DISSERTATION

A SYNTHESIS OF REINFORCEMENT LEARNING AND ROBUST CONTROL THEORY

Submitted by
R. Matthew Kretchmar
Department of Computer Science

In partial fulfillment of the requirements
for the Degree of Doctor of Philosophy
Colorado State University
Fort Collins, Colorado
Summer 2000

COLORADO STATE UNIVERSITY

July, 2000

WE HEREBY RECOMMEND THAT THE DISSERTATION PREPARED UNDER OUR SUPERVISION BY R. MATTHEW KRETCHMAR ENTITLED A SYNTHESIS OF REINFORCEMENT LEARNING AND ROBUST CONTROL THEORY BE ACCEPTED AS FULFILLING IN PART REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY.

Committee on Graduate Work

Adviser

Department Head

ABSTRACT OF DISSERTATION

A SYNTHESIS OF REINFORCEMENT LEARNING AND ROBUST CONTROL THEORY

The pursuit of control algorithms with improved performance drives the entire control research community as well as large parts of the mathematics, engineering, and artificial intelligence research communities. A fundamental limitation on achieving control performance is the conflicting requirement of maintaining system stability. In general, the more aggressive is the controller, the better the control performance but also the closer to system instability.

Robust control is a collection of theories, techniques, and tools that form one of the leading edge approaches to control. Most controllers are designed not on the physical plant to be controlled, but on a mathematical model of the plant; hence, these controllers often do not perform well on the physical plant and are sometimes unstable. Robust control overcomes this problem by adding uncertainty to the mathematical model. The result is a more general, less aggressive controller which performs well on both the model and the physical plant. However, the robust control method also sacrifices some control performance in order to achieve its guarantees of stability.

Reinforcement learning based neural networks offer some distinct advantages for improving control performance. Their nonlinearity enables the neural network to implement a wider range of control functions, and their adaptability permits them to improve control performance via on-line, trial-and-error learning. However, neuro-control is typically plagued by a lack of stability guarantees. Even momentary instability cannot be tolerated in most physical plants, and thus, the threat of instability prohibits the application of neuro-control in many situations.

In this dissertation, we develop a stable neuro-control scheme by synthesizing the two fields of reinforcement learning and robust control theory. We provide a learning system with many of the advantages of neuro-control. Using functional uncertainty to represent the nonlinear and time-varying components of the neural networks, we apply the robust control techniques to guarantee the stability of our neuro-controller. Our scheme provides stable control not only for a specific fixed-weight, neural network, but also for a neuro-controller in which the weights are changing during learning. Furthermore, we apply our stable neuro-controller to several control tasks to demonstrate that the theoretical stability guarantee is readily applicable to real-life control situations. We also discuss several problems we encounter and identify potential avenues of future research.

R. Matthew Kretchmar
Department of Computer Science
Colorado State University
Fort Collins, Colorado 80523
Summer 2000

ACKNOWLEDGEMENTS

An enormous credit is due to Dr. Charles Anderson of the Computer Science Department. Chuck served as my advisor for four and a half years, introduced me to reinforcement learning and nurtured me through my graduate career. Without his patient assistance, close collaboration, and careful guidance this dissertation would not have come to fruition.

Special thanks is due to Dr. Peter Young and Dr. Douglas Hittle of the Electrical Engineering and Mechanical Engineering departments, respectively. This dissertation would not have been possible without Peter's expertise in control theory and robust control theory, and without Doug's thorough knowledge of HVAC control systems. I would like to thank Dr. Adele Howe and Dr. Darrell Whitley of the Computer Science Department. While serving as committee members, they offered excellent technical advice to make the dissertation a more coherent document. Thanks is due to all five members of the committee for countless hours of document reading and editting.

From a financial standpoint, thanks is due to the National Science Foundation. The work in this dissertation is funded by grants CMS-9401249 and CMS-980474. Thanks is due to Charles Anderson, Douglas Hittle, and Peter Young who served as the principle investigators of these NSF grants and provided my research assistantship. Financial assistance was also provided by the Department of Computer Science in the form of computing equipment and a teaching assistantship. The Colorado State University is also thanked for additional scholarship funds.

I wish to thank my parents, R. Scott and Janet Kretchmar, and my sister, Jennifer Kretchmar, who provided support and motivation with a "dissertation race".

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

Introduction

1.1 Problem Statement

Automated controllers provide control signals to a plant in an attempt to cause the plant to exhibit a desired behavior. Here we use "plant" as a generic term for a device that is capable of being controlled. Together, the plant, the controller, and their interconnection comprise the system. The design of controllers is complicated by system instability which results in at least improper plant operation, and possibly, significant damage to equipment and/or injury to people. Fortunately, a properly designed controller prevents the system from operating in a dangerous, unstable mode. Therefore, it is imperative that the controller be engineered with stable operation as a primary goal; performance is a secondary design consideration to be pursued after stability is assured.

