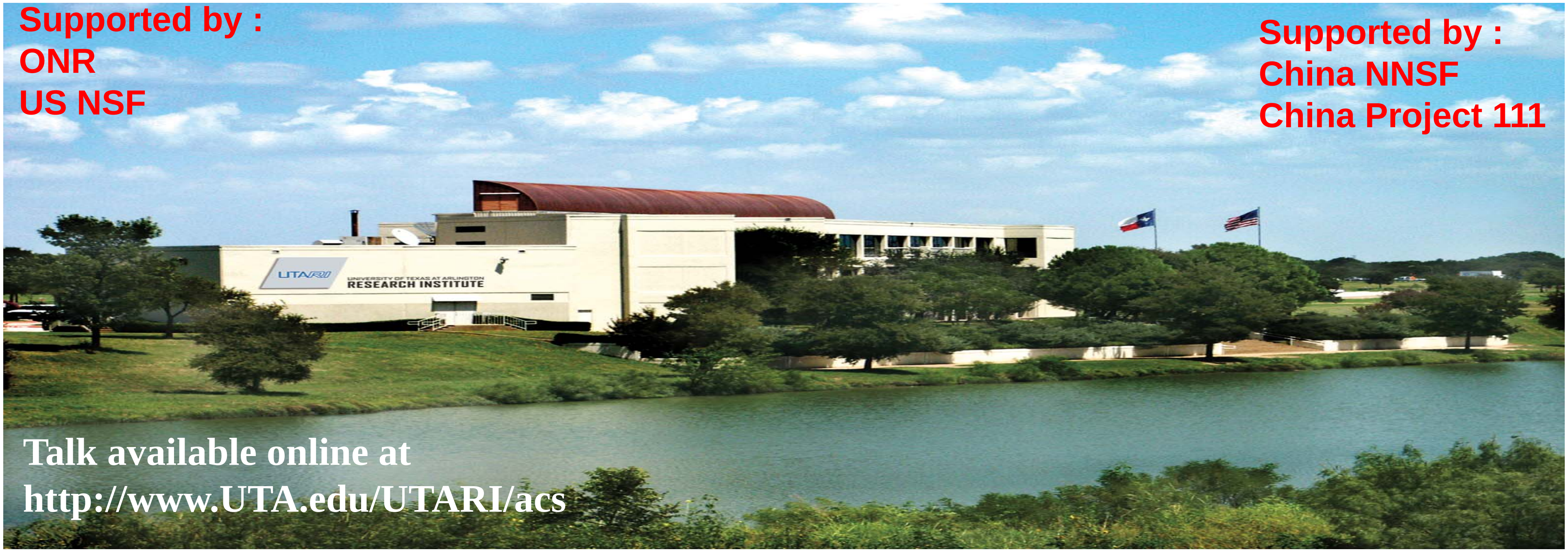
UNIVERSITY OF TEXAS AT ARLINGTON RESEARCH INSTITUTE

F.L. Lewis
National Academy of Inventors

The University of Texas ARLINGTON

Moncrief-O'Donnell Chair, UTA Research Institute (UTARI)
The University of Texas at Arlington, USA
and
Qian Ren Consulting Professor, State Key Laboratory of Synthetical Automation for Process Industries
Northeastern University, Shenyang, China

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




Invited by
Manfred Morari
Konstantinos Gatsis
Pramod Khargonekar
George Pappas

**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

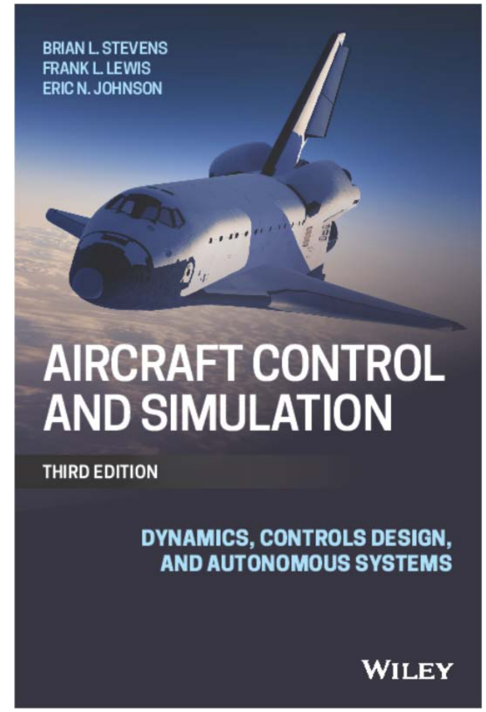
Optimal Control is Effective for:

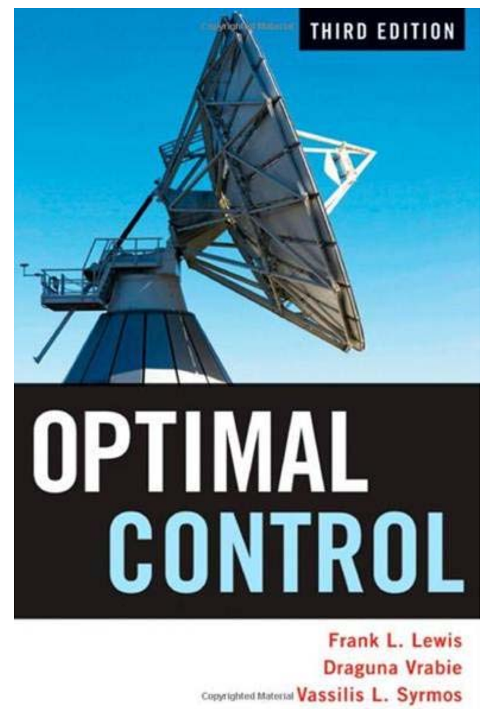
- Aircraft Autopilots
- Vehicle engine control
- Aerospace Vehicles
- Ship Control
- Industrial Process Control

Multi-player Games Occur in:

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

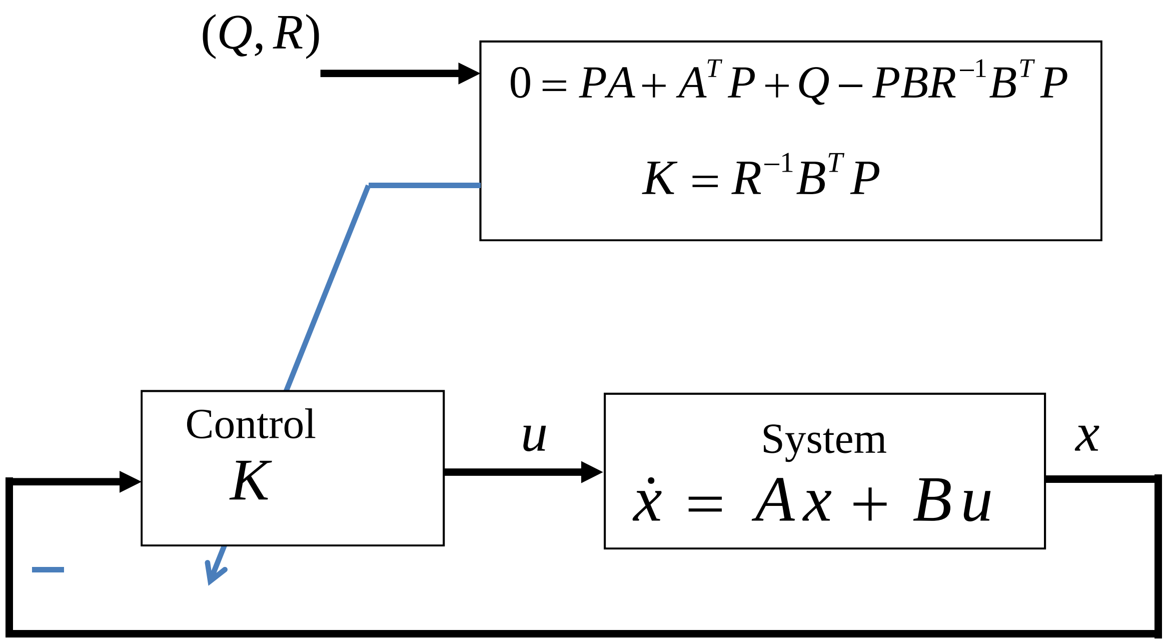
But, optimal control and game solutions are found by
- Offline solution of Matrix Design equations
- A full dynamical model of the system is needed



# 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

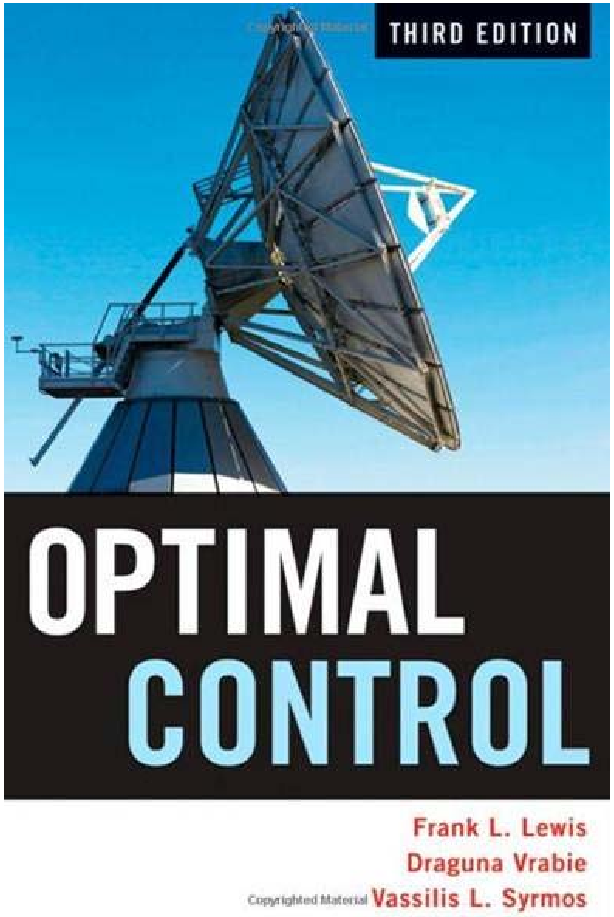
Optimal Control is off-line,
and needs to know the system dynamics to solve design eqs.

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!



