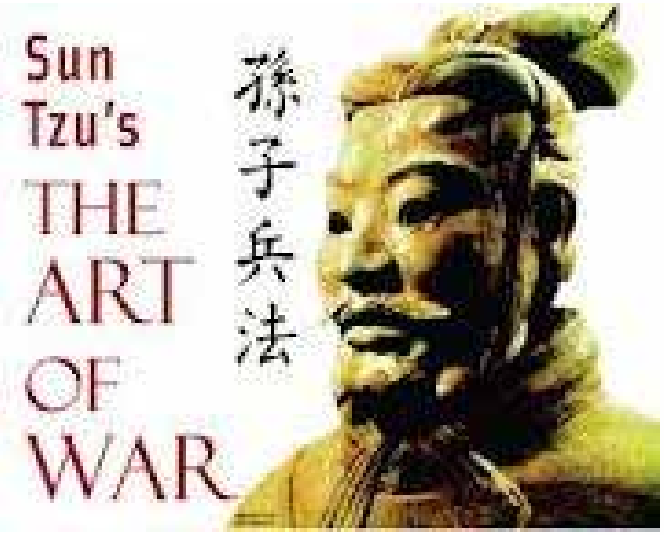
3D approximation error value for player 1.



3D approximation error of control for player 1.

Games on Communication Graphs







500 BC



孙**子兵法**

Sun Tz bin fa

F.L. Lewis, H. Zhang, A. Das, K. Hengster-Movric, *Cooperative Control of Multi-Agent Systems: Optimal Design and Adaptive Control*, Springer-Verlag, 2013



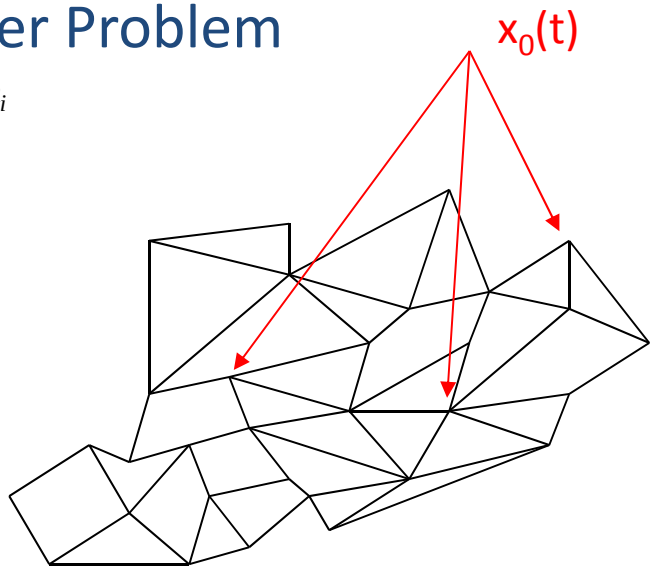
Key Point

Lyapunov Functions and Performance Indices
Must depend on graph topology

H. Zhang, F.L. Lewis, and Z. Qu, "Lyapunov, Adaptive, and Optimal Design Techniques for Cooperative Systems on Directed Communication Graphs," IEEE Trans. Industrial Electronics, vol. 59, no. 7, pp. 3026-3041, July 2012.

Hongwei Zhang, F.L. Lewis, and Abhijit Das
"Optimal design for synchronization of cooperative systems: state feedback, observer and output feedback," IEEE Trans. Automatic Control, vol. 56, no. 8, pp. 1948-1952, August 2011.

Graphical Games
Synchronization- Cooperative Tracker Problem



Node dynamics $\dot{x}_i = Ax_i + B_i u_i, \quad x_i(t) \in \mathbb{R}^n, \quad u_i(t) \in \mathbb{R}^{m_i}$

Target generator dynamics $\dot{x}_0 = Ax_0$

Synchronization problem $x_i(t) \to x_0(t), \forall i$

Local neighborhood tracking error (Lihua Xie)

$$\delta_i = \sum_{j \in N_i} e_{ij}(x_i - x_j) \quad + g_i(x_i - x_0),$$

Local nbhd. tracking error dynamics

$$\dot{\delta}_i = A\delta_i + (d_i + g_i)B_i u_i - \sum_{j \in N_i} e_{ij} B_j u_j$$

Local agent dynamics driven by neighbors' controls

Define Local nbhd. performance index

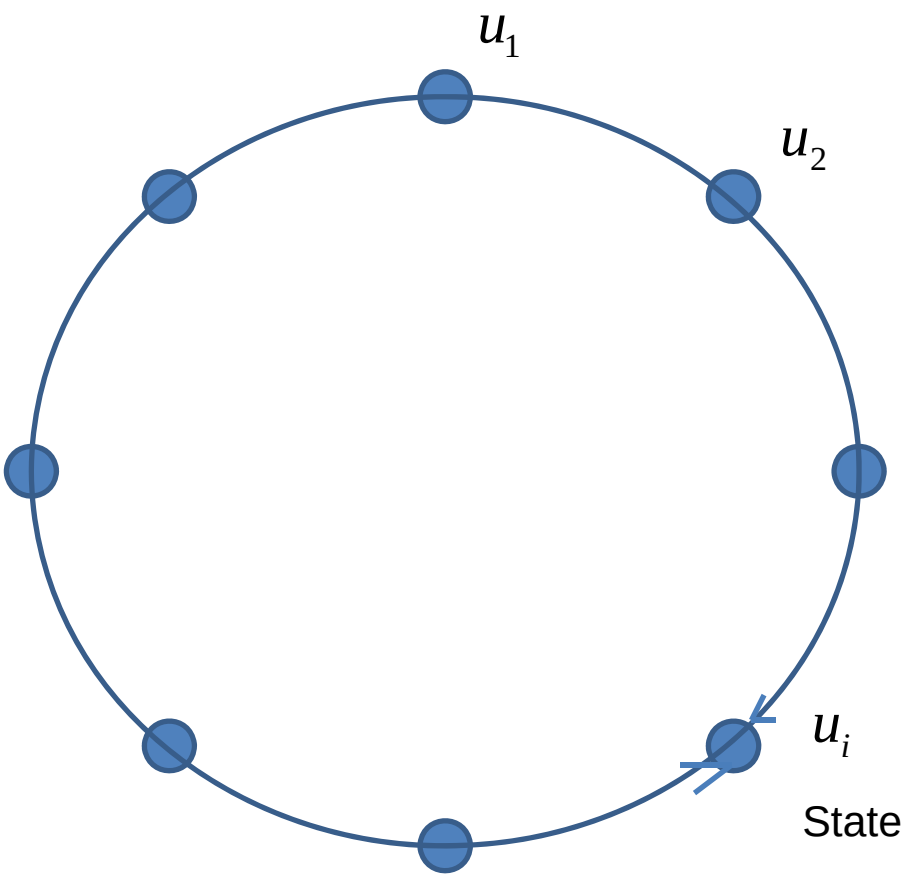
Values driven by neighbors' controls

$$J_i(\delta_i(0), u_i, u_{-i}) = \tfrac{1}{2}\int_0^\infty (\delta_i^T Q_{ii} \delta_i + u_i^T R_{ii} u_i + \sum_{j \in N_i} u_j^T R_{ij} u_j)\, dt \quad \equiv \tfrac{1}{2}\int_0^\infty L_i(\delta_i(t), u_i(t), u_{-i}(t))\, dt$$

K.G. Vamvoudakis, F.L. Lewis, and G.R. Hudas, "Multi-Agent Differential Graphical Games: online adaptive learning solution for synchronization with optimality," Automatica, vol. 48, no. 8, pp. 1598-1611, Aug. 2012.

M. Abouheaf, K. Vamvoudakis, F.L. Lewis, S. Haesaert, and R. Babuska, "Multi-Agent Discrete-Time Graphical Games and Reinforcement Learning Solutions," Automatica, Vol. 50, no. 12, pp. 3038-3053, 2014.

