

Source Coding for Controlling Cyber Physical Systems

A Dissertation Presented for the
Doctor of Philosophy
Degree

The University of Tennessee, Knoxville

Liang Li

December 2019

© by Liang Li, 2019
All Rights Reserved.

To all friends, thanks for all the supports.

Acknowledgments

Firstly, I would like to express my sincere gratitude to my advisor Prof. Husheng Li for the continuous support of my Ph.D. study and related researches, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of researches and writing of this dissertation. I could not have imagined having a better advisor for my Ph.D. study.

Besides my advisor, I would like to thank the rest of my dissertation committee: Prof. Hairong Qi, Prof. Qing Cao, and Prof. Xueping Li, for their insightful comments, encouragement, and especially their cooperation while I schedule this defense.

My sincere thanks also goes to Jiafu Mao and William Whiteley(Bill), who provided me opportunities to join their teams as an intern during my last two years in ORNL and Siemens. And I would also thank my fellow lab mates, Yifan Wang, Yawen Fan, Zhiyang Zhang and Jingchao Bao for all the fun we have had in the last five years.

Last but not the least, I would like to thank my families, especially my parents for raising me up in the early years, and my husband for supporting spiritually throughout writing this dissertation, and my daughter for motivating me in keeping working.

Abstract

Communication is an important issue in Cyber-Physical Systems(CPSs), which conveys information from sensor(s) to controller(s) in order to control the physical dynamics. However, the information exchanging from sensor(s) to controller(s) for controlling the physical dynamics are suffering critical problems, e.g., the lack of adequate bandwidth for bi-directional communication to handle increasing amount of data and the inefficiency of data transmissions with both high latency and high redundancy, and these critical problems have made the high communication cost and low performance of controlling in Cyber-Physical Systems(CPSs). In this dissertation, aiming at reducing the communication cost while maintaining the performance of controlling, we studied to design efficient source coding strategies (or quantization strategies) for controlling in the discrete-time and continuous-state CPSs, where the optimal source coding strategies are further formulated as the sequential coding optimization problems.

In a large scale, the goal of this dissertation is to find a deterministic but adaptive coding policy, as a series of mappings from the historical information to the quantization strategy for both the one sensor and multiple sensors case in CPSs. In particular, for the one sensor case, the optimization problem with the sequential coding is formulated and proposed to be solved by the Bellman Equation in dynamic programming (DP), and to overcome the challenge of continuous state space, a more practical solution is given by leveraging the Approximate Dynamic Programming (ADP). When it goes to the multiple sensors case, since the sensors receive correlated observations, we design an efficient and practical distributed source coding scheme by implementing reinforcement Q-learning with the learned side information available at the decoder. Numerical simulations with applications in smart grids demonstrate that the proposed two schemes for both the one sensor and the multiple sensors case can efficiently

compress the information source, which can significantly outperform previous strategies, such as fixed quantization strategies in CPSs.

Table of Contents

1	Introduction	1
1.1	Motivation	1
1.2	Prior Work	5
1.2.1	One Sensor Case	6
1.2.2	Multiple Sensors Case	8
1.3	Contributions	10
1.3.1	Features of Cyber-Physical System	11
1.3.2	One Sensor	12
1.3.3	Multiple Sensors	12
1.4	Structure of Dissertation	13
2	Dynamic State Aware Adaptive Source Coding (One Sensor)	14
2.1	System Model	14
2.1.1	Physical and Cyber Components	14
2.1.2	‘State Aware’ Quantizer and Decoder	17
2.1.3	Cost and Constraints	19
2.2	Optimization of Quantizer	21
2.2.1	DP Framework	21
2.2.2	ADP and Collocation	22
2.2.3	Performance of ADP	24
2.2.4	Summary of Algorithms	26
2.3	Experiment and Numerical Results	26
2.3.1	Two CPSs	27

2.3.2	Applications in mobile/vehicle communication	41
3	Dynamic Adaptive Distributed Source Coding (Multiple Sensors)	46
3.1	System Model	46
3.1.1	System Dynamics	46
3.1.2	Communication	49
3.1.3	Feedback from Physical Dynamics	49
3.2	Feedback Adaptive Distributed Source Coding	52
3.2.1	Quantization and Distortion	52
3.2.2	Distributed Source Coding	57
3.2.3	Theoretical Communication Bounds for Feedback Control	58
3.3	Cooperative Reinforcement Q-learning	62
3.3.1	The Optimization Problem	62
3.3.2	Cooperative Reinforcement Q-learning	64
3.4	Numerical Results	67
3.4.1	Feedback Adaptive Distributed Source Coding	67
3.4.2	Cooperative Reinforcement Q-learning	73
4	Conclusion	78
	Bibliography	79
	Appendix	87
A	Solar Power Truck	88
A.1	Key Components	88
A.2	Truck Circuits	96
	Vita	102

List of Tables

2.1	Key notations of the Chapter 2	15
3.1	Key notations of the Chapter 3	47
A.1	The pins for L298N	92
A.2	PINs distribution	101

List of Figures

1.1	An illustration of the vital components in Cyber-Physical Systems	3
1.2	An illustration of a microgrid [60]	4
1.3	Sequential source coding in one sensor case	7
1.4	Sequential source coding in multiple sensor case	9
2.1	A feedback control dynamic system with communications	17
2.2	Illustration of the quantizer at the sensor	18
2.3	Illustration of the decoder at the controller	18
2.4	A small truck balancing system	29
2.5	Quantization strategy for scalar case of Markov dynamical system	30
2.6	Quantization distortion for scalar case of Markov dynamical system	30
2.7	Absolute value of $\mathbf{R}(\mathbf{s}_i, \Phi)$ for scalar case of Markov dynamical system	31
2.8	Comparison of overall performance for scalar case of Markov dynamical system	32
2.9	Distribution of total cost for uniform and Lloyd quantizer	34
2.10	Selected strategy $\mathbf{Q}_t$ for uniform quantizer	35
2.11	Corresponding $\mathbf{R}(\mathbf{s}_i, \Phi)$ for uniform quantizer	36
2.12	Selected strategy $\mathbf{Q}_t$ for Lloyd quantizer	37
2.13	Corresponding $\mathbf{R}(\mathbf{s}_i, \Phi)$ for Lloyd quantizer	38
2.14	Average bits per system state in every dimension	39
2.15	Average distortion per system state in every dimension	39
2.16	Elapsed CPU time for different values of K	40
2.17	Sensing distance	42
2.18	Quantization error	43

2.19 Overall average cost	44
3.1 The feedback aware dynamic system	50
3.2 Sequential structure in distributed source coding model	50
3.3 An illustration of unstable and stable subspaces	56
3.4 An illustration of transformation to the desired lattice	56
3.5 An illustration of super-rectangles	58
3.6 Cooperative reinforcement Q-learning network architecture	63
3.7 Power-grid system	69
3.8 Kalman Filtering results for fifty samples	69
3.9 Prediction variance based on different quantization levels	70
3.10 State evolution	71
3.11 State error variance	72
3.12 The cart-pole system in GYM toolbox	74
3.13 The duration of the cart	75
3.14 The stabilization of the system with two sensors	76
3.15 The distribution of the total bits needed for each episode	77
A.1 Solar battery	89
A.2 Robot car chassis	89
A.3 L298N	91
A.4 MP430 by TI	93
A.5 The device pin out for MSP430G2	93
A.6 Wireless communication module CC2500	94
A.7 CC2500 module	95
A.8 HY-SRF05	97
A.9 HY-SRF05 measurement procedure	97
A.10 Female to female jumper	98
A.11 Programming line with MSP430	98
A.12 Digital multimeter	99
A.13 Whole truck circuit	100

Chapter 1

Introduction

1.1 Motivation

In the last decade, the advances of digital portable devices have resulted in an explosive growth of embedded systems. This growth trend of the embedded systems triggered the development of autonomous systems, which are capable of collecting and processing an enormous amount of data within a very short time [64]. The Cyber-Physical Systems (CPSs), known as one of these autonomous systems, have attracted substantial studies in recent years as the pressing demand of innovations in its advancing Cyber-Physical System (CPS) technologies and wide applications [27, 50, 66, 24]. The CPSs, usually defined as the smart networked systems with embedded sensors, processors and actuators, are designed to sense and interact with the physical world (including human users), and support various kinds of applications [22], such as assistive monitoring, feedback in tele-rehabilitation, energy management, and negotiation [8]. And in a typical CPS, these sensor(s) and controller(s) would observe or control the physical dynamics of digital systems. In fact, many practical systems can be categorized as the Cyber-Physical Systems, such as smart grids [2], unmanned aerial vehicles (UAVs) [4], robotic networks [7] and internet of things [52, 34].

As one of key important components for networked control in CPSs, communication provides vital functions for conveying the necessary information from the sensors to the controllers while the sensors and the controllers are not located at the same place(location). The physical dynamics is sensed by one or more sensors, and transferred from the sensors

to the controllers via the communication channels, where the communication channels can be wired or wireless as illustrated in Fig. 1.1. In order to support real-time and reliable critical services, the latency and reliability of the communication networks are usually of great importance for CPSs, which makes the communication networks play an crucial role in the development of CPSs. For example, certain applications, such as real-time state estimation using phasor measurement unit (PMU) data in smart grids, require very low latency (few tens of milliseconds) [40] and high reliability (e.g., the packet error rate should be less than 10^{-5} [1]). Hence, the communication networks, are the subject of many research efforts, delving into the most efficient topology of the communication networks, physical media, protocols, etc [33, 58, 59].

Indeed, from the viewpoint of communications, the existing CPSs are suffering from several critical problems, e.g., the fragment of the system architectures in the CPSs; the absence of adequate bandwidth for two-way communications; the lack of interoperability between system components; and the inability to handle increasing amount of data from smart devices [40]. Moreover, we know that the most important and crucial property of physical world is the continuous physical dynamics, and transmitting this continuous physical dynamics demands a large size of bandwidth in communication to transmit the sensed necessary information from the sensors to the controllers. Due to these critical issues, it is important to use efficient communication schemes (e.g. source coding scheme) to compress the information, while maintaining the minimal latency and sufficient reliability of the CPSs.

Fig. 1.2 shows one concrete example of CPSs as a microgrid. The microgrid is a group of interconnected loads and distributed energy resources that acts as a single controllable entity with respect to the grid. The microgrid can connect and disconnect from the grid, and to run an automated optimization, microgrid devices have be outfitted with sensors and communication links that can provide real-time information to a central energy management system. However, nowadays, the communication in microgrid still suffer heavy loads with the redundancy not being removed in the data transmitting. While microgrid requires low latency and high reliability in data transmitting, the efficient communication schemes will be very beneficial to satisfy these strict requirements .

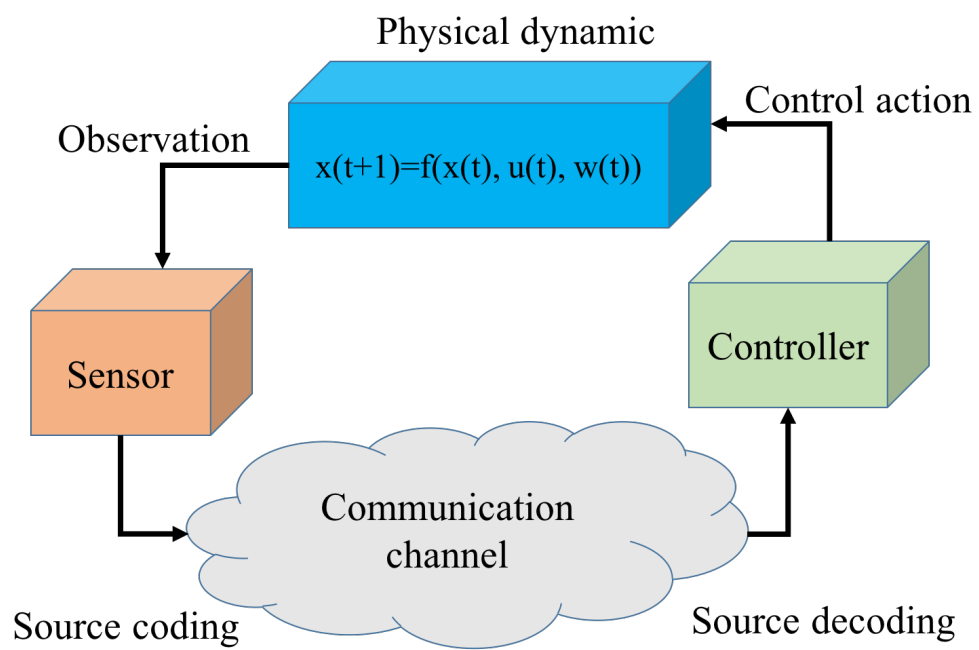


Figure 1.1: An illustration of the vital components in Cyber-Physical Systems

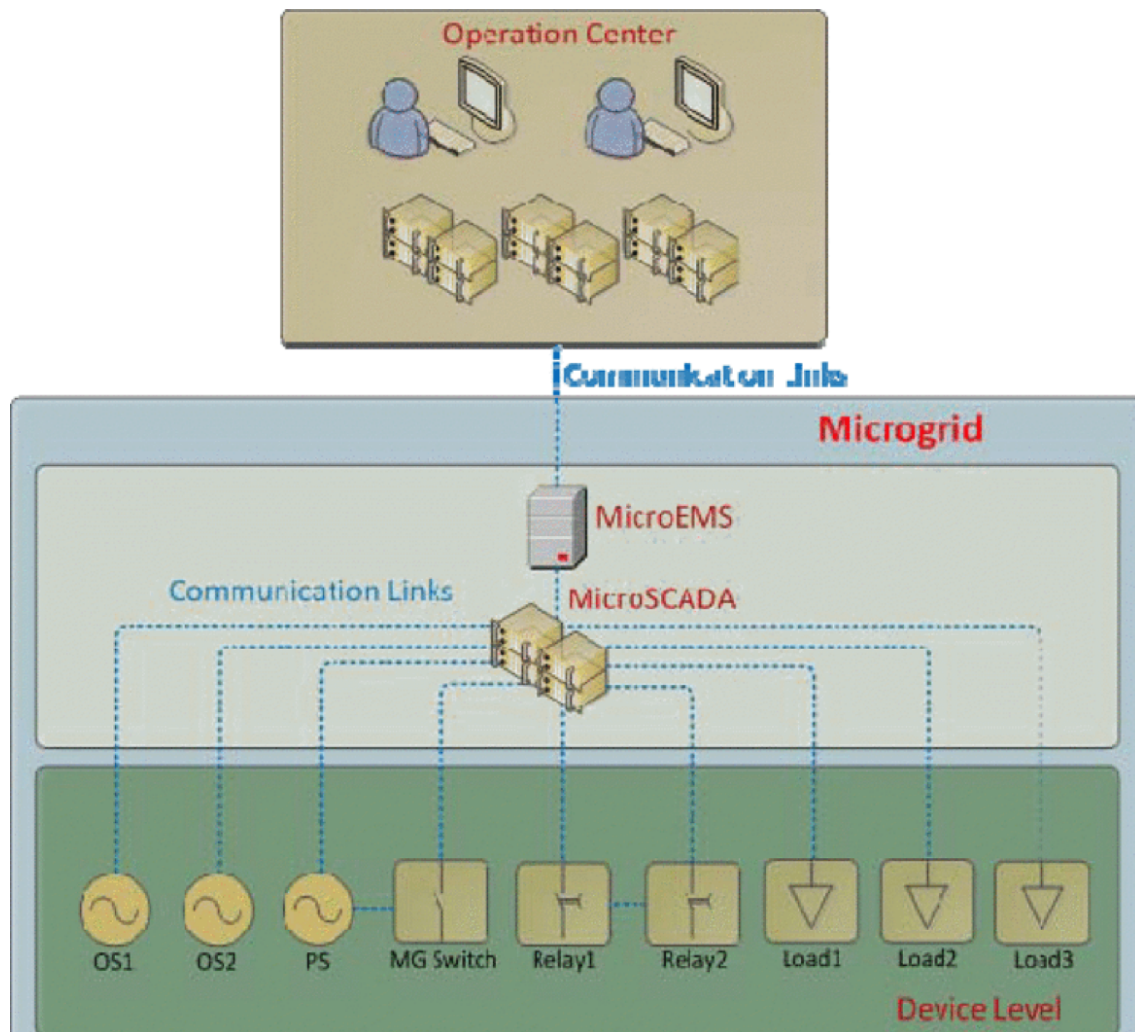


Figure 1.2: An illustration of a microgrid [60]

In regard to the source coding schemes, the most straightforward approach is to follow the traditional procedures of separating source coding and channel coding and adopting a fixed coding scheme for CPS. In this dissertation, we will study the lossy source coding (or equivalently quantization) as one type of source coding schemes, where the sensors in CPSs would monitor the physical dynamics and compress the necessary information meanwhile maintaining the sufficient reliability of the CPS. In general, we will try to work on two scenarios, e.g. one sensor is existing, and multiple sensors are existing, where they collaborate with each other in the CPSs. For the one sensor case, the sensor will try to compress the observed information through combining the state information available at the decoder side, and for the multiple sensors case, the sensors will try to collaborate with each other to reduce the average transmission cost by using the side information available at the decoder side. Since in CPSs the information source (namely physical dynamic in CPSs) is no longer i.i.d., it is desirable to make the source coding adaptive to the system state of physical dynamics and the side information available at the decoder. Driven by this motivation, we aim at studying the policy of the state (side information) aware sequential source coding schemes, instead of traditional fixed coding schemes, to propose the solutions of communications for CPSs for these two scenarios.

1.2 Prior Work

Although there have been substantial and successful studies on pure data communication networks, and the communication networks for CPSs have many new features. These new features add more challenges and require a full revisit to every aspect of the design and analysis of communications in CPSs. To the author's best knowledge, there are few studies on the practical implementations of protocols and algorithms for both the one sensor and the multiple sensors' communication in controlling dynamics of CPSs. To be specific, our goal is to find a solution for sequential source coding in CPS where we would utilize the state or side information available at the decoder to reduce the communication cost. To begin with, we start with the background with the sequential stochastic control, which has been studied in the community of both control and communications.

1.2.1 One Sensor Case

In fact, when only one sensor is considered, some interesting studies have been published to solve the sequential stochastic control problem theoretically focusing on sequential source coding (or equivalently quantization). In most cases, the idea of the sequential source coding is to find an optimal quantizer changing in time, which could be used to quantize the necessary information to be transmitted later. In Fig. 1.3, the target quantizer has been illustrated, where the designed quantizer would quantize the physical dynamics aiming at compressing the information and reducing the cost. For example, in [6], Bokar et al. formulated a stochastic control problem equivalent to the optimal sequential vector quantization one, which is jointly optimized over the average entropy rate and distortion, and the optimal strategy is solved via dynamic programming (DP). Later, Tatikonda et al. examined the role of communication on the classical linear quadratic Gaussian problem and presented the bounds on the achievable theoretical rate for the inherent tradeoffs between control and communication costs [56, 55]. Then, the optimal control schemes of sequential encoder and decoder, which could minimize an average distortion measure over a finite horizon, were proposed in [57]. In [38], Mahajan and Teneketzis formulated a zero-delay communication problem as a dynamic game with nonclassical information structure and proposed the methodology of the search for an optimal communication strategy to minimize the total expected symbol-by-symbol distribution between the source output and its reconstruction in real-time communication systems. In a large scale, these researches are leading in sequential source coding for feedback control systems, they are more focused in the theoretical studies, e.g., the optimal vector quantization schemes with the complete information of probability distributions or conditional distributions of the information sources.

In addition, several other strategies have shown some insights and been proposed to compress the information while maintaining the stability in networked control systems. In [17, 16], Fu et al. studied the quantized output feedback control of discrete-time linear systems using a finite-level quantizer, where the authors gave a suboptimal scheme for the optimization of quantization levels, and they also presented the design of a corresponding

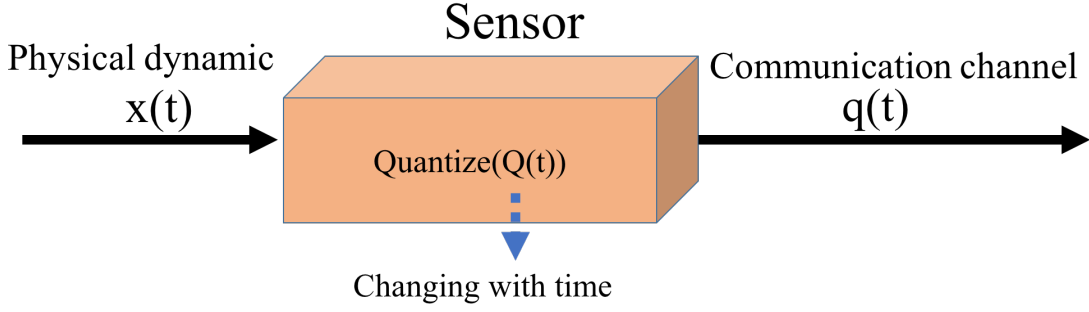


Figure 1.3: Sequential source coding in one sensor case

quantized dynamic output feedback controller. Rasool et al. investigated the case with multiple quantizers with different quantization levels and different network-induced delays in the networked control systems, and they showed that using multiple quantizers is more natural for the network with a limited capacity in [49]. [37] addressed the problem of a reset state observer-based controller by using the quantized measurement to reduce the estimation error. These works involved optimal quantizer are mainly related to either logarithmic quantizer or uniform quantizer for linear systems, where nonlinear system with strong latency constraints is still open in networked control systems.

The very recent researches also proposed some interesting results. In [62], the authors summarized previous works and established the structure of optimal quantization policies under various information structures for general cost functions. For example, the quantization output at time t can be generated by a function of its local information and some additional side information available at the encoder. And in [48], the authors considered three basic formats of encoder outputs, quantized samples, real-valued samples at event-triggered times, and real-valued samples over additive noise channels, and they showed that if the controller and encoder are dynamic, then the performance cost can be minimized by a separated design. Later, the authors in [48] again proved that it is optimal to apply separation and certainty equivalence for the dynamic encoder and controller design problem. In [31, 32], aiming at reducing communication cost, the authors obtained a suboptimal quantization strategy based on the history state information by using approximate dynamic programming. Later the joint channel decoding and state estimation was proposed for a single sensor in [20, 30] by using the idea of belief propagation.

1.2.2 Multiple Sensors Case

When it turns into multiple sensors case as shown in Fig. 1.4, the source coding schemes would be of more complexities and difficulties. Luckily, we will not need to study this all by standing on the shoulders of giants. It has been found in information theory that multiple sources can carry out independent source coding without exchanging information such that the sum transmission rate equals to that of the single source case, by exploiting the seminal Slepian–Wolf coding scheme [51] and Wyner–Ziv coding [3] scheme. The information theoretic argument is then extended to distributed source coding schemes by employing parity check based channel coding [46, 47, 53, 18, 39], nested lattice coding [63, 35], or network aware source coding [14]. A survey can be found in [12, 13], where multiple applications have been enumerated. In [61], the author proves the existence of a stationary measure, asymptotic mean stationarity, and ergodicity conditions when they are presenting for a class of non-Markovian processes for the applications in the feedback quantization and stochastic control. Nevertheless, it is extremely difficult to find the optimal solution due to the complexity of systems and controlling requirements in CPSs.

Based on Slepian–Wolf coding and Wyner–Ziv coding schemes, we also give one possible application of two sensors case. Here we denote the two sensors(encoders) as sensor 1 and sensor 2. The two sensors are observing the physical dynamics, and they will transmit correlated observations to the controller(decoder). We let the sensor 1 transmit ‘necessary information’ while the sensor 2 will transmit ‘encoded information’ to the decoder. In this case, the ‘necessary information’ sent by sensor 1 can be viewed as the side information available at the decoder, and the encoded information sent by sensor 2 will be the encoded one based on the side information. Note that in order to encode the observations, sensor 2 needs to learn the ‘necessary information’ sent by sensor 1 in advance. Since there are no communications between the sensors in Slepian–Wolf coding and Wyner–Ziv coding schemes, the validation in sharing of ‘necessary information’ between sensor 2 and the decoder impedes the development of distributed source coding in practical applications.