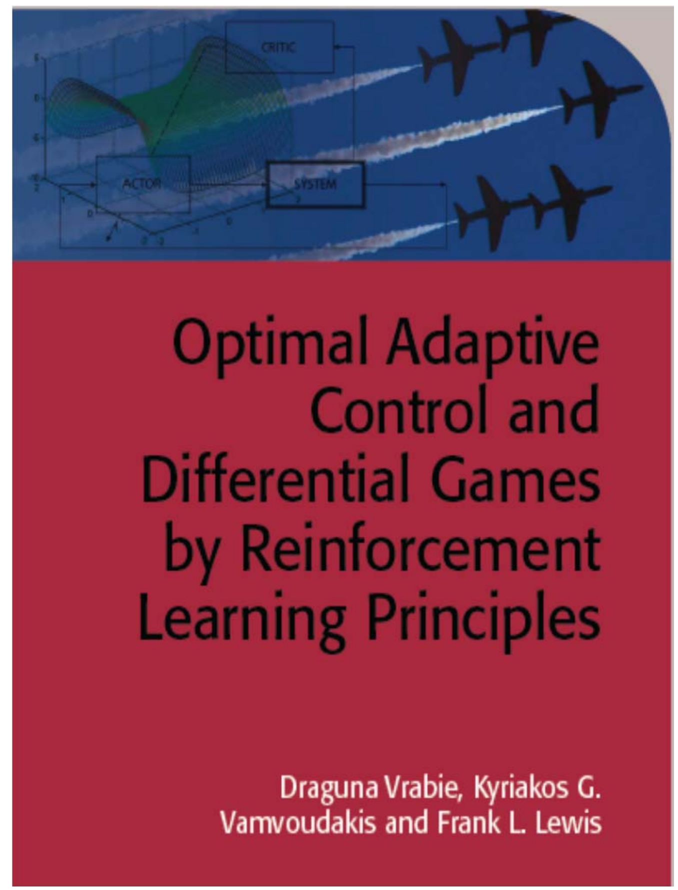

# Books

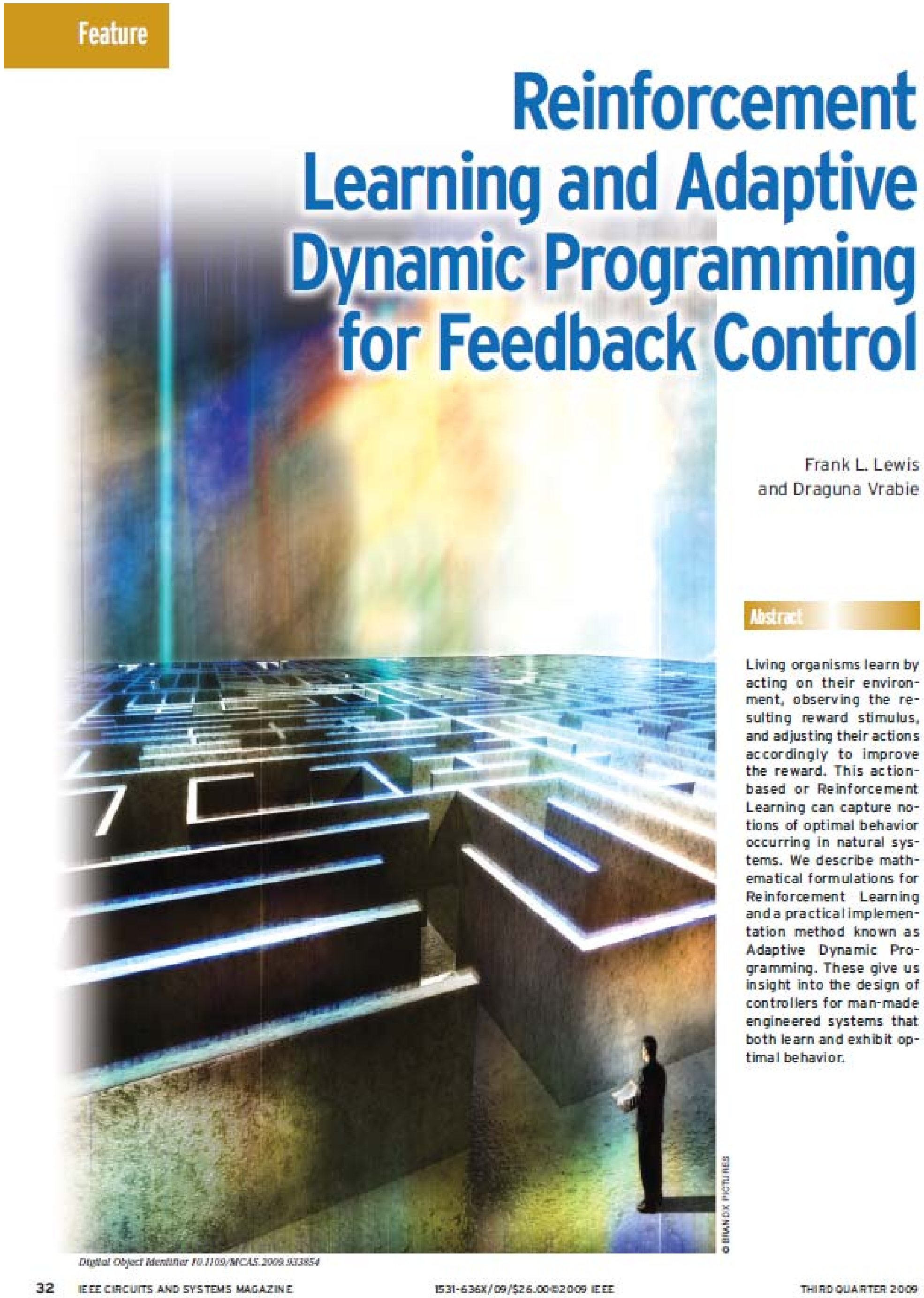
F.L. Lewis, D. Vrabie, and V. Syrmos, *Optimal Control*, third edition, John Wiley and Sons, New York, 2012.

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.

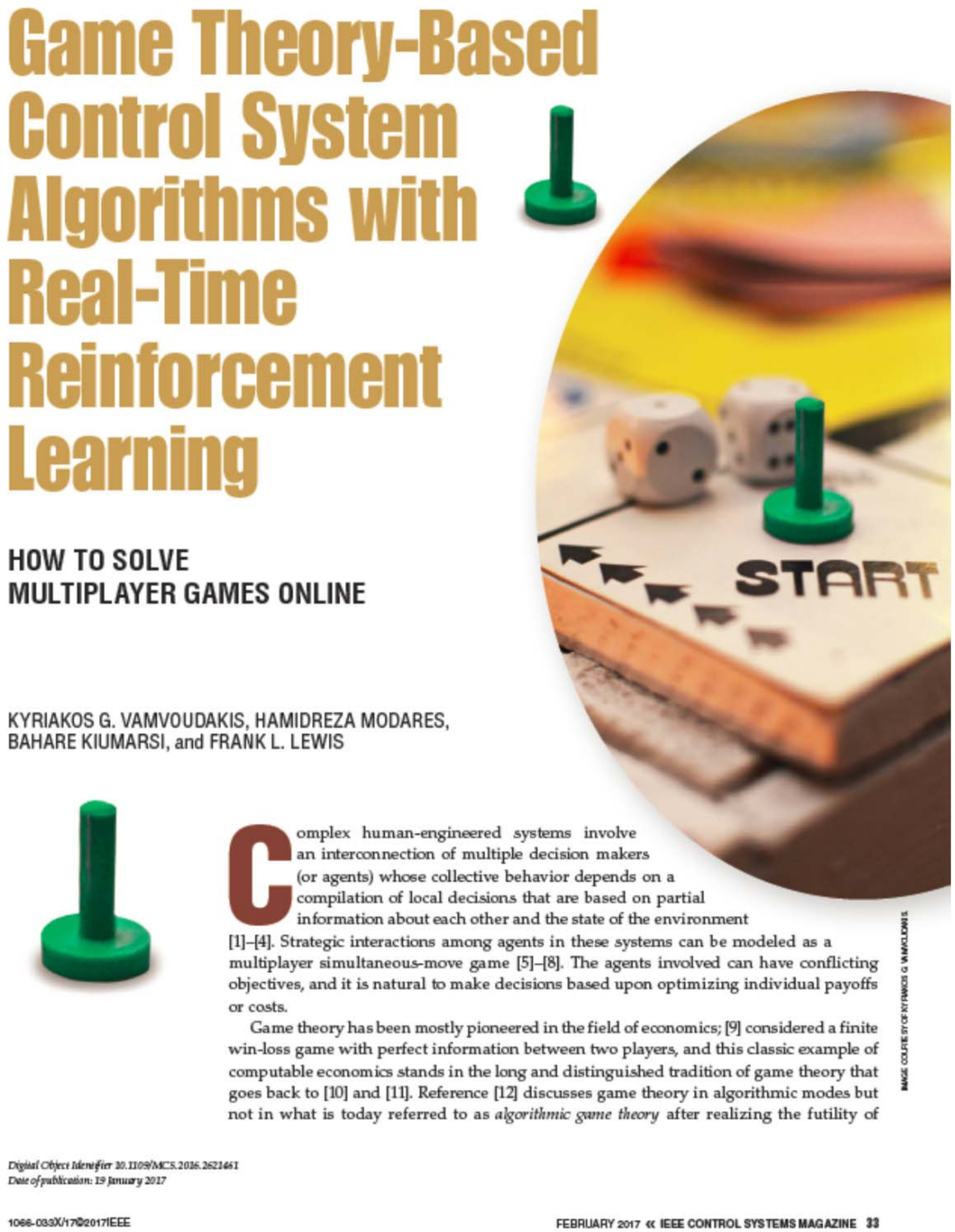
© BRAND X PICTURES

Digital Object Identifier 10.1109/MCAS.2009.933854



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*

1066-033X/17©2017IEEE



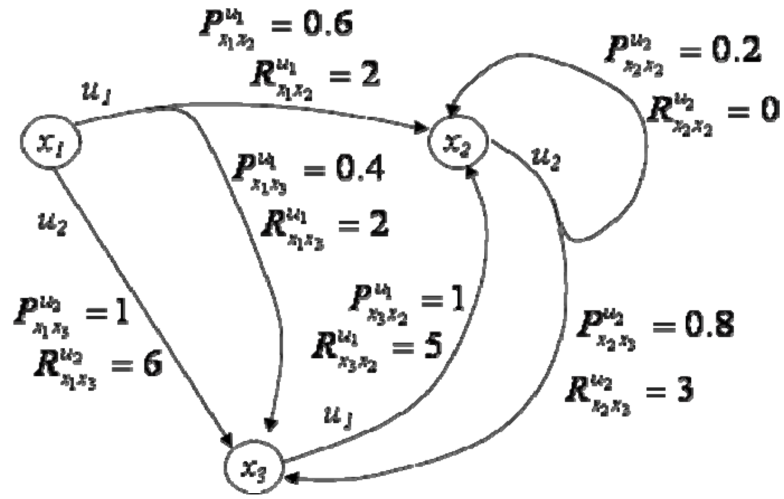
Multi-player Game Solutions
IEEE Control Systems Magazine, Dec 2017

# RL for Markov Decision Processes $(X, U, P, R)$

*X*= states, *U*= controls

P= Probability of going to state x' from state x given that the control is u

R= Expected reward on going to state x' from state x given that the control is u




*Expected Value* of a policy $\pi(x,u)$

$$V_k^\pi(x) = E_\pi\{J_{k,T} \mid x_k = x\} = E_\pi\{\sum_{i=k}^{k+T} \gamma^{i-k} r_i \mid x_k = x\}$$

*Optimal control problem*

determine a policy $\pi(x,u)$ to minimize the expected future cost

*optimal policy* $\pi^*(x,u) = \arg\min_\pi V_k^\pi(s) = \arg\min_\pi E_\pi\{\sum_{i=k}^{k+T} \gamma^{i-k} r_i \mid x_k = x\}$.

*optimal value* $V_k^*(x) = \min_\pi V_k^\pi(x) = \min_\pi E_\pi\{\sum_{i=k}^{k+T} \gamma^{i-k} r_i \mid x_k = x\}$.

Discrete State

## Policy Iteration

Policy evaluation by Bellman eq. $V_j(x) = \sum_u \pi_j(x,u) \sum_{x'} P^u_{xx'} \left[ R^u_{xx'} + \gamma V_j(x') \right]$ *for all* $x \in X$ .

Policy Improvement $\pi_{j+1}(x,u) = \arg\min_u \sum_{x'} P^u_{xx'} \left[ R^u_{xx'} + \gamma V_j(x') \right]$ *for all* $x \in X$ .

Policy Evaluation equation is a system of *N* simultaneous linear equations, one for each state.

Policy Improvement makes $V^{\pi'}(x) \le V^\pi(x)$


R.S. Sutton and A.G. Barto, Reinforcement Learning– An Introduction, MIT Press, Cambridge, Massachusetts, 1998.
D.P. Bertsekas and J. N. Tsitsiklis, Neuro-Dynamic Programming, Athena Scientific, MA, 1996.
W.B. Powell, Approximate Dynamic Programming: Solving the Curses of Dimensionality, Wiley, New York, 2009.

# 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?

Bertsekas- Neurodynamic Programming

# 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

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

HJB equation

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

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.

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

$$V_j(0) = 0$$

Policy improvement - Update control

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

Converges to solution of HJB

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

- 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

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

---

Differential equivalent is the Bellman equation

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

Given any stabilizing FB policy $u = -Kx$

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

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

---

Optimal Control is

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

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 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

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

# 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

# Approximate Dynamic Programming Implementation

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

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

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

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

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

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

regression vector

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 or batch least-squares 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 $\begin{bmatrix} p_{11} & p_{12} \\ p_{12} & p_{22} \end{bmatrix}$ $\qquad W^T = [p_{11} \quad p_{12} \quad p_{22}]$

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)

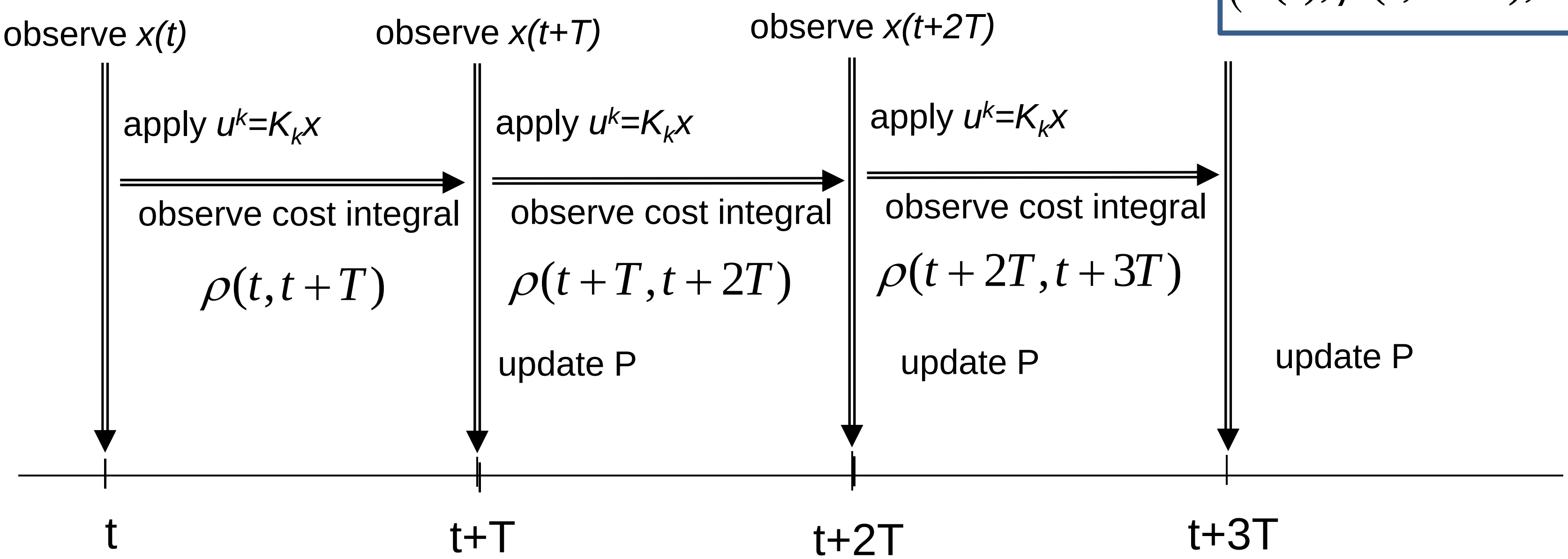
# Integral Reinforcement Learning (IRL)