New Differential Graphical Game



Local Dynamics
Local Value Function
Only depends on
graph neighbors

u_i Control action of player *i*

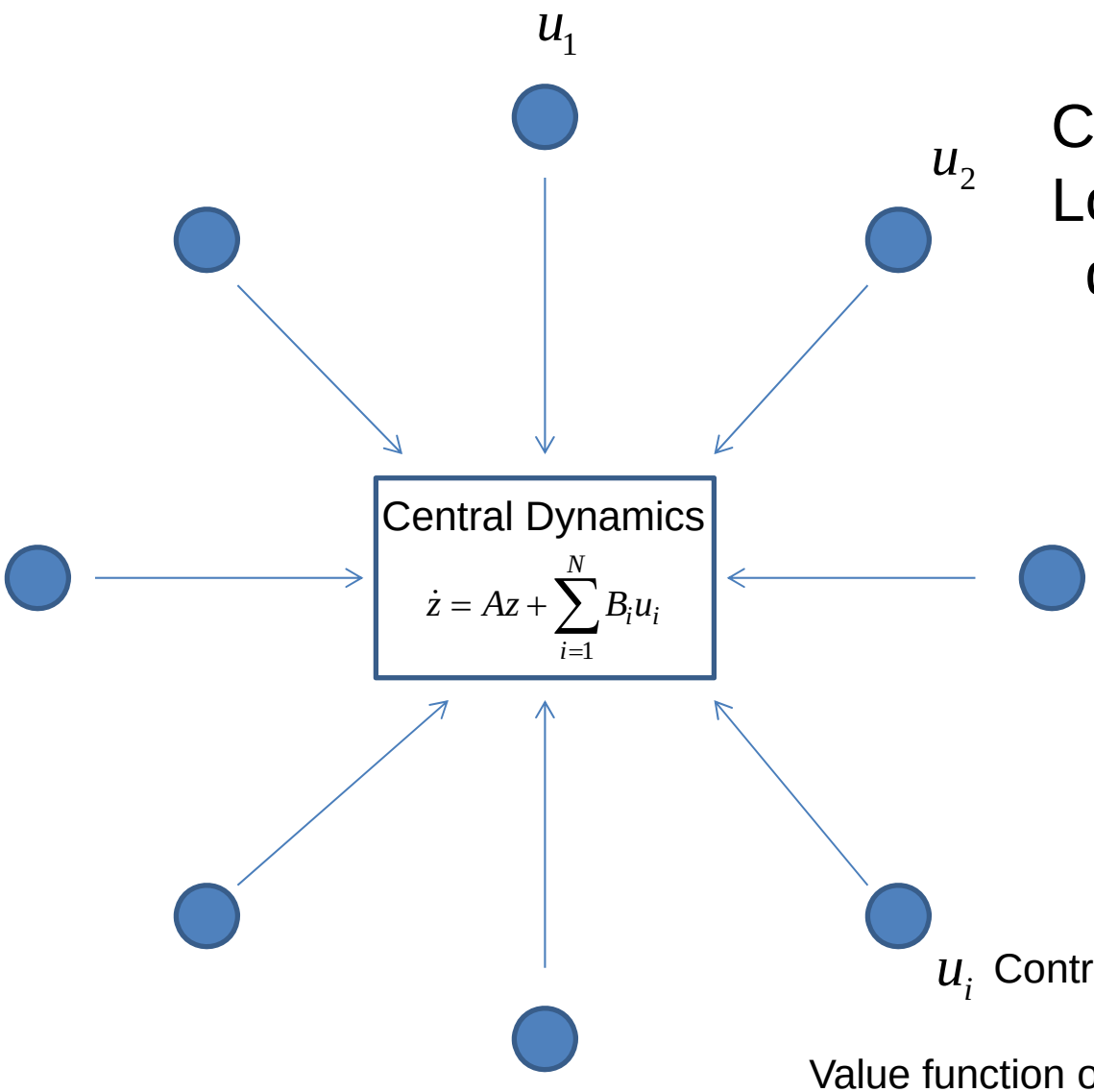
State dynamics of agent *i*

$$\dot{\delta}_i = A\delta_i + (d_i + g_i)B_i u_i - \sum_{j \in N_i} e_{ij} B_j u_j$$

Value function of player *i*

$$J_i(\delta_i(0), u_i, u_{-i}) = \tfrac{1}{2}\int_0^\infty (\delta_i^T Q_{ii}\delta_i + u_i^T R_{ii} u_i + \sum_{j \in N_i} u_j^T R_{ij} u_j)\, dt$$

Standard Multi-Agent Differential Game



Central Dynamics
Local Value Function
depends on ALL
other control actions

u_i Control action of player *i*

Value function of player *i*

$$J_i(z(0), u_i, u_{-i}) = \frac{1}{2}\int_0^{\infty} (z^T Q z + \sum_{j=1}^{N} u_j^T R_{ij} u_j)\, dt$$

New Definition of Nash Equilibrium for Graphical Games

Def. Local Best response.

u_i^* is said to be agent *i's* local best response to fixed policies u_{-i} of its neighbors if

$$J_i(u_i^*, u_{-i}) \leq J_i(u_i, u_{-i}), \quad \forall u_i$$

Def: Interactive Nash equilibrium

$\{u_1^*, u_2^*, \ldots, u_N^*\}$ are in Interactive Nash equilibrium if

1. $J_i^* \triangleq J_i(u_i^*, u_{G-i}^*) \leq J_i(u_i, u_{G-i}^*), \ \forall i \in N$ i.e. they are in Nash equilibrium

2. There exists a policy u_j such that

$$J_i(u_j, u_{G-j}^*) \neq J_i(u_j^*, u_{G-j}^*), \quad \forall i, j \in N$$

That is, every player can find a policy that changes the value of every other player.

A restriction on what sorts of performance indices can be selected in multi-player graph games.

A condition on the reaction curves (Basar and Olsder) of the agents

This rules out the disconnected counterexample.

Graphical Game Solution Equations

Value function

$$V_i(\delta_i(t)) = \tfrac{1}{2}\int_t^\infty (\delta_i^T Q_{ii}\delta_i + u_i^T R_{ii} u_i + \sum_{j\in N_i} u_j^T R_{ij} u_j)\, dt$$

Differential equivalent (Leibniz formula) is Bellman's Equation

$$H_i(\delta_i, \frac{\partial V_i}{\partial \delta_i}, u_i, u_{-i}) \equiv \frac{\partial V_i}{\partial \delta_i}^T \left(A\delta_i + (d_i + g_i) B_i u_i - \sum_{j\in N_i} e_{ij} B_j u_j \right) + \tfrac{1}{2}\delta_i^T Q_{ii}\delta_i + \tfrac{1}{2} u_i^T R_{ii} u_i + \tfrac{1}{2}\sum_{j\in N_i} u_j^T R_{ij} u_j = 0$$

Stationarity Condition

$$0 = \frac{\partial H_i}{\partial u_i} \quad \Rightarrow \quad u_i = -(d_i + g_i) R_{ii}^{-1} B_i^T \frac{\partial V_i}{\partial \delta_i}$$

1. Coupled HJ equations

$$\frac{\partial V_i}{\partial \delta_i}^T A_i^c + \tfrac{1}{2}\delta_i^T Q_{ii}\delta_i + \tfrac{1}{2}(d_i + g_i)^2 \frac{\partial V_i}{\partial \delta_i}^T B_i R_{ii}^{-1} B_i^T \frac{\partial V_i}{\partial \delta_i} + \tfrac{1}{2}\sum_{j\in N_i} (d_j + g_j)^2 \frac{\partial V_j}{\partial \delta_j}^T B_j R_{jj}^{-1} R_{ij} R_{jj}^{-1} B_j^T \frac{\partial V_j}{\partial \delta_j} = 0, i \in N$$

$$H_i(\delta_i, \frac{\partial V_i}{\partial \delta_i}, u_i^*, u_{-i}^*) = 0$$

where $$A_i^c = A\delta_i - (d_i + g_i)^2 B_i R_{ii}^{-1} B_i^T \frac{\partial V_i}{\partial \delta_i} + \sum_{j\in N_i} e_{ij}(d_j + g_j) B_j R_{jj}^{-1} B_j^T \frac{\partial V_j}{\partial \delta_j}, i \in N$$

Now use Synchronous PI to learn optimal Nash policies online in real-time as players interact

Distributed Multi-Agent Learning Proofs

Online Solution of Graphical Games
Multi-agent Learning Convergence proofs

Kyriakos Vamvoudakis

Use Reinforcement Learning

POLICY ITERATION

Algorithm 1. Policy Iteration (PI) Solution for N-player distributed games.

Step 0: Start with admissible initial policies $u_i^0, \forall i$.

Step 1: (Policy Evaluation) Solve for V_i^k using (14)

$$H_i(\delta_i, \frac{\partial V_i}{\partial \delta_i}^k, u_i^k, u_{-i}^k) = 0, \forall i = 1, \ldots, N \qquad (38)$$

Step 2: (Policy Improvement) Update the N-tuple of control policies using

$$u_i^{k+1} = \arg\min_{u_i} H_i(\delta_i, \frac{\partial V_i}{\partial \delta_i}^k, u_i, u_{-i}^k), \forall i = 1, \ldots, N$$

which explicitly is

$$u_i^{k+1} = -(d_i + g_i) R_{ii}^{-1} B_i^T \frac{\partial V_i}{\partial \delta_i}^k, \forall i = 1, \ldots, N. \qquad (39)$$

Go to step 1.

On convergence End ■

Convergence Results

Theorem 3. Convergence of Policy Iteration algorithm when only i^{th} agent updates its policy and all players u_{-i} in the neighborhood do not change. Given fixed neighbors policies u_{-i}, assume there exists an admissible policy u_i. Assume that agent i performs Algorithm 1 and the its neighbors do not update their control policies. Then the algorithm converges to the best response u_i to policies u_{-i} of the neighbors and to the solution V_i to the best response HJ equation (36).

The next result concerns the case where all nodes update their policies at each step of the algorithm. Define the relative control weighting as $\rho_{ij} = \bar{\sigma}(R_{jj}^{-1} R_{ij})$, where $\bar{\sigma}(R_{jj}^{-1} R_{ij})$ is the maximum singular value of $R_{jj}^{-1} R_{ij}$.

Theorem 4. Convergence of Policy Iteration algorithm when all agents update their policies. Assume all nodes i update their policies at each iteration of PI. Then for small enough edge weights e_{ij} and ρ_{ij}, μ_i converges to the global Nash equilibrium and for all i, and the values converge to the optimal game values $V_i^k \to V_i^*$.

