# 332:515 Reinforcement Learning for Engineers – Fall 2023

(Open to all graduate and advanced senior undergraduate students in engineering and sciences with solid mathematical undergraduate background)

**Instructor:** Zoran Gajic, ECE 222. Email: zgajic@soe.rutgers.edu

**Office Hours:** Tu 3:00 – 5:00 pm and F 3:00 – 5:00 pm, or in person in the ECE 222 office or via WebEx https://rutgers.webex.com/meet/zgajic. Additional office hours will be given before exams and project due dates.

**Textbook**: No textbook is available for his class. The course will follow the basic RL terminology snd methodology from the textbook by Sutton and Barto, *Reinforcement Learning: An Introduction,* and the typed class notes by Professor Gajic based on his 2021 course that evolved around the dynamic programming methodology of Bellman.

**Course Coverage**: This course is on the engineering approach to Reinforcement Learning (RL) (important area of machine learning) control systems perspective, which is based on iterative nonlinear, adaptive, and optimal feedback control of dynamic systems. The central theme will evolve around the approximate dynamic programming technique. As an introduction to the course, a brief coverage of the essence of some chapters from the Sutton and Barto's textbook (the main computer science textbook on reinforcement learning) will be presented with the purpose to learn RL terminology and essential procedures used in RL. That book presents the computer science approach to reinforcement learning (mostly tabular approach with Monte Carlo and Markovian statistics incorporated (Markov Decision Processes, MDP), and the use of neural networks for generating learning policies). In this class, the controls and systems approach will be used to generate learning policies (optimal feedback strategies, optimal feedback controls).

**Specific topics to be covered**:

- Dynamic programming in continuous- and discrete-time.
- Summary of the first six chapters from Sutton and Barto (2020) book.
- Approximate dynamic programming in continuous-and discrete-time (deterministic policy iterations and value iterations).
- Markov decision processes and RL.
- General approximate stochastic programming and reinforcement learning (if time permits).
- Introduction to the RL MATLAB Tool Box (last week of the semester if time permits).
  No MATLAB or any specific software like Phyton is required for the class, unless some students want to use it for their term papers or projects.

**Background:** Solid mathematical background is expected, such as optimization, differential and difference equations, linear algebra, some knowledge of dynamic systems and controls (state space approach), and basics of probability and Markov processes. The course will be introductory in nature, and advanced senior undergraduate students in engineering with strong mathematical background will be able to master successfully the course material.

**Grading** will be based on theoretical exam questions, term papers or projects. Project = 30%, two Midterm Exams 20% each (one exam on knowledge of theory, and the other one on derivations (presentations) of important results (concepts) – list of potential questions for each exam will be available to the students, Final term paper = 30%.

**Grading Scale:** A >= 90%, B+ >= 82%, B >= 75%, C+ >= 67%, C >= 60.

**Relevant Books and Magazine Article Overview Papers:**

R. Bellman, *Dynamic Programming*, Princeton University Press, 1957. (both deterministic and stochastic multi-stage decision processes, including MDP). Republished by Dover Publications, 2003.

F. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE Circuits and Systems Magazine*, 32-50, Third Quarter, 2009. (deterministic only, good starting point).

F. Lewis, D. Vrabie, and K. Vamvoudakis, "Reinforcement learning and feedback control," *IEEE Control Systems Magazine*, 76-105, Dec. 2012. (stochastic (MDP) and deterministic)

R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 2020. (stochastic MDP; NN use for decision making with applications to games, psychology, neuroscience)

D. Bertsekas, *Reinforcement Learning and Optimal Control*, Athena Scientific, 2019. (high level, mostly stochastic).

**Some Selected Journal Papers** will be assigned for reading as term papers and/or presented in class.

**Additional books on Reinforcements Learning:**

D. Bertsekas and J. Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, Nahua, NH, 1996. (fundamental, high level, all stochastic)

W. Powell, *Reinforcement Learning and Stochastic Optimization: A Unified Framework for Sequential Decisions*, Wiley Interscience, 2020. (all stochastic, available on line)

L. Graesser and W. L. Keng, *Foundations of Deep Reinforcement Learning*, Addison Wesley (Pearson Educations, 2020 (computer science, MDP based)

M. Sewak, *Deep Reinforcement Learning*, Springer Nature, Singapore, 2019 (easy to read, in Rutgers Library).

## Comments on the Spring 2021 ECE Reinforcement Learning Course

***Comment 1:*** The class material on engineering approach to reinforcement learning (RL) was presented in 26 eighty-minute lectures with the corresponding videos. For each lecture class notes (word or pdf formats) were provided. The lecture videos were based on the class notes. The students were responsible only for the material covered in class (the course load equivalent to a 3-credit course), even though a lot of additional material was posted on Sakai as future references for students who were interested to learn beyond an introductory engineering course on RL, and who will either potentially use some RL results in future or directly be involved in RL research. Very comprehensive website on RL, at that time was: https://wiki.pathmind.com/deep-reinforcement-learning.