Solve Bellman Equation - Solves Lyapunov eq. without knowing dynamics

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

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

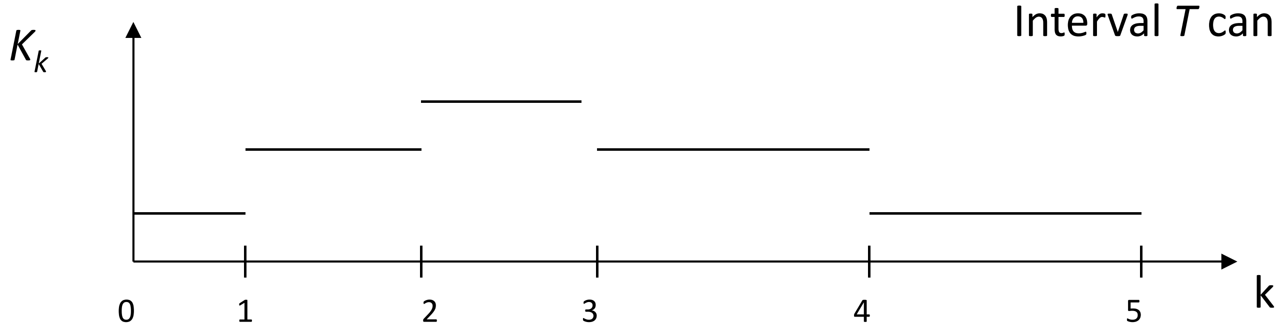
This is a data-based approach that uses measurements of *x(t), u(t)*
Instead of the plant dynamical model.

update control gain

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

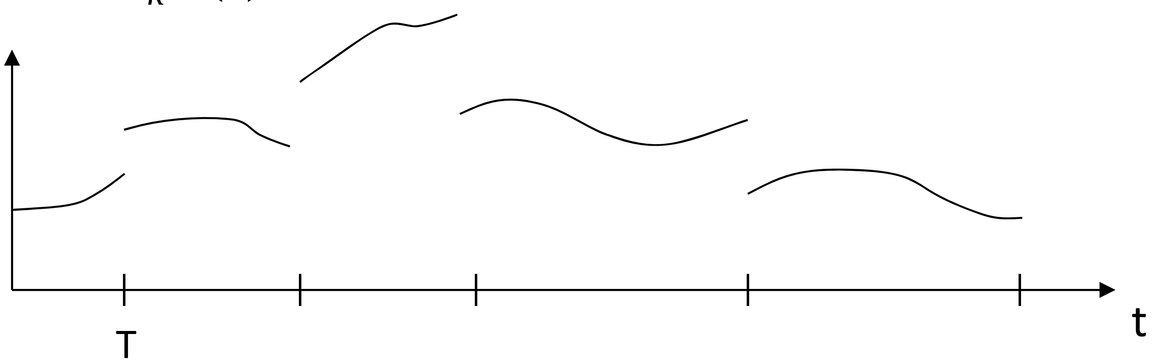
Gain update (Policy)

Interval $T$ can vary




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

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

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

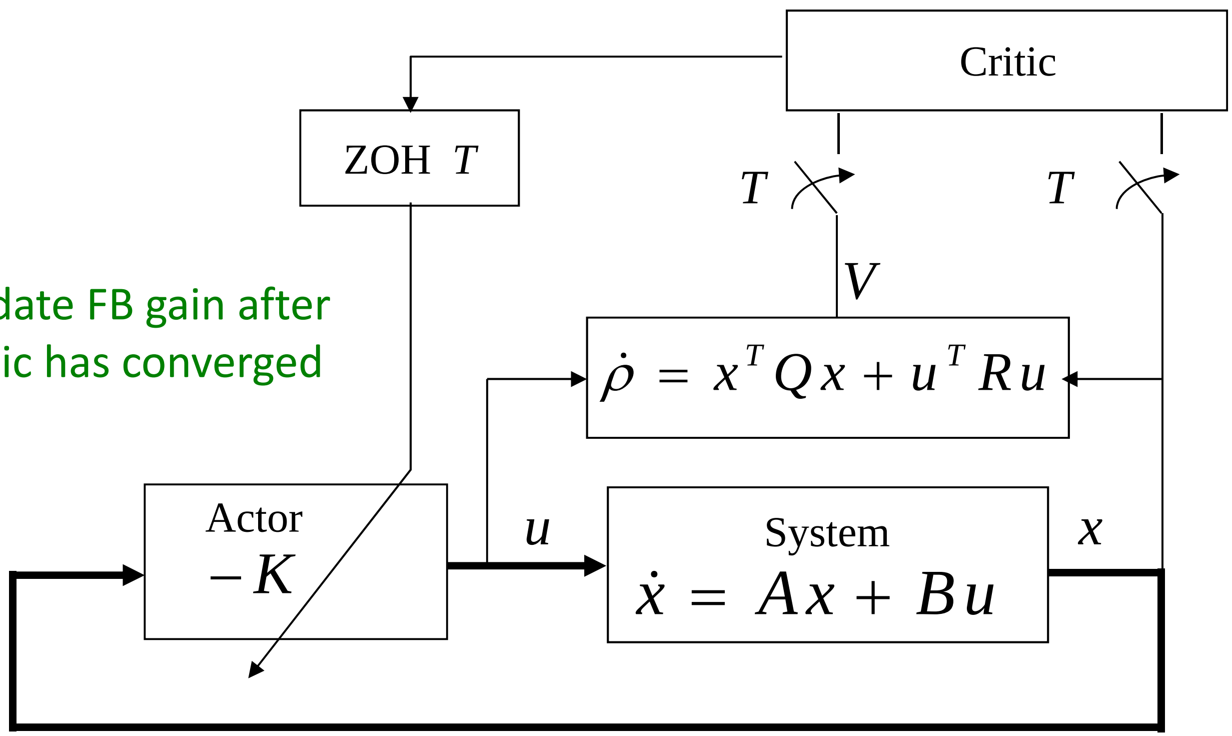
# Direct Optimal Adaptive Controller

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

Update FB gain after
Critic has converged




IRL requires a
Dynamic
Control
System
w/ MEMORY

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

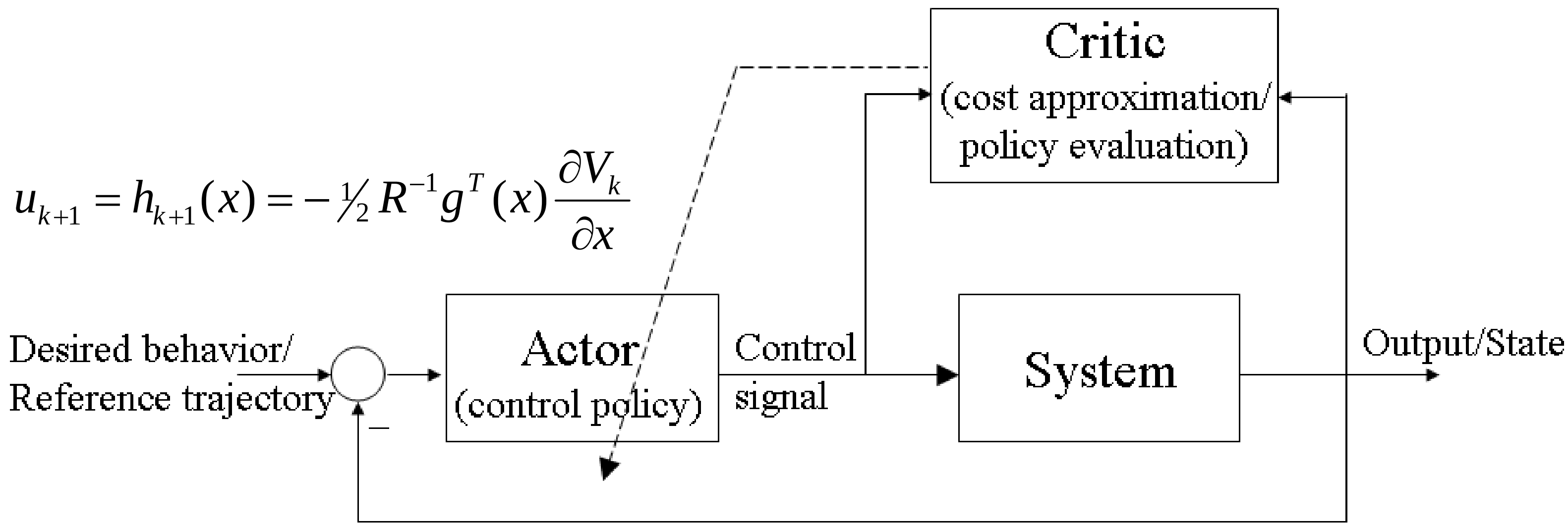
## Actor / Critic structure for CT Systems