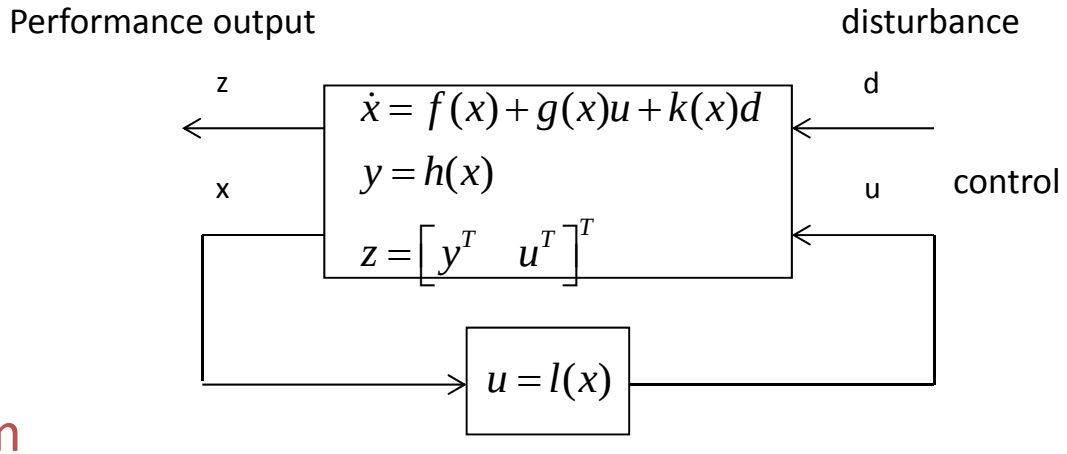
Data-driven Online Solution of Differential Games
Zero-sum 2-Player Games and H-infinity Control



H-Infinity Control Using Reinforcement Learning

Disturbance Rejection

System



L_2 Gain Problem

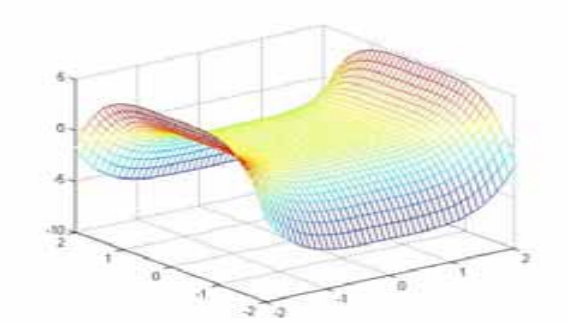
Find control *u(t)* so that

$$\frac{\int_0^\infty \|z(t)\|^2 dt}{\int_0^\infty \|d(t)\|^2 dt} = \frac{\int_0^\infty (h^T h + \|u\|^2) dt}{\int_0^\infty \|d(t)\|^2 dt} \leq \gamma^2$$

For all L_2 disturbances
And a prescribed gain γ^2

Zero-Sum differential game - Nature as the opposing player

The game has a unique value (saddle-point solution)
iff the Nash condition holds



Online Zero-Sum Differential Games

H-infinity Control

System $\dot{x} = f(x,u) = f(x) + g(x)u + k(x)d$

$y = h(x)$

2 players

Cost $V(x(t),u,d) = \int_t^\infty \left(h^T h + u^T R u - \gamma^2 \|d\|^2 \right) dt \equiv \int_t^\infty r(x,u,d)\, dt$

Leibniz gives Differential equivalent

Game saddle point solution found from Hamiltonian - ZS Game BELLMAN EQUATION

$$H(x, \frac{\partial V}{\partial x}, u, d) = h^T h + u^T R u - \gamma^2 \|d\|^2 + (\nabla V)^T (f(x) + g(x)u + k(x)d) = 0$$

Optimal control/dist. policies found by stationarity conditions $0 = \frac{\partial H}{\partial u}, \; 0 = \frac{\partial H}{\partial d}$

$$u = -\frac{1}{2} R^{-1} g^T(x) \nabla V \qquad d = \frac{1}{2\gamma^2} k^T(x) \nabla V$$

HJI equation $0 = H(x, \nabla V, u^*, d^*)$

$$= h^T h + \nabla V^T(x) f(x) - \frac{1}{4} \nabla V^T(x) g(x) R^{-1} g^T(x) \nabla V(x) + \frac{1}{4\gamma^2} \nabla V^T(x) k k^T \nabla V(x)$$

D. Vrabie and F.L. Lewis, “Adaptive dynamic programming for online solution of a zero-sum differential game,” *J Control Theory App.,* vol. 9, no. 3, pp. 353–360, 2011

K.G. Vamvoudakis and F.L. Lewis, “Online solution of nonlinear two-player zero-sum games using synchronous policy iteration,” Int. J. Robust and Nonlinear Control, vol. 22, pp. 1460-1483, 2012.

Double Policy Iteration Algorithm to Solve HJI

Add inner loop to solve for available storage

Start with stabilizing initial policy $u_0(x)$

Murad Abukhalaf
Draguna Vrabie

1. For a given control policy $u_j(x)$ solve for the value $V_{j+1}(x(t))$

2. Set $d^0=0$. For i=0,1,… solve for $V_j^i(x(t)), d^{i+1}$

$$0=h^T h+\nabla V_j^{iT}(x)(f+gu_j+kd^i)+u_j{}^T Ru_j-\gamma^2\left\|d^i\right\|^2$$

ZS game Bellman eq.

$$d^{i+1}=\frac{1}{2\gamma^2}k^T(x)\nabla V_j^i$$

On convergence set $V_{j+1}(x)=V_j^i(x)$

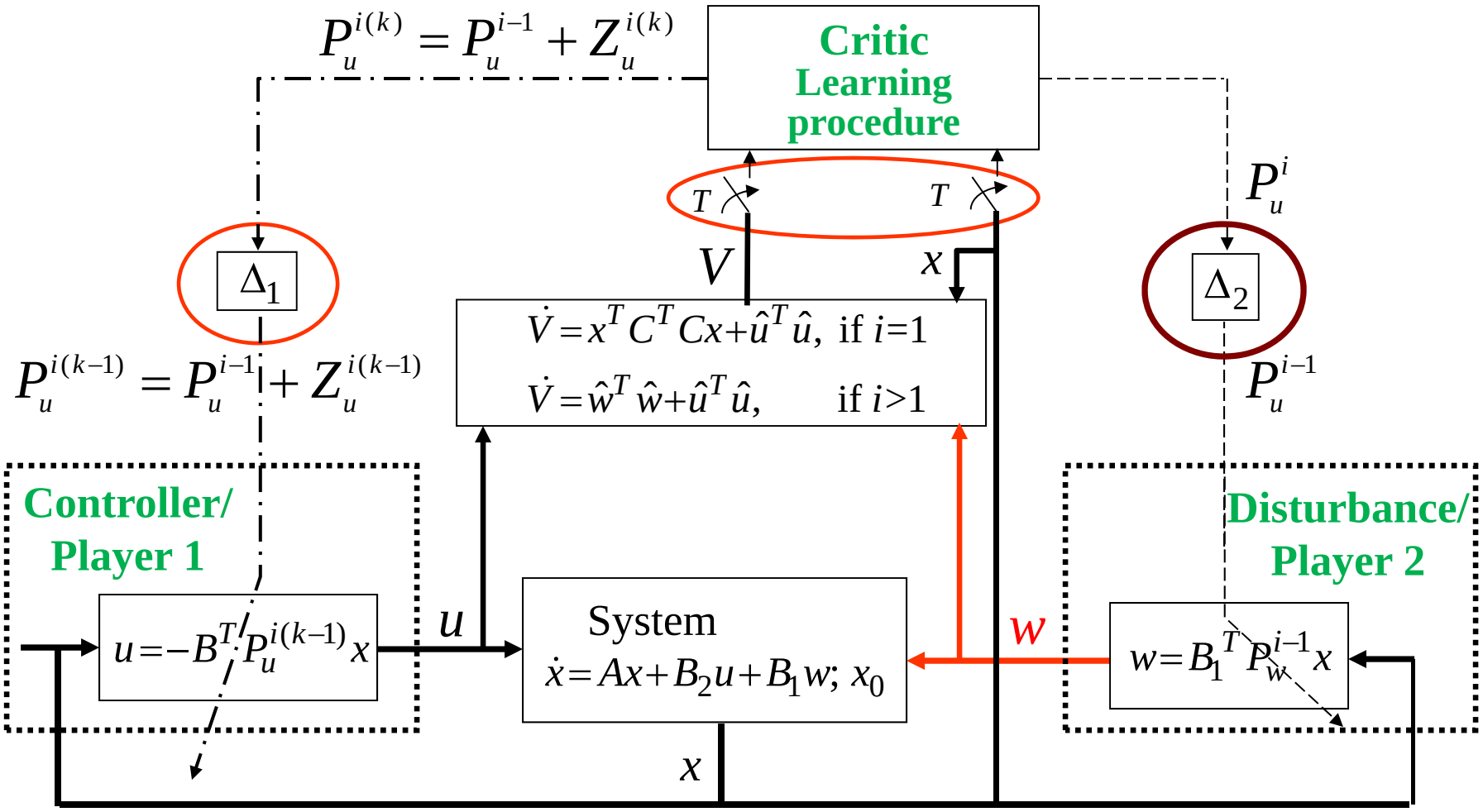
3. Improve policy:

$$u_{j+1}(x)=-\tfrac{1}{2}R^{-1}g^T(x)\nabla V_{j+1}$$

- Convergence proved by Van der Schaft if can solve nonlinear Lyapunov equation exactly
- Abu Khalaf & Lewis used NN to approximate V for nonlinear systems and proved convergence

ONLINE SOLUTION- Use IRL to solve the ZS Bellman eq. and update the actors at each step

Actor-Critic structure - three time scales




New Developments in IRL for CT Systems
Q Learning for CT Systems
Experience Replay
Off-Policy IRL