TOPICS COVERED IN 2021

Lecture 1: Course Information and Introduction to Reinforcement Learning

Lecture 2: Dynamic Programming (DP) in Continuous Time

Lecture 3: Kleinman's LQ Case Algorithm, Affine Nonlinear Systems, and Successive Approximations

Lecture 4: Basic System Stability, Controllability, and Observability Concepts

Lecture 5: Dynamic Programming in Discrete Time. Project 1 Assigned.

Lecture 6: Discrete Linear-Quadratic (LQ) Case and the hewer Algorithm

Lecture 7: Discrete-Time Hamiltonians and the LQ Case. MATLAB RL Toolbox

Lecture 8: Sutton and Barto's (S&B's) Chapters 1, 2, and 3 Summary of RL Terminology

Lecture 9: S&B's Chapter 3 Summary Continued. Markov Decision Processes Introduction.

Lecture 10: S&B's Chapter 4 Summary. Paper by Sutton, Barto, and Watkins Stochastic DP and MDP.

Lecture 11: S&B's Chapter 5 Summary. Introduction to Systems Identification and Adaptive Control.

Lecture 12: S&B's Chapter 6 Summary

Lecture 13: S&B's Chapter 7 Summary. Separation Principle and Kalman Filter.

Lecture 14: Successive Approximations in Continuous Time

Lecture 15: Successive Approximations in Discrete Time

Lecture1 6: Discrete-Time Policy Iterations and Value Iterations

Lecture 17: Course Review: Lectures 1-16

Lecture 18: Dynamic Programming for Differential Games

Lecture 19: Midterm Exam

Lecture 20: Dynamic Programming for Nash Differential Games

Lecture 21: RL via Policy Iterations for Zero-Sum Games and Anderson-Feng-Chen Algorithm

Lecture 22: Reinforcement Learning for Nash Differential Games via the Li-Gajic Algorithm

Lecture 23: Reinforcement learning for Partially Model Free Dynamic Systems

Lecture 24: Vrabie-Lewis-et-al Automatica Paper for RL Model Free Dynamic Systems

Lecture 25: Reinforcement Learning for Completely Model Free Dynamic Systems

Lecture 26: Model Free reinforcement learning for Zero-Sum Games

Lecture 27: Overview of Journal Papers to be Used for the Final term Paper

Lecture 28: Introduction to Reinforcement Learning MATLAB Toolbox

***Comment 2*:** (for 2023 students) Even though in this course we will see a lot of mathematical formulas and algorithms, we will make an attempt to reduce mathematics to a bare minimum by being less rigorous, but still provide complete presentation and clear picture of the main ideas and algorithms of engineering approach to RL. Selected examples in (optimal) control of dynamic systems will be used to demonstrate the RL mechanism. Neural Networks (NN) will not be seen in this course since the decision making will be done via the developed approximate (iterative, adaptive) optimal control system algorithms that converge in time (continuous or discrete) to the best (optimal) solutions. The tabular approach to RL (often used in the computer science filed) will be avoided. Instead the "learning environment" will be defined by differential and difference equations whose solutions will define evolution of learning dynamics.

***Comment 3:*** (for 2023 students) There is no textbook on the material covered in this course. Up to the instructor's best knowledge there is no any graduate/undergraduate class that covers topics presented and the methodology used in this class. During preparation for this class, the instructor read books (Sutton and Barto 2020, Bertsekas and Tsitsiklis 1996, Bertsekas, 2019; Graesser and Keng 2020, Sewak 2019, Powell 2020), glance the MATLAB Manuals for its Reinforcement Learning Toolbox, (MATLAB R2020a), and visited several website where RL teaching and research have be done: Texas Tech University, MIT, Carnegie Mellon, NYU, and Princeton University. The textbook by Sutton and Barto (2020) is considered as the standard textbook in a computer science course on RL. It can be uploaded from Professor Barto's website http://www.incompleteideas.net/book/the-book-2nd.html. In the first several weeks of this class, we will cover the essential terminology and techniques of RL from the first several chapters of the book by Sutton and Barto (2020), supplemented by some sections from the books (Graesser and Keng 2020, Sewak, 2019). Book by Sewak (2019) electronics' version can be found in the Rutgers University Library. Powell's book can be also downloaded from his website. In addition, in the past couple of years, the instructor read numerous journal papers on RL control systems oriented. Some of these journal papers will be considered in the second part of the course. Finally, many years ago the instructor did research on iterative/approximate optimal control technique, the work referenced in RL papers during the past ten years. If time permits some of these results will be presented in this class, especially Li-Gajic algorithm (1995) used for optimal control of linear-quadratic Nash games, which is a generalization of the well known Kleinman algorithm (1968) used often in RL as a policy iteration technique.