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

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

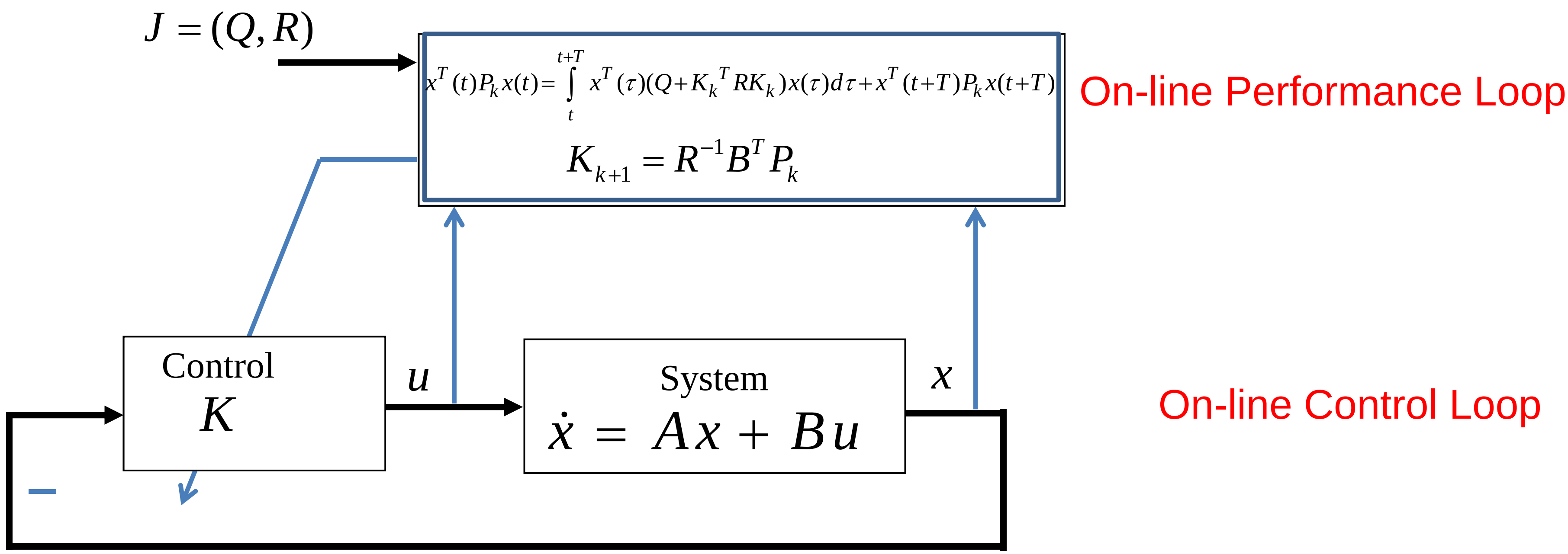



Motor control 200 Hz

## A new structure of adaptive controllers

# Data-driven Online Adaptive Optimal Control
# DDO

User prescribed optimization criterion




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

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

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

# 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

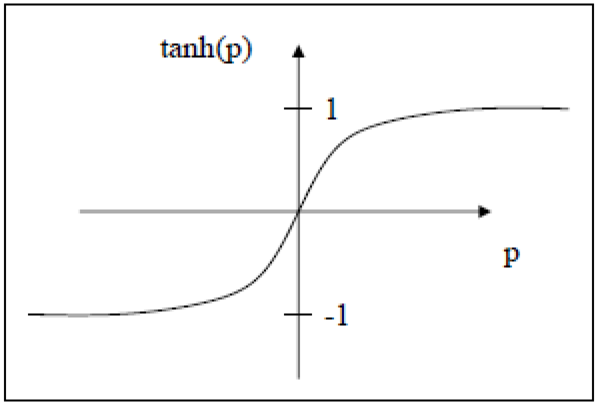
$$J = \int_0^T 1 \, dt = T$$

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

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

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

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

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**

# Optimal Adaptive IRL for CT systems

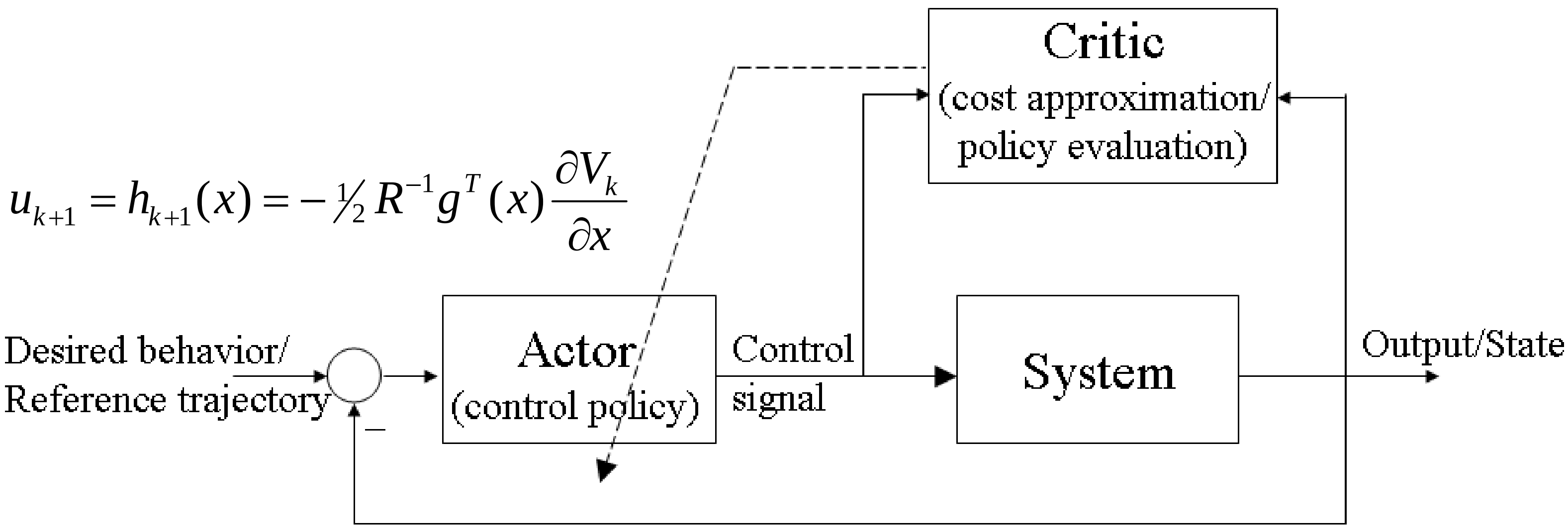
D. Vrabie, 2009

## 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

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




Motor control 200 Hz

# A new structure of adaptive controllers

Deliberative evaluation

Limbic system

control

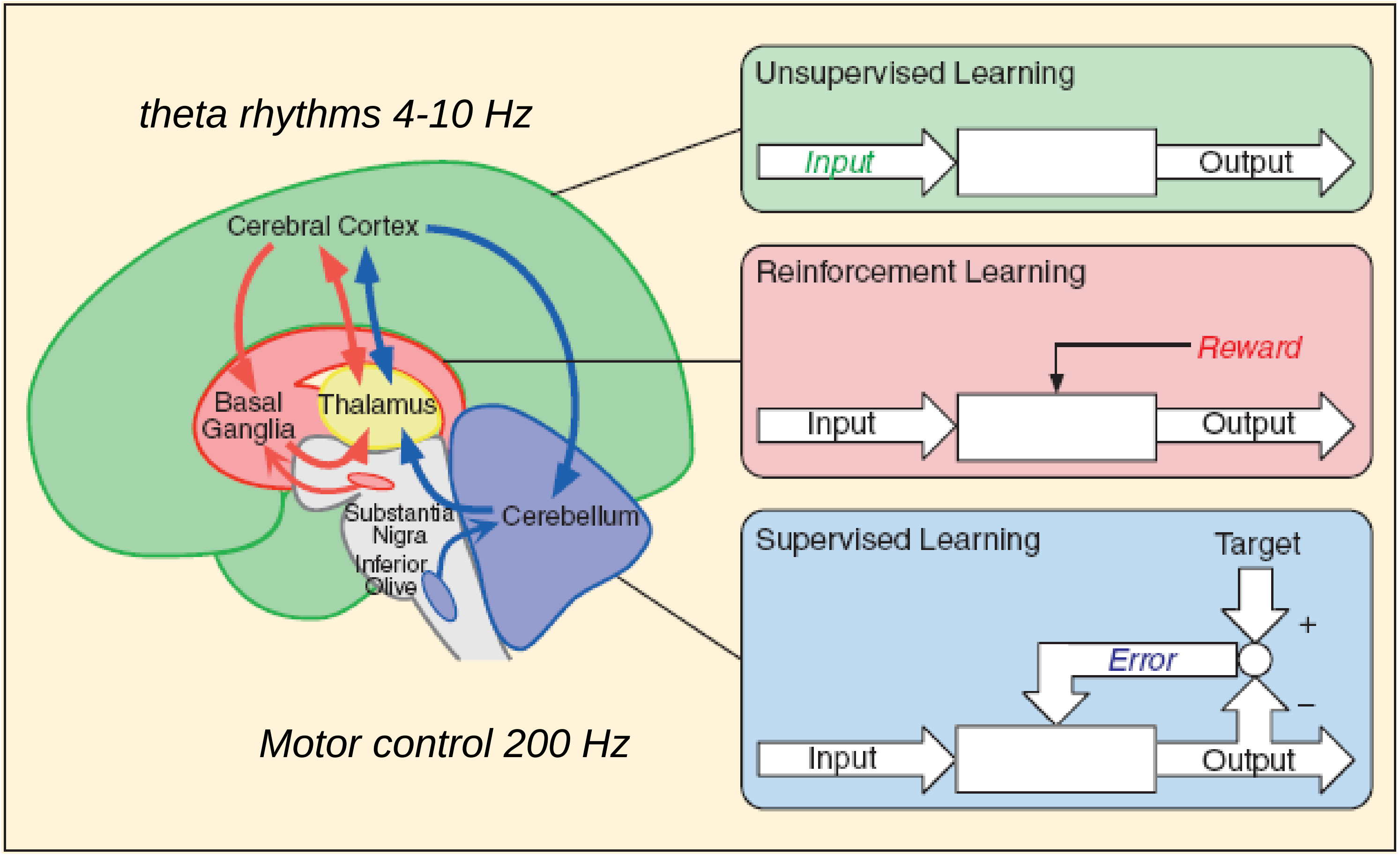
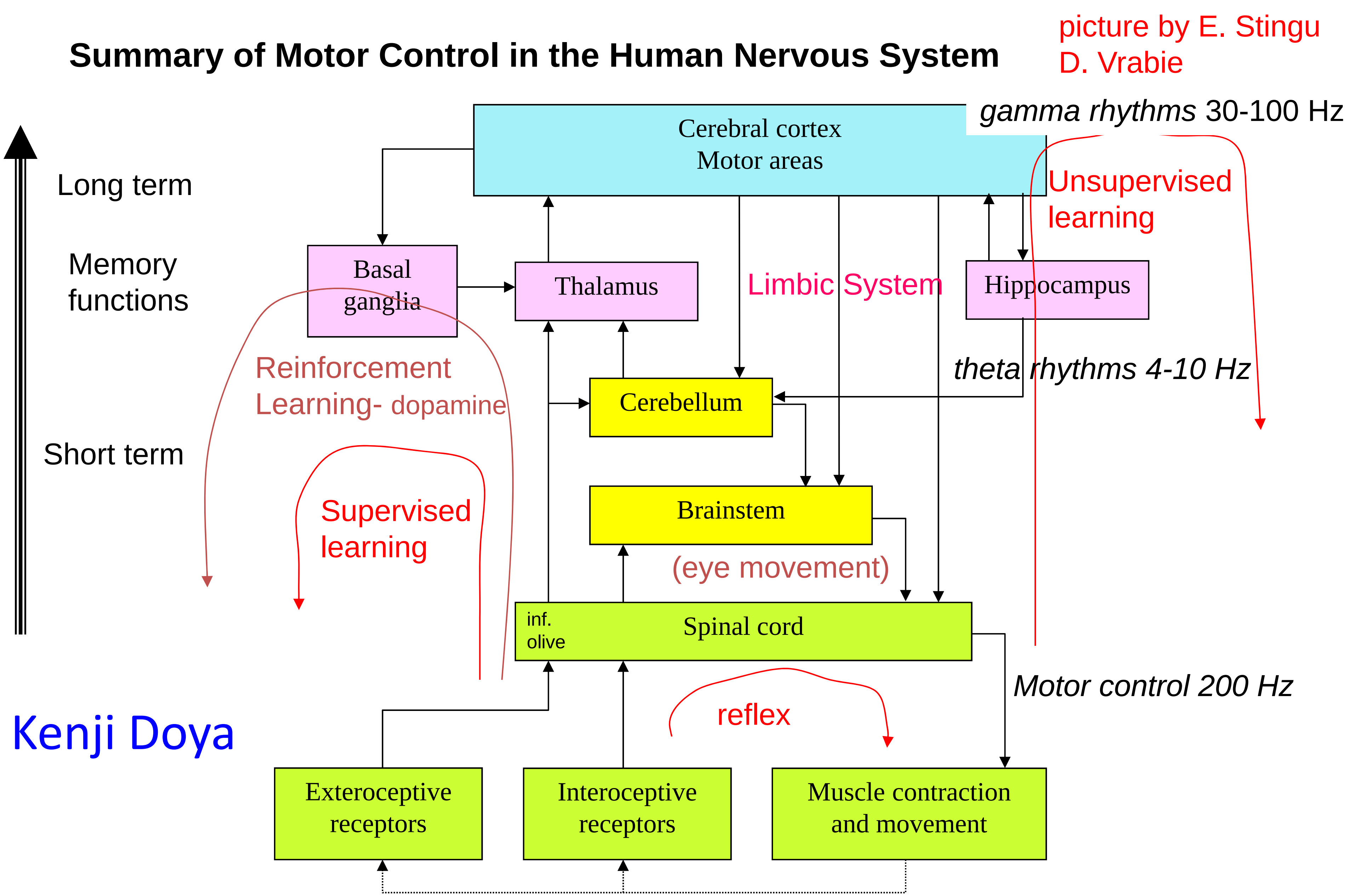



**Figure 1.** *Learning-oriented specialization of the cerebellum, the basal ganglia, and the cerebral cortex [1], [2]. The cerebellum is specialized for supervised learning based on the error signal encoded in the climbing fibers from the inferior olive. The basal ganglia are specialized for reinforcement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




Hierarchy of multiple parallel loops


Synchronous Real-time Data-driven Optimal Control

# Optimal Adaptive
# Integral Reinforcement Learning for CT systems

D. Vrabie, 2009

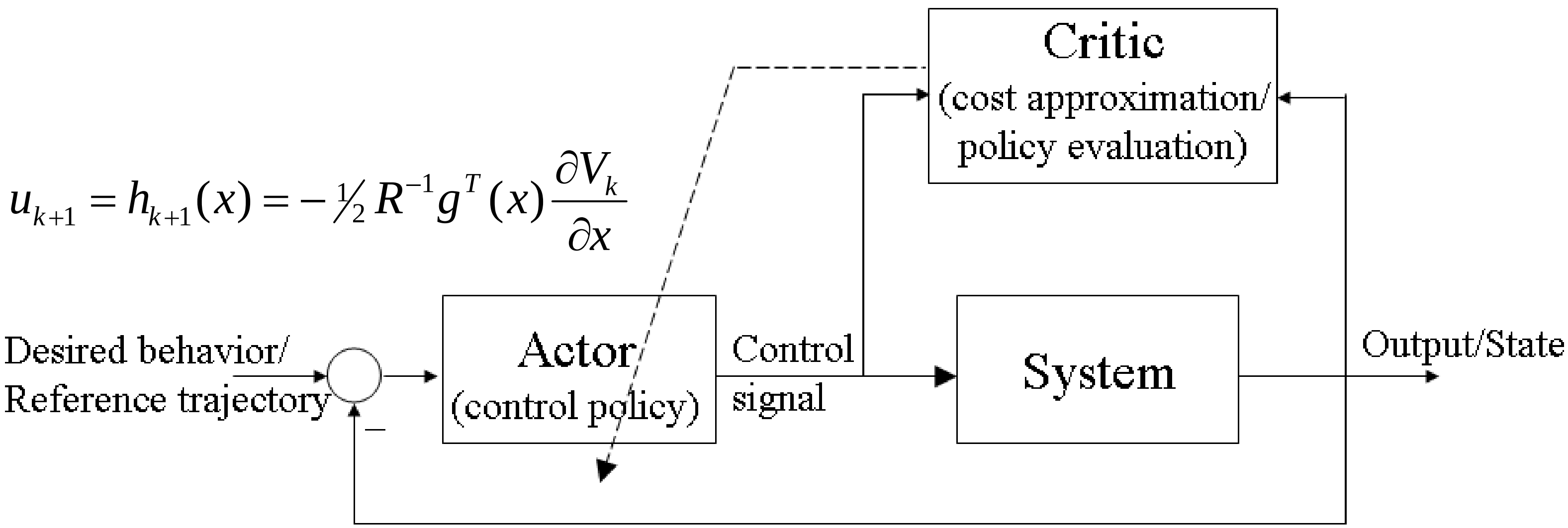
Policy Iteration gives the structure needed for online optimal solution

## 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

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




Motor control 200 Hz

A new structure of adaptive controllers

# Synchronous Online Solution of Optimal Control for Nonlinear Systems

Kyriakos Vamvoudakis

## 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

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

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.