IRL with Experience Replay

Humans use memories of past experiences to tune current policies

Modares and Lewis, Automatica 2014
Girish Chowdhary- concurrent learning
Sutton and Barto book

system $\dot{x}(t) = f(x(t)) + g(x(t))u(t)$

Value $V(x(t)) = \int_t^{\infty} \left(Q(x(\tau)) + 2\int_0^u (\lambda \tanh^{-1}(v/\lambda))^T R\, dv \right) d\tau$

Bellman Equation $Q(x) + 2\int_0^u (\lambda \tanh^{-1}(v/\lambda))^T R\, dv + \nabla V^T(x)\,(f(x) + g(x)\,u) = 0, \;\; V(0) = 0$

IRL Bellman Equation $V(x(t-T)) = \int_{t-T}^{t} \left(Q(x(\tau)) + 2\int_0^u (\lambda \tanh^{-1}(v/\lambda))^T R dv \right) d\tau + V(x(t))$

Action Update $u^* = -\lambda \tanh\left((1/2\lambda) R^{-1} g^T(x)\, \nabla V^*(x) \right)$

VFA- Value Function Approximation $\hat{V}(x) = \hat{W}_1^T \phi(x)$

Bellman Eq gives Linear Equation for Weights $\int_{t-T}^{t} \left(Q(x(\tau)) + 2\int_0^u (\lambda \tanh^{-1}(v/\lambda))^T R\, dv \right) d\tau + W_1^T\, \Delta\phi(x(t)) \equiv \varepsilon_B(t)$

i/o Data Measurements $\Delta\phi(x(t)) = \phi(x(t)) - \phi(x(t-T))$

$$p(t) = \int_{t-T}^{t} \left(Q + 2\int_0^u (\lambda \tanh^{-1}(v/\lambda))^T R\, dv \right) d\tau$$

Standard Critic Weight Tuning

$$\dot{\hat{W}}_1(t) = -\alpha_1 \frac{\Delta\phi(t)}{(1 + \Delta\phi(t)^T \Delta\phi(t))^2} \left(p(t) + \Delta\phi(t)^T \hat{W}_1(t) \right)$$

IRL with Experience Replay

Humans use memories of past experiences to tune current policies

VFA- Value Function Approximation $\hat{V}(x) = \hat{W}_1^T \phi(x)$

Modares and Lewis, Automatica 2014
Girish Chowdhary- concurrent learning
Sutton and Barto book

i/o Data Measurements

$$\Delta\phi(x(t)) = \phi(x(t)) - \phi(x(t-T))$$

$$p(t) = \int_{t-T}^{t} \left(Q + 2\int_0^u (\lambda \tanh^{-1}(v/\lambda))^T R\, dv\right) d\tau$$

Data from Previous time intervals

The samples are stored in a history stack. To collect data in the history stack, consider $\Delta\phi_j$ and p_j as evaluated values of $\Delta\phi(t)$ and $p(t)$ (see (17) and (26)) at the recorded time t_j. That is,

$$\Delta\phi_j = \Delta\phi(t_j) = \phi(x(t_j)) - \phi(x(t_j - T)) \qquad (27)$$

and

$$p_j = p(t_j) = \int_{t_j-T}^{t_j} \left(Q + 2\int_0^u (\lambda \tanh^{-1}(v/\lambda))^T R\, dv\right) d\tau \qquad (28)$$

Improvements
1. Speeds up convergence
2. PE condition is milder

Previous data

NN weight tuning uses past samples

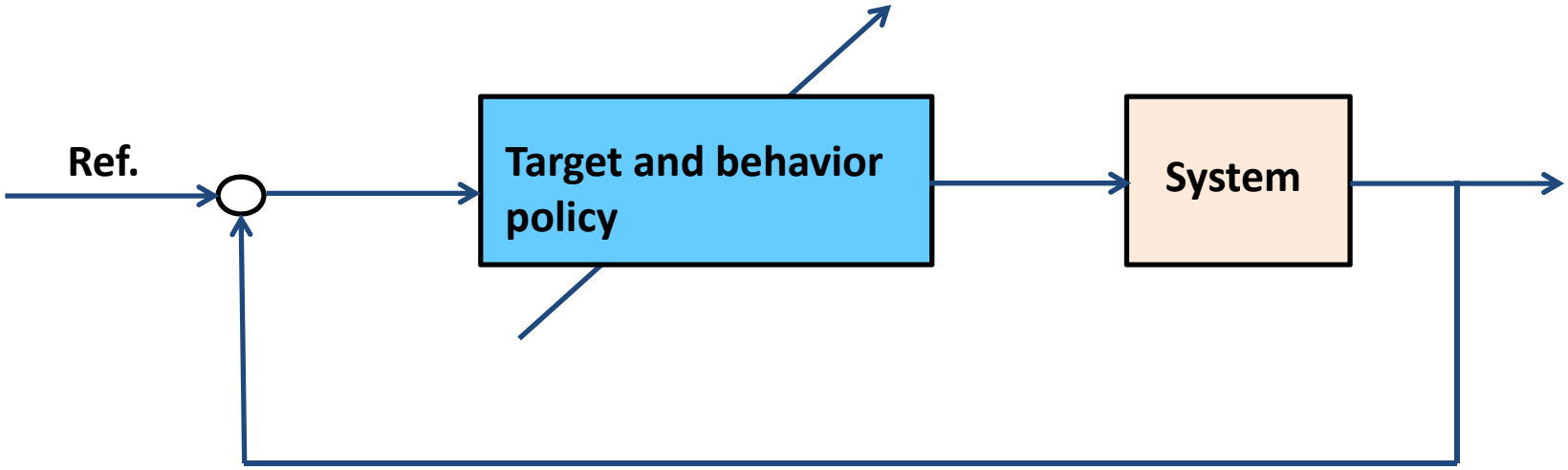
$$\dot{\hat{W}}_1(t) = -\alpha_1 \frac{\Delta\phi(t)}{(1+\Delta\phi(t)^T \Delta\phi(t))^2}\left(p(t) + \Delta\phi(t)^T \hat{W}_1(t)\right) - \alpha_1 \sum_{j=1}^{l} \frac{\Delta\phi_j}{(1+\Delta\phi_j{}^T \Delta\phi_j)^2}\left(p_j + \Delta\phi_j^T \hat{W}_1(t)\right)$$

Off-Policy Reinforcement Learning

Humans can learn optimal policies while actually playing suboptimal policies

On-policy RL

Target policy: The policy that we are learning about.
Behavior policy: The policy that generates actions and behavior

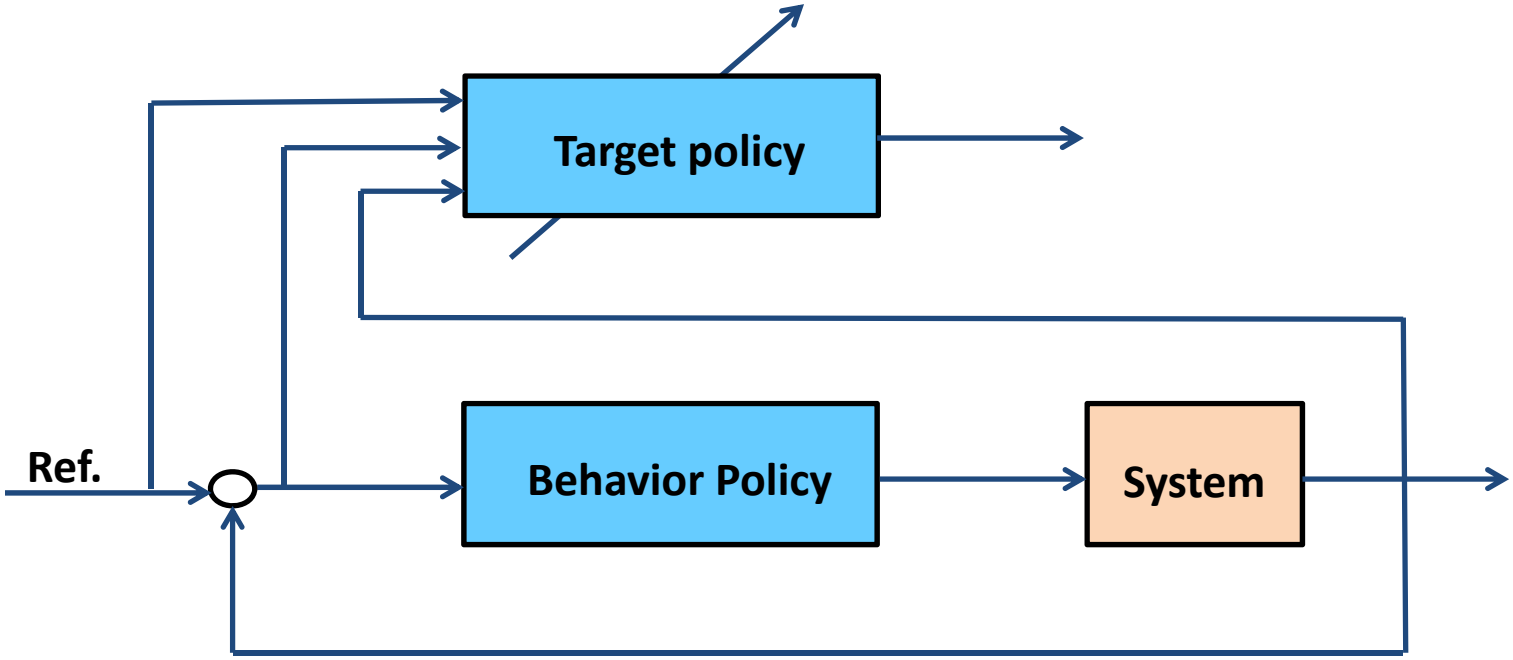


Target policy and behavior policy are the same

Sutton and Barto Book

Off-policy RL

Humans can learn optimal policies while actually applying suboptimal policies



Target policy and behavior policy are different

H. Modares, F.L. Lewis, and Z.-P. Jiang, "H-infinity Tracking Control of Completely-unknown Continuous-time Systems via Off-policy Reinforcement Learning ," IEEE Transactions on Neural Networks and Learning Systems, vol. 26, no. 10, pp. 2550-2562, Oct. 2015.