***Comment 4:*** (for 2023 students) No programming in MATLAB or Python will be required, unless a particular student is interested to do a term paper or a project using MATLAB or Python (Chan, 2017). No lecture time is planned to be dedicated to MATLAB and/or Python, maybe just the last week if the time permits. MATLAB introduction to RL slides are posted on Sakai. Occasionally ,we will refer to those slides to demonstrate/clarify engineering approach to RL from the control systems point of view. Students familiar with MATLAB can easily learn Python. More ever, Python is free to use and accessible to anyone https://python.org/downloads. Even though, many people consider Python as a language of artificial intelligence, these days it is used by everyone for many applications. It is interesting to point out that the book on undergraduate linear feedback control systems, (Beard *et al*. 2016), has controller design techniques implemented in both Python and MATLAB/Simulink.

***Comment #5***: (for 2023) I have been contacted by both undergraduate and graduate students from electrical and computer engineering, mechanical and aerospace engineering, industrial engineering, computer science, and business school (operational research) referring to this courses. Since in this class we have students with different background, it will be very helpful for planning of the material to be covered and types of term papers and projects if each of the students registered for the class fills in the attached table about your academic background and either email it to me or just upload to their class drop boxes at your earlier convenience.

PLEASE RETURN THE NEXT PAGE TO PROFESSOR GAJIC ON FRIDAY DURING HIS CLASS

| | Yes | No |
|---|---|---|
| Are you a graduate student | | |
| Are you an engineering student | | |
| Are you a computer science student | | |
| Are you the business school (operational research, finance) student | | |
| Have you had a course in control systems | | |
| Have you taken a course on dynamic programming (or optimal control) | | |
| Are you familiar with the state space approach for system analysis | | |
| Have you had a course in probability | | |
| Have you had a course in stochastic processes | | |
| Have you taken any course on Reinforcement Learning | | |
| Have you taken applied mathematics courses 642:527 or 642:528 or equivalent | | |
| Have you taken a numerical analysis course 642:573 or 642:574 or equivalent | | |
| Have you taken linear algebra courses 642:250 or 642:550 equivalent | | |
| Are you familiar with MATLAB | | |
| Are you familiar with Python | | |
| Do you plan to use RL in your research | | |
| Are you registered for this class or only plan to attend the lectures. | | |

As an introduction to RL, citations from several standard books on RL (Bertsekas and Tsitsklis 1996, Sewak 2019, Sutton and Barto 2020, Graesser and Keng 2020, Powel 2020) are provided. At some places the instructor comments are inserted. The cited text is set in "Calibri" font, and the rest of the lectures are typed using "Times New Roman" font and put in the bracket [ … ].

**RL Relations to Artificial Intelligence (AI)**

"Of the many definitions given by different researchers and authors of Artificial Intelligence, the criteria for calling an agent an AI agent is that it should possess ability to demonstrate "*through-process and reasoning*", "*intelligent-behavior*", "*success* in terms of human performance", and "*rationality.*", page 1, [7].

"Among the different Artificial Intelligence agents, Reinforcement Learning agents are considered to be among the most advanced and very capable of demonstrating high level of intelligence and rational behavior. A reinforcement agent interacts with its environment. The environment itself could demonstrate multiple states. The agent acts upon the environment to change the environment's state, thereby also receiving a reward or penalty as demonstrated by the achieved state and the objective of the agent.", page 1, [7].

"It [RL] is a subfield of artificial intelligence that dates back to the optimal control theory and Markov Decision processes (MDP). It was first worked on by Richard Bellman in the 1950s in the context of dynamic programming and quasilinear equations [15] [reference [2] of this document]. We will see this name again when we study a famous equation in reinforcement learning-the Bellman equation.", page 2, [7].

**General Statement Regarding Reinforcement Learning**

"Reinforcement learning is an area of machine learning that is useful for solving sequential decision-making problems–that is, problems that are solved over the time.", Foreword page, [6]. *Comment*: Iterative decisions (controls) at every discrete-time step, or more general iterative decisions (controls) in continuous-time.

"In particular, we will look at how an agent interacts with an environment to optimize an objective. We will then define these more formally and define reinforcement learning as a Markov Decision Process. This is the theoretical foundation of reinforcement learning.", page 1, [6]. *Comment:* Reference [6] is based on stochastic approach to RL.

"Reinforcement learning (RL) is concerned with solving sequential decision making problems.", page 1, [6].

"Essentially, a reinforcement learning system is a feedback control loop where an agent and an environment interact and exchange signals, while the agent tries to maximize the objective.", page 3, [6].

"Although the design blue print as given in Fig. 1.1 looks simple, but from the discussion of this section, it must be clear that the problem of converting a real-life problem to a reinforcement learning one itself is very challenging.", page 7, [7].

"… the data would be gathered to continuously train the agent and prepare it for any production deployment." page 8, [7]

"The agent is by far the most important piece of the Reinforcement Learning as it contains the intelligence to take decisions and recommend the optimal action in any given situation.", page 14, [7].

" … [agent] focuses on identifying which is the next best state to be in (reachable from the current state) as determined from the history of present and future rewards that the agent has received when it was in this particular state earlier.", page 15, [7]

"… we are [the agent] are trying to predict the "value" (or utility) of any state (or state-action combination), even the unseen ones, based on the ones that we have seen.", page 15, [7].