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

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

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

## 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.

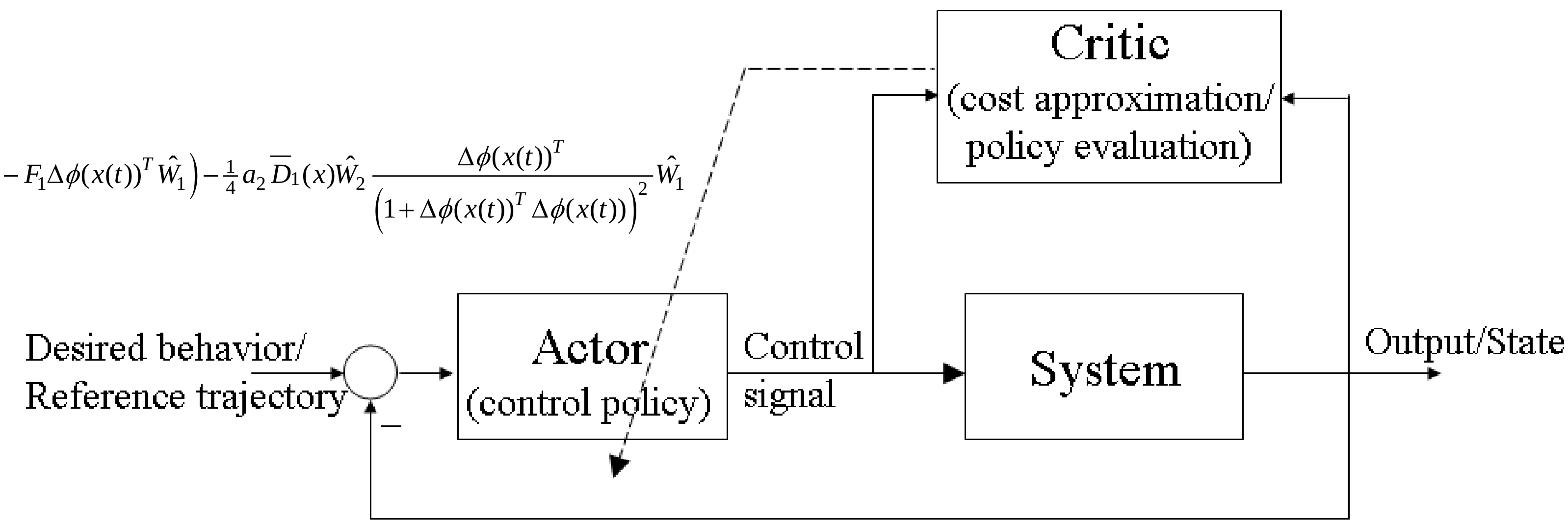
A new form of Adaptive Control with TWO tunable networks

# Adaptive Critic structure

Reinforcement learning

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

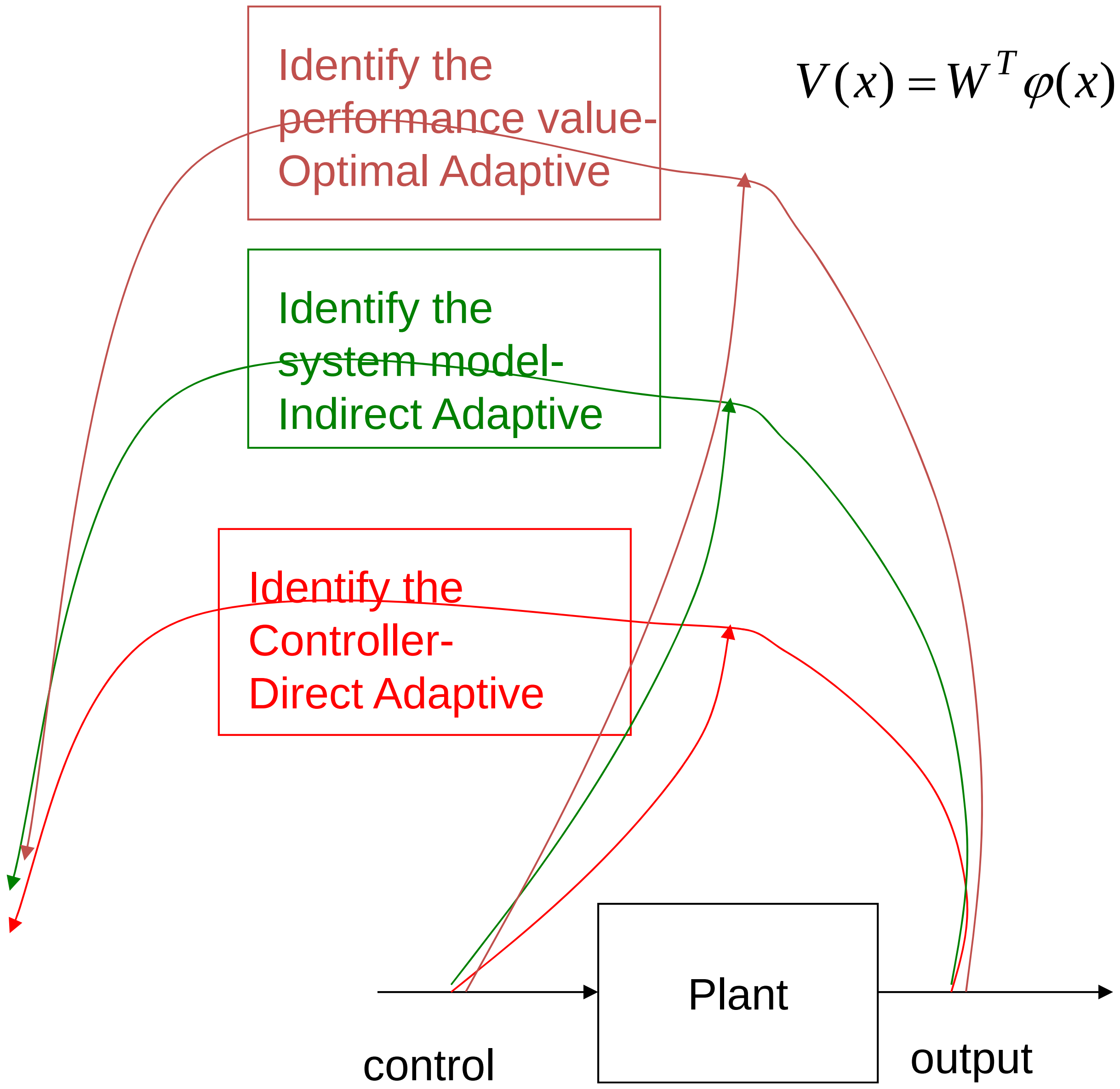
$$\dot{\hat{W}}_2 = -a_2\left(F_2\hat{W}_2 - F_1\Delta\phi(x(t))^T\hat{W}_1\right) - \tfrac{1}{4}a_2\bar{D}_1(x)\hat{W}_2\frac{\Delta\phi(x(t))^T}{\left(1+\Delta\phi(x(t))^T\Delta\phi(x(t))\right)^2}\hat{W}_1$$




Two Learning Networks
Tune them Simultaneously

## A new structure of adaptive controllers

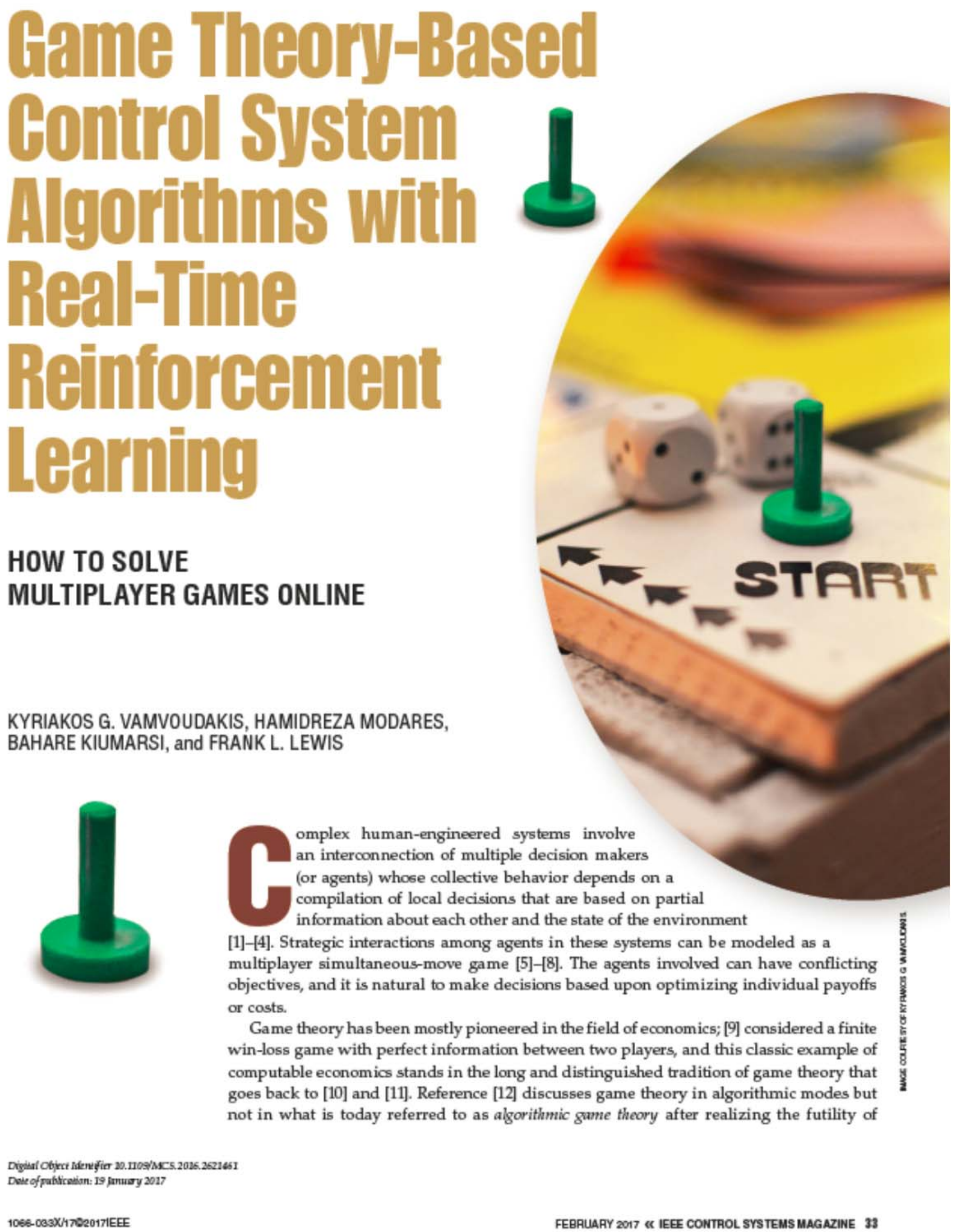
# A New Class of Adaptive Control



# 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

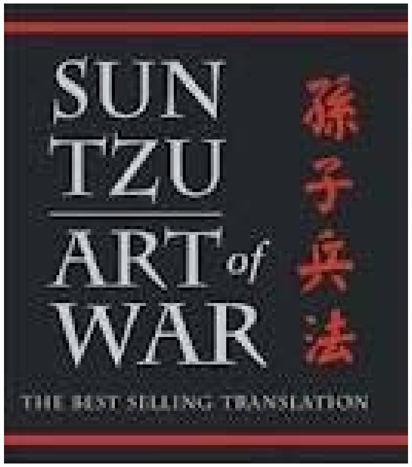
*Digital Object Identifier 10.1109/MCS.2016.2621461*
*Date of publication: 19 January 2017*