The design of such controllers can be approached from a number of design philosophies. In this dissertation, we focus on two diametric design philosophies. The first design philosophy, *robust control*, exploits significant a priori system knowledge in order to construct a high-performing controller that still guarantees stability. The other design philosophy, *reinforcement learning*, builds a controller assuming little initial knowledge of the system but is capable of learning and adapting to find better control functions. The robust controller uses extensive system knowledge but is fixed and rigid for all time; the reinforcement learning controller uses limited system knowledge but is capable of continuous adaptation to find better control schemes. The opposition of these two design approaches is illustrated in Figure 1.1.

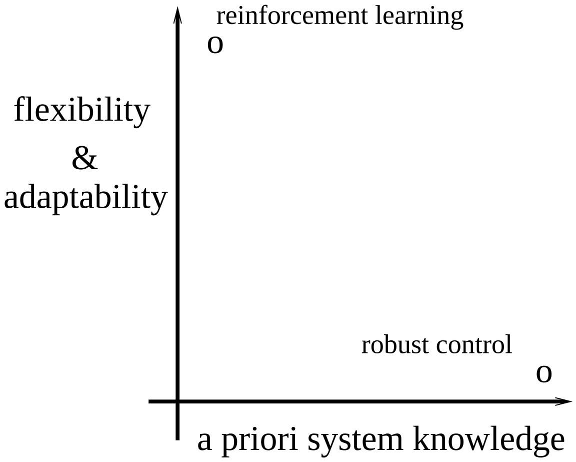

Figure 1.1: Controller Design Philosophies

In robust control, we analyze the dynamics of the plant in an attempt to build a controller that is mathematically guaranteed to provide stable control behavior. Often we discover an entire set of stable controllers; we then select the best performing controller from this stable set. However, most plants of practical interest possess enough complexity to prohibit the precise specification of the plant dynamics; usually we are forced to compute a mathematical model of the plant to serve as an approximation to the real plant. Consequently, the robust control design process is complicated because we must not only construct a controller that is stable for our mathematical model, but is also stable for the real plant. Necessarily, this limits the aggressiveness of the controller design and thus, results in suboptimal control performance.

On the other hand, reinforcement learning assumes little about the dynamics of the system. Instead, it develops a good control function through on-line, trial and error learning. The challenge of this approach is to establish a framework with enough flexibility to allow the controller to adapt to a good control strategy. However, this flexibility may result in numerous undesirable control strategies; the engineer must be willing to allow the controller to temporarily assume many of these poorer control strategies as it searches for the better ones. It is important to note that many of the undesirable strategies may provide unstable control behavior. We can envision the implausibility of the reinforcement learning approach in designing a controller for a helicopter rotor; the adaptive controller may crash thousands, even tens of thousands of helicopters before it finds a stable control function. However, once the reinforcement learner settles in on reasonably good control strategies, it can refine its search and often discover an exemplary controller that is not only stable, but also outperforms controllers designed using robust control techniques.

Thus, the problem this dissertation examines can be summarized as follows. We desire a robust control approach because this approach guarantees stable control behavior and exploits known system knowledge to achieve relatively good initial control performance. But, a robust control approach sacrifices some control performance in order to achieve the stability guarantee. A reinforcement learning approach is attractive because it is able to discover excellently performing controllers via trial-and-error search, but might temporarily implement a variety of unstable control functions. The problem is to combine these two techniques to guarantee stability and also perform safe trial-and-error search in order to adaptively improve control performance. We now present this problem in more detail by defining plants, stability, performance and numerous other terms.

1.2 Problem Details

Figure 1.2 shows the basic components of a typical system. The plant, G, is the device to be controlled. A controller, K, produces the control signal, u, used to modify the behavior of the plant. In this dissertation, we focus on a broad category of tasks known as *tracking tasks*: the controller must provide control signals so that the plant output, y, mimics an external, time-varying, input signal called the reference input,

r. Performance is measured by the error signal, e, which is the difference between the reference signal and the plant output: $e = r - y$. We also require that the controller maintain system stability: for a finite and static reference signal, r, we require the system signals, u and y, remain finite and furthermore move asymptotically toward stable fixed points, $\bar{u}$ and $\bar{y}$. We are more precise about the notion of stability in subsequent chapters.