"Reinforcement Learning has gradually become one of the most active research areas in machine learning, artificial intelligence, and neural network research.", Preface [to the first edition 1999], page xvii, [5].

"The overall problem of learning from interaction to achieve goals is still far from being solved, but our understanding of it has improved significantly.", Preface, page xviii, [5].

"Reinforcement learning is learning what to do—how to map situations to actions—so as to maximize a numerical award signals.", page 1, [5],

"We formalize the problem of reinforcement learning using ideas from dynamic systems theory, specifically, as the optimal control of incompletely-known Markov decision processes.", page 2, [5].

## Introduction to Reinforcement Learning for Engineers

### 1. Basic of Reinforcement Learning, Definitions, and Relations to Optimal Control of Systems

In this introductory lecture, we will present some basic historical facts and provide some fundamental information regarding the reinforcement learning (RL) method, the machine learning technique that is related to control of dynamic systems, and that has recently considered as one of the most promising tools to solve many challenging problems in artificial intelligence. In general, RL is a very broad multidisciplinary area involving researchers and practitioners from diverse fields such as phycology, neuroscience, computer science, mathematics, statistics, engineering, operational research, economy, finance, and other scientific and social fields. Its history is pretty long with the first fundamental results dating back hundred years ago. The field of reinforcement learning is very reach due to its long history and a larger number of researchers from different fields that participated in. Due to its complexity and diversity, it has been slowly evolving during the past fifties or sixty years owning to the fundamental work of Richard Bellman on dynamic programming (Bellman 1957) and the pioneering work on artificial intelligence by Marvin Minsky in the 1950s and 1960s (Minsky 1954, 1961), and gain its maturity in the past twenty or thirty years. Moreover, RL has become a very active research area in the past ten or so years especially within computers science and (control) systems engineering.

In these lectures, we will present RL from the perspective and prospective of control system engineering, in general real physical dynamic systems. For every previously mentioned scientific field, corresponding presentations can be done using their own perspectives. Our intention is to be general so that most of the material presented in these lectures be useful for the other scientific fields that either use RL in applications or develop new theoretical results within it. *The engineering approach to RL will be based on approximate dynamic programming*. It has been known since the 1950s that dynamic programming solves optimization (optimal control) problems of continuous- and discrete-time, deterministic and stochastic, dynamic systems.

RL in general stands for learning from feedback received through interactions with environment. It can be model based (when the mathematical model of environment exists) and non-model based when no such models are available. The approach taken in this course will be either *model based* or *partially model based* RL, which assumes that either complete or at least partial knowledge of the system dynamic mathematical model described by differential and/or difference equations. RL for stochastic dynamic systems will be mostly restricted to Markov decision processes (MDP) either in continuous- or discrete-time domains. The MDP appear to represent the main mathematical and computational tool for RL within the computer

science, operational research, and finance (business) areas. *The central theme in all presented methods will be discovering built-in mechanics that iteratively improve control strategies through RL schemes (known also as reinforcers) such that they converge to the optimal solutions (optimal control, optimal system state trajectories, optimal system performance).*

RL achievements and results have recently been very remarkable having in mind that MATLAB introduced in 2019 a toolbox on Reinforcement Learning (its two manuals have more than 700 pages). In 2016 Deep Mind's program Alpha Go used RL to defeat the human World Champion of the game of Go. Its sister program Alpha Zero became the World Chess Champion among computers in 2019 (computer chess programs defeated the Human World Chess Champion long time ago, in 1996, using the brute force computational power of computers – humans in average can precisely calculate in average 4-5 chess moves ahead, but computers even in 1996 were able to do so in average 10-15 chess moves ahead). Toyota Research Center in Polo Alto is presently doing research based on reinforcement learning to make crashless cars irrespective on decisions made by the driver and situations on the road, many computer science algorithms use RL, many artificial intelligence problems were solved using RL, and so on, we have witnessed a large number of practical achievements of reinforcement learning.

**Brief History of Reinforcement Learning**

RL originated in Phycology within experimental and theoretical studies of a Russian Phycologist Ivan Pavlov on animal learning during the 1920s that culminated in his famous book titled *Conditioned Reflexes* (Pavlov 1927). It is interesting to observe that around the same time a mathematical discipline known as Calculus of Variations made first attempts to solve what is today known as the optimal control problem. After thirty or so years, researchers in calculus of variations produced the fundamental results for the solution of a general optimal control problem: Optimize an integral functional (scalar function of vector variables, called also the performance criterion) along trajectories of a nonlinear dynamic system in general defined by

$$\frac{dx(t)}{dt} = f(x(t), u(t), t), \qquad x(t_0) = x_0 \tag{1}$$

$f(x(t), u(t), t) \in R^n$ is a vector function, $x(t) \in R^n$ is the system state space vector, $u(t) \in R^m$ is the system control input vector, $x(t_0) = x_0$ is the system initial state, and $t$ stands for continuous time with $t_0$ representing the initial time. A scalar performance criterion, to be optimized (minimized or maximized), associated with this dynamic system, is given by