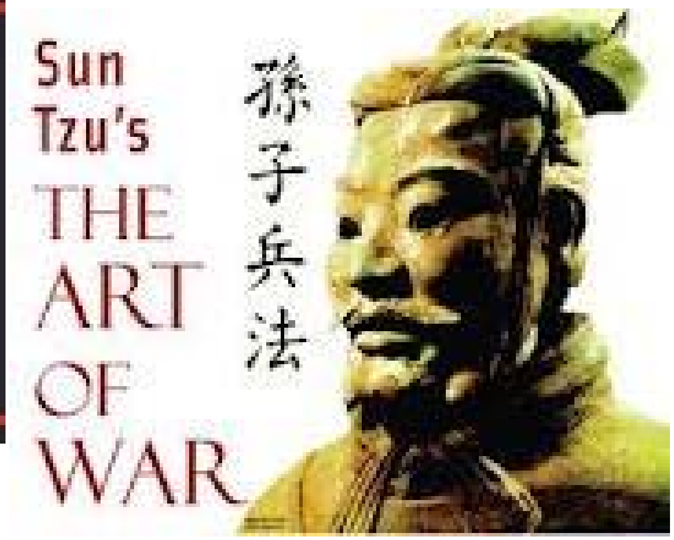




Multi-player Game Solutions
IEEE Control Systems Magazine, Dec 2017

# 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

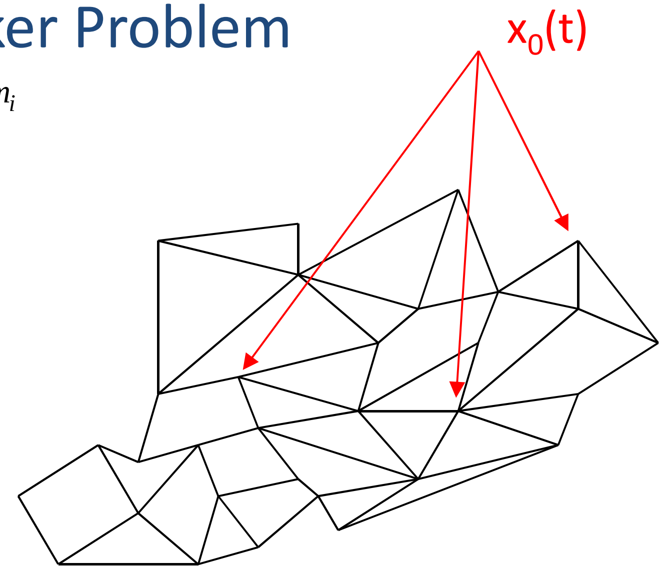

Key Point

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

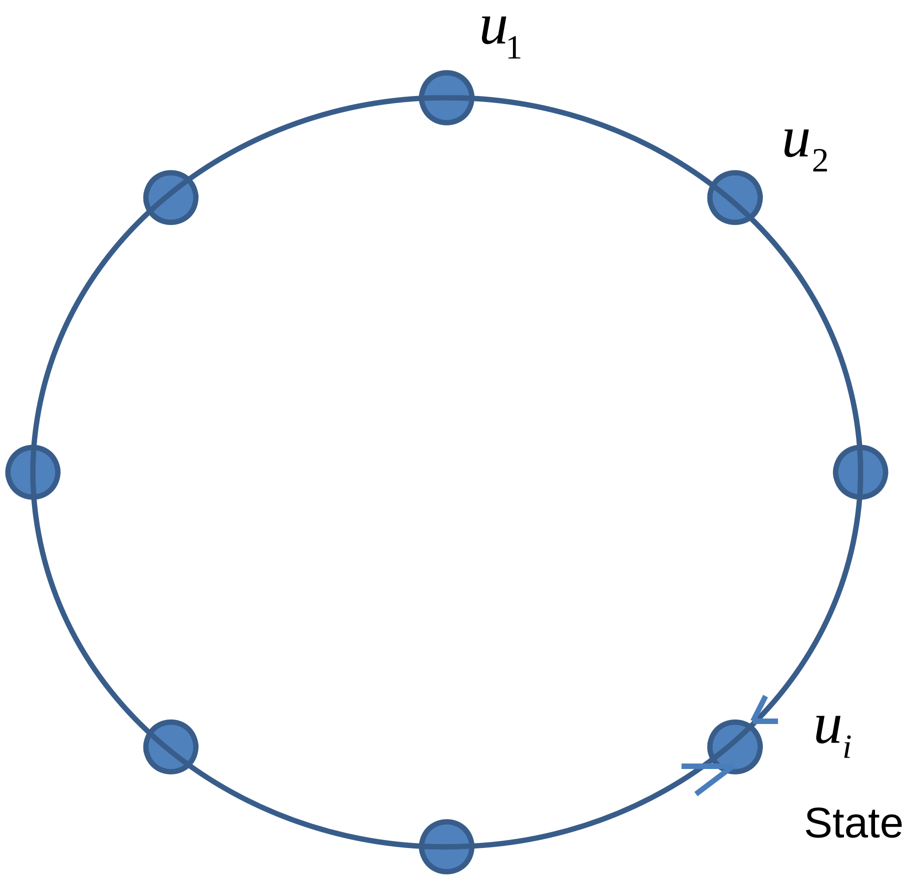
Values driven by neighbors' controls

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

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*

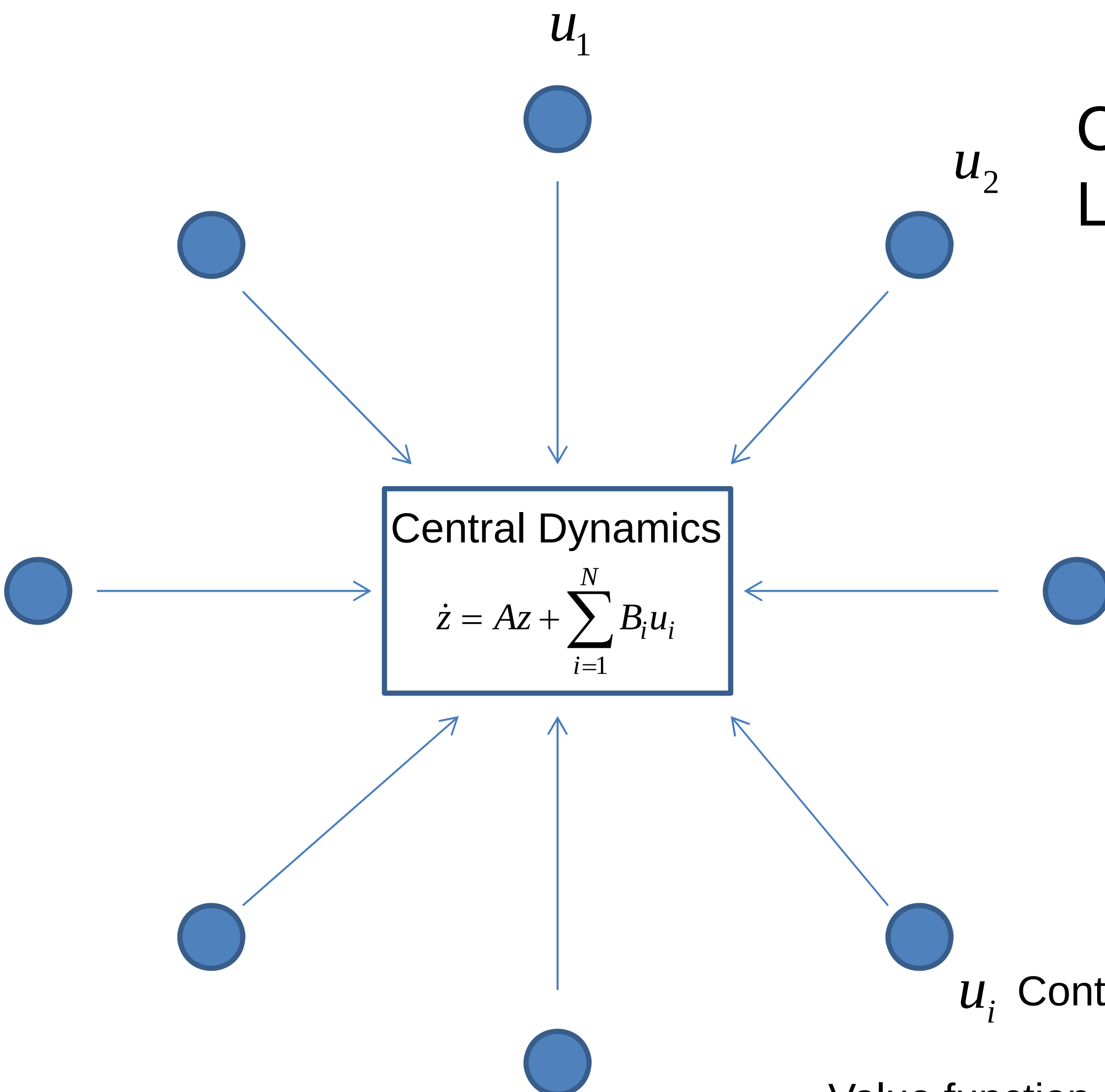
State dynamics of agent *i*

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

Value function of player *i*

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

# 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*

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

# New Definition of Nash Equilibrium for Graphical Games

## To restore symmetry of Nash Equilibrium

Def: Interactive Nash equilibrium

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

1. $J_i^* \triangleq J_i\,(u_i^*, u_{G-i}^*) \le J_i\,(u_i\,, u_{G-i}^*), \;\; \forall i \in N$

1. They are in Nash equilibrium

2. There exists a policy $u_j$ such that

2. Interaction Condition

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

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

---

Theorem 3. Let *(A,B$_i$)* be reachable for all *i*.
Let agent *i* be in local best response

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

Then $\{u_1^*, u_2^*, \ldots, u_N^*\}$ are in global Interactive Nash iff the graph is strongly connected.

# 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

Kyriakos Vamvoudakis

# Multi-agent Learning Convergence proofs

## 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)

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

*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

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

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 $R_{jj}^{-1} R_{ij}$.

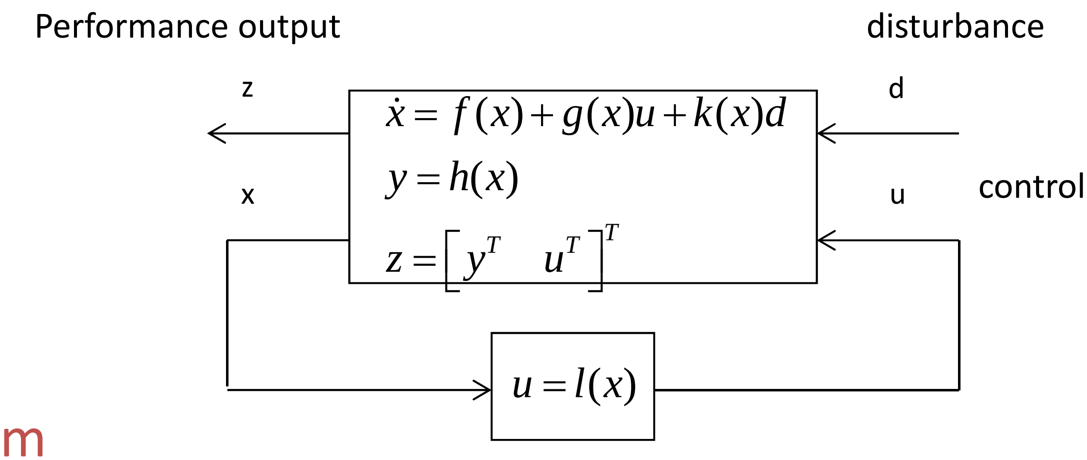
**Theorem 4. Convergence of Policy Iteration algorithm when all agents update their policies.** Assume all nodes $i$ update their policies at each iteration of PI. Then for small enough edge weights $e_{ij}$ and $\rho_{ij}$, $\mu_i$ converges to the global Nash equilibrium and for all $i$, and the values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