The truth is that compared with the theoretic studies, there are few studies on the practical applications of distributed source coding. To the author’s best knowledges, some

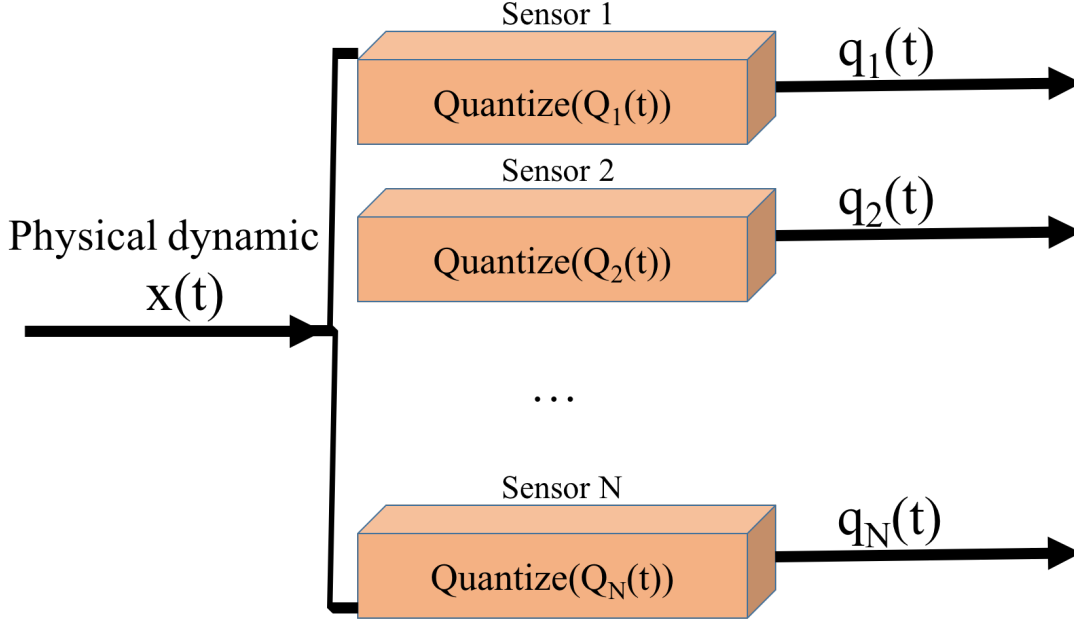


Figure 1.4: Sequential source coding in multiple sensor case

authors have tried to make use of the side information, such as the state information from past time, but multiple sensors case is still left open to discuss. In [31, 32], a one-sensor case is studied to design the optimal quantizer for CPSs in order to reduce the average communication by using approximate dynamic programming, where they utilize the past states information as the side information. The joint channel decoding and state estimation is proposed for a single sensor in [20, 30]. In [11], the authors consider the remote state estimation by using a remote estimator over a shared channel, and a multi-player non-cooperative game framework is proposed to achieve the better estimation performance. The multicast routing is studied for networking of decentralized sensors and controllers aiming at controlling the physical dynamics in [29], and simulations show that the proposed framework and algorithm can effectively stabilize cyber physical systems.

In summary, to realize the distributed source coding with utilization of Slepian–Wolf coding or Wyner–Ziv coding directly, we will need to make several basic assumptions.

- We need to assume that the error-free side information would be available at the decoder. In fact, it would be hard to realize the distributed source coding since the error in the side information will render the source coding un-decodable.

- We need to assume that the error-free side information must be synchronized in both the encoder and the decoder.

In fact, in most scenarios for CPSs, the sensors are battery-constrained and randomly scattered in a large area. The error-free communication might require complex energy consuming channel coding algorithms (such as LDPC and Turbo codes), which is not practical. Moreover, the synchronization between the encode and the decoder might take an extra cost, which would further burden the channels with extra communication cost.

Deep learning is one of the hottest topics in recent five years. As one of the main components, the back-propagation algorithm used in deep learning help discover how a machine should change its internal parameters to 'remember' the intricate structure in large data set [26]. And reinforcement learning is to take suitable actions to maximize the reward in a long time term with deep learning. Different from one agent reinforcement learning, cooperative reinforcement learning [23] is used in a multi-agent and cooperative environment with continuous observable state information, where multiple agents will cooperate by sharing episodes, learned policies and so on [54]. Due to validation of deep learning techniques, it has been widely used in computer vision, speech recognition, nature language processing.

Different from traditional methods, deep learning [26] (or known as deep structured learning or hierarchical learning) allows computational models that are composed of multiple processing layers to learn representation of data with multiple levels of abstraction. It is more simple without strict requirements once the models are trained, while they are still not touched in the field of source coding in CPSs. Hence, in this dissertation, we also explore to use deep learning (especially reinforcement Q-learning) for the distributed source coding problem in multiple sensors case.

1.3 Contributions

The existing results shed some ideas on how to design source coding for networked control systems in the sense of theoretical study, but the detailed algorithm design and performance evaluation for practical applications are still open. In this dissertation, we focus on the source coding scheme of CPS, in particular the optimal solutions of quantization policies

of CPS. We try to design the source coding in two cases, for the case of one sensor with correlated system states and the case of multiple sensors with correlated observations, and we also try to work out the practical solutions. From a large scale, we fit the source coding in CPSs into the framework of distributed lossy coding by defining the control aware criterion of distortion and applying the framework of lattices as quantization. Before we start with these two cases, we first revisit the features of CPSs.

1.3.1 Features of Cyber-Physical System

Although lots of theoretical studies have been deployed to design the communication network, new features in CPSs have added new challenges to the deployments of conventional strategies. In the following, we try to reconsider the new features in CPSs.

- Complicated relationship: The relationships among the distributed observations could be more complicated than many existing studies in which $Y(t) = CX(t) + N(t)$, where $Y(t)$ is the observation, C is the observation mapping matrix, $X(t)$ is the system state and $N(t)$ is a random noise.
- Control aware distortion: The distortion function could be directly related to the purpose of controlling the CPSs, which is more complex than simple distortion criterion such as minimum mean square error (MMSE) used in many lossy source coding scenarios.
- Time correlations: Many existing studies assume observations independent in time. However, the observations are usually correlated in time in CPSs due to the system state evolution.
- High dimension: The system dynamics might be high in dimensions for the certain systems. The dimension of the practical dynamics might be prohibitively difficult to obtain the optimal solution of quantization policies.

Note that if the computation complexity is still high in CPSs, it will finally lead to high latency and low reliability, which is intolerable for some strict applications. Thus,

considering the new features of CPSs, we try to propose more practical and effect schemes in source coding for those strict applications in CPSs while taking the communication cost into account.

1.3.2 One Sensor

For the one sensor case, we will focus on the design of practical algorithms for quantization policies, by leveraging approximate dynamic programming (ADP) [5, 45, 41] while taking advantages of the correlated physical dynamics. In summary, the major contributions of this part as shown in Chapter 2 include

- The source coding/quantization scheme is made aware of the physical dynamics, and the optimality of the quantization strategy is evaluated in a provable manner.
- The suboptimal but practical quantizers are designed by using ADP for the two practical dynamic systems.

1.3.3 Multiple Sensors

For multiple sensors case, as we assume that there are no communications among sensors, we first fit source coding problem for multiple sensors as a dynamic programing framework by following Slepian–Wolf coding and Wyner–Ziv coding schemes. This framework would be similar to our one sensor case, where the difference is that the side information is no long the past state information but a feedback information from the decoder. Next, to further reduce the complexity of source coding, we leverage reinforcement Q-learning to find the optimal solution of quantization policies. To this end, the contributions of this part as shown in Chapter 3 can be summarized as following:

- The practical distributed source coding/quantization method is proposed, where more than two sensors are considered with correlated observations.
- Two proposed methods are implemented in practical CPSs, which are dynamic programing method and cooperative reinforcement Q-learning method, and the

simulations are given to illustrate the performance with the theoretical bounds being compared.

1.4 Structure of Dissertation

The remainder of the dissertation is organized as following. We present the one sensor case in Chapter 2, where the system model for one sensor is shown in Section 2.1 to explain the schemes of encoder and decoder. The methodology for the design of the source coding as well as the theoretical performance evaluation, is elaborated in Section 2.2. Experiment and simulation results for two practical applications are shown in Section 2.3. We present the multiple sensors case in Chapter 3, where the system model is introduced in Section 3.1. To adapt the dynamic system, we consider the distortion criterion and quantization scheme in Section 3.2 and the proposed cooperative learning will be discussed in Section 3.3. An estimation of theoretical communication is given in Section 3.2.3. The numerical results is given in Section 3.4. Finally, the conclusions are drawn in Chapter 4.

Chapter 2

Dynamic State Aware Adaptive Source Coding (One Sensor)

In this chapter, we will give the source coding for one sensor case. Some key notations are listed in Table 2.1 to help understand the following chapter.

2.1 System Model

In this section, we provide a brief introduction to the system model of CPSs, in particular the communication system architecture.

2.1.1 Physical and Cyber Components

We consider a discrete-time and continuous-state feedback system in CPSs. The dynamics of the system is given by

$$\mathbf{x}_{t+1} = g(\mathbf{x}_t, \mathbf{u}_t, \epsilon_t) \quad (2.1)$$

where $\mathbf{x}_t \in \mathbb{R}^n$ is the system state vector, $\mathbf{u}_t \in \mathbb{R}$ is the control input, and the noise $\epsilon_t \in \mathbb{R}^n$ is a random process independent of the system state. A sensor measures the system state $\mathbf{x}_t$, and reports to a controller via a communication channel by quantizing the measurement. For simplicity, we assume that the measurement is noise free, which is reasonable when the

Table 2.1: Key notations of the Chapter 2

Symbols	Key units in CPS	Notes (introduced in equation)
$\mathbf{x}_t$	System state vector	$\mathbb{R}^n$, (2.1)
$\mathbf{u}_t$	System control input	$\mathbb{R}^n$, (2.1)
ϵ_t	System noise	$\mathbb{R}^n$, (2.1)
$g(\cdot)$	State transition function	(2.1)
q_t	Output of quantizer	$\mathbb{R}$, (2.2)
$\mathbf{Q}_t$	Quantization strategy	$\mathbf{Q}_t = f(\{q_s\}_{s < t})$, (2.2)
$\mathbf{Q}_t^{-1}$	Reconstruction strategy	(2.2)
$\hat{\mathbf{x}}_t$	Estimated system state vector	$\mathbb{R}^n$, (2.2)
$h(\cdot)$	Control strategy	(2.2)
$N_{\mathbf{Q}_t}$	Number of quantization regions	$\mathbb{R}$, (2.3)
$r_{\mathbf{Q}_t}$	Quantization rate	$\mathbb{R}$, (2.3)
α	Discount Factor	$0 < \alpha < 1$, (2.3)
D	Distortion threshold	$D \geq 0$, (2.3)
$d(\cdot, \cdot)$	Distortion	(2.3)
$R_T^{QDRD}(D)$	Quantization Discount Rate Distortion	$\mathbb{R}$, (2.3)
$E_\epsilon[\cdot]$	Expectation of system state over the noise	(2.3)
λ	Lagrange multiplier	$\lambda > 0$, (2.4)
$\mathbf{b}_t$	Quantization bits(strategy)	$\mathbf{b}_t = [b_t^1, b_t^2, \dots, b_t^n]$, (2.5)
ϕ	Pre-assigned basis function	$[\phi_1, \dots, \phi_K]$, (2.11)
c	Basis function coefficients	$[c_1, \dots, c_K]$, (2.11)
$v^*(\mathbf{x}_t)$	Solution of Bellman Equation(BE)	Value Function, (2.7)
$v^a(\mathbf{x}_t, \mathbf{c})$	Approximated solution of BE	Value Function, (2.11)
$\mathbf{s}_t$	Sampled system state vector	$\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_K]$, (2.12)
$\mathbf{x}_{min}$	Minimal Bound of system state	$\mathbb{R}^n$, (2.22)
$\mathbf{x}_{max}$	Maximal Bound of system state	$\mathbb{R}^n$, (2.22)
d_t	Distance state of the truck	$\mathbb{R}$, (2.24)
w_t	Random perturbation of state	$\mathbb{R}$, (2.24)

measurement noise is negligible, and the communication is free of transmission errors which is also practical since the existing encoding algorithm are well designed to obtain errorless information through CPS network. In case an error will happen, a packet-request step from the decoder and a packet-retransmit step from the quantizer can be proceeded sequentially. Furthermore, we assume that the quantized state and the feedback control of the system are given by

$$\begin{cases} q_t = \mathbf{Q}_t(\mathbf{x}_t) \\ \hat{\mathbf{x}}_t = \mathbf{Q}_t^{-1}(q_t) , \\ \mathbf{u}_t = h(\hat{\mathbf{x}}_t) \end{cases} \quad (2.2)$$

where $q_t \in \mathbb{R}$ is the output of the quantizer which will be sent to the controller, $\mathbf{Q}_t(\cdot)$ is the quantization strategy at time slot t , which is adaptive to the system state and obtained from a policy $f(\cdot)$, $\mathbf{Q}_t^{-1}(\cdot)$ represents the reconstruction of the state $\mathbf{x}_t$ from the quantized value q_t , and the function $h(\cdot)$ is the control strategy.

Note that for the generic case, the quantization strategies $\{\mathbf{Q}_s(\cdot)\}_{s=1,\dots,t}$ are based on all the history information, namely $\mathbf{Q}_t = f(\{q_s\}_{s<t})$, where the policy $f(\cdot)$ is a ‘state aware’ mapping from the history to the quantizer. Here the term ‘state aware’ means that the policy $f(\cdot)$ maps from the past quantized system states to the quantizer. Different from the adaptive quantization scheme in [15], which focuses on sequentially updating the local codebook, our strategies of quantization are based on the deterministic policy $f(\cdot)$; hence, the time variance of the quantization strategy stems from only the change of system state. Hence, we say that the source coding/quantization is aware of the system state. An illustration of the mode is shown in Fig. 2.1.

To simplify the analysis, we assume a Markov strategy in the simulation; i.e., $\mathbf{Q}_t = f(\hat{q}_t)$, where $\hat{q}_t$ denotes the prediction of the current system state at time t , which is obtained based on the observation of the previous system state at time $t - 1$. Since q_{t-1} is known to both the sensor and controller, so is $\hat{q}_t$. This can also make the system more practical, since a function based on all previous history requires an infinite memory. To be more generic, the case, i.e. $\mathbf{Q}_t = f(\{q_s\}_{s<t})$, will be discussed in the later section.

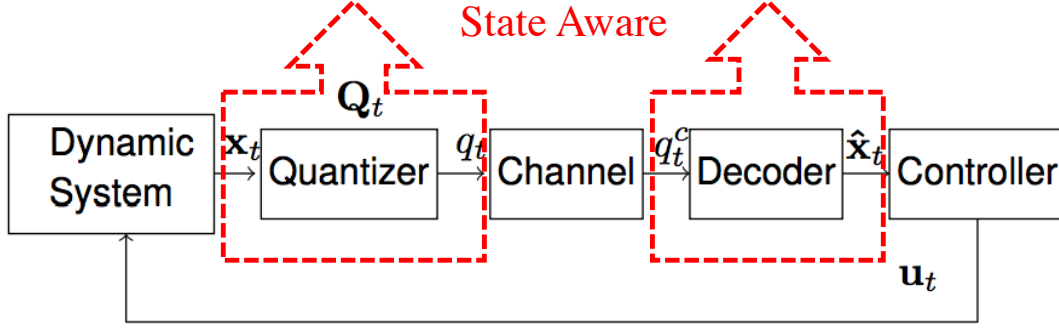


Figure 2.1: A feedback control dynamic system with communications

2.1.2 ‘State Aware’ Quantizer and Decoder

Quantizer

Fig. 2.2 illustrates the structure of the ‘state aware’ quantizer in Fig. 2.1. It first extracts the past quantized state $\{q_{s < t}\}$ of the dynamical system, estimates the current system state $\hat{q}_t$ based on the system state prediction, then obtains the quantization strategy $\mathbf{Q}_t = f(\hat{q}_t)$, quantizes the real current state $\mathbf{x}_t$, and finally outputs $q_t = \mathbf{Q}_t(\mathbf{x}_t)$ to the communication channel.

Note that since both the quantizer and the decoder share the same information of past quantized states at each time slot, the sender and receiver need no extra information to synchronize with each other; i.e., the quantizer and decoder can always share the same codebook according to the same quantization strategy. We will not specify the detailed structure of the quantizer since our analysis can handle the generic case.

Decoder

After receiving the quantized report q_t from the sensor, the decoder at the controller directly decodes the information based on the scheduled strategy $\mathbf{Q}_t^{-1}(q_t)$, yielding the estimate $\hat{\mathbf{x}}_t$ as shown in Fig. 2.3, and then calculates the control action $\mathbf{u}_t$ to regulate the dynamic system. Here $\mathbf{Q}_t^{-1}$ is obtained by the strategy $f(\cdot)$, which is based on the information of past quantized state, or equivalently the estimated state of current time slot, namely $\hat{q}_t$.

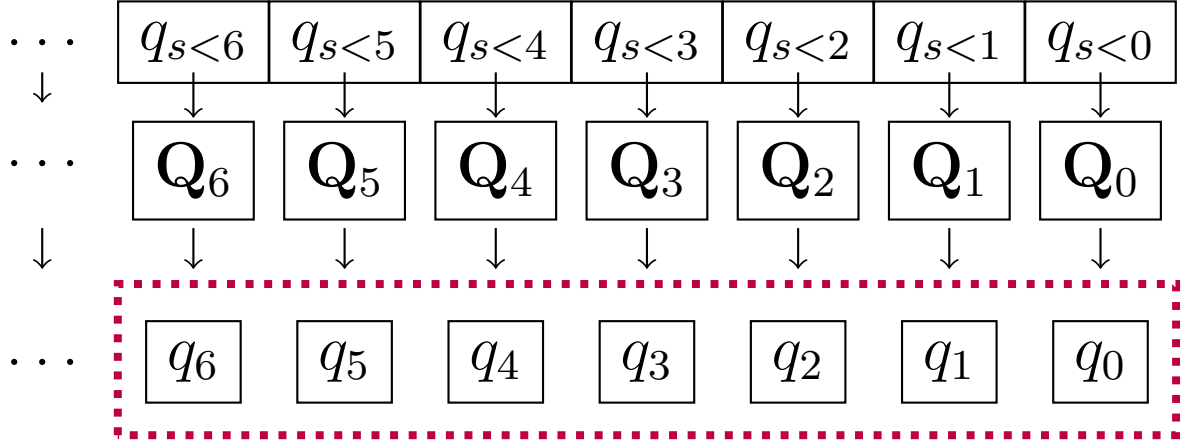


Figure 2.2: Illustration of the quantizer at the sensor

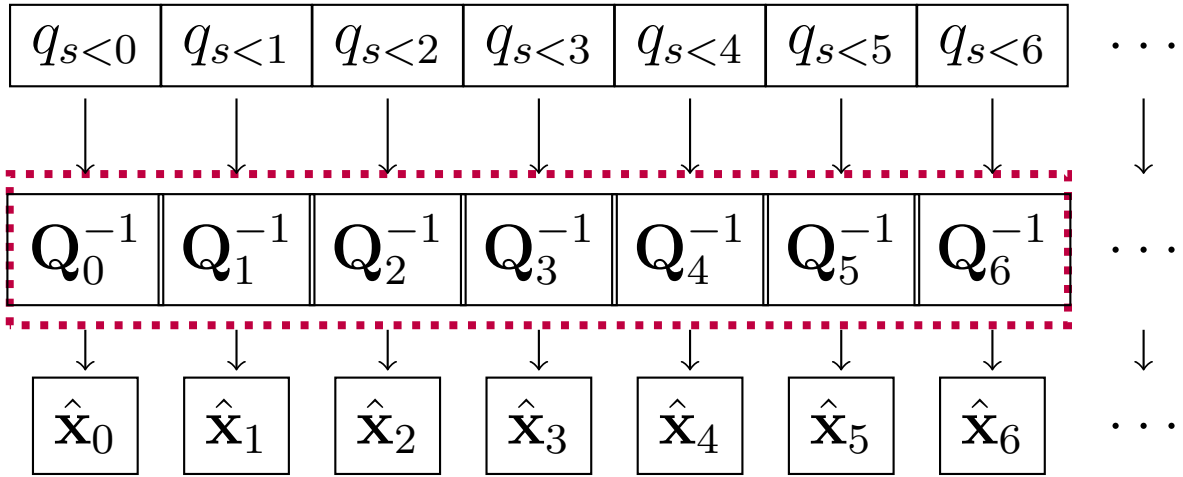


Figure 2.3: Illustration of the decoder at the controller

Note that, even though both quantizer and decoder share the estimated system state $\hat{q}_t$, the distribution of $\hat{q}_t$ is still unknown. To alleviate the unknown distribution, we still design the source coding policy Q_t by using the system state $\mathbf{x}_t$, while both the encoder and the decoder estimates Q_t using the prediction $\hat{q}_t$. The rationale is that the prediction error $x_t - \hat{q}_t$ is usually small, thus making the quantization policies calculated by both ways close to each other. As we will see, such an approximation yields an excellent performance.

2.1.3 Cost and Constraints

Now we formulate the quantization strategy design as an optimization problem with constraints. The philosophy of the formulation is to set the discounted sum of information bits as the cost function and the expected quantization error as the constraint.

To this end, we define the coding rate of the quantizer as $r_{\mathbf{Q}_t}$. Here $r_{\mathbf{Q}_t}$ stands for the overall coding rate to transmit the information, which includes both the overheads and the information bits in the packets. To be simplify, we assume that the size of overheads in the packets is fixed while the information bits are variable. This means that in our optimization we assume that $r_{\mathbf{Q}_t} = \lceil \log_2 N_{\mathbf{Q}_t} \rceil$, where $r_{\mathbf{Q}_t}$ is the number of bits used to represent the input sample via strategy $\mathbf{Q}_t$ [19], and $N_{\mathbf{Q}_t}$ is the number of quantization regions.

Plus, we assume a discount factor $0 < \alpha < 1$ for the cost such that the near future is more emphasized. Then for a time horizon T , the total cost of the system is defined as

$$\min_{\{\mathbf{Q}_0, \dots, \mathbf{Q}_{T-1}\}} \sum_{t=0}^{T-1} \alpha^t r_{\mathbf{Q}_t}$$

subject to a ‘hard’ constraint on the distortion as follows

$$\sum_{t=0}^{T-1} \alpha^t E_{\epsilon}[d(\mathbf{x}_t, \hat{\mathbf{x}}_t)] \leq D, t = 0, 1, \dots,$$

where D is a predetermined threshold, $E_{\epsilon}[\cdot]$ is the expectation of system state over the noise ϵ , and $d(\cdot, \cdot)$ is the distortion due to quantization error.

Then, as $T \rightarrow \infty$, we define the *Quantization Discount Rate Distortion* (QDRD) function as

$$R_T^{QDRD}(D) = \min_{F^{QDRD}} \sum_{t=0}^{\infty} \alpha^t \lceil \log_2 N_{\mathbf{Q}_t} \rceil, \quad (2.3)$$

where $F^{QDRD}(D) = \{ \{ \mathbf{Q}_0, \dots, \mathbf{Q}_\infty \}; \sum_{t=0}^{\infty} \alpha^t E_\epsilon[d(\mathbf{x}_t, \hat{\mathbf{x}}_t)] \leq D \}$.

To simplify the process here, we shall consider the Lagrange distortion measure and rewrite (2.3) as

$$\min_{\{\mathbf{Q}_0, \dots, \mathbf{Q}_\infty\}} \sum_{t=0}^{\infty} \alpha^t \{ \lceil \log_2 N_{\mathbf{Q}_t} \rceil + \lambda E_\epsilon[d(\mathbf{x}_t, \hat{\mathbf{x}}_t)] \} \quad (2.4)$$

where the weighting constant $\lambda > 0$ is the Lagrange multiplier.

Then the problem turns out to be finding the optimal strategy $\{\mathbf{Q}_0, \dots, \mathbf{Q}_\infty\}$ to minimize (2.4), where the constraint in (2.3) is ‘softened’. Note that the value of λ to equalize the hard and soft versions of the optimization in (2.3) and (2.4) is unknown and difficult to obtain. However, we can set a reasonable value for λ for a good tradeoff between the rate and the distortion.

If the structure of the quantizer $\mathbf{Q}_t$ is arbitrary, then we need to carry out the optimization in the function space, which requires variational analysis and is prohibitively difficult. Hence, for practical purposes, we assume a certain structure of the quantizer and consider $\mathbf{b}_t = [b_t^1, b_t^2, \dots, b_t^n]$, where b_t^i is the number of bits needed to quantize i -th component of the vector state. In this manner, we reduce the variable to be optimized to a limited set of integers. Then we have $\lceil \log_2 N_{\mathbf{Q}_t} \rceil = \log_2 \prod_i 2^{b_t^i} = \sum_i b_t^i$ and rewrite (2.4) as

$$\min_{\{\mathbf{b}_0, \dots, \mathbf{b}_\infty\}} \sum_{t=0}^{\infty} \alpha^t \left\{ \sum_i^n b_t^i + \lambda E_\epsilon[d(\mathbf{x}_t, \hat{\mathbf{x}}_t)] \right\}. \quad (2.5)$$

If we further assume that for every component $b_t^i = b_t$, then we have

$$\min_{\{b_0, \dots, b_\infty\}} \sum_{t=0}^{\infty} \alpha^t \{ n b_t + \lambda E_\epsilon[d(\mathbf{x}_t, \hat{\mathbf{x}}_t)] \}. \quad (2.6)$$

which finalizes our cost function.

2.2 Optimization of Quantizer

To solve the optimization problem in (2.4), we need to find a set $\{\mathbf{Q}_0, \dots, \mathbf{Q}_\infty\}$ to minimize the Lagrange distortion measure, which can be viewed as an infinite-horizon DP problem [5]. In this section, we will first propose the framework of DP for the optimization. Then, we will make the algorithm practical by exploiting ADP techniques. Finally we will evaluate the performance in a theoretical manner.

2.2.1 DP Framework

Bellman Equation

As $T \rightarrow \infty$, (2.4) can be solved by the Bellman equation given by

$$v^*(\mathbf{x}_t) = \min_{\mathbf{Q}_t} \{ \lceil \log_2 N_{\mathbf{Q}_t} \rceil + \lambda d(\mathbf{x}_t, \hat{\mathbf{x}}_t) + \alpha E_\epsilon[v^*(\mathbf{x}_{t+1})] \}, \quad (2.7)$$

where $\mathbf{x}_{t+1} = g(\mathbf{x}_t, \mathbf{u}_t, \epsilon_t)$ as shown in (2.1), the value function $v(\mathbf{x}_t)$ can be interpreted as the minimum cost starting from the current state using the optimal strategy $\mathbf{Q}_t$. Here, we can see the optimal strategy for calculating $\mathbf{Q}_t$, namely the policy functional $f(\cdot)$, does not depend on the time t as shown in DP.