$$J\left(x(t_0), t_0\right) = \int_{t_0}^{t_f} g(x(\tau), u(\tau), \tau) d\tau + \gamma(x(t_f)) \tag{2}$$

where $g(x(t), u(t), t)$ is a nonnegative scalar function, representing the value of the performance criterion (cost) at the continuous time instant $t$ (instant cost), $t_f$ is the final time instant assumed to be fixed, and $\gamma(x(t_f))$ is the final time constraint imposed on the state variables. As a matter of fact, the first solution to the dynamic optimization problem defined in (1)-(2) was derived by Bellman in the first part of the 1950s using the technique developed by himself that he called dynamic programming. The results of Bellman were published his famous book titled *Dynamic Programming* (Bellman 1957), the technique that solved also the general optimal control problem in both deterministic and stochastic frameworks and in both continuous- and discrete-time domains.

These days RL researchers consider *RL as an approximate optimal control technique*. The title of the paper (Sutton, Barto, and Williams 1992) is "Reinforcement Learning is Direct Adaptive Optimal Control."

In addition to *approximate dynamic programming*, RL is also known under several different names: *adaptive dynamic programming* or *heuristic programming* or *neuro-dynamic programming* (Bertsekas and Tsitsiklis 1996) or *iterative dynamic programming* or *incremental dynamic programming* or *forward dynamic programming* (Powell 2020). Within automatic control systems, research on reinforcement learning started at the beginning of the 1960s, see for example (Waltz and Fu 1965). Automatic control system approach to RL culminated in the past decade in the work of Frank Lewis (originally at Georgia Tech) and his coworkers at Texas Tech University (for example, Lewis and Vrabie 2009, Lewis *et al*. 2012, Lewis and Liu 2013, Vrabie *et al*. 2009, Vrabie *et al*. 2013, Vamvoudakis and Lewis 2011, Vamvoudakis 2015, Vamvoudakis *et al*. 2017, Kiumarsi *et al*. 2018). Several journal papers from that research group will be presented in this course.

In the middle of the 1950s Bellman published his famous book titled *Dynamic Programming* (Bellman 1957) on the technique that he solely developed, the technique that solved also the general optimal control problem in both deterministic and stochastic frameworks.

RL is often called an *approximate dynamic programming technique*. RL is also known under several different names: *adaptive dynamic programming* or *heuristic programming* or *neuro-dynamic programming* (Bertsekas and Tsitsiklis 1996) or *iterative dynamic programming* or *incremental dynamic programming* or *forward dynamic programming* (Powell 2020). Within automatic control systems, research on reinforcement learning started at the beginning of the 1960s, see for example (Waltz and Fu 1965). Automatic control system approach to RL culminated in the past decade in the work of Frank Lewis and his coworkers at Texas Tech University (for example, Lewis and Vrabie 2009, Lewis *et al*. 2012, Lewis and Liu 2013, Vrabie *et al*. 2009, Vrabie *et al*. 2013, Vamvoudakis and Lewis 2011, Vamvoudakis 2015, Vamvoudakis *et al*. 2017, Bahare *et al*. 2018). Several journal papers from that research group will be presented in this course. At the end of 1980s and the beginning of 1990s, the temporal-difference methods based on MDP were developed, which is considered as the beginning of the modern approach to reinforcement learning within the computer science (Watkins 1989; Watkins and Dayan, 1992).

Researchers in computer science consider that the Bellman's book (Bellman, 1957) and its Chapter XI on Markovian Decision Processes (MDP) as the beginning of modern era of computational RL, which might be called (together with the Monte Carlo search methodology and use of Neural Networks (NN)) the foundation of computer science approach to RL (Sutton and Barto, 2020; Graesser and Keng, 2020). Nowadays, classes on artificial intelligence taught in computer science undergraduate and graduate programs very often include several lectures on reinforcement learning. The engineering approach to RL deals with both the deterministic and stochastic RL in both continuous and discrete time domains. The control systems approach to RL is constrained to learning along trajectories of a dynamic system evolving in the *n*-dimensional vector space.

**RL Relations to Artificial Intelligence (AI)**

"Of the many definitions given by different researchers and authors of Artificial Intelligence, the criteria for calling an agent an AI agent is that it should possess ability to demonstrate "*through-process and reasoning*", "*intelligent-behavior*", "*success* in terms of human performance", and "*rationality*.", page 1, [7].

"Among the different Artificial Intelligence agents, Reinforcement Learning agents are considered to be among the most advanced and very capable of demonstrating high level of intelligence and rational behavior. A reinforcement agent interacts with its environment. The environment itself could demonstrate multiple states. The agent acts upon the environment to change the environment's state, thereby also receiving a reward or penalty as demonstrated by the achieved state and the objective of the agent.", page 1, [7].