Ruizhuo Song, F.L. Lewis, Qinglai Wei, "Off-Policy Integral Reinforcement Learning Method to Solve Nonlinear Continuous-Time Multi-Player Non-Zero-Sum Games," IEEE Transactions on Neural Networks and Learning Systems, vol. 28, no. 3, pp. 704-713, 2017.

Bahare Kiumarsi, Frank L. Lewis, Zhong-Ping Jiang, "H-infinity Control of Linear Discrete-time Systems: Off-policy Reinforcement Learning," Automatica, vol. 78, pp. 144-152, 2017.

Off-policy IRL

Yu Jiang & Zhong-Ping Jiang, Automatica 2012

Humans can learn optimal policies while actually applying suboptimal policies

system $\dot{x} = f(x) + g(x)u$

value $J(x) = \int_t^\infty r\left(x(\tau), u(\tau)\right) d\tau$

On-policy IRL

$$J^{[i]}(x(t)) - J^{[i]}(x(t-T)) = -\int_{t-T}^{t} Q(x) d\tau - \int_{t-T}^{t} u^{[i]T} R u^{[i]} d\tau$$

$$u^{[i+1]} = -\frac{1}{2} R^{-1} g^T J_x^{[i]}$$

Must know *g(x)*

Off-policy IRL

$$\dot{x} = f + g u^{[i]} + g(u - u^{[i]})$$

$$J^{[i]}(x(t)) - J^{[i]}(x(t-T)) = -\int_{t-T}^{t} Q(x) d\tau - \int_{t-T}^{t} u^{[i]T} R u^{[i]} d\tau + 2\int_{t-T}^{t} u^{[i+1]T} R(u^{[i]} - u)\, d\tau$$

DDO

This is a linear equation for $J^{[i]}$ and $u^{[i+1]}$

They can be found simultaneously online using measured data using Kronecker product and VFA

1. Completely unknown system dynamics
2. Can use applied u(t) for –
   disturbance rejection – Z.P. Jiang - R. Song and Lewis, 2015
   robust control – Y. Jiang & Z.P. Jiang, IEEE TCS 2012
   exploring probing noise – without bias ! - J.Y. Lee, J.B. Park, Y.H. Choi 2012

Off-policy for Multi-player NZS Games

Angela Song and Lewis

$$\dot{x} = f(x) + \sum_{j=1}^{N} g(x)u_j$$

$$V_i(x(t)) = \int_t^\infty (r_i(x,u_1,u_2,\ldots,u_N))d\tau = \int_t^\infty (Q_i(x) + \sum_{j=1}^{N} u_j^T R_{ij} u_j)d\tau$$

On-policy

Algorithm 1:

Step 1: Start with stabilizing initial policies $u_1^{[0]}, u_2^{[0]}, \ldots, u_N^{[0]}$

Step 2: Given the N-tuple of policies $u_1^{[k]}, u_2^{[k]}, \ldots, u_N^{[k]}$, solve for the N-tuple of costs $V_1^{[k]}(x(t)), V_2^{[k]}(x(t)), \ldots, V_N^{[k]}(x(t))$ using

$$0 = \nabla V_i^{[k]T}(f(x) + \sum_{j=1}^{N} g(x)u_j^{[k]}) + r_i(x, u_1^{[k]}, u_2^{[k]}, \ldots, u_N^{[k]}) \quad (9)$$

with $V_i^{[k]}(0) = 0$.

Step 3: Update the N-tuple of control policies using:

$$u_i^{[k+1]} = \arg\min_{u_i}[H_i(x, \nabla V_i, u_1, \ldots, u_N)] \quad (10)$$

which explicitly is

$$u_i^{[k+1]} = -\frac{1}{2} R_{ii}^{-1} g^T(x) \nabla V_i^{[k]} \quad (11)$$

Off-policy

$$\dot{x} = f(x) + \sum_{j=1}^{N} g(x)u_j^{[k]} + \sum_{j=1}^{N} g(x)(u_j - u_j^{[k]})$$

$$V_i^{[k]}(x(t+T)) - V_i^{[k]}(x(t)) = -\int_t^{t+T} Q_i(x)d\tau - \int_t^{t+T} \sum_{j=1}^{N} u_j^{[k]T} R_{ij} u_j^{[k]} d\tau \quad -2\int_t^{t+T} u_i^{[k+1]T} R_{ii} \sum_{j=1}^{N} (u_j - u_j^{[k]})d\tau$$

DDO

1. Solve online using measured data for $V_i^{[k]}, u_i^{[k+1]}$
2. Completely unknown dynamics
3. Add exploring noise with no bias

Off-policy IRL for ZS Games – H-infinity Control

Optimal tracker

Extended state $X(t) = [e_d(t)^T \; r(t)^T]^T \in \mathbb{R}^{2n}$

Including ref. traj. dynamics

$$\dot{X}(t) = F(X(t)) + G(X(t))u(t) + K(X(t))d(t)$$

$$V(u,d) = \int_t^\infty e^{-\alpha(\tau - t)}(X^T Q_T \; X + u^T R \, u - \gamma^2 d^T d) d\tau$$

Off-policy

$$\dot{X} = F + G\,u_i + K\,d_i + G\,(u - u_i) + K\,(d - d_i)$$

DDO

1. Solve online using data for V_i, u_{i+1}, d_{i+1}
2. Completely unknown dynamics
3. Disturbance does not need to be specified
4. Add exploring noise with no bias

Biao Luo, H.N. Wu, T. Huang, IEEE T. Cybern.
Modares, Lewis, Z.P. Jiang, TNNLS 2015
H. Li, Derong Liu, D. Wang, IEEE TASE 2015

Algorithm 2 . *Online Off-policy RL algorithm for solving tracking HJI equation*

1. *Initialization*: Start with a control policy u_0
2. Solve the following Bellman equation for V_i and u_{i+1} simultaneously

$$e^{-\alpha T} V_i(X(t+T)) - V_i(X(t)) = \int_t^{t+T} e^{-\alpha(\tau - t)}(-X^T Q_T \; X - u_i{}^T R \, u_i + \gamma^2 d_i{}^T d_i) \, d\tau + \int_t^{t+T} e^{-\alpha(\tau - t)}(-2u_{i+1}{}^T R(u - u_i) + 2\gamma^2 d_{i+1}{}^T (d - d_i)) d\tau \quad (67)$$

Observe played policy of disturbance
Compute his worst case malicious attack
should he choose to play it



Applications of Reinforcement Learning

- Microgrid Control
- Human-Robot Interactive Learning
- Industrial process control- Mineral grinding in Gansu, China
- Resilient Control to Cyber-Attacks in Networked Multi-agent Systems
- Decision & Control for Heterogeneous MAS (different dynamics)



Advances in Industrial Control

Ali Bidram · Vahidreza Nasirian · Ali Davoudi · Frank L. Lewis

Cooperative Synchronization in Distributed Microgrid Control

This book brings together emerging objectives and paradigms in the control of both AC and DC microgrids; further, it facilitates the integration of renewable-energy and distribution systems through localization of generation, storage and consumption. The control objectives in a microgrid are addressed through the hierarchical control structure.

After providing a comprehensive survey on the state of the art in microgrid control, the book goes on to address the most recent control schemes for both AC and DC microgrids, which are based on the distributed cooperative control of multi-agent systems. The cooperative control structure discussed distributes the co-ordination and optimization tasks across all distributed generators. This does away with the need for a central controller, and the control system will not collapse in response to the outage of a single unit. This avoids adverse effects on system flexibility and configurability, as well as the reliability concerns in connection with single points of failure that arise in traditional, centralized microgrid control schemes.

Rigorous proofs develop each control methodology covered in the book, and simulation examples are provided to justify all of the proposed algorithms. Given its extensive yet self-contained content, the book offers a comprehensive source of information for graduate students, academic researchers, and practicing engineers working in the field of microgrid control and optimization.

Advances in Industrial Control aims to report and encourage the transfer of technology in control engineering. The rapid development of control technology has an impact on all areas of the control discipline. The series offers an opportunity for researchers to present an extended exposition of new work in all aspects of industrial control.