For notational simplicity, we can make (2.7) compact by defining

$$\mathbf{T}(v^*(\mathbf{x}_t)) = \min_{\mathbf{Q}_t} \{ \lceil \log_2 N_{\mathbf{Q}_t} \rceil + \lambda d(\mathbf{x}_t, \hat{\mathbf{x}}_t) + \alpha E_\epsilon[v^*(\mathbf{x}_{t+1})] \} \quad (2.8)$$

which results in

$$v^*(\mathbf{x}_t) = \mathbf{T}(v^*(\mathbf{x}_t)). \quad (2.9)$$

Following the same philosophy, we define $\mathbf{T}^k(v^*(\mathbf{x}_t)) = \mathbf{T}(\mathbf{T}^{k-1}(v^*(\mathbf{x}_t)))$ and $\mathbf{T}^0(v^*(\mathbf{x}_t)) = v^*(\mathbf{x}_t)$. Once the unique bounded solution $v^*(\cdot)$ is found, the strategy $\mathbf{Q}_t$ can be obtained from

$$\mathbf{Q}_t^* = \arg \min_{\mathbf{Q}_t} \mathbf{T}(v^*(\mathbf{x}_t)). \quad (2.10)$$

Major Challenge

Although DP provides the optimal solution, if we look into (2.10), we find that the computation complexity is of the order $O(|\mathbf{x}_t| \times |\mathbf{Q}_t|)$, where $|\mathbf{x}_t|$ is the total number of states and $|\mathbf{Q}_t|$ denotes the number of all existing quantization strategies. Since the system state to be quantized is continuous in our setup, namely $|\mathbf{x}_t| = \infty$, a precise solution to (2.10) is impossible except for very special cases.

Thanks to the collocation method [41] in ADP, instead of taking account of all the states, we can obtain a suboptimal but feasible solution based on some predetermined sampled states. Hence, only finitely many states are required to satisfy the equation of $v^*(\cdot)$ in (2.9). Through solving the equation of finite states we can attain a suboptimal mapping from the state $\mathbf{x}_t$ to the quantization strategy $\mathbf{Q}_t$, where the sampled states are called the ‘collocation states’. In the following discussion, we will explain the ADP approach of collocation.

2.2.2 ADP and Collocation

As one effective method of ADP, the collocation method described in [41] is widely applied for solving continuous state Markov decision problems in economics and finance. The basic idea of the collocation approach is to approximate the value function $v^*(\mathbf{x}_t)$ by a linear combination of K pre-assigned basis functions $\phi_1, \phi_2, \dots, \phi_K$, as well as the coefficients $c_1, c_2, \dots, c_K$ for the basis functions. That is, we assume

$$v^*(\mathbf{x}_t) \approx v^a(\mathbf{x}_t, \mathbf{c}) = \sum_{j=1}^K c_j \phi_j(\mathbf{x}_t) \quad (2.11)$$

where $v^a(\mathbf{x}_t, \mathbf{c})$ is the approximate value function based on the system state $\mathbf{x}_t$ and the basis coefficients $\mathbf{c}$.

Meanwhile, rather than computing all the possible states in $\mathbb{R}^n$, the number of states to be considered in ADP becomes finite. Hence, the optimization of functions is converted to that of real numbers, which substantially simplifies the problem. Then K sample states will be selected, which are denoted by $\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_K]$. Thus, the collocation method changes

the Bellman equation into K nonlinear equations with unknowns $c_1, c_2, \dots, c_K$ as follows:

$$\begin{cases} v^a(\mathbf{s}_1, \mathbf{c}) = \sum_{j=1}^K c_j \phi_j(\mathbf{s}_1) \\ v^a(\mathbf{s}_2, \mathbf{c}) = \sum_{j=1}^K c_j \phi_j(\mathbf{s}_2) \\ \dots \\ v^a(\mathbf{s}_K, \mathbf{c}) = \sum_{j=1}^K c_j \phi_j(\mathbf{s}_K) \end{cases}$$

and the function $v^a(\mathbf{s}_i, \mathbf{c})$ satisfies

$$v^a(\mathbf{s}_i, \mathbf{c}) = \min_{\mathbf{Q}_i} \left\{ [\log_2 N_{\mathbf{Q}_i}] + \lambda d(\mathbf{s}_i, \hat{\mathbf{s}}_i) + \alpha E_\epsilon \left[\sum_{j=1}^K c_j \phi_j(\mathbf{s}'_{i+1}) \right] \right\} \quad (2.12)$$

where $\mathbf{s}'_{i+1} = g(\mathbf{s}_i, u_i, \epsilon_i)$. For simplicity it can be rewritten as

$$\mathbf{v}^a(\mathbf{c}) = \Phi \mathbf{c} \quad (2.13)$$

where $\mathbf{v}^a_i(\mathbf{c}) = v^a(\mathbf{s}_i, \mathbf{c})$, $\mathbf{c}_i = c_i$, and Φ is a matrix with elements $\Phi_{ij} = \phi_j(\mathbf{s}_i)$.

Since the collocation method with K as the number of the collocation node substitutes the infinite-dimensional functional equation problem(DP) into a finite-dimensional linear or nonlinear equation problem(ADP), the computation complexity of calculating the finite-dimensional equations(K equations in K unknowns) is easy to confirm, which is the order of $O(K \times |\mathbf{Q}_t|)$.

Data: Finding the coefficient vector $\mathbf{c}$
for $iter=1:maxiter$ **do**
 $\mathbf{oldc} = \mathbf{c};$
 $[\mathbf{v}^a(\mathbf{c}), \mathbf{Q}, \mathbf{v}^a(\mathbf{c})'] = v_{min}(\mathbf{s}, \mathbf{c});$
 $\mathbf{c} = \mathbf{oldc} - \frac{[\Phi \mathbf{c} - \mathbf{v}^a(\mathbf{c})]}{[\Phi - \mathbf{v}^a(\mathbf{c})']};$
 If $\|\mathbf{c} - \mathbf{oldc}\| < tol$, break, end;
end

Algorithm 1: Newton iteration for solving the ADP

Actually, given the collocation samples, the basis functions, and an initial setting for the value vector $\mathbf{v}^a(\mathbf{c})$, various numerical methods can be applied to solve (2.13) and find the

basic coefficient vector $\mathbf{c}$, which can be viewed as a mapping from the solution of the finite-dimensional nonlinear equations back to the solution of infinite-dimensional DP. Standard linear or nonlinear equation schemes, such as the Newton method, can be employed to find this mapping, the computation complexity of which depends on the features of equations (the basis functions ϕ , the number of collocation nodes K), the initial states, which is not the focus of our work but has been discussed well in [10]. Here we employ the iterative Newton method in Algorithm 1 to find the solution $\mathbf{c}$, whose updating rule is

$$\mathbf{c} \leftarrow \mathbf{c} - [\Phi - \mathbf{v}^a(\mathbf{c})']^{-1}[\Phi\mathbf{c} - \mathbf{v}^a(\mathbf{c})]. \quad (2.14)$$

where $\mathbf{v}^a(\mathbf{c})'_{ij} = \frac{\partial v^a(\mathbf{s}_i, \mathbf{c})}{\partial c_j} = \alpha E_\epsilon[\phi_j(\mathbf{s}'_{i+1})]$ as $\mathbf{v}^a(\mathbf{c})'$ is the $K \times K$ Jacobian matrix of $\mathbf{v}^a(\mathbf{c})$. Here, tol and $maxiter$ are the pre-determined iteration control parameters, which specify the convergence tolerance and the maximum number of iterations. v_{min} is the optimization problem embedded in the Bellman equations, which will be defined later in Algorithm 2

However, the exact solution for $\mathbf{c}$ might not be available, especially when there are too many unknowns. Hence, it will be of great importance to evaluate the performance of the solution $\mathbf{c}$. We will analyze the performance of the collocation method subsequently.

2.2.3 Performance of ADP

Even though we can obtain the solution for the ADP, it is important to evaluate the gap between the solutions of ADP and DP, such that we can know the performance loss incurred by the simplification. To this end, we need the following lemma (Proposition 4, Sec. 5.2 [5]).

Proposition 2.1. *For every vector $v^*(\mathbf{x}_t)$, system state $\mathbf{x}_t$, and iteration time k , a bound between the approximate value $v^a(\mathbf{c}, \mathbf{x}_t)$ in ADP and the true solution $v^*(\mathbf{x}_t)$ to DP is given by*

$$\mathbf{T}^k(v^a(\mathbf{x}_t, \mathbf{c})) + \gamma_k \leq v^*(\mathbf{x}_t) \leq \mathbf{T}^k(v^a(\mathbf{x}_t, \mathbf{c})) + \bar{\gamma}_k \quad (2.15)$$

where $\gamma_k = \frac{\alpha}{1-\alpha} \min_{\mathbf{x}_t} [\mathbf{T}^k(v^a(\mathbf{x}_t, \mathbf{c})) - \mathbf{T}^{k-1}(v^a(\mathbf{x}_t, \mathbf{c}))]$ and $\bar{\gamma}_k = \frac{\alpha}{1-\alpha} \max_{\mathbf{x}_t} [\mathbf{T}^k(v^a(\mathbf{x}_t, \mathbf{c})) - \mathbf{T}^{k-1}(v^a(\mathbf{x}_t, \mathbf{c}))]$.

To fully exploit the error bound, we have to evaluate all the $\mathbf{x}_t$ to find $\max_{\mathbf{x}_t} \gamma_k$ and $\min_{\mathbf{x}_t} \gamma_k$, which is prohibitively difficult to attain for the continuous state $\mathbf{x}_t$ and unknown value function. Hence, instead of considering continuous state, in this dissertation we consider the system where only finitely many sampled states are available. Then we define the absolute divergence of the solution for the ADP as

$$\begin{aligned} \mathbf{R}(\mathbf{s}_i, \Phi) &= \mathbf{T}^k(v^a(\mathbf{s}_i, \mathbf{c})) - \mathbf{T}^{k-1}(v^a(\mathbf{s}_i, \mathbf{c})) \\ &= \sum_{j=1}^K c_j \phi_j(\mathbf{s}_i) - \min_{\mathbf{Q}_i} \{ \lceil \log_2 N_{\mathbf{Q}_i} \rceil \\ &\quad + \lambda d(\mathbf{s}_i, \hat{\mathbf{s}}_i) + \alpha E_\epsilon \left[\sum_{j=1}^K c_j \phi_j(\mathbf{s}'_{i+1}) \right] \}, \end{aligned} \quad (2.16)$$

which can be compactly expressed as

$$\mathbf{R}(\mathbf{s}, \Phi) = \Phi \mathbf{c} - \mathbf{v}^a(\mathbf{c}). \quad (2.17)$$

Based on this definition, we propose the following sufficient condition for the solution of ADP in finite sample state system.

Theorem 2.2. *In ADP with finite states $\mathbf{S} = \{\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_K\}$, if $\mathbf{R}(\mathbf{s}_i, \Phi) \rightarrow 0$, then an approximate function $\mathbf{v}^a(\mathbf{c})$ can be obtained such that $\mathbf{v}^a(\mathbf{c}) \rightarrow \mathbf{v}^*$ for the sampled states.*

Proof. We can rewrite the (2.15) as

$$\begin{aligned} &\mathbf{T}^k(v^a(\mathbf{s}_i, \mathbf{c})) + \min_{\mathbf{s}_i \in \mathbf{S}} \mathbf{R}(\mathbf{s}_i, \Phi) \\ &\leq v^*(\mathbf{s}_i) \leq \mathbf{T}^k(v^a(\mathbf{s}_i, \mathbf{c})) + \max_{\mathbf{s}_i \in \mathbf{S}} \mathbf{R}(\mathbf{s}_i, \Phi). \end{aligned} \quad (2.18)$$

Since $0 \leq |\mathbf{R}(\mathbf{s}_i, \Phi)| \rightarrow 0$, we have

$$\begin{aligned} \mathbf{T}^k(v^a(\mathbf{s}_i, \mathbf{c})) \leftarrow \mathbf{T}^k(v^a(\mathbf{s}_i, \mathbf{c})) + \min_{\mathbf{s}_i \in \mathbf{S}} \mathbf{R}(\mathbf{s}_i, \Phi) &\leq v^*(\mathbf{s}_i) \leq \\ \mathbf{T}^k(v^a(\mathbf{s}_i, \mathbf{c})) + \max_{\mathbf{s}_i \in \mathbf{S}} \mathbf{R}(\mathbf{s}_i, \Phi) &\rightarrow \mathbf{T}^k(v^a(\mathbf{s}_i, \mathbf{c})), \end{aligned} \quad (2.19)$$

which results in

$$v^*(\mathbf{s}_i) \rightarrow \mathbf{T}^k(v^a(\mathbf{s}_i, \mathbf{c})). \quad (2.20)$$

This concludes the proof. $\square$

Remark 1. *This theorem provides a tool to verify the performance of ADP, since the term $\mathbf{R}(\mathbf{s}_i, \Phi)$ can be evaluated numerically.*

2.2.4 Summary of Algorithms

Now we summarize the proposed algorithms for computing the quantization strategy. First, we apply the Newton iteration to calculate (2.14) using Algorithm 1. Then we define a $v_{\min}$ function to solve the Bellman equation (2.7) based on Algorithm 2, where $r(:, i)$ denotes the reward of i -th strategy (e.g. $\lceil \log_2 N_{\mathbf{Q}_i} \rceil + \lambda \|\hat{\mathbf{s}} - \mathbf{s}\|^2$) and $Ev(:, i)$ denotes the expected future reward of i -th strategy (e.g. $E_\epsilon[\sum_{j=1}^K c_j \phi_j(g(\mathbf{s}, u, \epsilon))]$), $g(\mathbf{s}, u, \epsilon)$ is the defined dynamics of the system state in (2.1).

```

Data: Function  $[\mathbf{v}^a(\mathbf{c}), \mathbf{Q}, \mathbf{v}^a(\mathbf{c})'] = v_{\min}(\mathbf{s}, \mathbf{c})$ 
for  $i = 1 : 1 : |\mathbf{Q}|$  do
     $r(:, i) = \lceil \log_2 N_{\mathbf{Q}_i} \rceil + \lambda \|\hat{\mathbf{s}} - \mathbf{s}\|^2$ ;
     $Ev(:, i) = E_\epsilon[\sum_{j=1}^K c_j \phi_j(g(\mathbf{s}, u, \epsilon))]$ ;
end
 $[\mathbf{v}^a(\mathbf{c}), \mathbf{Q}] = \min_{\mathbf{Q}} (r(:, :) + \alpha Ev(:, :))$ ;
 $\mathbf{v}^a(\mathbf{c})' = \alpha E_\epsilon[\phi_j(g(\mathbf{s}, u, \epsilon))]$ ;

```

Algorithm 2: Solving the Bellman equation

2.3 Experiment and Numerical Results

In this section, we introduce two CPSs, provide numerical results for the performance evaluation and finally propose the potential applications in mobile/vehicle communication.

2.3.1 Two CPSs

Other than just theoretical derivation, we apply the proposed ‘state aware’ encoder and decoder in two physical dynamics, namely a general continuous-state Markov dynamic system and a small truck balancing system, in order to verify the validity of the proposed scheme.

Continuous-state Markov Dynamical System

As illustrated in Fig. 1.1, here we assume that the sensor is monitoring a continuous-state Markov dynamical system, where the physical dynamics is given by

$$\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \mathbf{B}h(\hat{\mathbf{x}}_t) + \mathbf{C}\epsilon_t, \quad (2.21)$$

where $h(\hat{\mathbf{x}}_t) = \mathbf{F}\hat{\mathbf{x}}_t$ (where $\mathbf{F}$ is the feedback gain matrix), and ϵ_t has a standard Gaussian distribution, namely $\epsilon_t \sim \mathcal{N}(0, \mathbf{I})$. Although the system state is boundless in the theoretical meaning, in practical applications the system state is usually bounded by $\mathbf{x}_{\min} = (x_{\min}^1, \dots, x_{\min}^n)$ and $\mathbf{x}_{\max} = (x_{\max}^1, \dots, x_{\max}^n)$; i.e., the i -th dimension of the system state is confined by $[x_{\min}^i, x_{\max}^i]$, $1 \leq i \leq n$.

As stated, in order to simplify the problem, we choose the Chebychev nodes in [41] as the sampled system states by interpolating the following function:

$$\mathbf{s}_i = \frac{\mathbf{x}_{\max} + \mathbf{x}_{\min}}{2} + \frac{\mathbf{x}_{\max} - \mathbf{x}_{\min}}{2} \cos\left(\frac{K - i + 0.5}{K}\pi\right), \quad (2.22)$$

where $i = 1, 2, \dots, K$. Two strategies are available for the quantizer, where $\mathbf{Q}_t \in \mathbf{Q} = \{b_t = 4, b_t = 5\}$; i.e., we employ 4 bits or 5 bits to represent each dimension of the system state with $N_{\mathbf{Q}_t} = 2^4$ or 2^5 intervals.

Two categories of quantizers will be considered on both scalar and vector cases, where the scalar denotes the system state variable with one-dimension and the vector denotes the system state with n -dimension ($n \geq 2$) :

- **Uniform Quantizer:** We divide the interval $[x_{\min}, x_{\max}]$ uniformly in each dimension. For example, for the scalar case, the quantization step is defined as

$$\begin{aligned} \text{if } x_t \in \left[x_{\min} + \frac{j(x_{\max} - x_{\min})}{N_{\mathbf{Q}_t}}, x_{\min} + \frac{(j+1)(x_{\max} - x_{\min})}{N_{\mathbf{Q}_t}} \right] \\ \text{then } q_t = \mathbf{Q}_t(x_t) = j \end{aligned} \quad (2.23)$$

where $j \in \{0, \dots, N_{\mathbf{Q}_t} - 1\}$; for the vector case, we carry out the scalar quantization for each dimension.

- **Lloyd Quantizer:** We divide the quantization regions using the Lloyd's algorithm in [36] based on a certain training set. The training dataset could be generated by the estimated distribution of the system state. To simplify the simulation, we assume that for the scalar case, the training dataset is Gaussian distributed with the same expectation and variance as the samples ones of $\mathbf{x}_t$. For the vector case, the training dataset is multivariate Gaussian distribution with the same expectation and covariance matrix of $\mathbf{x}_t$.

Truck Balancing System

A more practical truck balancing system is developed for measuring the performance. As illustrated in Fig. 2.4, the small truck is programmed with two options, either running toward the wall or away from the wall, with a distance ultrasonic sensor installed in the front. The ultrasonic sensor senses the distance and sends the sensed distance data to the controller. The controller will receive the distance data, compute the action and control the truck to keep a fixed distance (which is set to 50cm) from the wall. Here both the truck and the controller are implemented on MSP430. The dynamics of the balancing truck is formulated as

$$d_t = d_{t-1} + (v_t * \Delta t)u_t + w_t \quad (2.24)$$

where d_t is the distance between the wall and the front of the small truck (which can be considered as the system state), v_t is the speed of the truck, which is driven and adjusted by pulse width modulation (PWM), Δt is the time interval between every two control actions, and w_t is the random perturbation, which is assumed to be Gaussian.

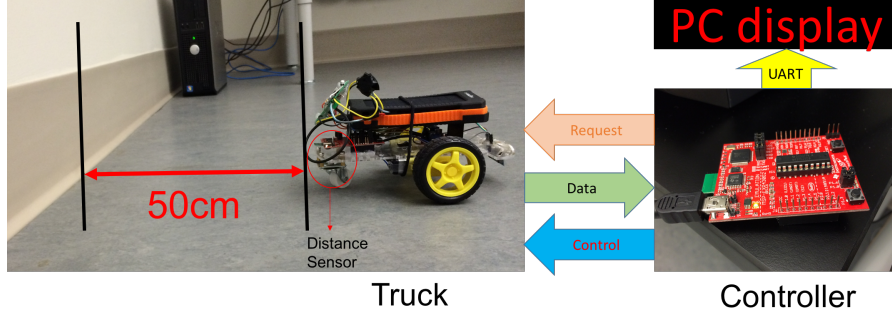


Figure 2.4: A small truck balancing system

The controller can change the truck running direction by determining u_t :

$$\begin{cases} \text{Run away from the wall: } u_t = 1 & \text{if } d_t \leq 50\text{cm} \\ \text{Run toward the wall: } u_t = -1 & \text{otherwise} \end{cases} \quad (2.25)$$

Now we provide numerical results for the above practical systems, based on the cost function in (2.6), where $\lceil \log_2(N_{\mathbf{Q}_t}) \rceil = nb_t$ and $d(\mathbf{x}_t, \hat{\mathbf{x}}_t) = \|\hat{\mathbf{x}}_t - \mathbf{x}_t\|^2$, $\lambda = 1300$ and $\alpha = 0.9$.

Continuous-state Markov Dynamical System

We first consider scalar dynamics in (2.21), where the parameters are set as $A = -0.5$, $F = 1$, $C = 0.05$, $B = 1$, $x_{\min} = 0$ and $x_{\max} = 1$. Fig. 2.5 shows the selective strategy $\mathbf{Q}_i \in \{b_i = 4, b_i = 5\}$ for both the uniform quantizer and the Lloyd quantizer with the same expectation as that of x_t but different variances. The corresponding distortion and divergence are shown in Fig. 2.6 and Fig. 2.7, respectively when $K = 30$. Here, σ^2 is the variance of Gaussian distribution used in the training dataset. Finally, a summary of total cost for both uniform quantizer and Lloyd quantizer with different variances is shown in Fig. 2.8. As shown in Fig. 2.8, Lloyd quantizer can obtain better performance when both the quantizer and decoder have priori information about the system state (e.g. smaller variance $\sigma^2 = 0.1$). However, the uniform quantizer can always attain good approximate results with the smallest divergence as shown in Fig. 2.7, which is straightforward for its linear properties.