"It [RL] is a subfield of artificial intelligence that dates back to the optimal control theory and Markov Decision processes (MDP). It was first worked on by Richard Bellman in the 1950s in the context of dynamic programming and quasilinear equations [15] [reference [2] of this document]. We will see this name again when we study a famous equation in reinforcement learning-the Bellman equation.", page 2, [7].

"Of all the forms of machine learning, reinforcement learning is the closest to the kind of learning that humans and other animals do, and many of the core algorithms of reinforcement learning were originally inspired by biological learning systems." page 4, [5].

"One of the most pressing areas for future reinforcement learning research is to adopt and extend methods developed in control engineering with the goal of making it acceptably safe to fully embed reinforcement learning agents into physical environments.", page 478, [5].

**General Statement Regarding Reinforcement Learning**

"Reinforcement learning is an area of machine learning that is useful for solving sequential decision-making problems—that is, problems that are solved over the time.", Foreword page, [6]. *Comment*: Iterative decisions (controls) at every discrete-time step, or more general iterative decisions (controls) in continuous-time.

"In particular, we will look at how an agent interacts with an environment to optimize an objective. We will then define these more formally and define reinforcement learning as a Markov Decision Process. This is the theoretical foundation of reinforcement learning.", page 1, [6]. *Comment:* Reference [6] is based on stochastic approach to RL.

"Reinforcement learning (RL) is concerned with solving sequential decision making problems.", page 1, [6].

"Essentially, a reinforcement learning system is a feedback control loop where an agent and an environment interact and exchange signals, while the agent tries to maximize the objective.", page 3, [6].

"Although the design blue print as given in Fig. 1.1 looks simple, but from the discussion of this section, it must be clear that the problem of converting a real-life problem to a reinforcement learning one itself is very challenging.", page 7, [7].

(Fig. 1.1: Design for a Reinforcement Learning System, page 2 of [6], or from MATLAB manual)

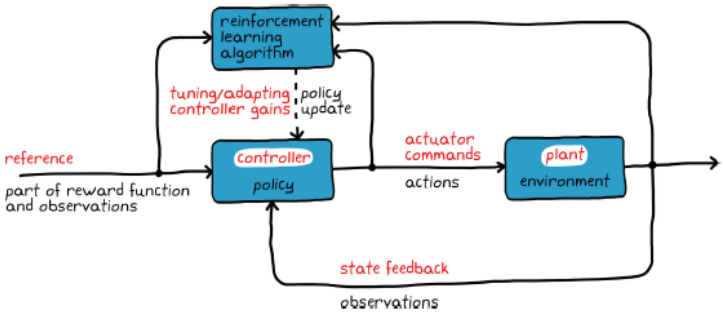
## How Is Reinforcement Learning Similar to Traditional Controls?

The goal of reinforcement learning is similar to the control problem; it's just a different approach and uses different terms to represent the same concepts.

With both methods, you want to determine the correct inputs into a system that will generate the desired system behavior.

You are trying to figure out how to design the policy (or the controller) that maps the observed state of the environment (or the plant) to the best actions (the actuator commands).

The state feedback signal is the observations from the environment, and the reference signal is built into both the reward function and the environment observations.



"… the data would be gathered to continuously train the agent and prepare it for any production deployment." page 8, [7]

"The agent is by far the most important piece of the Reinforcement Learning as it contains the intelligence to take decisions and recommend the optimal action in any given situation.", page 14, [7].

" … [agent] focuses on identifying which is the next best state to be in (reachable from the current state) as determined from the history of present and future rewards that the agent has received when it was in this particular state earlier.", page 15, [7]

"… we are [the agent] are trying to predict the "value" (or utility) of any state (or state-action combination), even the unseen ones, based on the ones that we have seen.", page 15, [7].

"Reinforcement Learning has gradually become one of the most active research areas in machine learning, artificial intelligence, and neural network research.", Preface [to the first edition 1999], page xvii, [5].

"The overall problem of learning from interaction to achieve goals is still far from being solved, but our understanding of it has improved significantly.", Preface, page xviii, [5].

"Reinforcement learning is learning what to do—how to map situations to actions—so as to maximize a numerical award signals.", page 1, [5],

"We formalize the problem of reinforcement learning using ideas from dynamic systems theory, specifically, as the optimal control of incompletely-known Markov decision processes.", page 2, [5].

"Of all the forms of machine learning, reinforcement learning is the closest to the kind of learning that humans and other animals do, and many of the core algorithms of reinforcement learning were originally inspired by biological learning systems." page 4, [5].

"One of the most pressing areas for future reinforcement learning research is to adopt and extend methods developed in control engineering with the goal of making it acceptably safe to fully embed reinforcement learning agents into physical environments.", page 478, [5].