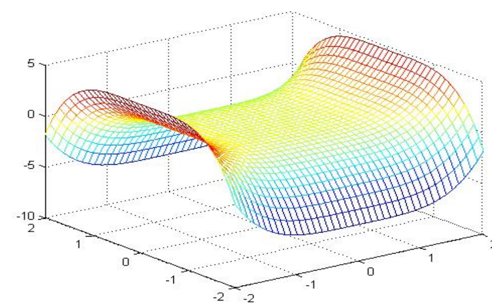
Find control *u(t)* so that

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

For all $L_2$ disturbances
And a prescribed gain $\gamma^2$

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

Leibniz gives
Differential equivalent

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

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

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

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

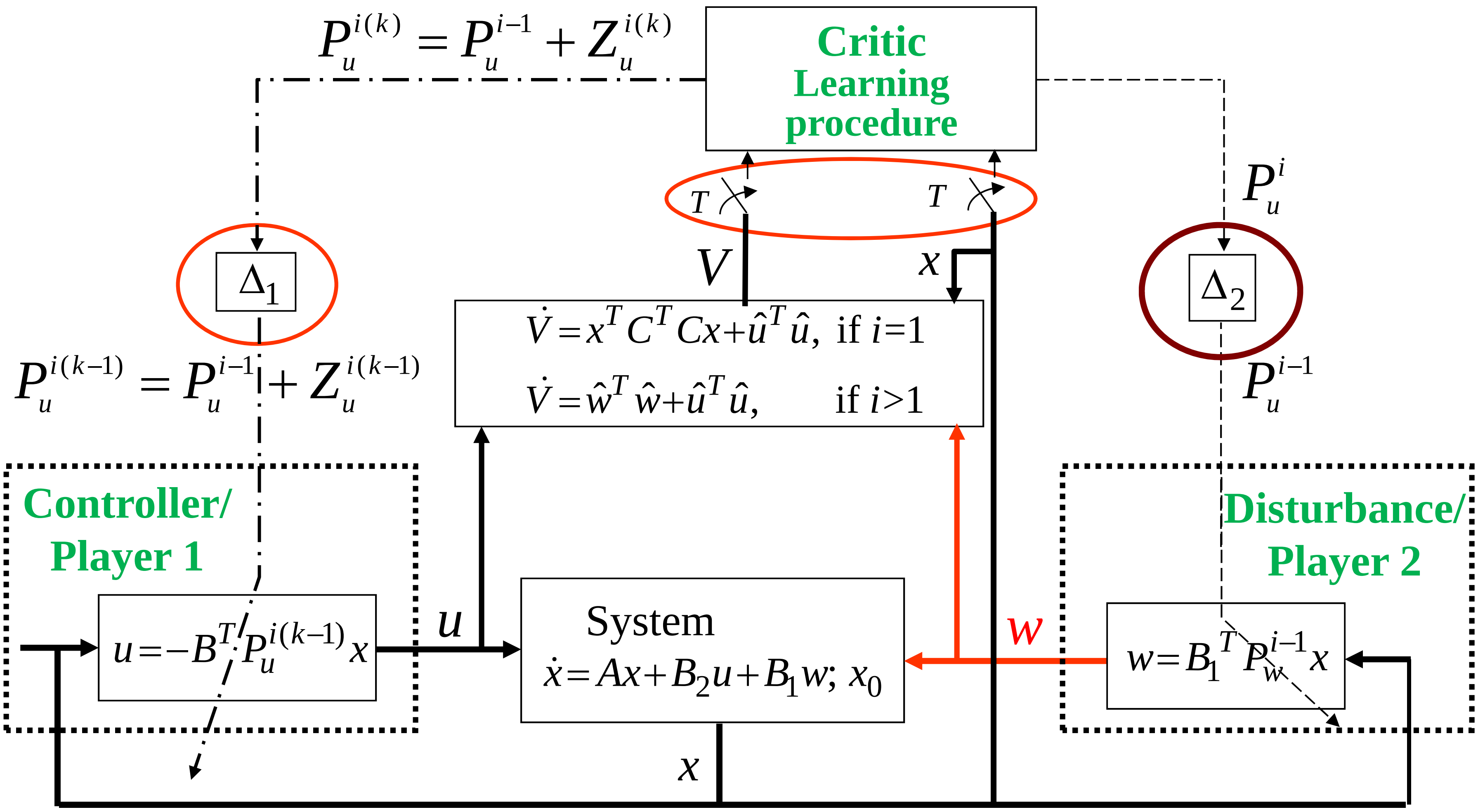
HJI equation $0 = H(x, \nabla V, u^*, d^*)$

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

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.

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

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

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

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

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

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

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

Standard Critic Weight Tuning

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

# 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

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

i/o Data Measurements

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

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

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

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

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

NN weight tuning uses past samples

Previous data

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

# New Principles

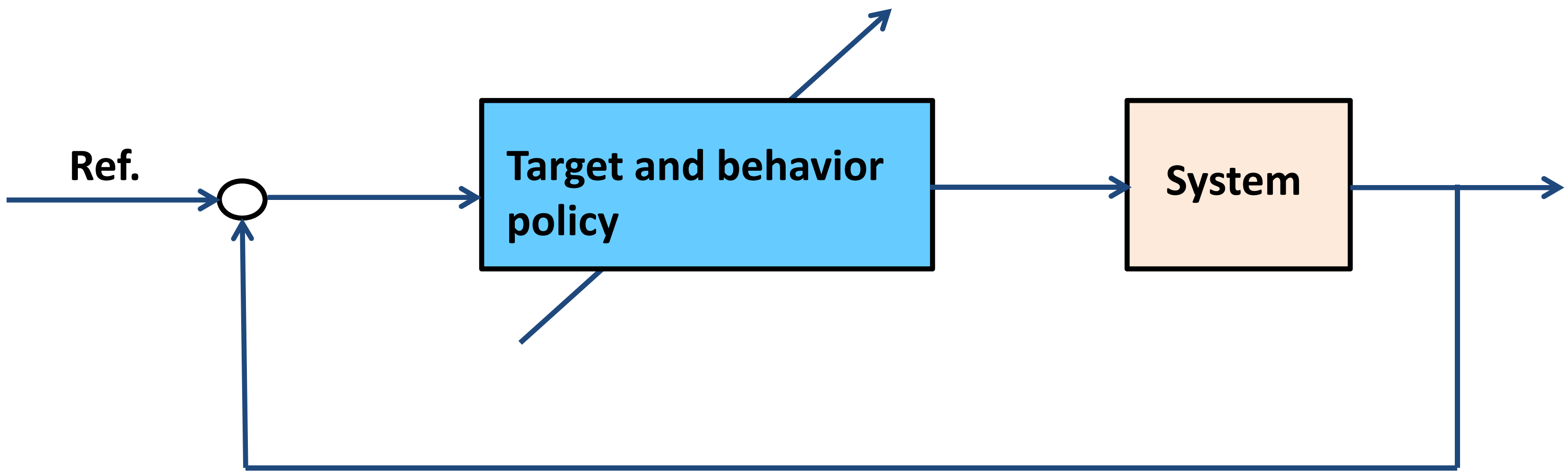
## Off-Policy Learning

## 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

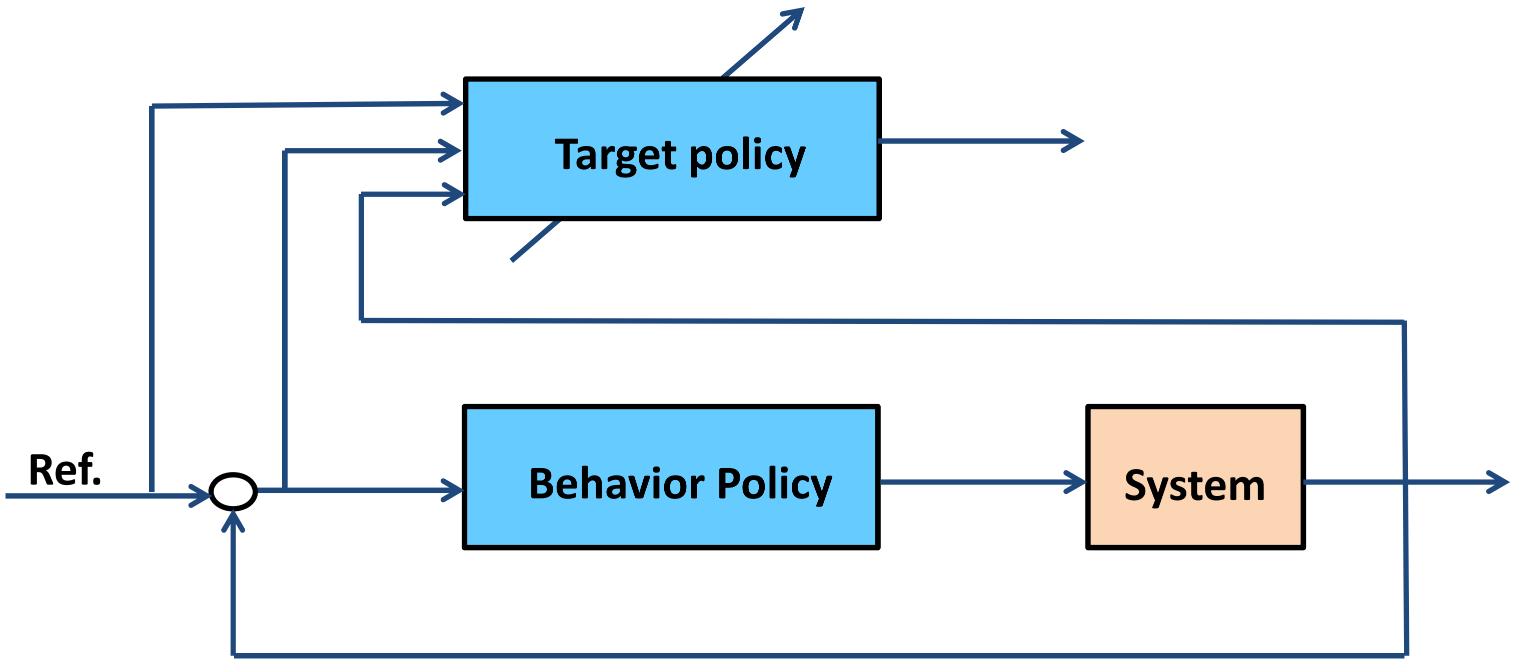



Target policy and behavior policy are the same

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

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

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

**On-policy** →

---

Algorithm 1:

---

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

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

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

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

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

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

which explicitly is

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

---

**Off-policy**

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

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

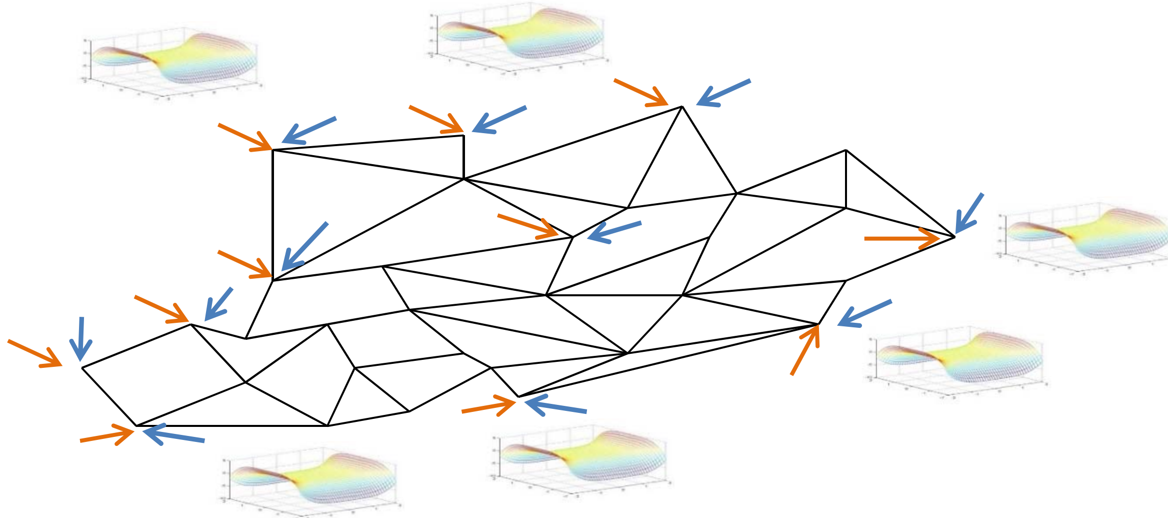
DDO

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 Learning for Estimating Malicious Adversaries' Hidden True Intent

MA Systems $\dot{x}_i = Ax_i + B_i u_i + D_i v_i$

Two Opposing Teams



MAS H-infinity control

Cost $V_i(x_i(t)) = \frac{1}{2}\int_t^\infty (x_i^T Q_{ii} x_i + u_i^T R_{ii} u_i + \sum_{j \in \mathcal{N}_i} u_j^T R_{ij} u_j - \gamma^2 v_i^T T_{ii} v_i - \gamma^2 \sum_{j \in \mathcal{N}_i} v_j^T T_{ij} v_j)\, dt \equiv \frac{1}{2}\int_t^\infty r_i(x_i, u_i, v_i)\, dt$

Off-policy IRL $\dot{x}_i = Ax_i + B_i u_i^k + D_i v_i^k + B(u_i - u_i^k) + D(v_i - v_i^k)$

$$u_i^{k+1} = -\frac{1}{2} B_i^T \frac{\partial V_i^k}{\partial x_i}, \quad v_i^{k+1} = \frac{1}{2\gamma^2} D_i^T \frac{\partial V_i^k}{\partial x_i}$$