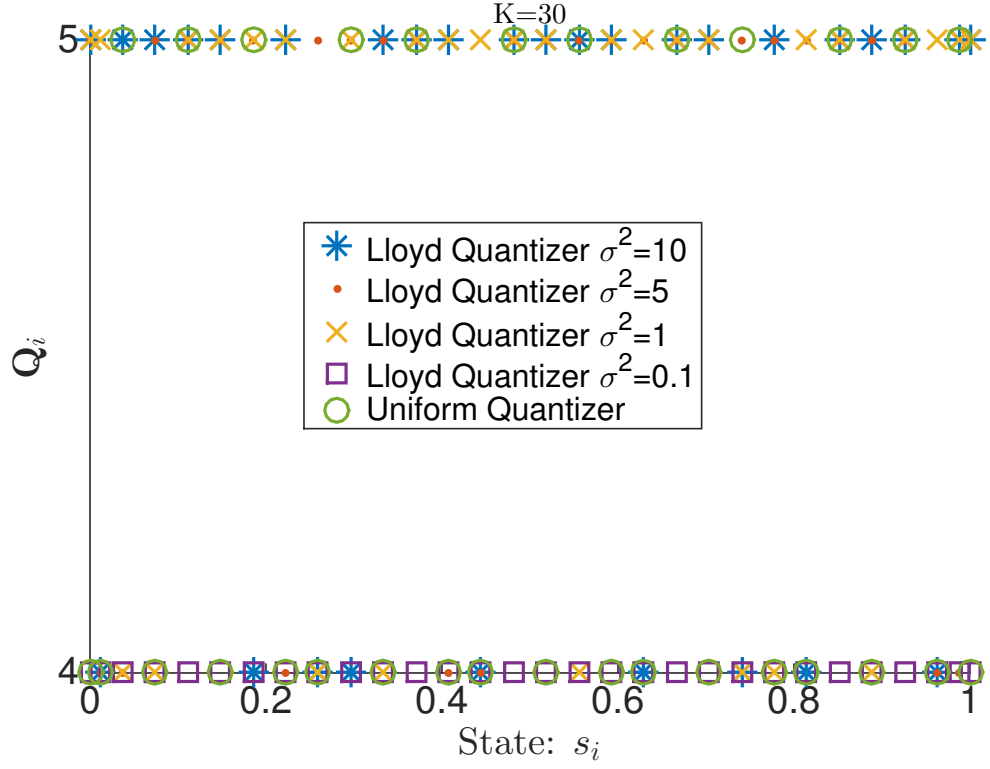


Figure 2.5: Quantization strategy for scalar case of Markov dynamical system

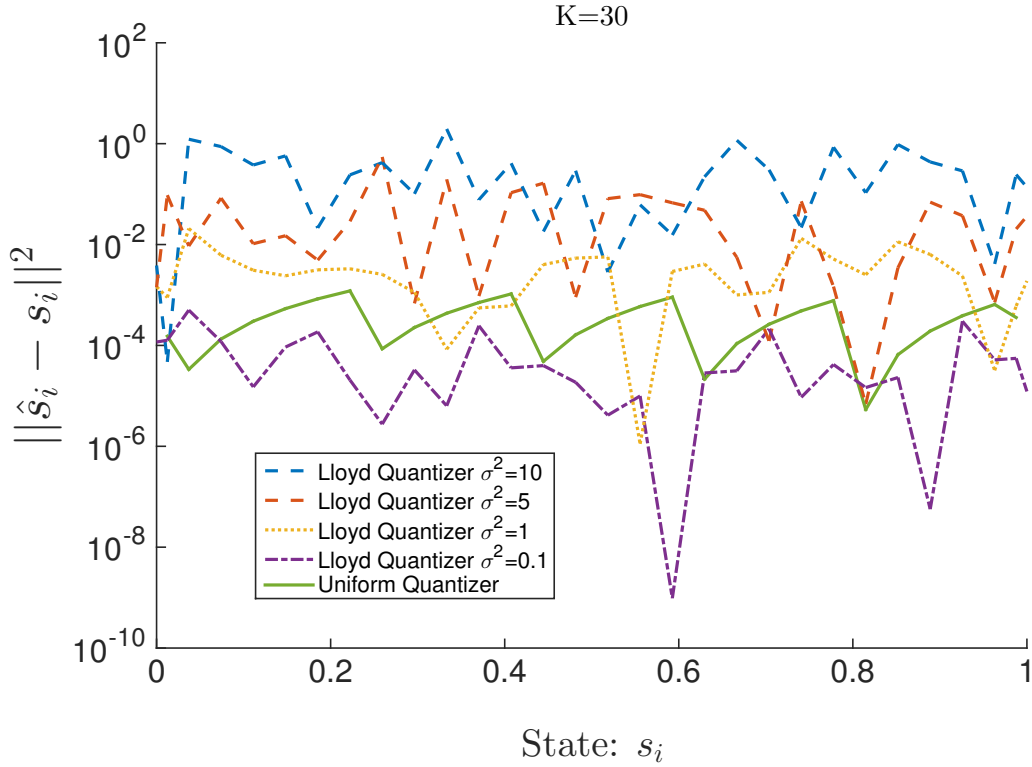


Figure 2.6: Quantization distortion for scalar case of Markov dynamical system

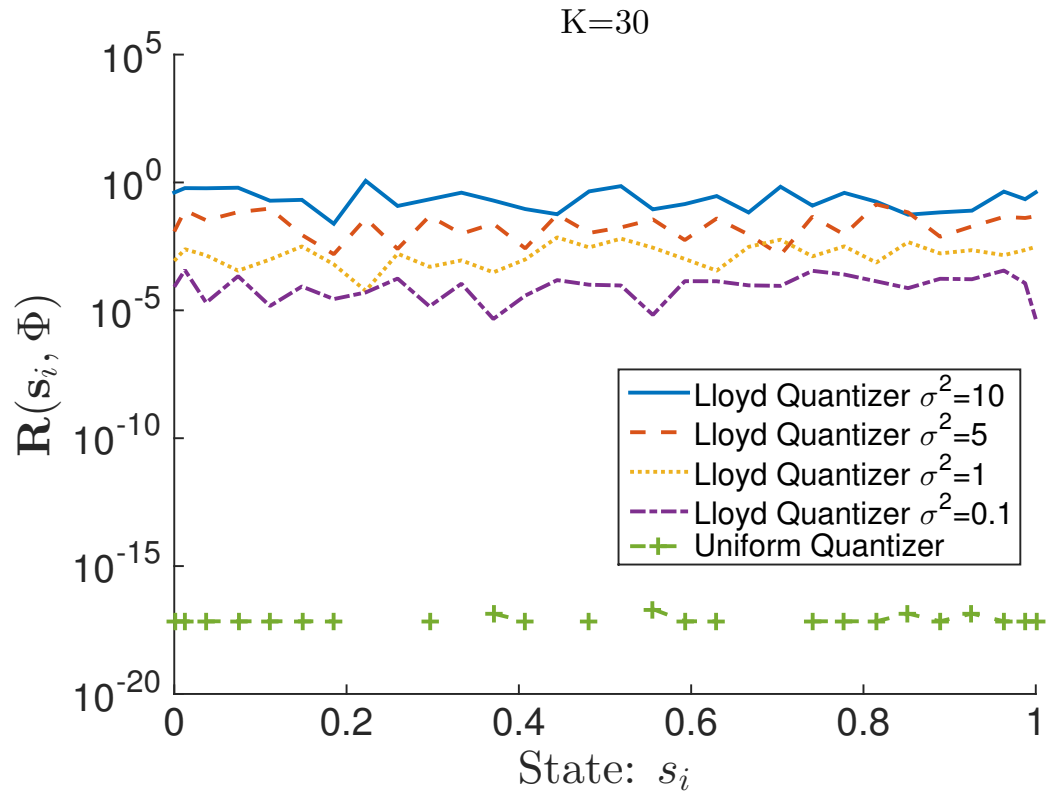


Figure 2.7: Absolute value of $\mathbf{R}(s_i, \Phi)$ for scalar case of Markov dynamical system

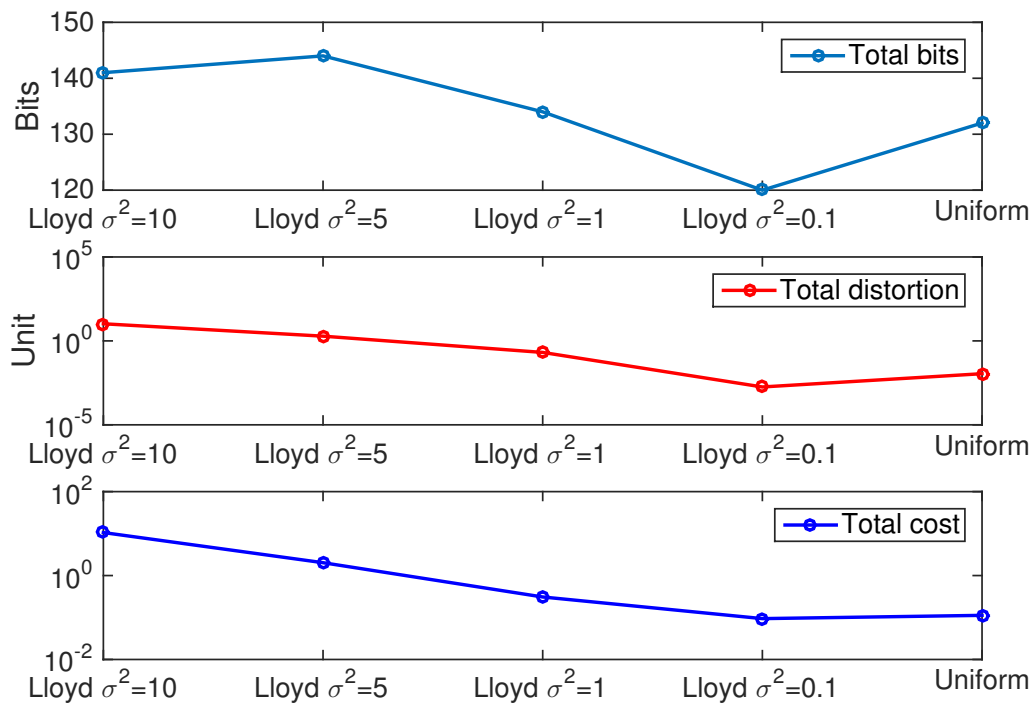


Figure 2.8: Comparison of overall performance for scalar case of Markov dynamical system

Then we enumerate all the combinations of strategies for the sampled state $\mathbf{s}_i$. To reduce the computation, we consider $K = 10$, denote all the combinations as $\{\mathbf{b}^j\}_{j=1,\dots,2^K}$, where $\mathbf{b}^j = (b_1^j, \dots, b_K^j)$, $b_i^j \in \{b_i = 4, b_i = 5\}$ and compare them with the obtained solution by using the collocation method. The cumulative distribution functions (CDFs) of $\left\{ \frac{[\sum_{i=1}^K b_i^j + \|\hat{\mathbf{s}}_i^j - \mathbf{s}_i\|^2]}{[\sum_{i=1}^K b_i^{opt} + \|\hat{\mathbf{s}}_i^{opt} - \mathbf{s}_i\|^2]} \right\}$, $\left\{ \frac{\sum_{i=1}^K b_i^j}{\sum_{i=1}^K b_i^{opt}} \right\}$ and $\left\{ \frac{\sum_{i=1}^K \|\hat{\mathbf{s}}_i^j - \mathbf{s}_i\|^2}{\sum_{i=1}^K \|\hat{\mathbf{s}}_i^{opt} - \mathbf{s}_i\|^2} \right\}$ are shown in Fig. 2.9 with each combination as one symbol.

We can see that the rate (the x-axis in Fig. 2.9) of uniform quantizer and Lloyd quantizer with $\sigma^2 = 0.1$ achieve the smallest cost. The reason for this might be insufficient approximation to the value function as shown in Fig. 2.7, which leads to the error in finding the optimal strategy. This phenomenon emphasizes the vital role of approximation in the proposed scheme.

For the vector state case, we consider vector system states in an n -dimensional sample space $\mathbf{s}_i = [s_i(1), s_i(2), \dots, s_i(n)]$. We use the parameters $\mathbf{A} = -0.5\mathbf{I}$, $\mathbf{F} = \mathbf{I}$, $\mathbf{C} = 0.05\mathbf{I}$, $B = 1$, $\mathbf{x}_{min} = \mathbf{0}$ and $\mathbf{x}_{max} = \mathbf{1}$, where $\mathbf{I}$ is an $n \times n$ identity matrix, $\mathbf{0}$ is an $n \times 1$ all zero vector and $\mathbf{1}$ an $n \times 1$ all one vector. We set $K = 30^n$ collocation grid for uniform quantizer and to simplify the computation complex by using a $K = 5^n$ collocation grid for Lloyd quantizer.

We simulated the cases when $n = 2$ for the uniform quantizer as shown in Fig. 2.10 and Fig. 2.11, and similarly $n = 2$ for Lloyd quantizer in Fig. 2.12 and Fig. 2.13. Both simulation results for selective optimal quantization strategy are similar to the scalar case in each dimension. However, the divergence for both cases is much bigger compared to Fig. 2.7, which means that the approximations for higher dimensions are much less reliable. The reason for this might be that the distortion factor of pre-assigned basis functions increases linearly with the dimensions of the features [21].

Furthermore, we also run simulations for higher dimensions as $n = 3, 4, 5$. The average bits and distortion per dimension for one system state are shown in Fig. 2.14 and Fig. 2.15, respectively, which show similar results as scalar cases. However, the computation complexity becomes higher as the dimension increases. Hence, in practical applications, the scalar quantizer is strongly recommended when the correlation of each dimension is low. To this end, the elapsed CPU time for scalar case with different values of K is shown in Fig. 2.16.