**Relation to Optimal Control Systems**

Reinforcement learning main variables have their analogs in control (optimal) of dynamic systems. They are summarized in the next table.

| Reinforcement Learning | Optimal Control Systems |
|---|---|
| $s(t)$ state of an abstract system | $x(t)$ state of a dynamic system |
| $a(t)$ - agent or actor action | $u(t)$ - control input |
| $r(t)$ - scalar reward | $J(x(t),u(t))$ - scalar performance criterion |
| Environment: in general it is an abstract space. It can be a state of a game, a subset of $R^n$, or anything else that changes in time. | $R^n$ - state space, with ODE system dynamics $\frac{dx(t)}{dt} = f(x(t),u(t), \qquad y(t) = g(x(t),u(t))$ or a dynamic system described by PDEs |

Table 1.1 Reinforcement Learning and Optimal Control

Graphically, using formulas and a flow chart, the reinforcement learning process paralleling iterative feedback control can be symbolically represented as follows.

| |
|---|
| RL: Agent (Policy) $\rightarrow$ Environment $\Big\vert_{\rightarrow \text{reward}}^{\rightarrow \text{state}}$ $\rightarrow$ Agent (Policy) $\rightarrow$ repeat many times$\cdots$ |
| $u_i(x(t)) \rightarrow \dfrac{dx(t)}{dt} = f\left(x(t), u_i(x(t))\right)\Big\vert_{\rightarrow J(x(t),u_i(x(t)))}^{\rightarrow x(t)} \rightarrow u_{i+1}(x(t), J(x(t), u_i(x(t))) = u_{i+1}(x(t)) \rightarrow \cdots$ |

Table 1.2 Reinforcement Learning (RL) as a Repetitive (Iterative) Feedback Control System Technique

## COMPUTER SCIENCE APPROACHES TO REINFORCEMENT LEARNING

Within computer science the most important steps have been inventions of two RL techniques:

**TEMPORAL DIFFERENCE LEARNING (Sutton, 1984)**

$$V(s_t) \leftarrow V(s_t) + \alpha\left[V(s_{t+1}) - V(s_t)\right] = (1-\alpha)V(s_t) + \alpha V(s_{t+1})$$

$$\text{NEW ESTIMATE} \leftarrow \text{OLD ESTIMATE} + \alpha \times \left[\text{ERROR ESTIMATE}\right]$$

$V(s_t)$ = value (or estimate) function at present state $s_t$

$V(s_{t+1})$ = value (or estimate) function at the next state $s_{t+1}$

$V(s_{t+1}) - V(s_t)$ = temporal difference

$t$ = present time; $t$+1 next (discrete) time (next iteration)

$\alpha$ = step size

**Q-LEARNING (Watkins, 1989) includes the action (control)**

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left[Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right]$$

$a_t$ = action (control) at time $t$

$a_{t+1}$ = action (control) at time $t+1$

Both temporal difference learning and Q-learning have some variants, especially the ones that incorporate expected values and discount factors.

***It will be seen that these simple formulas can be related to the continuous- and discrete-time Hamiltonians of the optimal control problem.***

The computer science approach is mostly tabular, array or tree based, ***stochastic in nature***, using Markov Decision Processes (MDP) or Monte Carlo tree searches.

## 2. Some Reinforcement Learning Terminology

- The **"(State) Value Function"** is denoted by $V(s)$. Most of the methods presented in [5] are "structured around estimating value functions".
- The **"Action-Value Function"** is denoted by $Q(s,a)$. It is also referred to as the "Q-Function".
  $V(s)$ and $Q(s,a)$ are improved via a lot of experiments over the training data/scenarios/episodes.
- **"Exploit"**: the agent uses the already learned information to makes a decision about the reward ("greedy strategy").
- **"Explore"**: the agent goes into uncharted territory to explore new information and eventually gets a better reward.
- **"Policy"**: denoted by $\pi(s)$ is the learning strategy used to determine the best action at state *s*. "It remains valid for the entire learning/training phase." page 16, [7].
- **"On-Policy"** learning approach when "training can only utilize data generated by the current policy $\pi$." page 16, [6].
- **"Off-Policy"** learning approach when any data collected can be used in training (memory base).
- **Model-based** reinforcement learning methods are used for planning, page 7, [5].
- **Model-free** reinforcement learning methods are used for trial-and-error learning, page 7, [5].

**Additional References:**


K. Bahare, K. Vamvoudakis, H. Modares, and F.L. Lewis, "Optimal and autonomous control using reinforcement learning: A survey," *IEEE Trans. Neural Networks and Learning Systems*, Vol. 29, 2042-2061, 2018.

R. Bellman, *Dynamic Programming*, Princeton University Press, Princeton NJ, 1957. Republished by Dover Publications Inc., Mineola New York, 2003.

R. Beard, T. McLain, C. Peterson, and M. Killpack, *Introduction to Feedback Control: Using Design Studies*, Published by Randal W. Beard, 2016, revised August 25, 2020.

D. Bertsekas, *Reinforcement Learning and Optimal Control*, Athena Scientific, Belmont, MA, 2019.

D. Bertsekas and J. Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, Nahua, NH, 1996.

J. Chan, *Learn Python in One Day and Learn it Well*, 2nd edition, published by Jamie Chan, 2017.

F. Lewis and D. Vrabie, "Reinforcement learning and feedback control," *IEEE Circuits and Systems Magazine*, 32-50, Third Quarter, 2009.