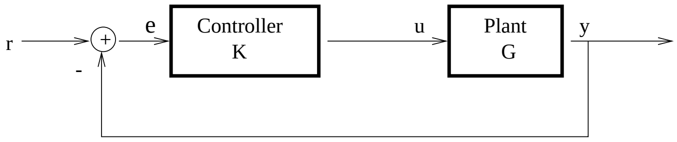

Figure 1.2: Nominal System

The vast majority of research in control theory applies to systems which are linear, time-invariant (LTI). The simple mathematics of LTI systems enables the application of the mature and extensive body of linear systems theory. Consequently, the design of stable controllers is straightforward. However, the LTI conditions place restrictive limits on the class of controllers available for use.

A non-LTI controller is often able to achieve greater performance because it is not saddled with the limitations of LTI. Two classes of non-LTI controllers are particularly useful for control; nonlinear controllers implement a wider range of control functions, and adaptive controllers self-modify to better match the system characteristics. However, nonlinear and adaptive controllers are difficult, and often impossible, to study analytically. Thus, the guarantee of stable control inherent in LTI designs is sacrificed for non-LTI controllers.

Neural networks, or neuro-controllers, constitute much of the recent non-LTI control research. Because neural networks are both nonlinear and adaptive, they often realize far superior control compared to LTI. However, dynamic analysis of neuro-controllers is mostly intractable thereby prohibiting control engineers from ascertaining their stability status. As a result, the use of neuro-controllers is primarily restricted to academic experiments; most industrial applications require guarantees of stable control which have not been possible with neural networks.

The stability issue for systems with neuro-controllers encompasses two aspects. *Static stability* is achieved when the system is proven stable provided that the neural network weights are constant. *Dynamic stability* implies that the system is stable even while the network weights are changing. Dynamic stability is required for networks which learn on-line in that it requires the system to be stable regardless of the sequence of weight values learned by the algorithm.

1.3 Objective

The primary objective of this work is to develop a framework in which we can ensure the stability of neuro-controllers. We then use this framework to prove the stability of both static and dynamic neuro-controllers. This objective is mainly a theoretical goal. While a few recent published results have shown some success with the static stability problem, we will provide a different proof that strengthens existing solutions. We also offer the first neuro-control scheme proven to solve the dynamic stability problem; the neuro-controller we propose is guaranteed to provide stable control even while the network is training.

As a secondary objective, we demonstrate that the theoretical stability proofs are practical to implement on real neuro-control systems tackling difficult control problems. Many theoretical results in artificial intelligence are not amenable to practical implementation; often technical assumptions of the theorems are violated in order to construct "reasonable and working" systems in practice. We show the assumptions of our stability theorems do not place unreasonable limitations on the practical implementation of a neuro-control system.

To demonstrate that we have achieved these objectives we will provide the following.

- A formal proof of the stability of a neuro-controller with fixed weights (static neural network).
- A formal proof of the stability of a neuro-controller undergoing weight changes during learning (dynamic neural network).
- A neural network architecture and a learning algorithm suitable for use in a general class of control tasks.
- A series of case studies showing that the neuro-control architecture and stability proofs are amenable to practical implementation on several control tasks.

1.4 Approach

In an effort to achieve our primary goal of verifying the stability of static and dynamic neuro-controllers, we employ an approach that combines reinforcement learning and robust control. We draw upon the reinforcement learning research literature to construct a learning algorithm and a neural network architecture that are suitable for application in a broad category of control tasks. Robust control provides the tools we require to guarantee the stability of the system.

Figure 1.3 depicts the high-level architecture of the proposed system. Again, r is the reference input to be tracked by the plant output, y. The tracking error is the difference between the reference signal and the plant output: $e = r - y$. A *nominal* controller, K, operates on the tracking error to produce a control signal u_c. A learning agent is included that also acts on the tracking error to produce a control signal $\hat{u}$. The two component control signals are added to arrive at the overall control signal: $u = u_c + \hat{u}$. Again, the goal of the controller(s) is twofold. The first goal is to guarantee system stability. The second goal is to produce the control signals to cause the plant to closely track the reference input over time. Specifically, this latter performance goal is to learn a control function to minimize the mean squared tracking error over time.

Importantly, the learning agent does not replace the nominal controller; rather, it adds to the control signal in an attempt to improve performance over the nominal LTI controller. This approach, retaining the LTI controller as opposed to replacing it, offers two advantages. First, if the neuro-controller fails, the LTI controller will still provide good control performance; the neuro-controller can be turned off without greatly affecting the system. Second, the control performance of the system is improved during the learning process. If the neuro-controller were operating alone, its initial control performance would most likely be extremely poor; the neural network would require substantial training time to return to the level of performance of the nominal LTI controller. Instead, the neuro-controller starts with the performance of the nominal controller and adds small adjustments to the control signal in an attempt to further improve control performance. The neuro-controller starts with an existing high-performance controller instead of starting tabula rasa.