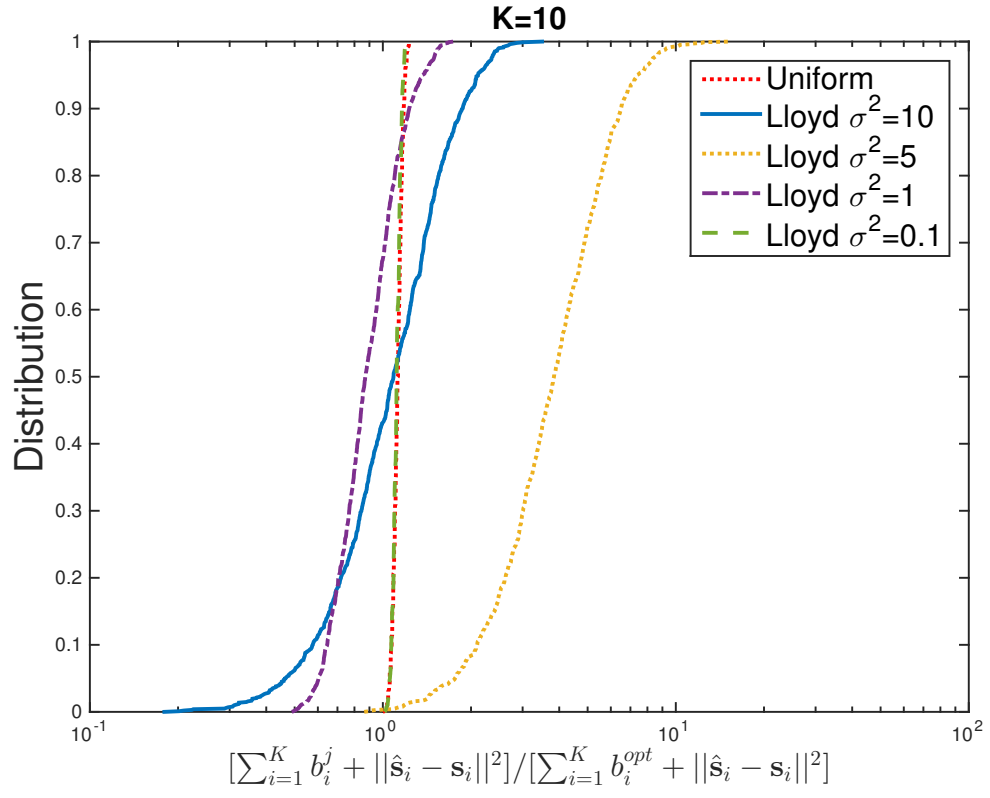


Figure 2.9: Distribution of total cost for uniform and Lloyd quantizer

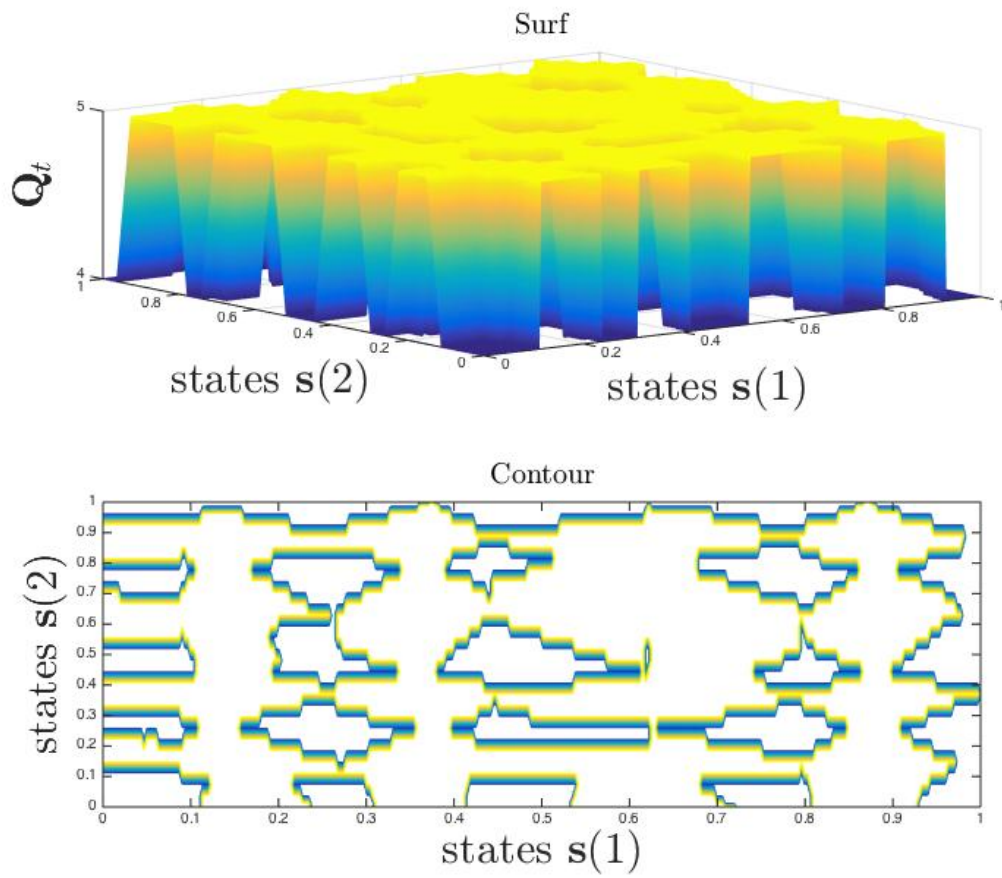


Figure 2.10: Selected strategy Q_t for uniform quantizer

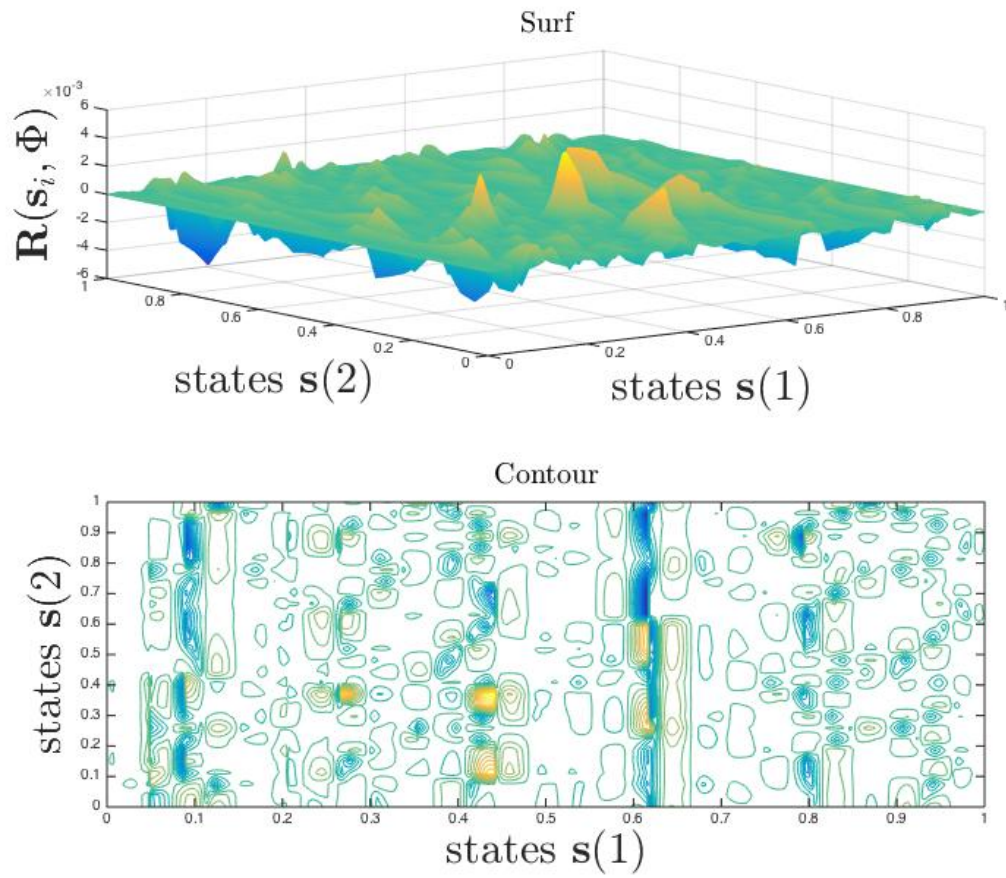


Figure 2.11: Corresponding $R(s_i, \Phi)$ for uniform quantizer

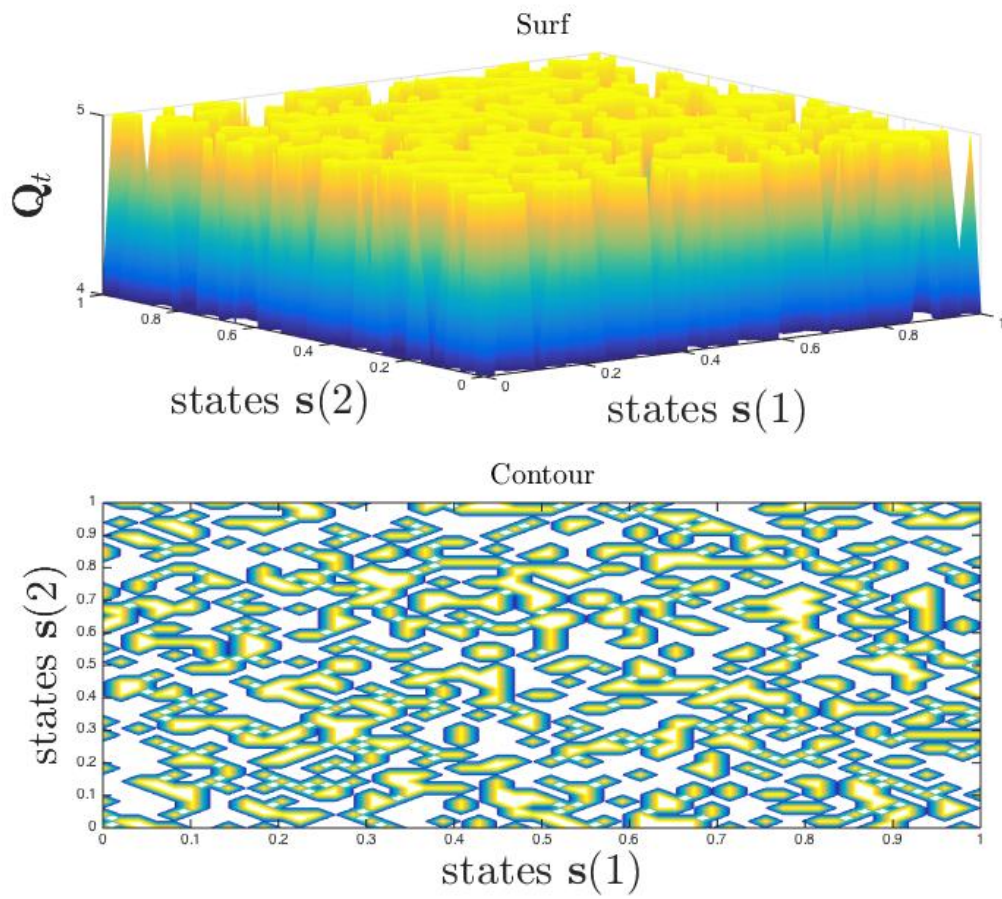


Figure 2.12: Selected strategy Q_t for Lloyd quantizer

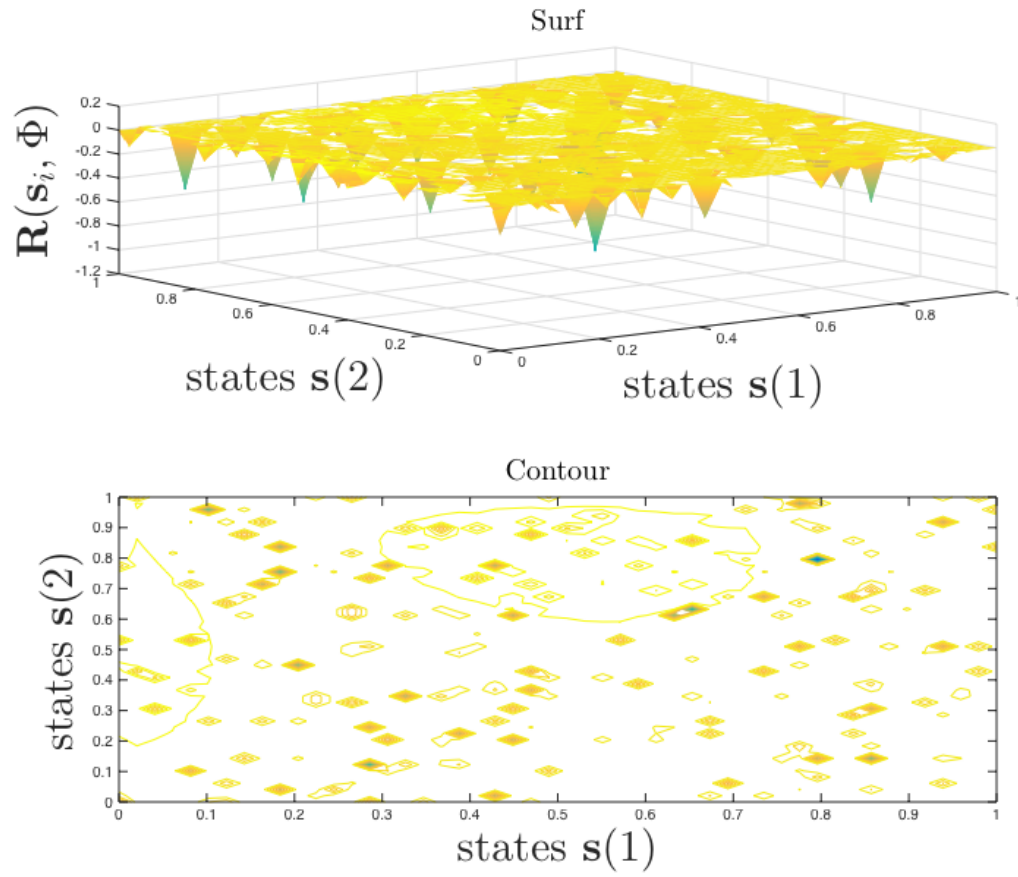


Figure 2.13: Corresponding $R(s_i, \Phi)$ for Lloyd quantizer

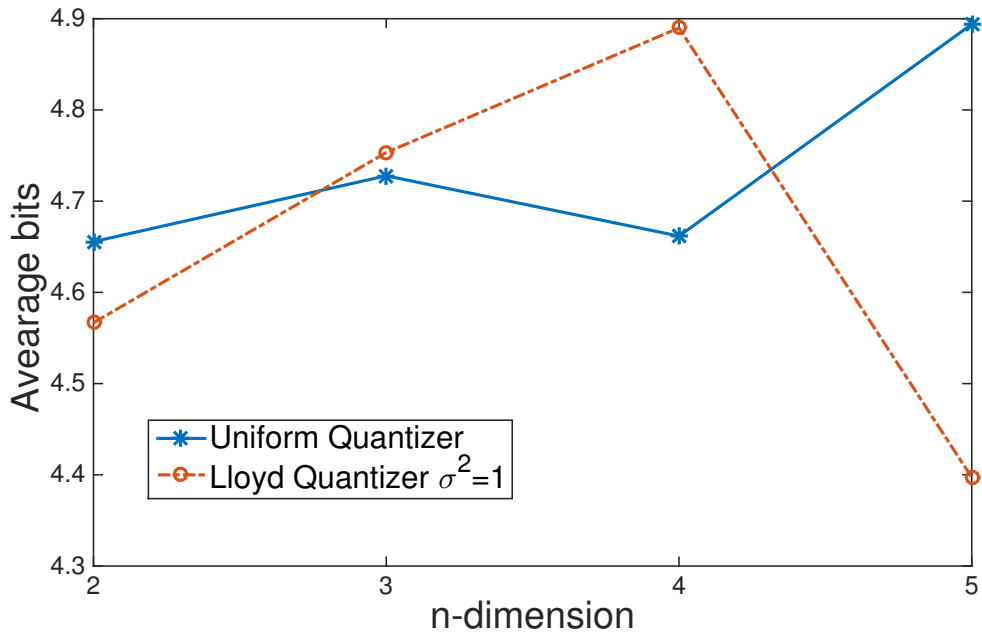


Figure 2.14: Average bits per system state in every dimension

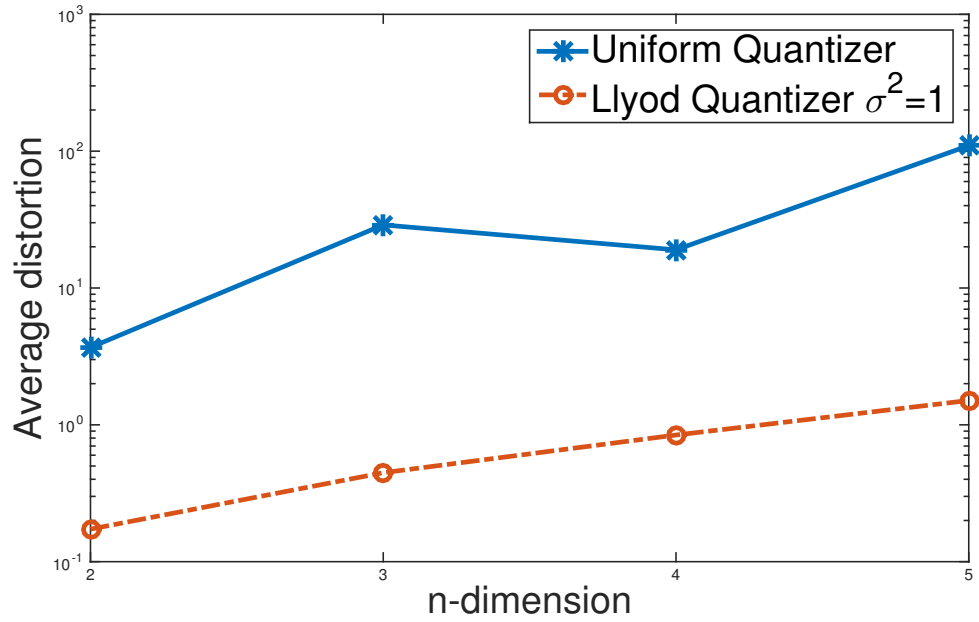


Figure 2.15: Average distortion per system state in every dimension

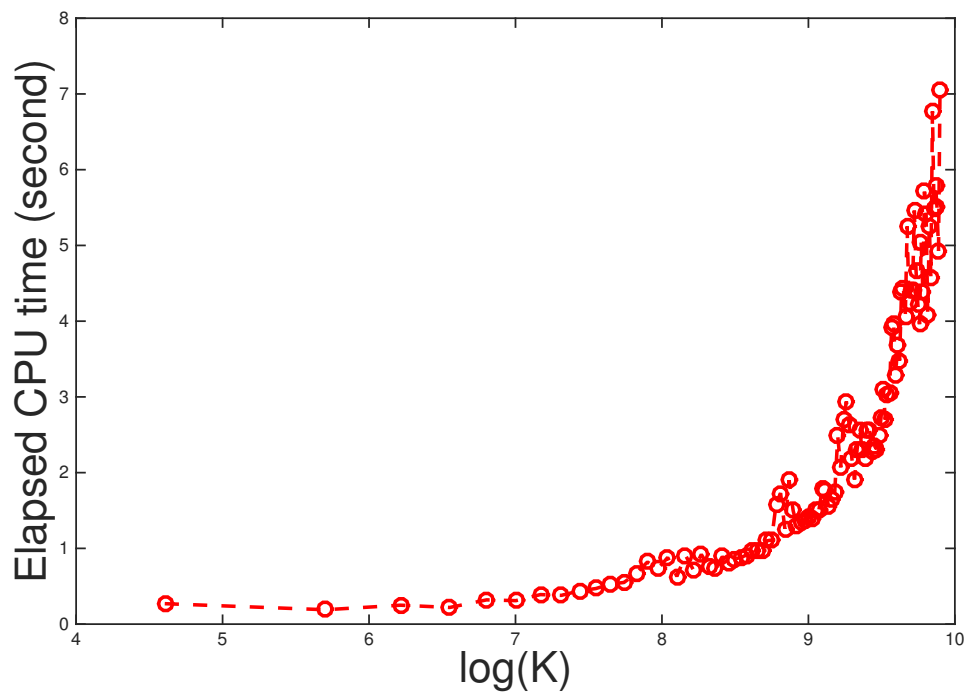


Figure 2.16: Elapsed CPU time for different values of K

Truck balancing system

We programmed the MSP430 for the controller in Fig. 2.4 to request for the distance information every $40ms$ from the truck. Upon receiving the distance information, the controller immediately sends out the control action in (2.25) to adjust the running direction of the truck. The real-time sensing distance, error and its average cost for each transmission based on (2.6) are shown in Fig. 2.17, Fig. 2.18 and Fig. 2.19, respectively.

As we can see in Fig. 2.17, at the macroscopic level the truck is successfully controlled running around $50cm$ even though there are still some collisions with the wall, and the average cost of the system is much lower than unadaptive quantizer shown in Fig. 2.19, which verifies our proposed scheme. An animation is also shown in Youtube via “<https://youtu.be/yyZNEBzX2kg>”.

2.3.2 Applications in mobile/vehicle communication

Our proposed scheme can also be applied to broader CPSs applications, such as the vehicle CPSs [65]. As one of the vital parts of the vehicle CPSs, the GPS navigation system has been widely used by drivers for efficient driving in the form of dedicated navigators or the smartphone navigator Apps. However, how to design of the communication among these devices to support good quality of service is still open in vehicle CPSs.

As shown in [25], the mobile cloud computing has been introduced to help the mobile or vehicular learning the around environments targeting at driving safety, navigation efficiency, and location-based services in road networks. And in order to to maintain good performance on sensing the road traffic conditions, vehicle traffic statistic (e.g., vehicle density and average speed) per road segment or intersection, and the navigation paths(called vehicle trajectories) of vehicles moving in road networks, the Traffic Control Center (TCC) has deployed multiple Road-Side Units(RSU) as wireless gateway to interconnect with the vehicular for either data forwarding to/from the vehicular cloud.

To reduce the total communication cost, the proposed scheme in our dissertation can be well applied to communication between the vehicular and the vehicular cloud, which can largely improve the efficacy and the accuracy of the information sharing. For instance, the

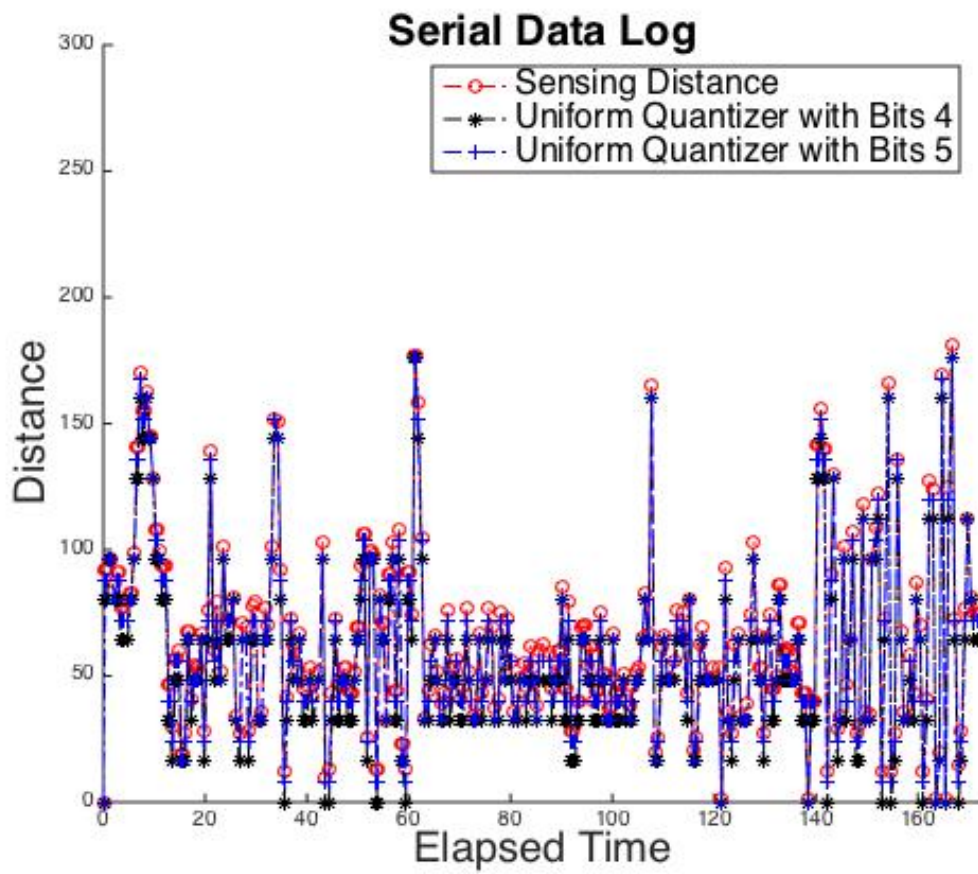


Figure 2.17: Sensing distance

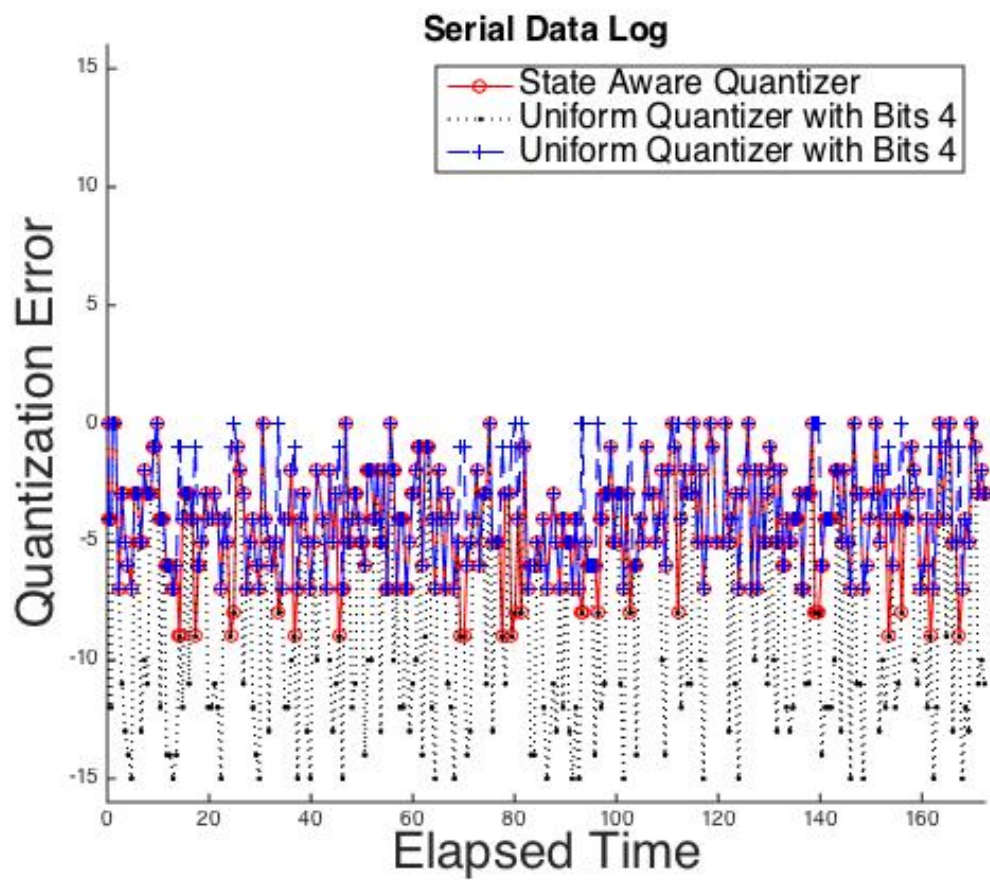


Figure 2.18: Quantization error

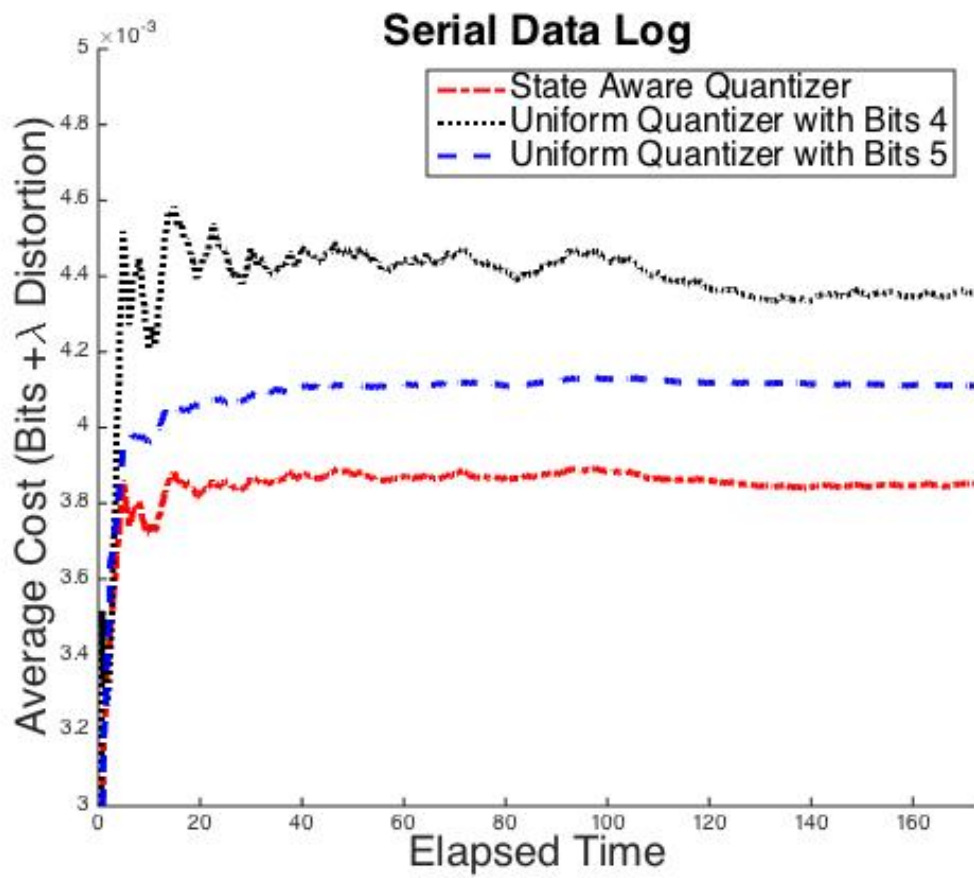


Figure 2.19: Overall average cost

dedicated navigators or the smartphone navigator Apps could collaborate with the RSU, and quantize their sensed road or traffic information by using the ‘state aware’ quantizer. Since no additional information needed to be transmitted to synchronize the navigator and the RSU, the power efficiency of battery-limited devices could also be improved.

Chapter 3

Dynamic Adaptive Distributed Source Coding (Multiple Sensors)

In this chapter, we will go through the multiple sensors case, specifically distributed source coding. In the coming sections, we first illustrate the system model. Next, we will give an incentive strategy to realize the distributed source coding, where we implement the predictive methods for system dynamics, which are deployed as the estimated feedback from the controller to sensors. Then, we will go to solve the distributed source coding by leveraging the reinforcement Q-learning. To clarify, we also use sensors as encoder and controller as decoder in the sequential sections. Some key notations are listed in Table [3.1](#).

3.1 System Model

First, we review a generalized system model [\[28\]](#) of CPSs, which contains the system dynamics and communication in settings of multiple sensors.

3.1.1 System Dynamics

We assume that the system dynamics is discrete-time, and the dimension of the system state is N with system state as $\mathbf{x}(t)$ at time t . The evolution law of $\mathbf{x}(t)$ is given as

$$\mathbf{x}(t+1) = f(\mathbf{x}(t), \mathbf{u}(t), \mathbf{n}(t)) \tag{3.1}$$

Table 3.1: Key notations of the Chapter 3

Symbols	Key units in CPS	Notes (introduced in equation)
$\mathbf{x}(t)$	System state vector	$\mathbb{R}^n$, (3.1)
$\mathbf{u}(t)$	System control input	$\mathbb{R}^n$, (3.1)
$\mathbf{n}(t)$	System noise	$\mathbb{R}^n$, (3.1)
$f(\cdot)$	State transition function	(3.1)
$g_i(\cdot)$	Observation transition function	(3.2)
$\mathbf{y}_i(t)$	Observations	$\mathbb{R}^n$, (3.2)
$g_i^{-1}(\cdot)$	Invertible transition function	(3.2)
$\mathbf{w}(t)$	Observation noise	$\mathbb{R}^n$, (3.2)
$h_{ij}(\cdot)$	State map function	(3.5)
$q(t)$	Output of quantizer	$\mathbb{R}$, (3.8)
$\mathbf{A}$	State transition matrix	(3.6)
$\mathbf{B}$	Control transition matrix	(3.6)
$\mathbf{C}$	Observation transition matrix	(3.6)
$\mathbf{1}$	Indicator function	(3.8)
$\mathbf{x}(t t)$	Estimated current system state	(3.32)
$\mathbf{x}(t+1 t)$	Estimated future system state	(3.32)
$\mathbf{K}(t)$	Kalman Filtering Matrix	(3.33)
Σ	Lattice	(3.14)
$\mathbf{G}$	Generate matrix	(3.14)
$V[\mathbf{x}(t)]$	Distortion function of $\mathbf{x}(t)$	(3.21)

where we denote $\mathbf{u}(t)$ as the control action and $\mathbf{n}(t)$ as noise at time slot t .

To start with, we simply consider two sensors with their observations being denoted by $\mathbf{y}_i, i = 1, 2$ and their dimensions are denoted by $N_i, i = 1, 2$, respectively. In this case, the observations are determined by the system dynamics and the observation noise; i.e.,

$$\mathbf{y}_i(t) = g_i(\mathbf{x}(t), \mathbf{w}_i(t)), i = 1, 2 \quad (3.2)$$

Next, the function g_i is assumed to be invertible, by fixing the noise $\mathbf{w}_i(t)$; i.e.,

$$\mathbf{x}(t) = g_i^{-1}(\mathbf{y}_i(t), \mathbf{w}_i(t)) \quad (3.3)$$

Then, the relationship of two observations can be written as:

$$\mathbf{y}_2(t) = g_2(g_1^{-1}(\mathbf{y}_1(t), \mathbf{w}_1(t)), \mathbf{w}_2(t)) \quad (3.4)$$

if a function h_{ij} and noise $\mathbf{w}_{ij}$ exist (as a function of $\mathbf{w}_1$ and $\mathbf{w}_2$) such that

$$\mathbf{y}_j = h_{ij}(\mathbf{y}_i) + \mathbf{w}_{ji} \quad (3.5)$$

which means that there is no nonlinear coupling between $\mathbf{y}_i$ and the noise, and $\mathbf{y}_1$ and $\mathbf{y}_2$ is separable.

One example of the linear system dynamics can be written as

$$\begin{cases} \mathbf{x}(t+1) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{n}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{w}(t) \end{cases} \quad (3.6)$$

where $\mathbf{y}(t) = (\mathbf{y}_1^T(t), \mathbf{y}_2^T(t))^T$ and $\mathbf{w}(t) = (\mathbf{w}_1^T(t), \mathbf{w}_2^T(t))^T$ and $\mathbf{C} = (\mathbf{C}_1^T, \mathbf{C}_2^T)^T$. And we assume that both $\mathbf{C}_1^T, \mathbf{C}_2^T$ have full column ranks. Then, it is easy to verify

$$\mathbf{y}_1(t) = \mathbf{C}_1(\mathbf{C}_2^T \mathbf{C}_2)^{-1} \mathbf{C}_2^T (\mathbf{y}_2(t) - \mathbf{w}_2(t)) + \mathbf{w}_1(t) \quad (3.7)$$

which is easy to tell that $\mathbf{y}_1$ and $\mathbf{y}_2$ are separable.

Since the controller receive observations from sensors and takes actions based on the received information, the correctness of the received information and the control action will influence the performance fo the system.

3.1.2 Communication

In most cases the sensors are battery-constrained and randomly scattered in CPSs, which makes them hard to communicate with each other directly. Hence, we assume no communications are allowed among the sensors. Because of this, to facilitate the distributed source coding, we assume that the side information as shown in (3.7) is required, e.g. computing $\mathbf{y}_1(t)$ given $\mathbf{y}_2(t)$ at time t , which necessitates the process of receiving feedback from the controller to the sensors as shown in Fig. 3.1. Actually, the challenges in our work are how to obtain and utilize the side information as the feedback and how to apply it into the distributed source coding. Note that we also assume there is no communication error through the communication channel, which is practical based on the existing well-designed encoding algorithms.

Before we discuss how to apply distributed source coding(or quantization) for the two sensors(sources). Next, we first propose the heuristic algorithm and Kalman Filtering algorithm, with which we used to estimate the prior information for the two sensors¹, and then the decoder could send it as the feedback to the sensors.

3.1.3 Feedback from Physical Dynamics

We first consider the sequential structure illustrated in Fig. 3.2. The most intuitive idea is that the sensor's side information in time slot t is provided by the decoder (controller) from the prior information of the decoding procedure in time slot $t - 1$; This procedure is repeated throughout the operations of the CPSs. Similar to [20], we list two methods to obtain the feedback or the estimated side information.

¹The prediction can be deployed at either controller or sensors, here we assume it is made at controller considering the battery constraint.

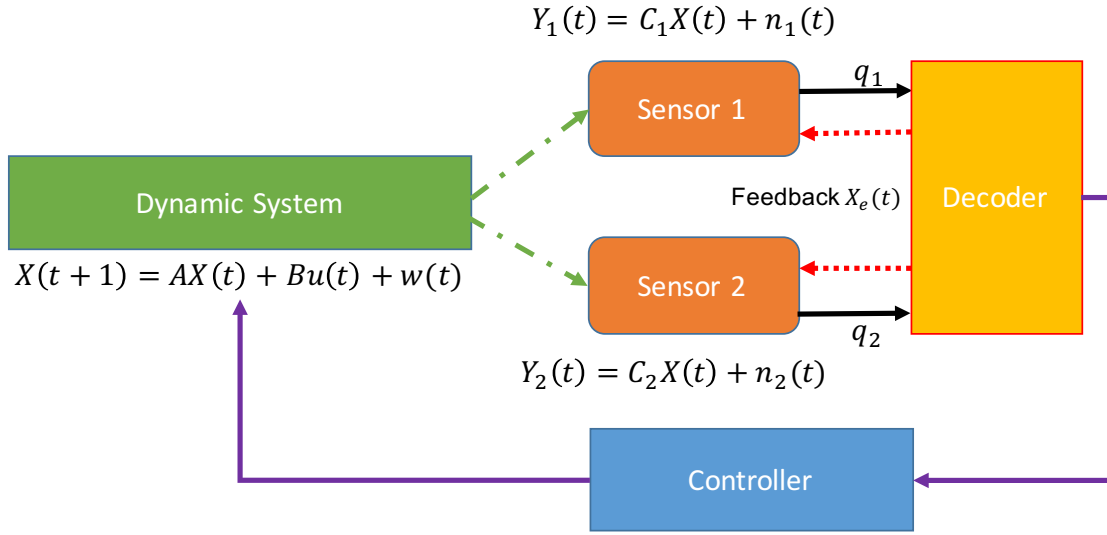


Figure 3.1: The feedback aware dynamic system

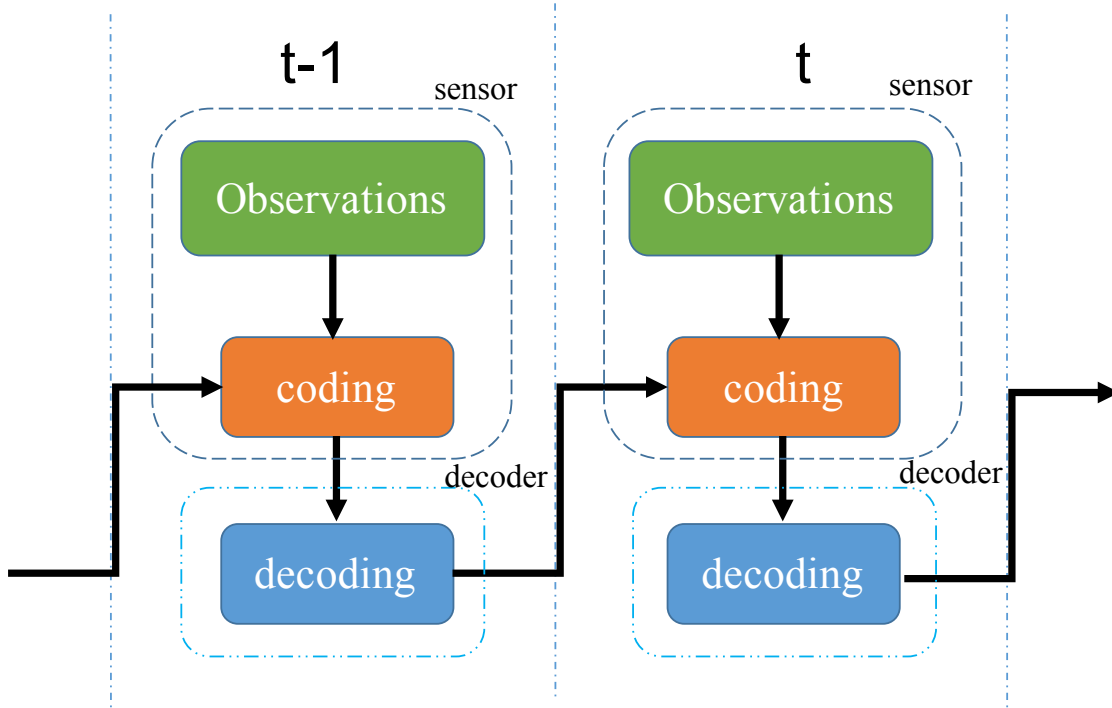


Figure 3.2: Sequential structure in distributed source coding model

Generic Case

In fact, based on the system dynamics of (3.2), the decoder can predict the observations of $\mathbf{y}(t)$ at one sensor if knowing all observations before time slot t , as

$$p_{\mathbf{y}}(t) = \int p_{\mathbf{w}}(\mathbf{w}) p_{\mathbf{x}}(\mathbf{x}(t) | \mathbf{q}(\tau)_{\tau < t}) \times \mathbb{1}(g(\mathbf{x}(t), \mathbf{w}) = \mathbf{y}(t)) d\mathbf{x} d\mathbf{w} \quad (3.8)$$

where $\mathbb{1}(g(\mathbf{x}(t), \mathbf{w}) = \mathbf{y}(t))$ is the indicator function for the event that the noise $\mathbf{w}(t)$ and system state $\mathbf{x}(t)$ generate the observation $\mathbf{y}(t)$, and $p_{\mathbf{x}}(\mathbf{x}(t) | \mathbf{q}(\tau)_{\tau < t})$ is the conditional probability of $\mathbf{x}(t)$ given all previously received information of $\mathbf{y}(t)$.