F. Lewis, D. Vrabie, and K. Vamvoudakis, "Reinforcement learning and feedback control," *IEEE Control Systems Magazine*, 76-105, Dec. 2012.

F. Lewis and D. Liu, Eds., *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*, Hoboken, New Jersey, 2013.

L. Graesser and W. L. Keng, *Foundations of Deep Reinforcement Learning*, Addison Wesley (Pearson Educations, 2020.

MATLAB R2020a, *Reinforcement Learning Tool Box – User's Guide*, The MathWorks Inc., Natick, MA, 2019-2020.

MATLAB R2020a, *Reinforcement Learning Tool Box – Reference*, The MathWorks Inc, Natick, MA, 2019-2020.

M. Minsky, *Theory of Neural-Analog Reinforcement Systems and Its Application to the Brain-Model Problem*, Ph. D. Dissertation, Princeton University, 1954.

M. Minsky, "Steps Toward Artificial Intelligence," *Proceedings of Radio Engineers*, Vol. 49, 8-30, 1961.

I. Pavlov, *Conditioned Reflexes*, Oxford University Press, London, 1927.

W. Powell, *Reinforcement Learning and Stochastic Optimization: A Unified Framework for Sequential Decisions*, Wiley Interscience, 2020.

M. Sewak, *Deep Reinforcement Learning*, Springer Nature, Singapore, 2019.

R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 2020.

K. Vamvoudakis, "Non-zero sum Nash Q-learning for unknown deterministic continuous-time linear systems," *Automatica*, Vol. 61, 274-281, 2015.

K. Vamvoudakis and F. Lewis, "Multi-player non-zero-sum games: Online adaptive learning solution of coupled Hamilton-Jacobi equations," *Automatica*, Vol. 47, 1556-1569, 2011.

K. Vamvoudakis, H. Modares, B. Kiumarsi, and F. Lewis, “Game theory-based control system algorithms with real-time reinforcement learning: How to solve multiplayer games on line,” *IEEE Control Systems Magazine*, 33-52, February 2017.

D. Vrabie, O. Pastravanu, M. Abu-Khalaf, and F. Lewis, “Adaptive optimal control for continuous-time linear systems based on policy iteration,” *Automatica*, 477-484, 2009.

D. Vrabie, K. Vamvoudakis, and F. Lewis, *Optimal Adaptive Control and Differential Games by Reinforcement Learning Principles*, IET, London, United Kingdom, 2013.

M. Waltz and K. Fu, “A heuristic approach to reinforcement learning,” *IEEE Transactions on Automatic Control*, Vol. 10, 390-398, 1965.

C. Watkins, *Learning from Delayed Rewards*, Ph. D. Dissertation, University of Cambridge, 1989.

C. Watkins and P. Dayan, “Q-Learning”, *Machine Learning*, Vol. 8, 279-292, 1992.

Bahare Kiumarsi, K. Vamvoudakis, H. Modares, and F.L. Lewis, “Optimal and Autonomous Control UsingReinforcement Learning: A Survey,” IEEE Trans. Neural Networks and Learning Systems, vol. 29, no. 6, pp. 2042-2061, June 2018.

Yi Jiang, Bahare Kiumarsi, Jialu Fan, Tianyou Chai, Jinna Li, and F.L. Lewis, “Optimal Output Regulation of Linear Discrete-Time Systems With Unknown Dynamics Using Reinforcement Learning,” IEEE Transactions on Cybernetics, Vol. 50, No. 7, Pp. 3147-3156, July 2020.

Bahare Kiumarsi, Bakur AlQaudi, Hamidreza Modares, Frank L. Lewis, Daniel S. Levine, “Optimal Control Using Adaptive Resonance Theory and Q-Learning,” Neurocomputing, vol. 361, p. 119-125 October 2019.

Ci Chen, H. Modares, Kan Xie, F.L. Lewis, Yan Wan, and Shengli Xie, “Reinforcement Learning-based Adaptive Optimal Exponential Tracking Control of Linear Systems with Unknown Dynamics,” IEEE Trans. Automatic Control, vol. 4423, no. 11, pp. 4423-4438, Nov. 2019. DOI 10.1109/TAC.2019.2905215.

Victor Lopez and F.L. Lewis, “Dynamic Multiobjective Control for Continuous-time Systems using Reinforcement Learning,” IEEE Trans. Automatic Control, vol. 64, no. 7, pp. 2869-2874, July 2019.

Qinglai Wei, F.L. Lewis, Derong Liu, Ruizhuo Song, and Hanquan Lin, “Discrete-Time Local Value Iteration Adaptive Dynamic Programming: Convergence Analysis,” IEEE Transactions on Systems, Man, and Cybernetics: Systems, Vol. 48, No. 6, pp. 875-891, June 2018.