Engineering

ISBN 978-3-319-50807-8



► springer.com

Bidram · Nasirian · Davoudi · Lewis



Cooperative Synchronization in Distributed Microgrid Control

Advances in Industrial Control

Ali Bidram
Vahidreza Nasirian
Ali Davoudi
Frank L. Lewis

Cooperative Synchronization in Distributed Microgrid Control

AIC

Springer

Intelligent Operational Control for Complex Industrial Processes

Jinliang Ding

Professor Chai Tianyou

State Key Laboratory of Synthetical Automation for Process Industries

Northeastern University

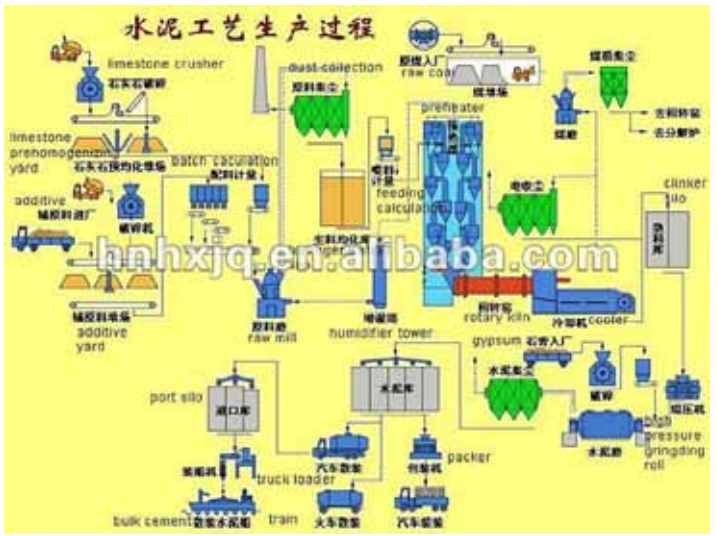
May 20, 2013

1. Jinliang Ding, H. Modares, Tianyou Chai, and F.L. Lewis, "Data-based Multi-objective Plant-wide Performance Optimization of Industrial Processes under Dynamic Environments," IEEE Trans. Industrial Informatics, vol.12, no. 2, pp. 454-465, April 2016.
2. Xinglong Lu, B. Kiumarsi, Tianyou Chai, and F.L. Lewis, "Data-driven Optimal Control of Operational Indices for a Class of Industrial Processes," IET Control Theory & Applications, vol. 10, no. 12, pp. 1348-1356, 2016.

Manufacturing as the Interactions of Multiple Agents

Each machine has it own dynamics and cost function
Neighboring machines influence each other most strongly
There are local optimization requirements as well as global necessities





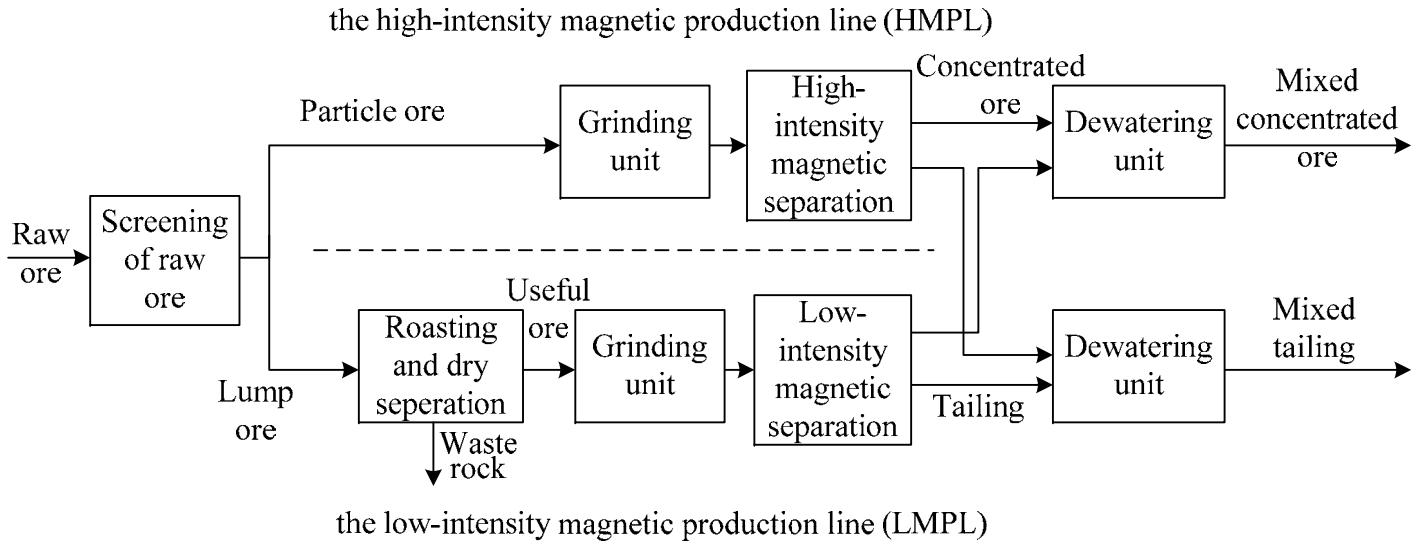




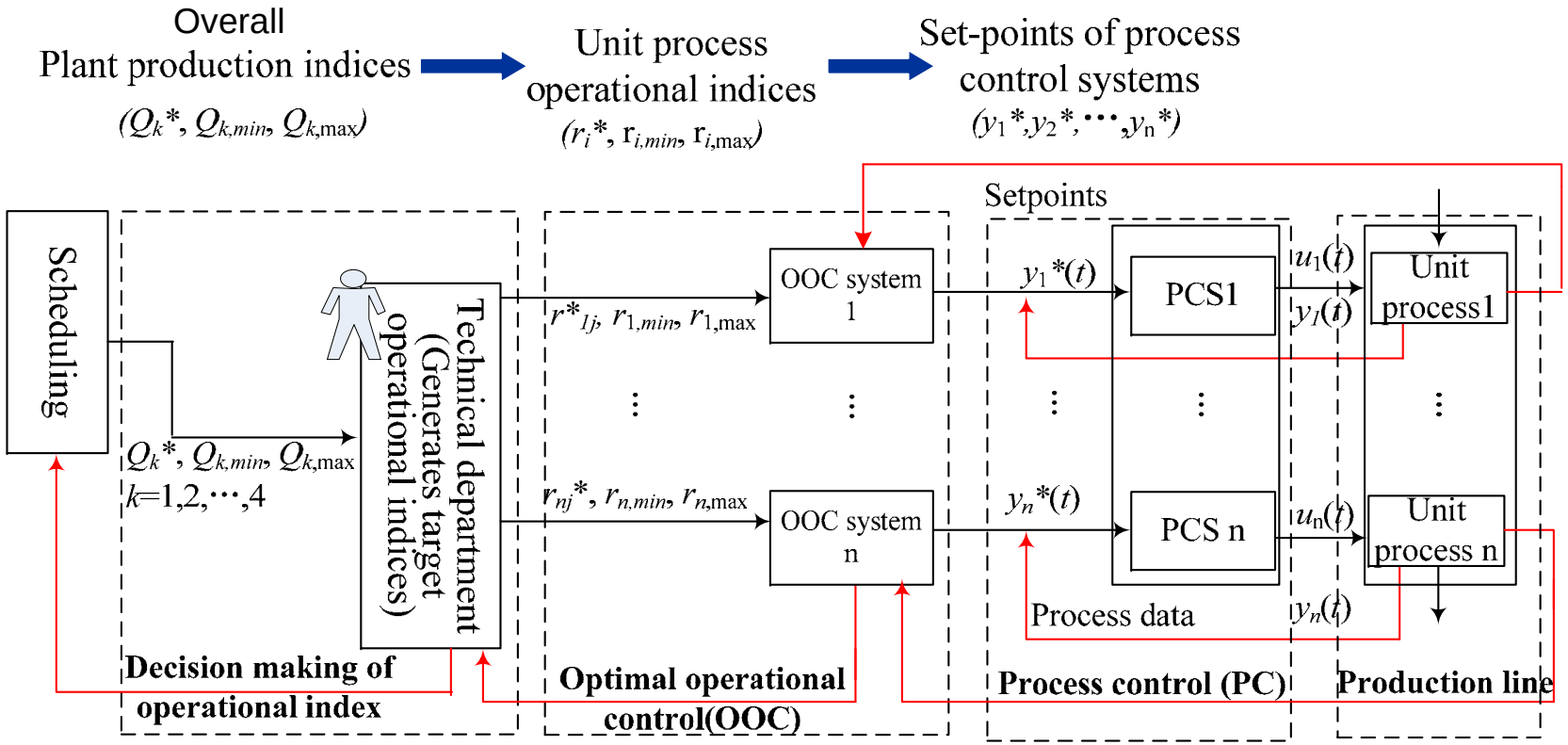
Production line for mineral processing plant