However, it is usually too hard to compute (3.8), thus for simplicity we redirect to a simple case of linear system by using Kalman Filtering.

Kalman Filtering for Linear Systems

When the quantizations $\mathbf{q}(t)$ of one sensor are sent to the controller from the sensors, Kalman Filtering (KF) can be applied to estimate the current system state and predict the future one by $\mathbf{x}(t|t)$ and $\mathbf{x}(t+1|t)$, where they are defined as $\mathbf{x}(t|t) \triangleq E[\mathbf{x}(t) | \mathbf{q}(\tau \leq t)]$, $\mathbf{x}(t+1|t) \triangleq E[\mathbf{x}(t+1) | \mathbf{q}(\tau \leq t)]$, respectively. If the quantizations were viewed as perfect observations, and they are fed to the controller recursively by the following equations,

$$\mathbf{x}(t+1|t) = \mathbf{A}\mathbf{x}(t|t) \quad (3.9)$$

where the current system state estimation $\mathbf{x}(t|t)$ is given by

$$\mathbf{x}(t|t) = \mathbf{x}(t|t-1) + \mathbf{K}(t)(\mathbf{q}(t) - \mathbf{C}\mathbf{x}(t|t-1)) \quad (3.10)$$

and

$$\mathbf{K}(t) = \Sigma(t|t-1)\mathbf{C}^T(\mathbf{C}\Sigma(t|t-1)\mathbf{C}^T + \Sigma_{\mathbf{w}})^{-1} \quad (3.11)$$

where $\Sigma(t|t-1) = \text{Cov}(\mathbf{x}(t)|\mathbf{q}(\tau \leq t-1))$, $\Sigma(t|t) = \text{Cov}(\mathbf{x}(t)|\mathbf{q}(\tau \leq t))$ and the covariance matrices of prediction error are given by

$$\Sigma(t|t) = \Sigma(t|t-1) - \mathbf{K}(t)\mathbf{C}\Sigma(t|t-1) \quad (3.12)$$

where

$$\Sigma(t|t-1) = \mathbf{A}\Sigma(t-1|t-1)\mathbf{A}^T + \mathbf{B}\mathbf{Q}_t\mathbf{B}^T + \Sigma_{\mathbf{n}} \quad (3.13)$$

where $\mathbf{Q}_t = E[\mathbf{u}_t\mathbf{u}_t'|\mathbf{q}(\tau < t)]$.

Note that, due to quantization errors or transmission errors, the input $\mathbf{q}(t)$ to the system state estimator at the decoder is not perfect. However, we consider it being in high resolution and directly apply $\mathbf{q}(t)$ in the system estimation in (3.10). As will be seen in the numerical results, this approximation results in good performance and also substantially simplifies the algorithm.

3.2 Feedback Adaptive Distributed Source Coding

In this section, we try to deploy the lattice quantization for the encoder and define the criterion of distortion for the decoder, where the distortion lays the foundation of the eventual purpose for controlling.

3.2.1 Quantization and Distortion

Since the sensors(encoder) could receive feedback from the controller(decoder), we assume the observations of the sensor as $\mathbf{y}_1$ and the feedback as $\mathbf{y}_2$, and we assume that lattices are used for quantization of observations $\mathbf{y}_1$ with $\mathbf{y}_2$.

Lattice Quantization

To begin with, we define the lattice as [63, 9, 28].

$$\Sigma = \{\mathbf{G}\mathbf{i}, \mathbf{i} \in \mathbf{Z}^n\} \quad (3.14)$$

where $\mathbf{Z} = (\dots, -2, -1, 0, 1, 2, \dots)$, n is the dimensional of the space and $\mathbf{G}$ is an $n \times n$ generator matrix.

Similar to our previous work [28], we assume that for $\mathbf{y}_1$ and $\mathbf{y}_2$, a higher dimension lattice generated by $\mathbf{G}_1$ and $\mathbf{G}_2$ as $\mathbf{G}_{12}$ as

$$\mathbf{G}_{12} = (\mathbf{G}_1^T, \mathbf{G}_2^T)^T \quad (3.15)$$

and

$$\begin{aligned} \Sigma_{12} &= \{\mathbf{G}_{12}\mathbf{i}, \mathbf{i} \in \mathbf{Z}^{N_1+N_2}\} \\ &= \left\{ ((\mathbf{G}_1\mathbf{i}_1)^T, (\mathbf{G}_2\mathbf{i}_2)^T)^T, \mathbf{i}_1 \in \mathbf{Z}^{N_1}, \mathbf{i}_2 \in \mathbf{Z}^{N_2} \right\} \end{aligned}$$

Here, $\mathbf{G}_{12}$ would generate the lattice, which has full observations of $\mathbf{y} = (\mathbf{y}_1^T, \mathbf{y}_2^T)^T$. In fact, in this distributed source coding setting, when we need to quantize $\mathbf{y}_1$ given $\mathbf{y}_2$, the side information $\mathbf{y}_2$ can also be viewed as the side information used for choosing different codebooks (hints to tell $\mathbf{y}_1$ to implement a more suitable codebook for the encoding, and these hints are shared by both encoder and decoder), where the side information $\mathbf{y}_2$ acts the gear in changing these quantization codebooks, say $\mathbf{G}_{12}$.

Note that, we apply the generating matrixes $\mathbf{G}_1$ and $\mathbf{G}_2$ as the ones that need to be adjusted as our coding strategies, and which can be optimized based on the distortion function we will discuss later since it will affect the distortion of decoder. For example, the system state estimated from full information $\mathbf{y}_1$ and $\mathbf{y}_2$ can be written as

$$\hat{\mathbf{x}} = (\mathbf{C}_1^T \mathbf{C}_1 + \mathbf{C}_2^T \mathbf{C}_2)^{-1} (\mathbf{C}_1^T, \mathbf{C}_2^T) (\mathbf{y}_1^T, \mathbf{y}_2^T) \quad (3.16)$$

Then, the lattice generated the estimation of $\hat{\mathbf{x}}$ is $\mathbf{G}_x$, which can be given as

$$\mathbf{G}_x = (\mathbf{C}_1^T \mathbf{C}_1 + \mathbf{C}_2^T \mathbf{C}_2)^{-1} (\mathbf{C}_1^T, \mathbf{C}_2^T) \mathbf{G}_{12} \quad (3.17)$$

Overall, the choice of $\mathbf{G}_1$ and $\mathbf{G}_2$ would directly affect $\mathbf{G}_{12} = (\mathbf{G}_1^T, \mathbf{G}_2^T)^T$ and then the distortion of controller or the performance of system.

Distortion for Decoder

After learning the quantization, we will delve into the distortion of decoder for the purpose of controlling. With the dynamics in Fig. 3.1 and (3.6), we assume that the controller takes a ‘blow-up’ control action, which is given by

$$\mathbf{B}\mathbf{u}(t) = -\mathbf{A}\hat{\mathbf{x}}(t) \quad (3.18)$$

where $\hat{\mathbf{x}}(t)$ is the estimation of $\mathbf{x}(t)$. We assume that $\mathbf{B}$ is full-rank, which guarantees the unique existence of $\mathbf{u}(t)$.

$$\mathbf{u}(t) = -\mathbf{B}^{-1}\mathbf{A}\hat{\mathbf{x}}(t) \quad (3.19)$$

It is obvious that if we have zero noise (e.g., $\mathbf{n}(t) = \mathbf{0}$) and the perfect estimation $\hat{\mathbf{x}}(t) = \mathbf{x}(t)$, the system state would be zero under this control. Then, we can obtain the variance of the system state by

$$V[\mathbf{x}(t)] = E[\mathbf{n}(t)^2] + \mathbf{A}E[||\hat{\mathbf{x}}(t) - \mathbf{x}(t)||^2] \quad (3.20)$$

And, the distortion criterion of the source coding can be ruled out as

$$D = V[\mathbf{x}(t)] \quad (3.21)$$

i.e., the variance of the system state estimation. In this work, we define this as the loss function of the decoder, while the distortion criterion can be different based on different CPSs.

In fact, the estimations of $\mathbf{x}(t)$ in different subspaces are playing different roles in the performance of controlling. Thus, decomposing the matrix $\mathbf{A}$ can help for the analysis of system performance,

$$\mathbf{A} = \mathbf{U}^T \mathbf{\Lambda} \mathbf{U} \quad (3.22)$$

where $\mathbf{U}$ is unitary and $\mathbf{\Lambda} = \text{diag}(\lambda_1, \dots, \lambda_N)$, where λ_n is the n -th eigenvalue, where we assume $\lambda_i \neq \lambda_j$, if $i \neq j$. In this way, the system state can be decomposed into N independent

sub-spaces, which are given by

$$z_n(t+1) = \lambda_n z_n(t) + v_n(t) + \tilde{n}(t) \quad (3.23)$$

where $\mathbf{z} = (z_1, \dots, z_N) = \mathbf{U}\mathbf{x}$, $\tilde{\mathbf{n}} = (\tilde{n}_1, \dots, \tilde{n}_N) = \mathbf{U}\mathbf{n}$, and $\mathbf{v} = (v_1, \dots, v_N) = \mathbf{U}\mathbf{B}\mathbf{u}$.

It is easy to tell that the subspaces of $|\lambda_n| \geq 1$ are not stable and if not properly controlled, the system state will diverge; and when $|\lambda_n| < 1$, the subspaces inherently degrade to zero and are kept from zero by the noise perturbations, and hence, even if without control, the system state is still stable. The decomposition is illustrated in Fig. 3.3.

Similar to the above, we also assume that the controller still takes the subspace ‘blow-up’ control action as

$$v_n(t) = -\lambda_n \hat{z}_n(t) \quad (3.24)$$

where $\hat{z}_n$ is the estimation of z_n . with full row rank $\mathbf{U}\mathbf{B}$ and the existence of $\mathbf{v}$, we can rewrite the variance of system state at the n -th subspace by

$$V[z_n] = E[\hat{n}_n^2] + |\lambda_n|^2 E[\epsilon_n^2] \quad (3.25)$$

where ϵ is the estimation error of z_n .

In this case, the distortion of the system is

$$\mathbf{D} = \sum_{n \in \Omega} V[z_n] \quad (3.26)$$

i.e., the sum of variances of the system state in the unstable subspaces, where Ω is the set of subspaces with $|\lambda_n| \geq 1$.

We also hope the variance caused by the quantization to be identical in all Ω . As shown in Fig. 3.4, the lattice would shape as being proportional to $\{\frac{1}{\lambda_n}\}_{n \in \Omega}$, where the quantization error is inversely proportional to the corresponding eigenvalue, since the larger the eigenvalue is the more impact the quantization error will cause on the system state.

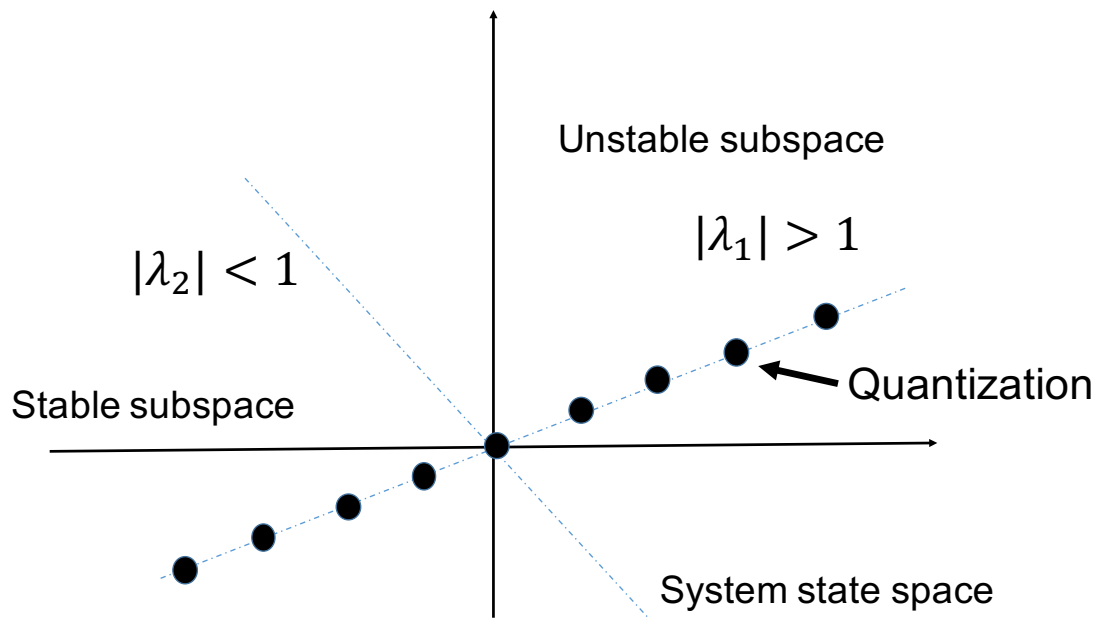


Figure 3.3: An illustration of unstable and stable subspaces

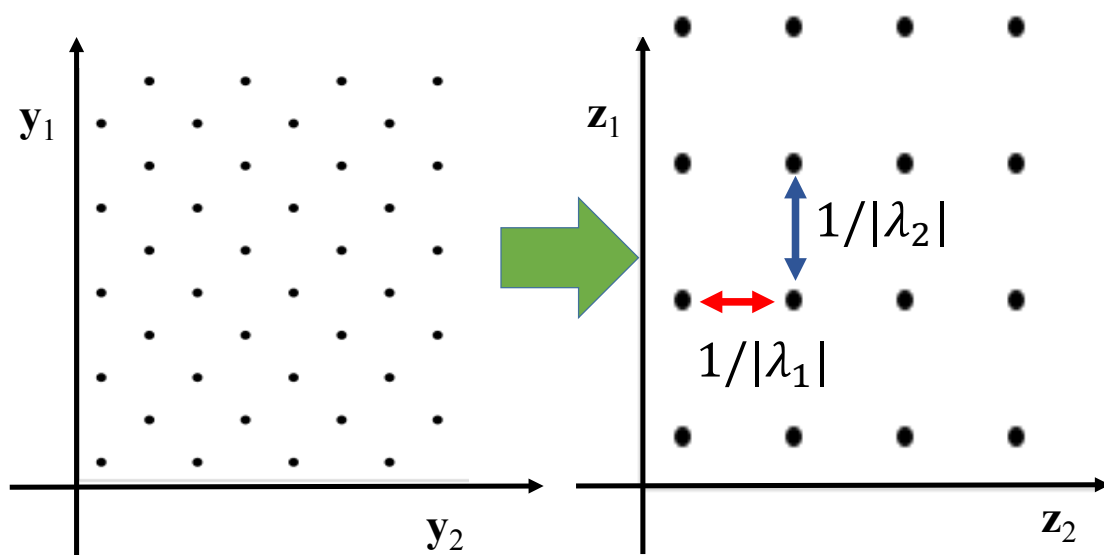


Figure 3.4: An illustration of transformation to the desired lattice

Hence, $\mathbf{G}_x$ could be rewritten as

$$\mathbf{G}_x = \Delta \begin{pmatrix} \frac{1}{\lambda_1} & 0 & \dots & 0 \\ 0 & \frac{1}{\lambda_2} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \frac{1}{\lambda_{N_1}} \end{pmatrix} \quad (3.27)$$

where Δ would be the quantization strategy for controlling the communication cost and performance of the system.

3.2.2 Distributed Source Coding

Intuitively, the decoder expect the choices of $\mathbf{G}_1$ and $\mathbf{G}_2$ make the distortion of the system to be small, which would by no means decrease the communication cost of the sensors. Thus, the optimal choice of $\mathbf{G}_1$ and $\mathbf{G}_2$ that results in small communication cost and good performance would our main target, which we will discuss later.

Moreover, to make sure the validation of the source coding, we also need the the relationship between lattices Σ_1 and Σ_2 follows the following rules:

$$E[\mathbf{y}_1|\mathbf{y}_2] \in \Sigma_1, \quad \forall \mathbf{y}_2 \in \Sigma_2 \quad (3.28)$$

and,

$$h_{21}(\mathbf{y}_2) + E[\mathbf{w}_{12}] \in \Sigma_1, \quad \forall \mathbf{y}_2 \in \Sigma_2 \quad (3.29)$$

if $\mathbf{y}_1$ and $\mathbf{y}_2$ are separable.

The procedure is illustrated in Fig. 3.5. The criterion for the encoder would be to find the optimal generating matrix $\mathbf{G}_1$ and $\mathbf{G}_2$ that can map $\mathbf{y}_1$ with $\mathbf{y}_2$, and to minimize the average cost in communication.

In this dissertation, we apply the same coloring based coding as shown in [28], and the details can be referred in Section IV of [28], so we will omit here. In summary, for sensor $\mathbf{y}_1$, the total number of colors is given by $\prod_{k=1}^{N_1} n_k^*$ and the average number of bits used to

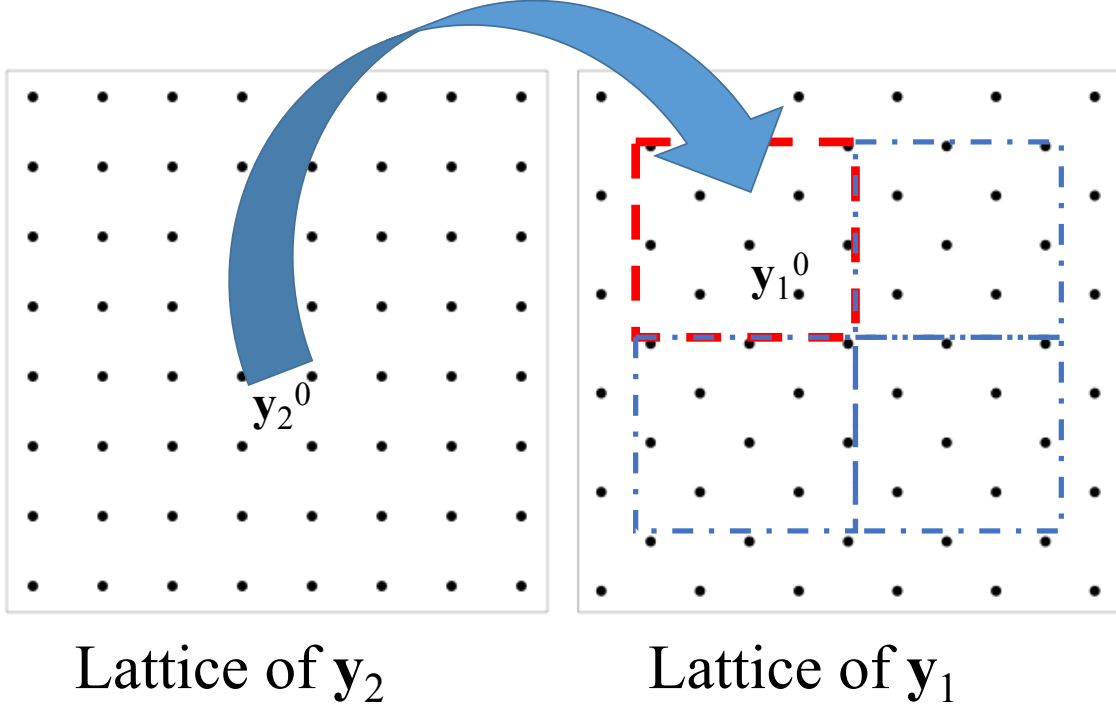


Figure 3.5: An illustration of super-rectangles

describe the color of $\mathbf{y}_1$ is given by

$$E[\text{bits of } \mathbf{y}_1] = \sum_{k=1}^{N_1} \log_2 n_k^* \quad (3.30)$$

which stands the communication cost of the encoder, and it actually depends on the $\mathbf{G}_1$ and γ_1 , where γ_1 is the threshold for the probability of corresponding values $\mathbf{y}_1$ fall in the lattice of Σ_1 given $\mathbf{y}_2$. For instance, with a fixed $\gamma_1 = 1$ and a fixed range of $\mathbf{y}_1$ in $[0, 1]$ in one dimension lattice $\mathbf{Z} = (\dots, -2, -1, 0, 1, 2, \dots)$, we will need 1 bit for $\mathbf{G}_1 = 1$ and 2 bits for $\mathbf{G}_1 = 0.25$ to transmit the quantized $\mathbf{y}_1$.

3.2.3 Theoretical Communication Bounds for Feedback Control

As we stated if a special case, such as the linear system dynamics, is considered, the theoretical bound for the communication cost can be estimated by delving into the induction

of Kalman Filtering with a simple assumption. For instance, we assume that the quantization error can be separated from observation error, and then (3.6) can be reformulated as

$$\begin{cases} \mathbf{x}(t+1) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{n}(t) \\ \mathbf{q}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{e}(t) + \mathbf{w}(t) \end{cases} \quad (3.31)$$

where we assume the quantization error is denoted as $\mathbf{e}(t)$, which complies to a Gaussian distribution with zero mean and covariance matrix as $\Sigma_{\mathbf{e}}$.

Message Passing from Sensors to Controller

The messages sent from sensors to controller is the quantizations $\mathbf{q}(t)$, while the controller works as a data fusion collecting all the quantizations $\mathbf{q}_i(t)$ ($i = 1, 2$) from the two sensors. If the Kalman Filtering at the controller is applied to estimate the current system state and predict the future system state based on the received quantizations, then system state can be estimated and predicted by $\mathbf{x}(t|t)$ and $\mathbf{x}(t+1|t)$, which are defined by $\mathbf{x}(t|t) \triangleq E[\mathbf{x}(t)|\mathbf{q}(\tau \leq t)]$, $\mathbf{x}(t+1|t) \triangleq E[\mathbf{x}(t+1)|\mathbf{q}(\tau \leq t)]$, respectively.

Next, the current and future system state could be calculated recursively by using the following equations derived from Kalman Filtering,

$$\mathbf{x}(t+1|t) = \mathbf{A}\mathbf{x}(t|t) \quad (3.32)$$

where the current system state estimation $\mathbf{x}(t|t)$ is given by

$$\mathbf{x}(t|t) = \mathbf{x}(t|t-1) + \mathbf{K}(t)(\mathbf{q}(t) - \mathbf{C}\mathbf{x}(t|t-1)) \quad (3.33)$$

and

$$\mathbf{K}(t) = \Sigma(t|t-1)\mathbf{C}^T(\mathbf{C}\Sigma(t|t-1)\mathbf{C}^T + \Sigma_{\mathbf{w}+\mathbf{e}})^{-1} \quad (3.34)$$

where $\Sigma(t|t-1) = Cov(\mathbf{x}(t)|\mathbf{q}(\tau \leq t-1))$, $\Sigma(t|t) = Cov(\mathbf{x}(t)|\mathbf{q}(\tau \leq t))$.