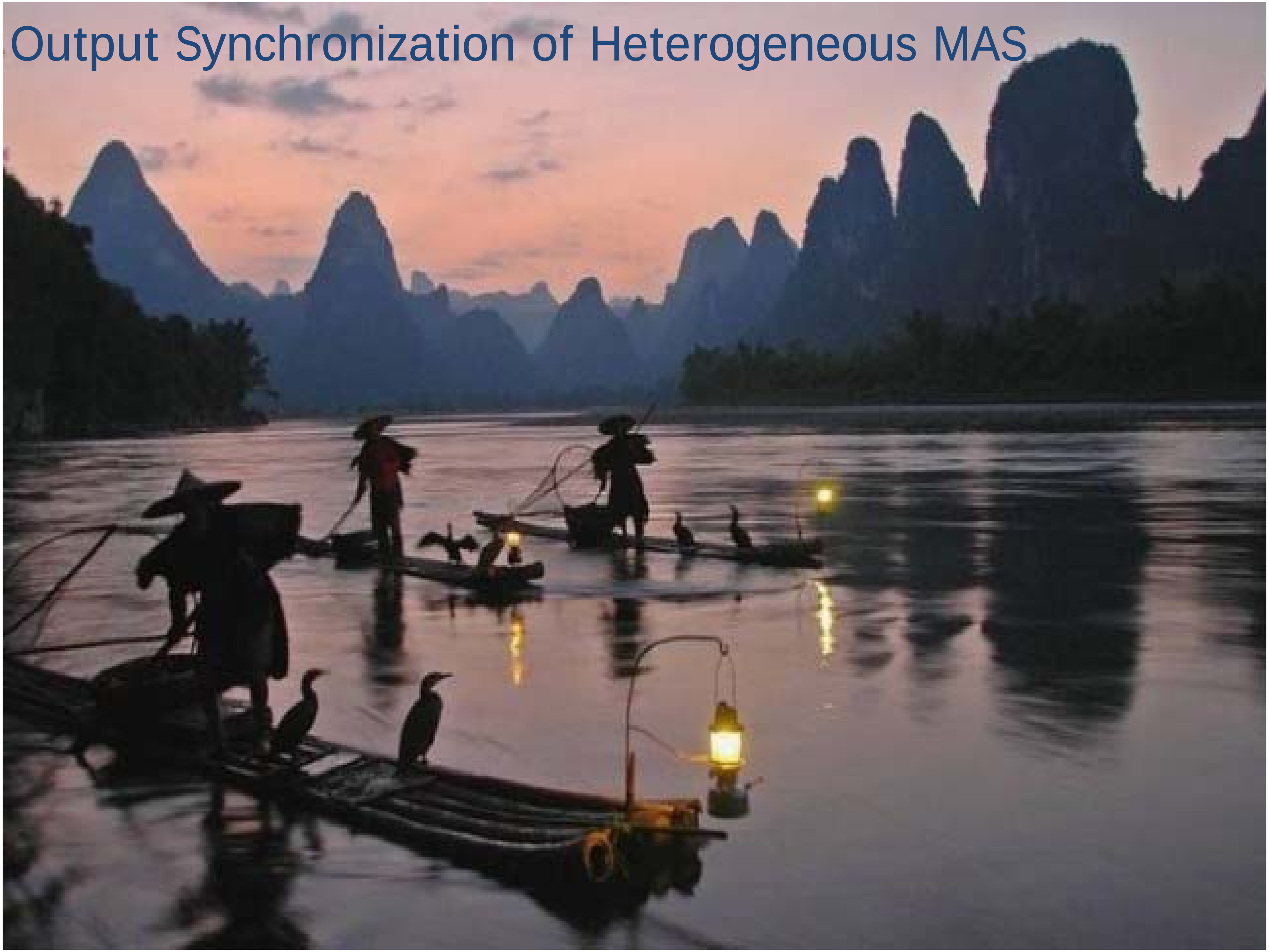
Optimal Target policies

Actual Behavior policies

**Off-Policy Bellman Eq.**

$$V_i^k(x_i(t)) - V_i^k(x_i(t+T)) = \frac{1}{2}\int_t^{t+T} r_i(x_i, u_i^k, v_i^k)\, dt - \int_t^{t+T} (u_i^{k+1})^T R_{ii}(u_i - u_i^k) - \gamma^2 (v_i^{k+1})^T R_{ii}(v_i - v_i^k)\, dt$$

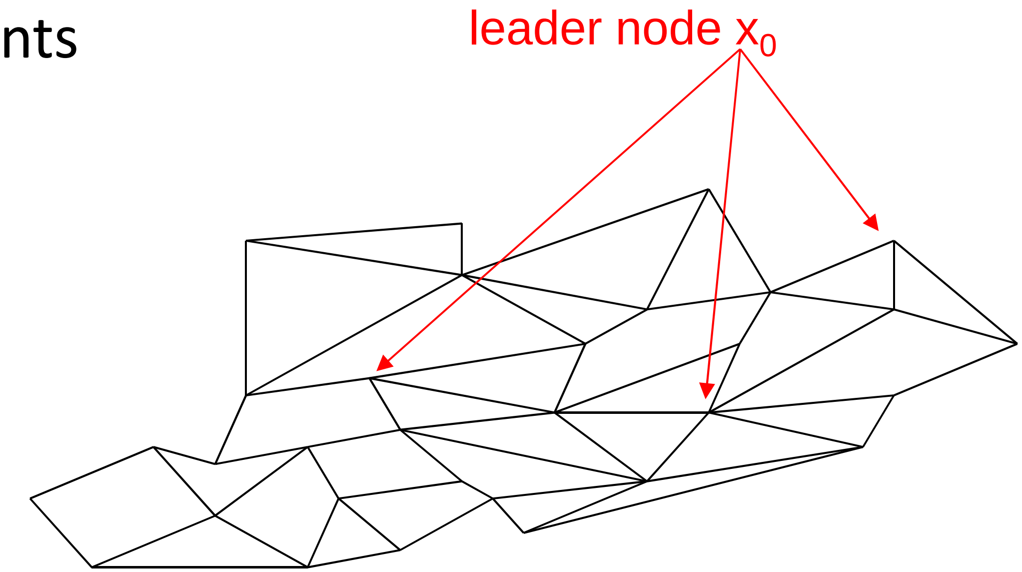
# Output Synchronization of Heterogeneous MAS



# Output Synchronization of Heterogeneous MAS

## Heterogeneous Multi-Agents

$$\dot{x}_i = A_i x_i + B_i u_i$$
$$y_i = C_i x_i$$




## Leader

$$\dot{\zeta}_0 = S\zeta_0$$
$$y_0 = R\zeta_0$$

## Output regulation error

$$\eta_i(t) = y_i(t) - y_0(t) \to 0$$

Output regulator equations

$$A_i\Pi_i + B_i\Gamma_i = \Pi_i S$$
$$C_i\Pi_i = R$$

Dynamics are different, state dimensions can be different
o/p reg eqs capture the common core of all the agents dynamics
And define a synchronization manifold

# Optimal Output Synchronization of Heterogeneous MAS Using Off-policy IRL

**Nageshrao, Modares, Lopes, Babuska, Lewis**

MAS $\dot{x}_i = A_i x_i + B_i u_i$

$y_i = C_i x_i$

Leader $\dot{\zeta}_0 = S\zeta_0$

$y_0 = R\zeta_0$

Our Solution

## Optimal Tracker Problem

Augmented Systems

$$X(t) = \left[ x_i(t)^T \;\; \zeta_0{}^T \right]^T \in \mathbb{R}^{n_i+p}$$

$$\dot{X}_i = T_i X_i + B_{1i} u_i$$

$$T_i = \begin{bmatrix} A_i & 0 \\ 0 & S \end{bmatrix}, B_{1i} = \begin{bmatrix} B_i \\ 0 \end{bmatrix}$$

Performance index

$$V(X_i(t)) = \int_t^\infty e^{-\gamma_i(\tau-t)} X_i{}^T (C_{1i}{}^T Q_i C_{1i} + K_i{}^T W_i K_i) X_i \, d\tau$$
$$= X_i(t)^T P_i X_i(t)$$

Control

$$u_i = K_{1i} x_i + K_{2i}\zeta_0 = K_i X_i$$

# Off-Policy RL

Tracker dynamics

$$\dot{X}_i = T_i X_i + B_{1i} u_i$$

Rewrite as

$$\dot{X}_i = (T_i + B_{1i} K_i^{\kappa}) X_i + B_{1i}(u_i - K_i^{\kappa} X_i) \equiv \bar{T}_i X_i + B_{1i}(u_i - K_i^{\kappa} X_i)$$

Now the Bellman equation becomes

$$e^{-\gamma_i \delta t} X_i(t+\delta t)^T P_i^{\kappa} X_i(t+\delta t) - X_i(t)^T P_i^{\kappa} X_i(t) = -\int_t^{t+\delta t} e^{-\gamma_i(\tau - t)} (y_i - y_0)^T Q_i (y_i - y_0)\, d\tau$$
$$+ \quad 2\int_t^{t+\delta t} e^{-\gamma_i(\tau - t)} (u_i - K_i^{\kappa} X_i)^T W_i K_i^{\kappa+1} X_i \, d\tau$$

Extra term containing $K_i^{\kappa+1}$

**Algorithm 2.** *Off-policy IRL Data-based algorithm*

Iterate on this equation and solve for $P_i^{\kappa}, K_i^{\kappa+1}$ simultaneously at each step

Note about probing noise If $u_i = K_i^{\kappa} X_i + e$ then $(u_i - K_i^{\kappa} X_i) = e$

Do not have to know any dynamics

agent
$$\dot{x}_i = A_i x_i + B_i u_i$$
$$y_i = C_i x_i$$

Or leader
$$\dot{\zeta}_0 = S\zeta_0$$
$$y_0 = R\zeta_0$$

Theorem- Off-policy Algorithm 2 converges to the solution to the ARE

$$T_i^T P_i + T_i P_i - \gamma_i P_i + C_{1i}{}^T Q_i C_{1i} - P_i B_{1i} W_i^{-1} B_{1i}{}^T P_i = 0$$

Theorem- o/p reg eq solution

Let $P_i = \begin{bmatrix} P^i{}_{11} & P^i{}_{12} \\ P^i{}_{21} & P^i{}_{22} \end{bmatrix}$

Then the solution to the output regulator equations

$$A_i \Pi_i + B_i \Gamma_i = \Pi_i S$$
$$C_i \Pi_i = R$$

Is given by

$$\Pi_i = -(P_{11}^i)^{-1} P_{12}^i$$
$$\Gamma_i = K_{2i} - K_{1i} (P_{11}^i)^{-1} P_{12}^i$$

Do not have to know the
Agent dynamics or the leader's dynamics (S,R)

## New Principles

There Appear to be Multiple Reinforcement Learning Loops in the Brain

**Multiple Actor-Critic Learning Structures**

**Narendra MMAC - Multiple Model Adaptive Control**

## 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)

# Intelligent Operational Control for Complex Industrial Processes

Jinliang Ding

**Professor Chai Tianyou**

**State Key Laboratory of Synthetical Automation for Process Industries**

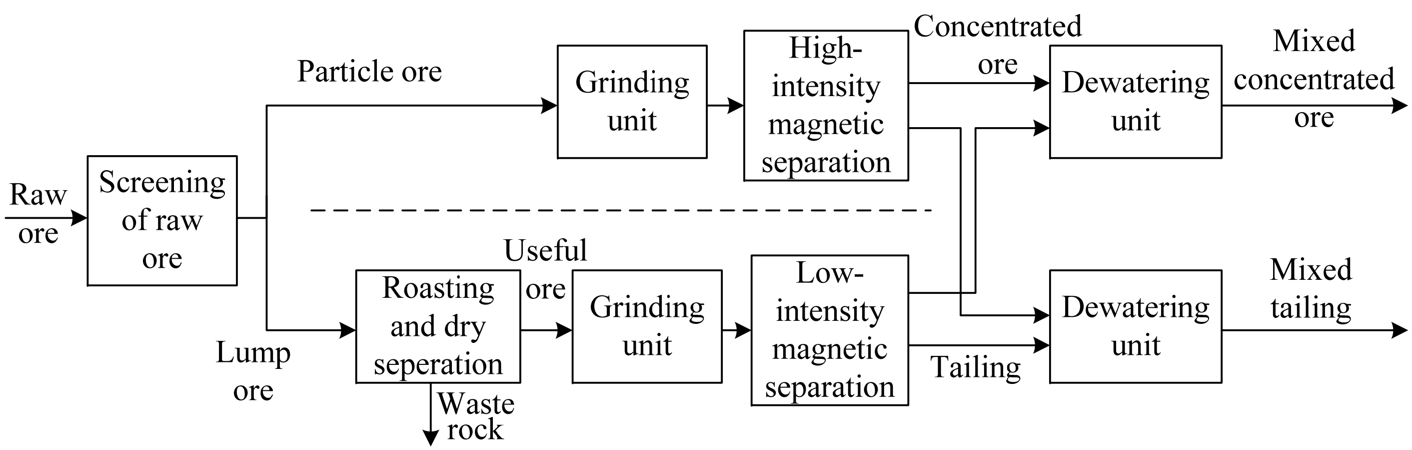
**Northeastern University**

**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.

# Production line for mineral processing plant


the high-intensity magnetic production line (HMPL)




the low-intensity magnetic production line (LMPL)

## Mineral Processing Plant in Gansu China

# 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.