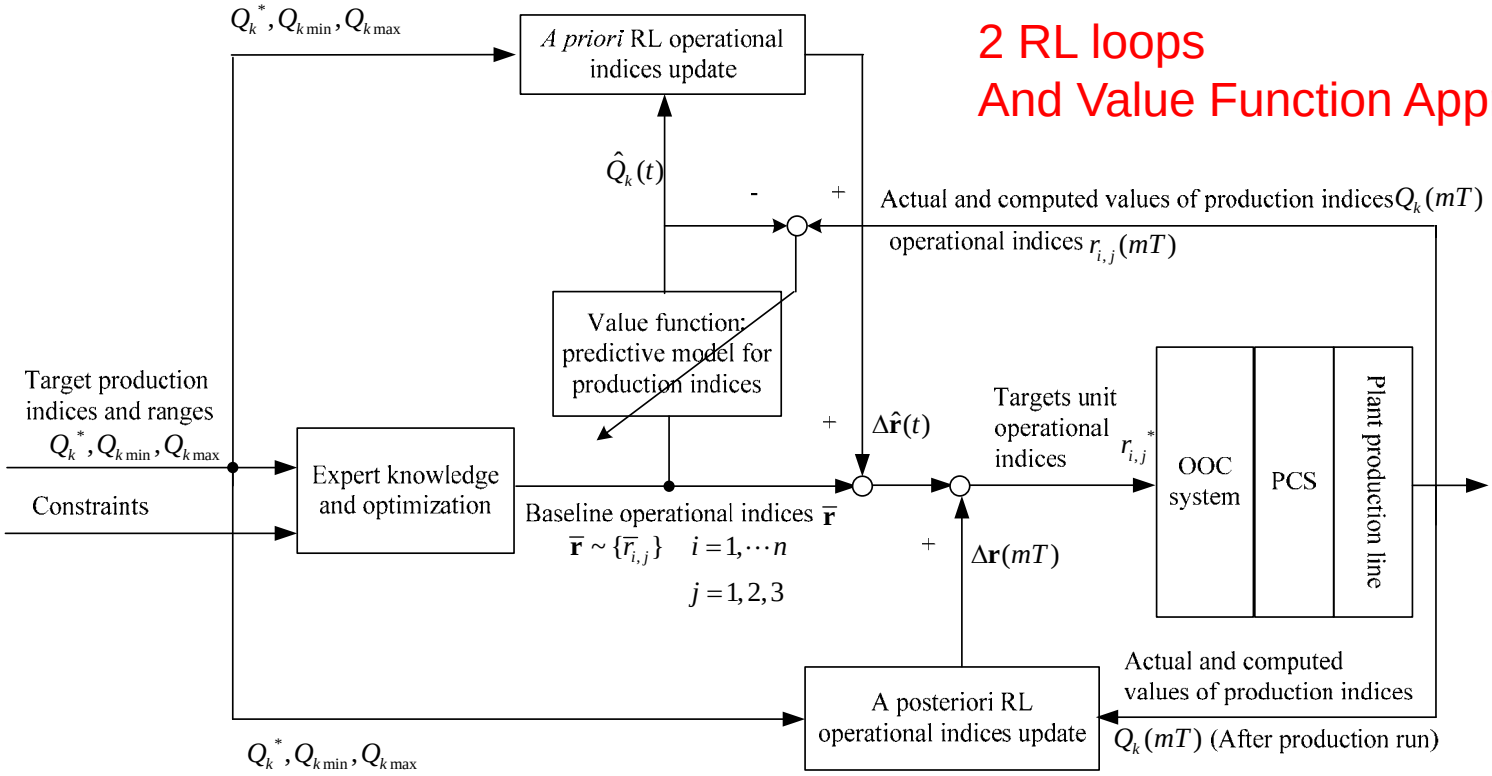


Mineral Processing Plant in Gansu China



Existing Manual Control for
Plant production indices, unit operational indices,
and unit process control for a production line

Automated online reinforcement learning for determining operational indices



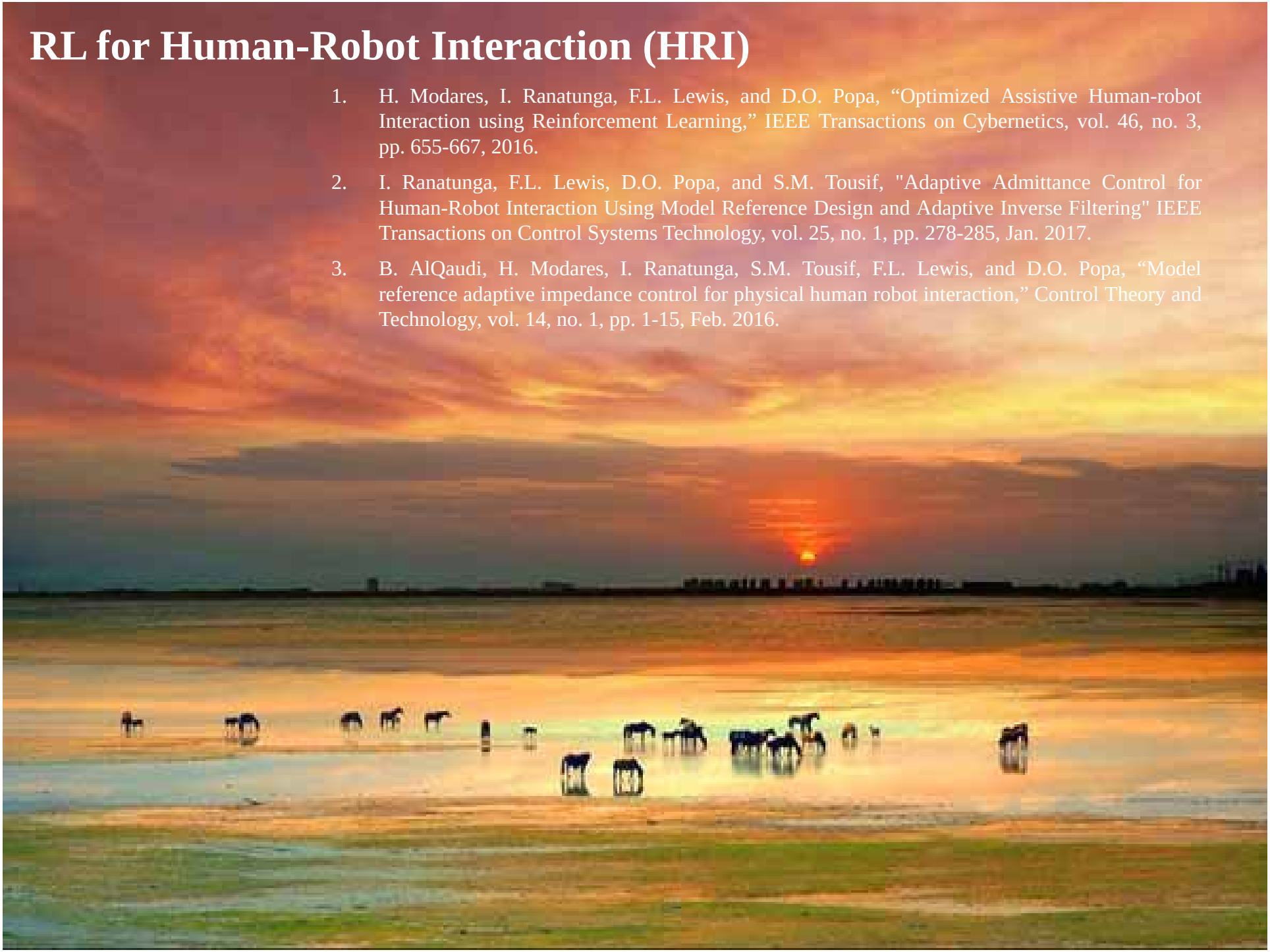
2 RL loops
And Value Function Approximation

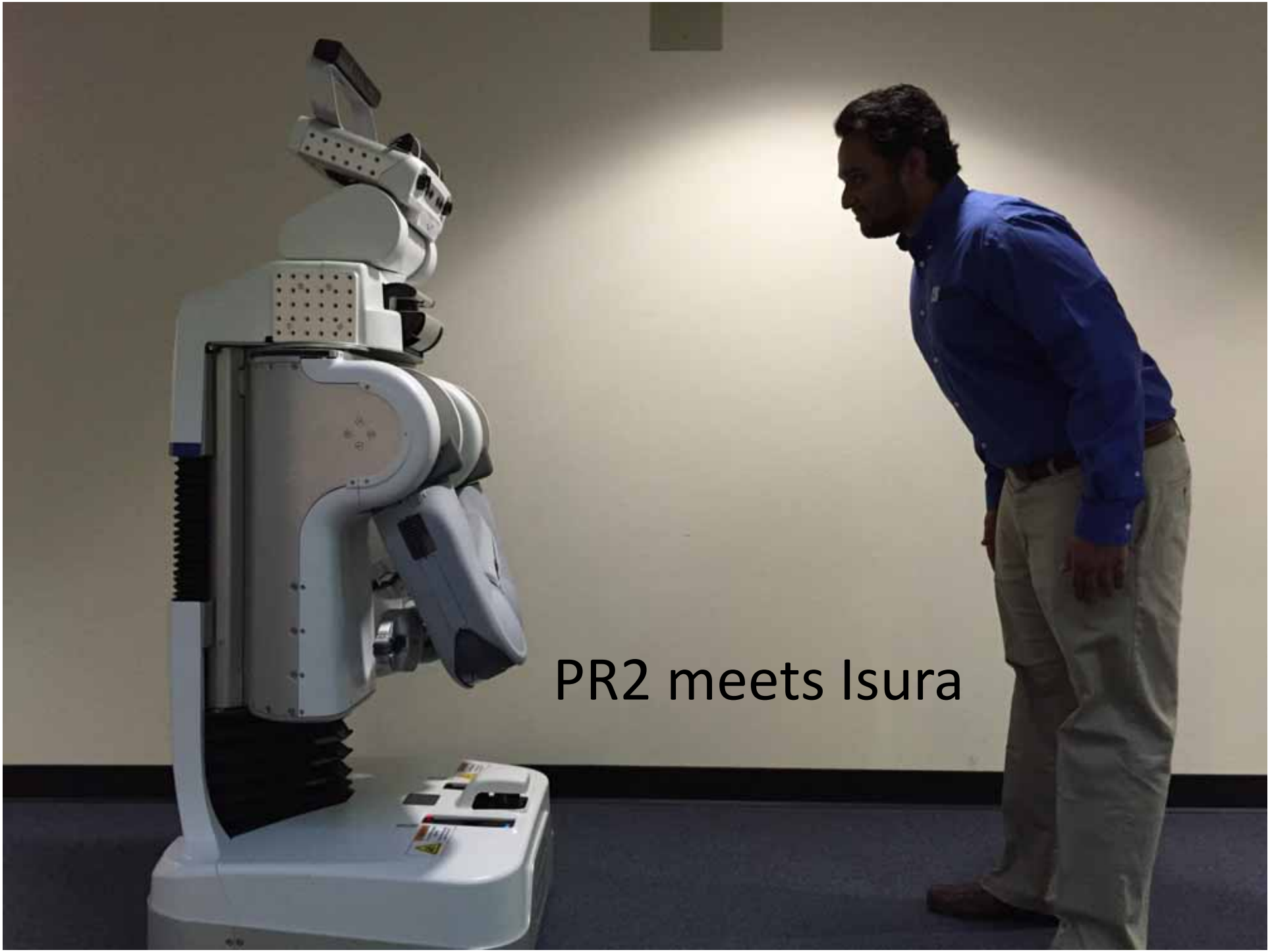
Implemented by Jingliang Ding and Chai Tianyou's group in biggest mineral processing factory of hematite iron ore in China, Gansu Province.