Then, the covariance matrices of prediction/estimation error, say $\mathbf{x}(t) - \mathbf{x}(t|t-1)$ given $\mathbf{q}(\tau \leq t-1)$, are given by

$$\Sigma(t|t) = \Sigma(t|t-1) - \mathbf{K}(t)\mathbf{C}\Sigma(t|t-1) \quad (3.35)$$

where

$$\Sigma(t|t-1) = \mathbf{A}\Sigma(t-1|t-1)\mathbf{A}^T + \mathbf{B}\mathbf{Q}_t\mathbf{B}^T + \Sigma_n \quad (3.36)$$

where $\mathbf{Q}_t = E[\mathbf{u}_t\mathbf{u}_t'|\mathbf{q}(\tau < t)]$.

Note that, due to the possible quantization errors and transmission errors, the input $\mathbf{q}(t)$ to the controller is not perfect either, and to further reduce the total cost of computation at the sensors, we also include the feedback from the controller in our analysis, which is defined as the estimation of system state $\mathbf{x}_e(t+1) = \mathbf{x}(t+1|t)$.

Feedback Sent from Controller to Sensors

To view the effect of the feedback sent from controller to sensors, we assume that a differential encoding is deployed at the sensors and then the real information being transmitted to the communication channel by the sensors are denoted as

$$\begin{aligned} \mathbf{e}(t+1) &= \mathbf{C}\mathbf{x}(t+1) - \mathbf{C}\mathbf{x}_e(t+1) \\ &= \mathbf{C}[\mathbf{x}(t+1) - \mathbf{x}(t+1|t)] \end{aligned} \quad (3.37)$$

To make it short, we define $\hat{\mathbf{e}}(t+1) = \mathbf{x}(t+1) - \mathbf{x}(t+1|t)$, where it could be further calculated as

$$\hat{\mathbf{e}}(t+1) = \mathbf{A}[\mathbf{x}(t) - \mathbf{x}(t|t)] + \mathbf{B}\mathbf{u}(t) + \mathbf{n}(t) \quad (3.38)$$

If we denote $\mathbf{e}_t = \mathbf{x}(t) - \mathbf{x}(t|t)$, which stands for the estimation error of current system state at the controller, then the variance of $\mathbf{e}_t$ can be formulated as

$$E[\mathbf{e}_t\mathbf{e}_t^T] = E[(\mathbf{x}(t) - \mathbf{x}(t|t))(\mathbf{x}(t) - \mathbf{x}(t|t))^T] \quad (3.39)$$

Here we denote $\mathbf{P}_t = E[\mathbf{e}_t \mathbf{e}_t^T]$, and we know that,

$$\begin{aligned} \mathbf{x}(t|t) &= \mathbf{x}(t|t-1) + \mathbf{K}(t)(\mathbf{q}(t) - \mathbf{C}\mathbf{x}(t|t-1)) \\ &= \mathbf{x}(t|t-1) + \mathbf{K}(t)(\mathbf{C}\mathbf{x}(t) + \mathbf{e}(t) + \mathbf{w}(t) - \mathbf{C}\mathbf{x}(t|t-1)) \end{aligned} \quad (3.40)$$

where we can obtain

$$\mathbf{P}_t = [\mathbf{I} - \mathbf{K}(t)\mathbf{C}]\mathbf{P}(t)[\mathbf{I} - \mathbf{K}(t)\mathbf{C}]^T - \mathbf{K}(t)\Sigma_{\mathbf{w}+\mathbf{e}}\mathbf{K}(t)^T \quad (3.41)$$

where $\mathbf{P}(t)$ is the prior estimation of $\mathbf{P}_t$, where $\mathbf{P}(t) = E[\hat{\mathbf{e}}(t)\hat{\mathbf{e}}(t)^T]$, which is also one of the updating rules in Kalman Filtering.

Overall, we learn that the variance of information $\hat{e}(t+1)$ being transmitted in the communication channel is

$$\mathbf{P}(t+1) = E[\hat{\mathbf{e}}(t+1)\hat{\mathbf{e}}(t+1)^T] = \mathbf{A}\mathbf{P}_t\mathbf{A}^T + \mathbf{B}\mathbf{Q}_t\mathbf{B}^T + \Sigma_{\mathbf{n}} \quad (3.42)$$

Then, combining (3.41) and (3.42), we can obtain the evolution of $\mathbf{P}(t)$, which is also the variance of $\hat{\mathbf{e}}(t)$.

Theoretical Bound

Since we assume that all the sources are complying the Gaussian distribution, then based on the Quadratic Gaussian rate function, which is defined by

$$\mathbf{R}(x) = \max\left\{\frac{1}{2}\log x, 0\right\} = \frac{1}{2}[\log x]^+ \quad (3.43)$$

where we can obtain the theoretical bound of communication in this feedback control setting as

$$\mathbf{R}(x) = \frac{1}{2}[\log P(t+1)]^+ \quad (3.44)$$

3.3 Cooperative Reinforcement Q-learning

Previously, we have given an incentive method to realize the distributed source coding by utilizing the feedback from the controller. But in practical cases, if error free communication is hard to be taken into effect, the error in feedback will render the the lattice encoding completely not decodable.

And hence in this section, we will use cooperative reinforcement Q-learning, which provides more easy solutions, to solve our distributed source coding problem. We first reformulate our problem as an optimization problem and then we give our network architecture in the cooperative reinforcement Q-learning to give a solution to the optimization.

3.3.1 The Optimization Problem

Here, we formulate the design of optimal quantization strategies as an optimization problem with both encoders and decoder as being illustrated in Fig. 3.6. Here we try to train the neural networks to find the mapping functions: for encoders, a mapping function from observations to coding strategies; for decoders, a mapping function from estimated system state to actions. Before we start to train the neural networks, we first formulate our rewards for our Q-learning and the philosophy of the formulation is to set the combination of both the discounted sum of communication cost and system variance as the overall cost.

Encoder

In (3.1), we recall the discrete-time and continuous-state dynamic system in CPSs. The sensor i measures the system state and reports to the controller via a communication channel by quantizing the observation $\mathbf{y}_i$. Different from previous method, here we assume that the quantization strategies are adaptive to the system state and obtained based on a quantization policy $\mathbf{G}_i(t) = f_{encoder}(\mathbf{y}_i(t))$, where $\mathbf{y}_i(t)$ is the observation at time t . Based on (3.30), we can write the total communication cost for the encoder i for a time horizon T as

$$\min_{\mathbf{G}_i(0), \mathbf{G}_i(1), \dots, \mathbf{G}_i(T)} \sum_{t=1}^{T-1} \alpha^t E[\text{bits of } \mathbf{y}_i(t)] \quad (3.45)$$

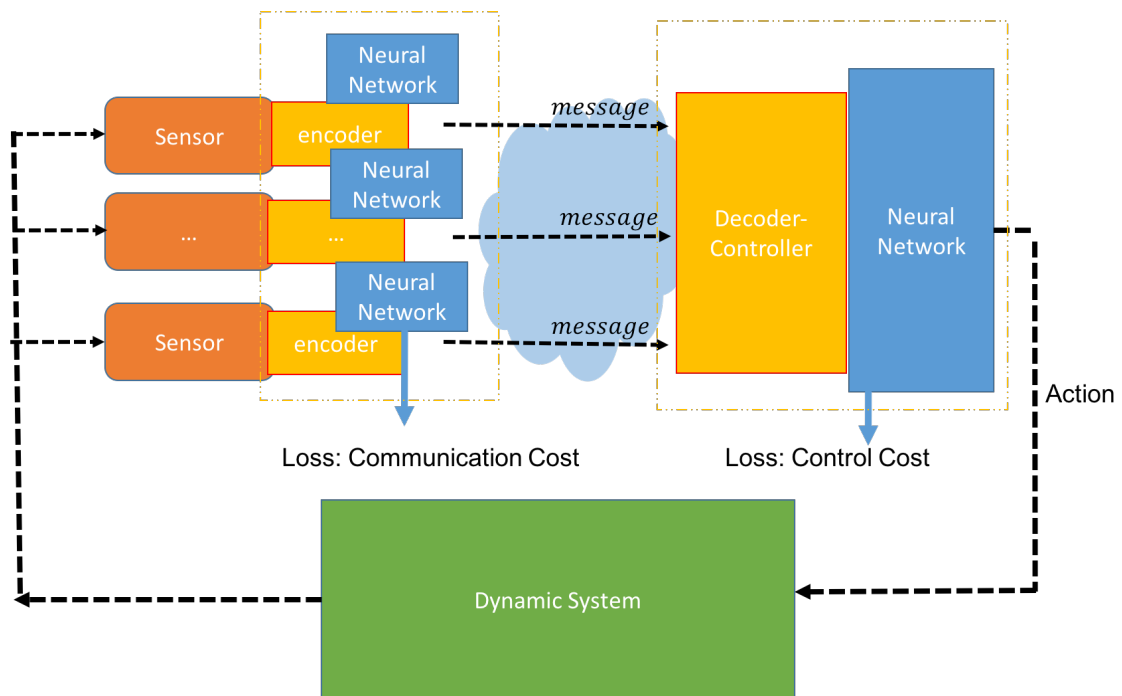


Figure 3.6: Cooperative reinforcement Q-learning network architecture

where $i = 1, 2$ and we assume $0 < \alpha < 1$ as the discount factor such that the near future is more emphasized. (3.45) also stands for the transmission rate of the encoder i .

Decoder

Similar to our previous method, we define that the controller takes a 'blow up' control action, then the performance of the system can be evaluated by the distortion criterion in (3.21). Then, for the encoder for a time horizon T , we can give the total cost/penalty of the dynamic system as

$$\min_{\mathbf{u}(0), \mathbf{u}(1), \dots, \mathbf{u}(T)} \sum_{t=1}^{T-1} \alpha^t V[\mathbf{x}(t)] \quad (3.46)$$

where $\mathbf{u}(t) = -\mathbf{B}^{-1}\mathbf{A}\hat{\mathbf{x}}(t)$, which means we try to decode the system state $\hat{\mathbf{x}}(t)$ based on all the received quantized $\mathbf{y}_i$ for $i = 1, 2$, where $\hat{\mathbf{x}}(t) = f_{\text{decoder}}(\{\mathbf{y}_i(t)\})$. To simplify the process, we let the network directly output the control $\mathbf{u}(t)$, which makes $\mathbf{u}(t) = f_{\text{decoder}}(\{\mathbf{y}_i(t)\})$.

System Cost

The cooperative setting here can be expressed as the two players α discounted sum Markov game [43]. The encoders and decoder will work cooperatively minimizing the system cost, where the system cost is defined as

$$\min_{\mathbf{u}(0), \dots, \mathbf{u}(T)} \min_{\mathbf{G}_1(0), \dots, \mathbf{G}_1(T), \mathbf{G}_2(0), \dots, \mathbf{G}_2(T)} \sum_{t=1}^{T-1} \alpha^t (E[\text{bits of } \mathbf{y}_1^t] + E[\text{bits of } \mathbf{y}_2^t] + \lambda V[\mathbf{x}(t)]) \quad (3.47)$$

where the weighting constant $\lambda > 0$ is the Lagrange multiplier.

To this end, we have all the routines to formulate the distributed source coding problem, and if we know $f_{\text{encoder}}(\cdot)$ and $f_{\text{decoder}}(\cdot)$, a solution can always be obtained. Next, we will apply the cooperative reinforcement Q-learning to find $f_{\text{encoder}}(\cdot)$ and $f_{\text{decoder}}(\cdot)$.

3.3.2 Cooperative Reinforcement Q-learning

Reinforcement Learning as a branch in machine learning aims to solve problem where the agents take actions in an environment so as to maximize some notion of cumulative reward. Since the cumulative reward can be rewritten to comply to the Bellman equation as shown in

[32], the reinforcement Q-learning from some point is to use the neural networks to find the universal function approximations that resemble the function Q (or say the value function in Bellman Equation), where we could easily construct a policy that maximizes our rewards or minimize our penalty. In our case, we use the deep neural networks to learn the approximated functions $f_{encoder}(\cdot)$ and $f_{decoder}(\cdot)$.

Network Architecture

The architecture of the proposed cooperative reinforcement Q-learning is summarized as below.

Encoder The inputs² of the neural network at the sensor are the observations of the sensor, and the output of the neural network is the quantization strategy. Here we use a two-layers linear network followed by one leaky-ReLU activation and one Softmax function to train the inputs and output the quantization strategy, such as the index number to select a certain $\mathbf{G}_i(t)$ from a group of $\{\mathbf{G}\}$ at time t , where $i = 1, 2$ for two sensors case.

Decoder The inputs of the neural network at the decoder/controller are all the received quantized information from all the sensors, and the output of the neural network is the action to control the system. Here we use the same structure of the neural network as the encoders to simplify the process, which is a two-layers linear network followed by one leaky-ReLU activation and one Softmax function to train the inputs, where the output of the network is the index number of available actions.

Network Training

During training, we implement the real dynamic system, and we sequentially feed the neural network at the encoder with the sensors' observations, and we feed the neural network at the decoder with all the quantized information sent from the sensors. At each training epoch, all the neural networks will try to keep the variance of the system to be small while decreasing the communication cost. That is, all neural networks would try to minimize the

²The feedback from the controller can also be added to further improve the results. Here we only use observations to simplify the simulations

cost function as shown in (3.47). Note that we train all the neural networks in a centralized setting, and after the training the networks could be implemented in a distributed manner. We illustrated the algorithm how we train the network with two sensors and one controller in Algorithm 3.

Data: Initiate networks at Sensors and Controller
Result: Save all the networks
Initialization of the dynamic system $\mathbf{x}_t$ and feedback;
Initialization the network at Sensor1 as Quantizer1;
Initialization the network at Sensor2 as Quantizer2;
Initialization the network at the Controller as Policy;
while *System not converge* **do**
 obtain $\mathbf{y}_1(t), \mathbf{y}_2(t)$;
 if *Done* **then**
 $\text{Strategy}_1 = \text{Quantizer1}(\mathbf{y}_1(t))$;
 $\text{Strategy}_2 = \text{Quantizer2}(\mathbf{y}_2(t))$;
 $\mathbf{q}_1(t) = Q(\mathbf{y}_1(t), \text{Strategy}_1)$;
 $\mathbf{q}_2(t) = Q(\mathbf{y}_2(t), \text{Strategy}_2)$;
 $\mathbf{u}(t) = \text{Policy}(\mathbf{q}_1(t), \mathbf{q}_2(t))$;
 Feed $\mathbf{u}(t)$ to the dynamic system;
 else
 Wait;
 end
end

Algorithm 3: Train the network at the encoders and decoder

The weight of all the neural networks are learning from back-propagation algorithm and the Adam optimizer by minimizing the loss function. The train will stop when a certain stable status of the system will be reached. Our neural network is implemented in PyTorch [42] and evaluated by OpenAI GYM [44]. We started to train with an initial learning rate of 10^{-2} . In this dissertation, we select a simple neural network to test the results, and for other CPSs other neural networks would also be a good try.

Evaluation Methods

Performance of the source coding method would be evaluated in the same system, and in our simulation we test our proposed method in the open source toolbox from OpenAI.

The evaluations of the neural networks would be omitted since it is not the focus of the dissertation.

3.4 Numerical Results

In this section, we use numerical results to demonstrate the performance of the proposed schemes, where we try to deploy two kinds of CPSs, one power-grid system to test the feedback adaptive distributed source coding and one cart-pole system to validate the cooperate reinforcement learning distributed source coding.

3.4.1 Feedback Adaptive Distributed Source Coding

For the power-grid system, we assume that the main bus conveys the sensor observations, i.e., the voltage measurements to the controller in order to stabilize the voltages to a reference value. As shown in Fig. 3.7, we assume there are two sensors in the power-grid system, and the evolution of the power grid is given as

$$\mathbf{A} = \begin{pmatrix} 1.1269 & -0.4940 & 0.1129 \\ 1.0000 & 0 & 0 \\ 0 & 1.0000 & 0 \end{pmatrix} \quad (3.48)$$

with the initial system state as $\mathbf{x}(0) = [0, 0, 0]$ and

$$\mathbf{B} = \begin{pmatrix} -0.3832 \\ 0.5919 \\ -0.5191 \end{pmatrix} \quad (3.49)$$

and

$$\mathbf{C}_1 = \begin{pmatrix} 0.9234 & 0.9049 & 0.1111 \\ 0.4302 & 0.9797 & 0.2581 \\ 0.1848 & 0.4389 & 0.4087 \end{pmatrix} \quad (3.50)$$

and

$$\mathbf{C}_2 = \begin{pmatrix} 0.5328 & 0.8759 & 0.5870 \\ 0.3507 & 0.5502 & 0.2077 \\ 0.9390 & 0.6225 & 0.3012 \end{pmatrix} \quad (3.51)$$

Prediction based heuristic approach

We can apply the Monte Carlo integration to calculate the integration in (3.8). Since this is not our focus and it has been tested in [20], we omit the details here.

Prediction based Kalman Filtering

We try to consider the predictions based on steady-state and time-varying Kalman Filtering, and the results of the two sensors are shown in Fig. 3.8. It shows that when we feed the Kalman filter with some inputs(quantizations) and random noise, the true system state and the filtered system state can be quite close with the errors being fluctuant around. However, this closeness between the true system state and the filtered system state might not be acceptable in practical CPSs.

Distributed Source Coding

To realize the distributed source coding, we consider the sequential structure in Fig. 3.2. Here, we use the ‘Quantize’ as the quantized output of the encoder(sensor), ‘Kalman Filter’ as the feedback of the controller, and ‘Estimation’ as the difference between the quantized output and the feedback. As shown in Fig. 3.9, as the growing of the quantization levels, the estimation error has been decreasing, which verified the validation of Kalman Filter.

Since the feedback from the controller can be very close to the quantized output, we no longer need to transmit all the information, where we add this feedback as side information to do the distributed source coding by using the lattices as shown in Fig. 3.4. The results of the distributed source coding are shown in Fig. 3.10 and Fig. 3.11, where Fig. 3.10 shows that the state of the system is stable within a certain bound while Fig. 3.11 shows it needs less communication cost.

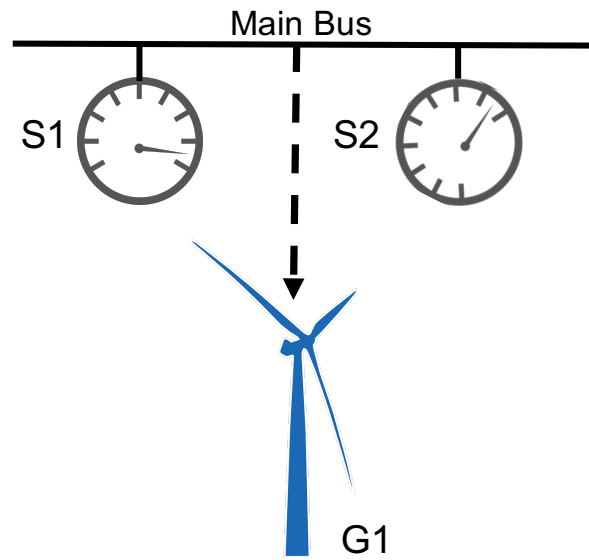


Figure 3.7: Power-grid system

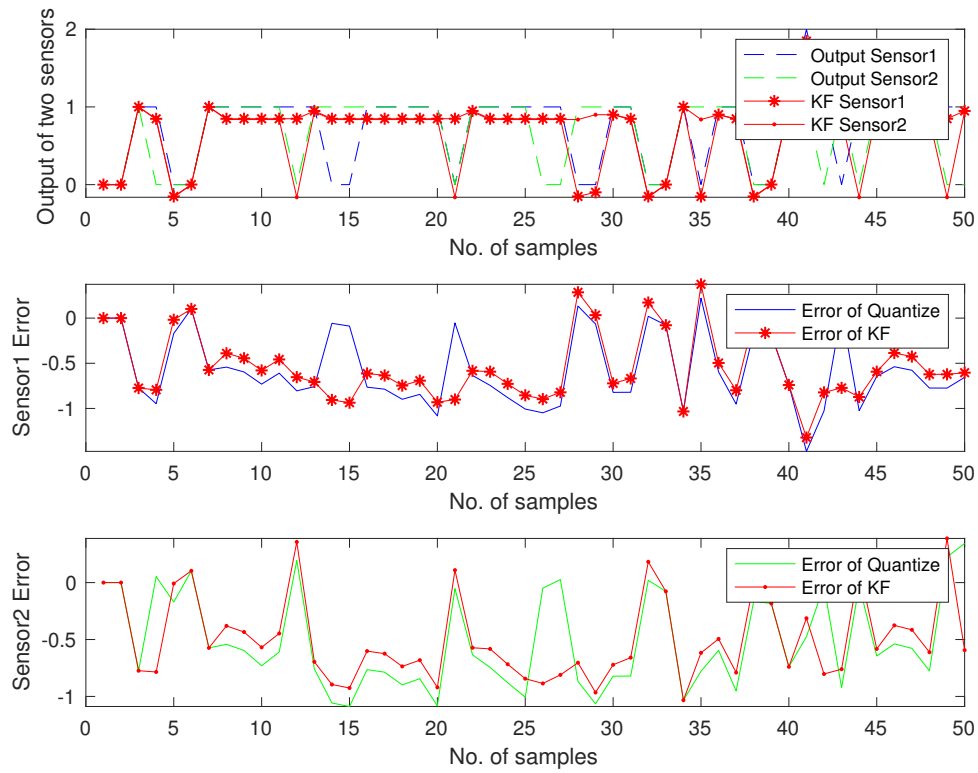


Figure 3.8: Kalman Filtering results for fifty samples

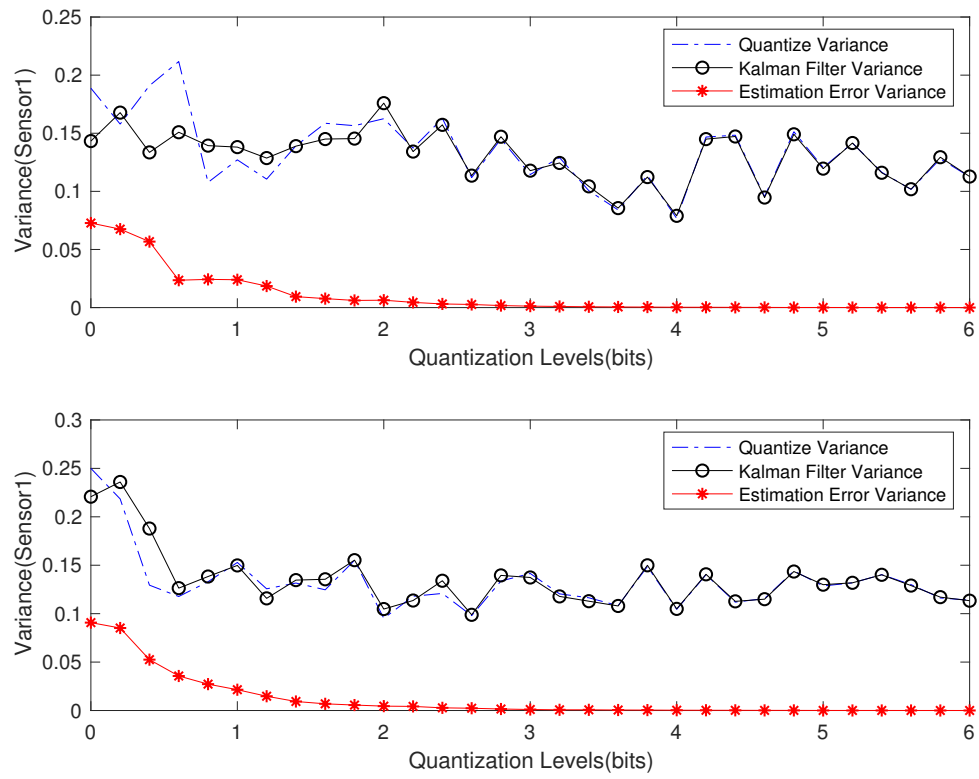


Figure 3.9: Prediction variance based on different quantization levels

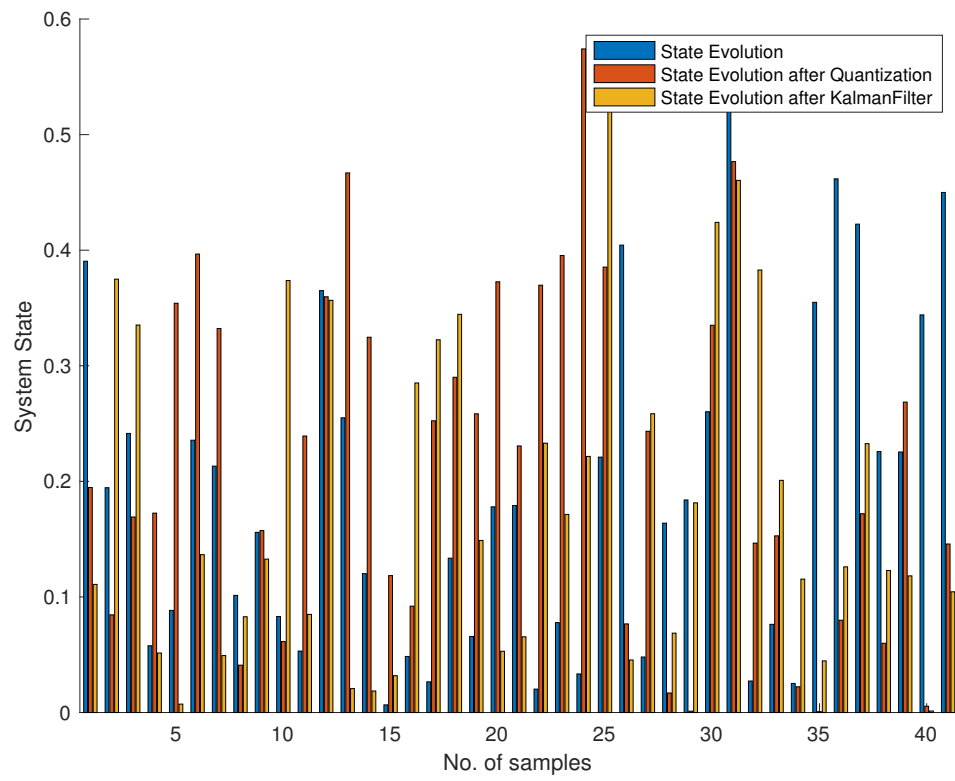


Figure 3.10: State evolution

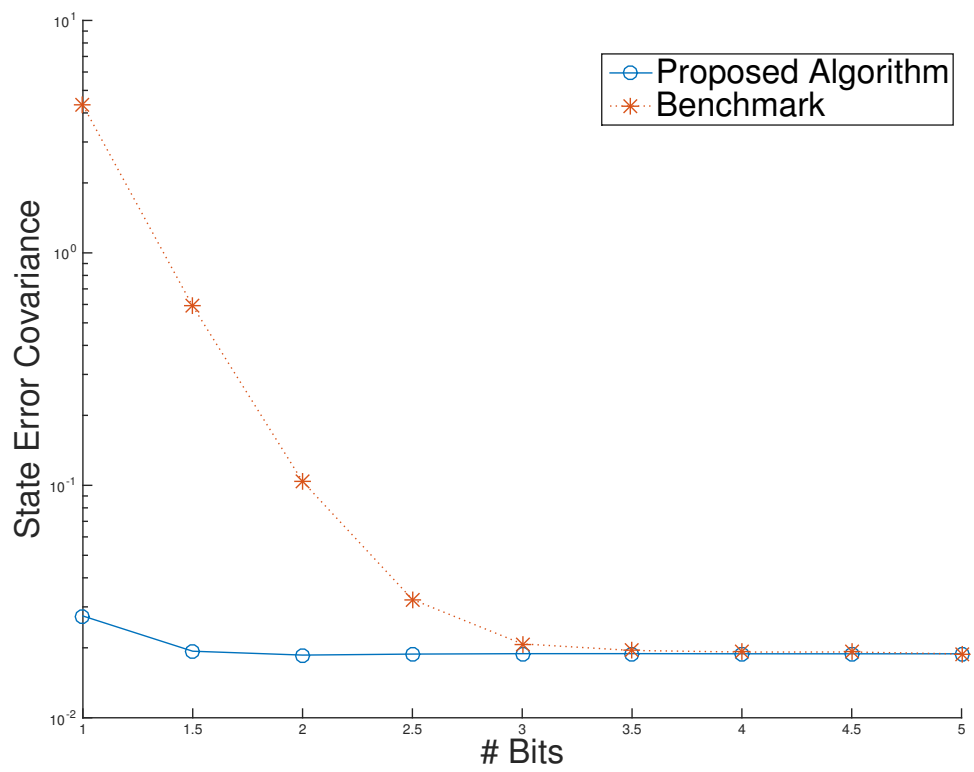


Figure 3.11: State error variance

3.4.2 Cooperative Reinforcement Q-learning

Here we test the methods by the cart-pole system shown in Fig. 3.12, and we assume that there are two sensors, while they are monitoring the observations of the cart-pole from two different angles, and one controller is trying to control the system by using two actions: push left, or push right. In the cart-pole system, the pole is attached by an un-actuated joint to a cart, which moves along a frictionless track.