Savings of 30.75 million RMB per year were realized by implementing this automated optimization procedure instead of the standard industry practice of human operator selection of process operational indices.

RL for Human-Robot Interaction (HRI)

1. H. Modares, I. Ranatunga, F.L. Lewis, and D.O. Popa, “Optimized Assistive Human-robot Interaction using Reinforcement Learning,” IEEE Transactions on Cybernetics, vol. 46, no. 3, pp. 655-667, 2016.
2. I. Ranatunga, F.L. Lewis, D.O. Popa, and S.M. Tousif, "Adaptive Admittance Control for Human-Robot Interaction Using Model Reference Design and Adaptive Inverse Filtering" IEEE Transactions on Control Systems Technology, vol. 25, no. 1, pp. 278-285, Jan. 2017.
3. B. AlQaudi, H. Modares, I. Ranatunga, S.M. Tousif, F.L. Lewis, and D.O. Popa, “Model reference adaptive impedance control for physical human robot interaction,” Control Theory and Technology, vol. 14, no. 1, pp. 1-15, Feb. 2016.




PR2 meets Isura

Impedance Control

Robot dynamics

The rigid body dynamics in task coordinates [5]^2 is:

$$\Lambda(x,\dot{x})\ddot{x} + \mu(x,\dot{x})\dot{x} + J^{-T}g(x) = J^{-T}\tau + f, \qquad (2)$$
$$\Lambda(x,\dot{x}) = J^{-T}MJ^{-1} \quad \mu(x,\dot{x}) = J^{-1}(C - MJ^{-1}\dot{J})J^{-1},$$

Prescribed Error system

Impedance control in the task space consists of the following control objective [6]:

$$\Lambda_d\ddot{e}_x + D_d\dot{e}_x + K_d e_x = e_f \qquad (3)$$

where $e_x = x - x_d$ is the position error between the actual position x and the reference position x_d; $e_f = f - f_d$ measures

Control torque depends on Impedance model parameters

$$\tau = u + J^T\tilde{K}e_x + J^T\tilde{D}\dot{e}_x + J^T\tilde{\Lambda}_d e_f, \qquad (4)$$

where $u = g + J^T(\Lambda\ddot{x}_d + \mu\dot{x}_d)$ (5)

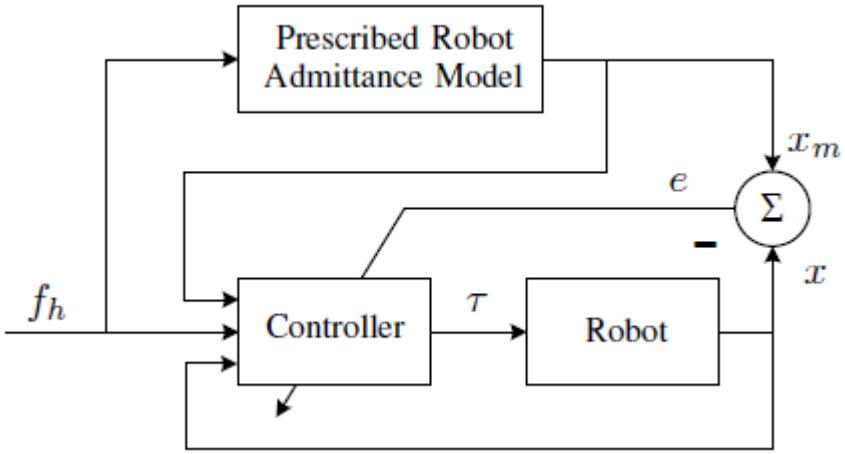
$$\tilde{K}_d = \Lambda\Lambda_d^{-1}K_d,\ \tilde{D}_d = \Lambda\Lambda_d^{-1}D_d + \mu,\ \tilde{\Lambda}_d = \Lambda\Lambda_d^{-1} - I.$$

Human Performance Factors Studies

Human task learning has 2 components:

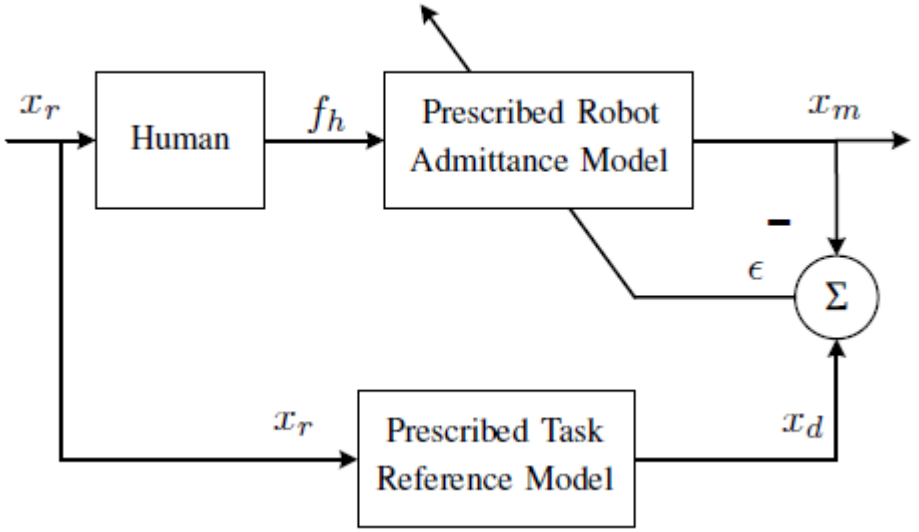
1. Human learns a robot dynamics model to compensate for robot nonlinearities
2. Human learns a task model to properly perform a task

Inner Robot Specific Control Loop
INDEPENDENT OF TASK



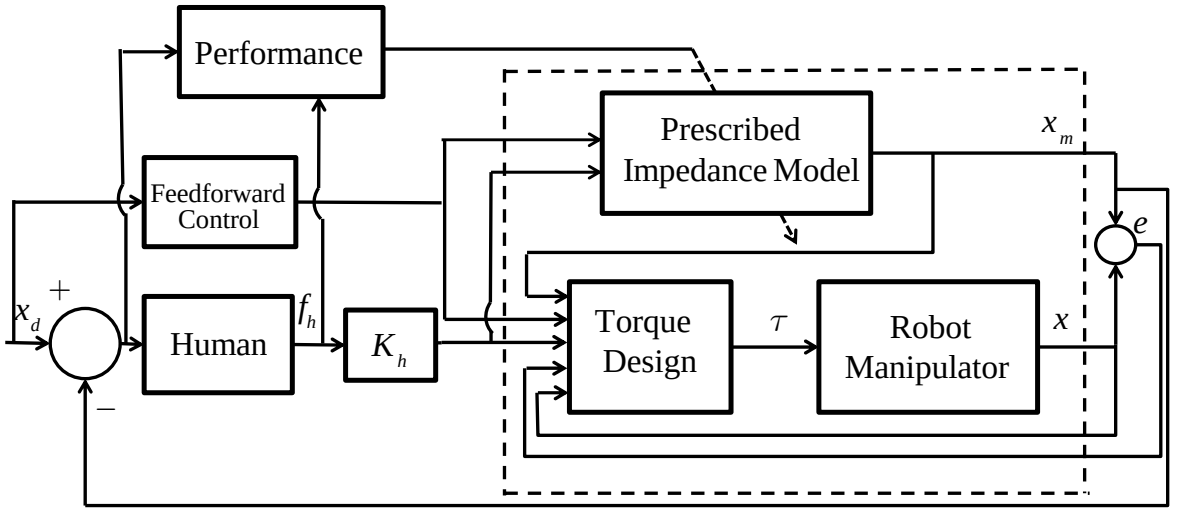
This paper develops a two-loop design approach for adaptive admittance control for human-robot interaction. This approach follows the human factors studies [44], [45] that indicate human learning in task performance has two components: a robot-specific component whereby a robot dynamics model is learned to compensate for robot nonlinearities, and a component where task-related details are learned. Here, an inner

Outer Task Specific Control Loop
INDEPENDENT OF ROBOT DETAILS



RL for Human-Robot Interactions

H. Modares, I. Ranatunga, F.L. Lewis, and D.O. Popa, "Optimized Assistive Human-robot Interaction using Reinforcement Learning," IEEE Transactions on Cybernetics, to appear 2015.



Robot-specific inner control loop

Task-specific outer-loop control design
Using IRL Optimal Tracking



Implemented on PR2 robot

Inspired by work of Sam S. Ge