System State and Feedback

The observations of the cart-pole system can be found in [44], and here we used all the four dimensions of observations as the observed system state of the dynamic system and feed it into the neural network at each encoder. As we mentioned before, we assume there is no communication between the two sensors. The system state of the four dimensions can be viewed as following

Num	Observation	Min	Max
Dimension 0	Cart Position	-2.4	2.4
Dimension 1	Cart Velocity	$-\text{Inf}$	Inf
Dimension 2	Pole Angle	-41.8	41.8
Dimension 3	Pole Velocity At Tip	$-\text{Inf}$	Inf

Note that, in the cart-pole system, the angle of pole is close to 90 degrees while the system is stable, and this phenomenon results in a limited number of the system states. The small number of system states makes the Q-table to be small, which makes it possible to run the small neural networks on this setting. For other CPSs, other neural networks might be a good try to learn the source coding schemes.

Actually, based on different settings of CPSs, the feedback from the controller performs differently. In this case, the feedback could largely reduce the communication cost (side information available). To simplify the process and focus on the performance of learning, we let the encode cache the previous quantized system state and use it as the feedback from the controller.

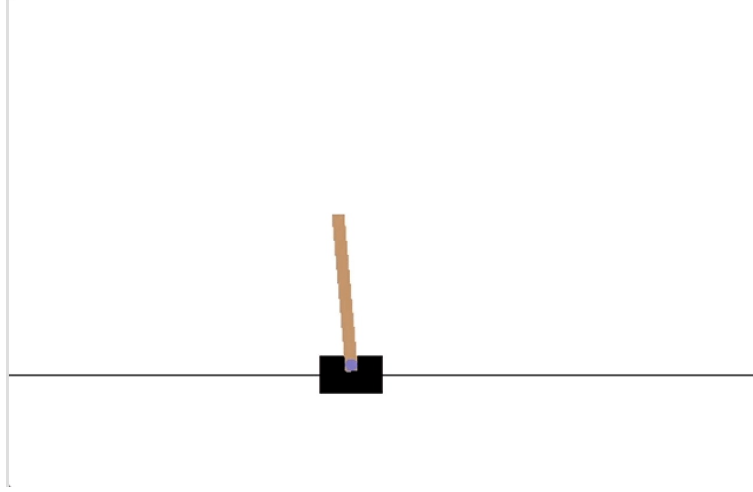


Figure 3.12: The cart-pole system in GYM toolbox

Training

Based on Algorithm 3, we simulate the system with the observation matrix $\mathbf{C}_1, \mathbf{C}_2 = [1, 1, 1, 1], [1, 1, 1, 1]$, which means the two sensors are observing full information from the dynamic system. And our quantization strategies are defined as the total bits to used to normally divide the range in each dimension of observations³.

In Fig. 3.13, we show the case where quantizations are applied, where the difference with the case of no quantization is that it is lossy in the sensing and communication. Here we denote horizontal axis as the total number of episodes passed and the vertical axis as the duration of the cart's running within the frame, which also means that the system is well controlled. And we use the blue line to denote the duration of each episode and red dot as the local mean of twenty episodes in order to view the trend of the train. Fig. 3.13 shows that after a certain number of episodes, a good approximation of action function $f_{decoder}$ can be obtained which is then lated used to control the cart and make the cart stable⁴. In summary, this simulation shows that to satisfy certain requirements of the control system, the lossless communication brings redundancy which might not even needed, which necessitated our work. And our proposed scheme does fulfill the requirements to achieve a good encoder from this point.

³In our simulation, we assume the maximum and minimum are fixed and known.

⁴Here, we set the target time duration as 200, and it is possible that we will never meet our goal. And in this case, we need to further improve the neural network.

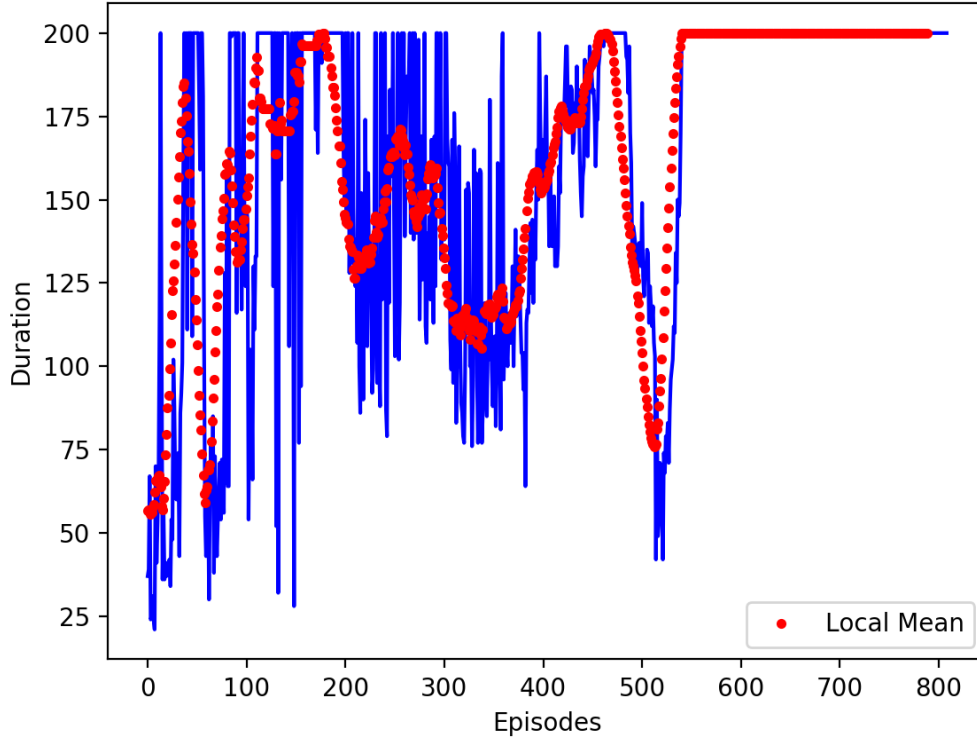


Figure 3.13: The duration of the cart

In Fig. 3.14 and Fig. 3.15, we show the total bits needed for each episode during the train and during the test. The system has been tested to show to reach the goal 149 for all different initial system states based on the trained $f_{encoder1}$, $f_{encoder2}$, $f_{decoder}$. Especially, in Fig. 3.14, we can tell that at the beginning of the training, the encoders are trying to minimize the total communication cost but since it hurts the system thus they turn to cooperate with the controller by increasing the revolution of the sensing results and finally the system turn out to be perfectly controlled with the cooperation. An animation is also shown in Youtube via “<https://youtu.be/diaRXgtvYOk>”.

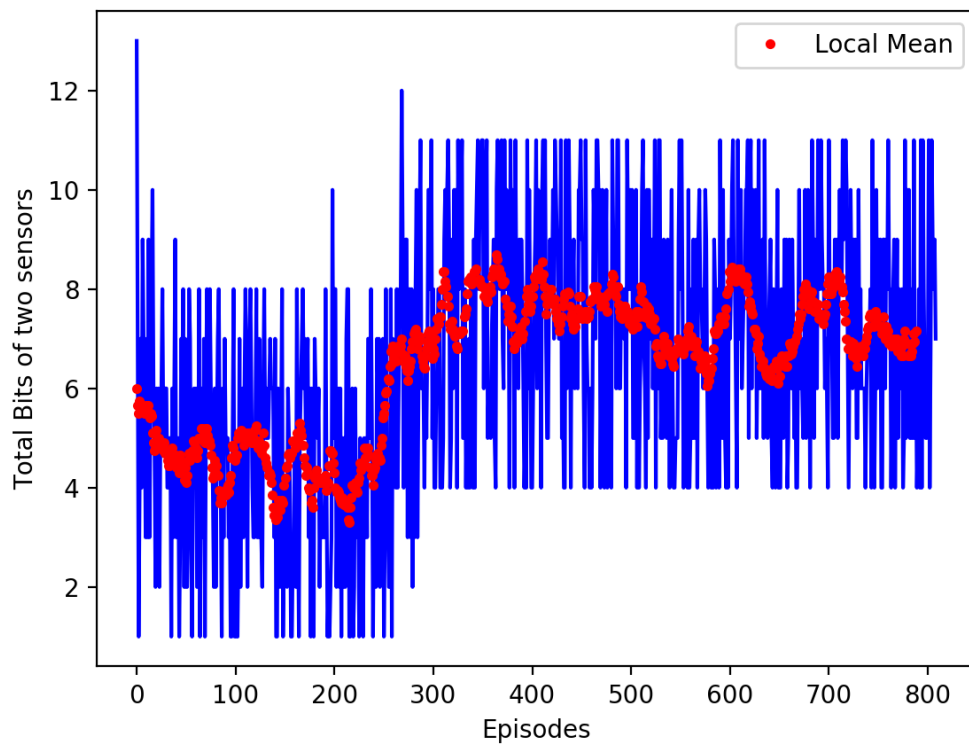


Figure 3.14: The stabilization of the system with two sensors

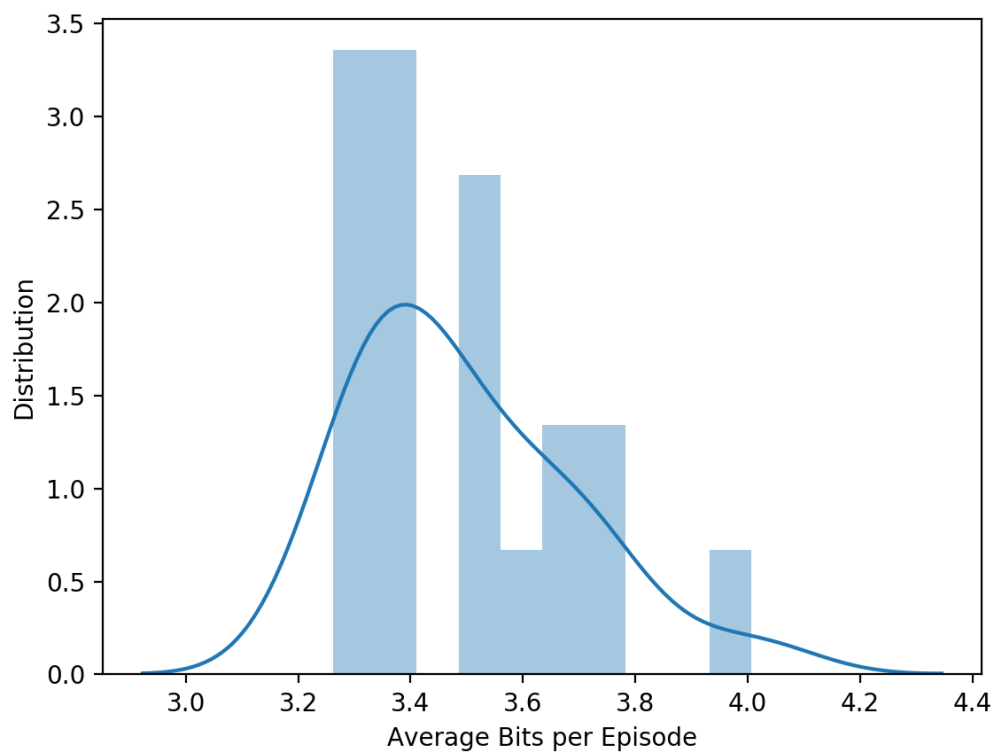


Figure 3.15: The distribution of the total bits needed for each episode

Chapter 4

Conclusion

In this dissertation, we have studied the source coding in CPS for two cases, i.e., one sensor and multiple sensors. First, we have formulated the system state aware source coding for one sensor case as a dynamic programming problem, and have exploited approximate dynamic programming to obtain a functional policy $f(\cdot)$, which is a deterministic but adaptive coding policy for feasible quantization. The auto-truck system has been used for one sensor case to test the proposed approximate dynamic programming method. Second, we have studied the distributed source coding for controlling the physical dynamics in CPS for the two sensors case. We have proposed two schemes to design the quantization lattices of observations at different sensors for the purpose of control. One is that we try to obtain the feedback from the controller and color the lattice points using the feedback, which is in a similar manner to Slepian-Wolf coding. The other is that we try to simplify the process by leveraging the reinforcement Q-learning, and we obtain a mapping function for each quantizer as the functional policy $f_{encoder}(\cdot)$. Numerical simulations based on cart-pole system have demonstrated the validity of the proposed scheme of distributed source coding in CPS.

In summary, by utilizing the mentioned methods to realize source coding in CPS, we have found the suboptimal solutions for both the one sensor case and the multiple sensors case in system controlling in CPS. By adopting these adaptive quantization policies in CPS, we show that these suboptimal solutions can reduce the communication cost while maintaining the stability of the system. These adaptive source coding policies are beneficial for practical applications with redundancy in data transmitting for CPS.

Bibliography

- [1] (2010). *SG Network System Requirements Specification*. Open Smart Grid(OpenSG). 2
- [2] (Accessed in August 2017). 20% wind energy by 2030: Increasing wind energy’s contribution to us electricity supply. US DOE Report DOE/GO-102008-2567, July 2008 [online], Available: <http://www1.eere.energy.gov/windandhydro/pdfs/41869.pdf>. 1
- [3] Aaron, A., Zhang, R., and Girod, B. (2002). Wyner-ziv coding of motion video. In *Conference Record of the Thirty-Sixth Asilomar Conference on Signals, Systems and Computers, 2002.*, volume 1, pages 240–244. IEEE. 8
- [4] Beard, R. W., McLain, T. W., Nelson, D. B., Kingston, D., and Johanson, D. (2006). Decentralized cooperative aerial surveillance using fixed-wing miniature uavs. *Proceedings of the IEEE*, 94(7):1306–1324. 1
- [5] Bertsekas, D. P. (1987). *Dynamic Programming: deterministic and stochastic models*. Prentice-Hall, Englewood Cliffs, NJ. 12, 21, 24
- [6] Borkar, V. S., Mitter, S. K., and Tatikonda, S. (2001). Optimal sequential vector quantization of markov sources. *SIAM Journal on Control and Optimization*, 40(1):135–148. 6
- [7] Bullo, F., Cortes, J., and Martinez, S. (2009). *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*. Princeton University Press. 1
- [8] Calvaresi, D., Marinoni, M., Sturm, A., Schumacher, M., and Buttazzo, G. (2017). The challenge of real-time multi-agent systems for enabling iot and cps. In *Proceedings of the International Conference on Web Intelligence*, pages 356–364. ACM. 1
- [9] Conway, J. H. and Sloane, N. J. A. (2013). *Sphere packings, lattices and groups*, volume 290. Springer Science & Business Media. 52
- [10] Dennis Jr, J. E. and Schnabel, R. B. (1996). *Numerical methods for unconstrained optimization and nonlinear equations*. SIAM. 24
- [11] Ding, K., Li, Y., Dey, S., and Shi, L. (2017). Multi-sensor transmission management for remote state estimation under coordination. *arXiv preprint arXiv:1703.09124*. 9

- [12] Dragotti, P. L. and Gastpar, M. (2009). *Distributed source coding: theory, algorithms and applications*. Academic Press. [8](#)
- [13] El Gamal, A. and Kim, Y.-H. (2011). *Network information theory*. Cambridge university press. [8](#)
- [14] Fleming, M., Zhao, Q., and Effros, M. (2004). Network vector quantization. *IEEE Transactions on Information Theory*, 50(8):1584–1604. [8](#)
- [15] Fowler, J. E. (1998). Generalized threshold replenishment: An adaptive vector quantization algorithm for the coding of nonstationary sources. *IEEE Transactions on Image Processing*, 7(10):1410–1424. [16](#)
- [16] Fu, M. and de Souza, C. E. (2009). State estimation for linear discrete-time systems using quantized measurements. *Automatica*, 45(12):2937–2945. [6](#)
- [17] Fu, M. and Xie, L. (2009). Finite-level quantized feedback control for linear systems. *IEEE Transactions on Automatic Control*, 54(5):1165–1170. [6](#)
- [18] Garcia-Frias, J. and Cabarcas, F. (2006). Approaching the slepian–wolf boundary using practical channel codes. *Signal Processing*, 86(11):3096–3101. [8](#)
- [19] Gersho, A. and Gray, R. M. (1992). *Vector quantization and signal compression*. Kluwer Academic Publishers. [19](#)
- [20] Gong, S., Li, L., Song, J. B., and Li, H. (2017). Joint channel decoding and state estimation in cyber-physical systems. *IEEE Transactions on Wireless Communications*, 16(11):7560–7573. [7](#), [9](#), [49](#), [68](#)
- [21] Grauman, K. and Darrell, T. (2007). Approximate correspondences in high dimensions. In *Advances in Neural Information Processing Systems*, pages 505–512. [33](#)
- [22] Group, C. S. S. (2015). Cps vision statement. [1](#)
- [23] Hadfield-Menell, D., Russell, S. J., Abbeel, P., and Dragan, A. (2016). Cooperative inverse reinforcement learning. In *Advances in neural information processing systems*, pages 3909–3917. [10](#)

- [24] Humayed, A., Lin, J., Li, F., and Luo, B. (2017). Cyber-physical systems security—a survey. *IEEE Internet of Things Journal*, PP(99):1–30. [1](#)
- [25] Jeong, J. and Lee, E. (2014). Vcps: vehicular cyber-physical systems for smart road services. In *Advanced Information Networking and Applications Workshops (WAINA), 2014 28th International Conference on*, pages 133–138. IEEE. [41](#)
- [26] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436. [10](#)
- [27] Lee, E. A. (2008). Cyber physical systems: Design challenges. In *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, pages 363–369. IEEE. [1](#)
- [28] Li, H. and Han, Z. (2014). Distributed source coding for controlling cyber physical systems with application in smart grids. In *2014 IEEE Global Communications Conference*, pages 1516–1521. IEEE. [46](#), [52](#), [53](#), [57](#)
- [29] Li, H., Lai, L., and Poor, H. V. (2012). Multicast routing for decentralized control of cyber physical systems with an application in smart grid. *IEEE Journal on Selected Areas in Communications*, 30(6):1097–1107. [9](#)
- [30] Li, L., Gong, S., Song, J. B., and Li, H. (2017a). Performance analysis on joint channel decoding and state estimation in cyber-physical systems. *EURASIP Journal on Wireless Communications and Networking*, 2017(1):158. [7](#), [9](#)
- [31] Li, L. and Li, H. (2016). Dynamic state aware source coding for networked control in cyber-physical systems. In *Global Communications Conference (GLOBECOM), 2016 IEEE*, pages 1–6. IEEE. [7](#), [9](#)
- [32] Li, L., Song, J. B., and Li, H. (2017b). Dynamic state aware adaptive source coding for networked control in cyberphysical systems. *IEEE Transactions on Vehicular Technology*, 66(11):10000–10010. [7](#), [9](#), [65](#)

- [33] Lin, H., Veda, S. S., Shukla, S. S., Mili, L., and Thorp, J. (2012). Geco: Global event-driven co-simulation framework for interconnected power system and communication network. *IEEE Transactions on Smart Grid*, 3(3):1444–1456. [2](#)
- [34] Lin, J., Yu, W., Zhang, N., Yang, X., Zhang, H., and Zhao, W. (2017). A survey on internet of things: architecture, enabling technologies, security and privacy, and applications. *IEEE Internet of Things Journal*, pages 1–17. [1](#)
- [35] Liu, Z., Cheng, S., Liveris, A. D., and Xiong, Z. (2006). Slepian-wolf coded nested lattice quantization for wyner-ziv coding: High-rate performance analysis and code design. *IEEE Transactions on Information Theory*, 52(10):4358–4379. [8](#)
- [36] Lloyd, S. (1982). Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137. [28](#)
- [37] Lu, R., Xu, Y., Xue, A., and Zheng, J. (2013). Networked control with state reset and quantized measurements: observer-based case. *IEEE Transactions on Industrial Electronics*, 60(11):5206–5213. [7](#)
- [38] Mahajan, A. and Teneketzis, D. (2009). Optimal design of sequential real-time communication systems. *IEEE Transactions on Information Theory*, 55(11):5317–5338. [6](#)
- [39] Matsuta, T., Uyematsu, T., and Matsumoto, R. (2010). Universal slepian-wolf source codes using low-density parity-check matrices. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 93(11):1878–1888. [8](#)
- [40] Mets, K., Ojea, J. A., and Develder, C. (2014). Combining power and communication network simulation for cost-effective smart grid analysis. *IEEE Communications Surveys & Tutorials*, 16(3):1771–1796. [2](#)
- [41] Miranda, M. J. and Fackler, P. L. (2002). *Applied computational economics and finance*. MIT press. [12](#), [22](#), [27](#)
- [42] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch. In *NIPS-W*. [66](#)

- [43] Pinto, L., Davidson, J., Sukthankar, R., and Gupta, A. (2017). Robust adversarial reinforcement learning. *arXiv preprint arXiv:1703.02702*. [64](#)
- [44] Plappert, M., Andrychowicz, M., Ray, A., McGrew, B., Baker, B., Powell, G., Schneider, J., Tobin, J., Chociej, M., Welinder, P., Kumar, V., and Zaremba, W. (2018). Multi-goal reinforcement learning: Challenging robotics environments and request for research. [66](#), [73](#)
- [45] Powell, W. B. (2007). *Approximate Dynamic Programming: Solving the curses of dimensionality*. John Wiley & Sons. [12](#)
- [46] Pradhan, S. S. and Ramchandran, K. (2003). Distributed source coding using syndromes (discuss): Design and construction. *IEEE Transactions on Information Theory*, 49(3):626–643. [8](#)
- [47] Pradhan, S. S. and Ramchandran, K. (2005). Generalized coset codes for distributed binning. *IEEE Transactions on Information Theory*, 51(10):3457–3474. [8](#)
- [48] Rabi, M., Ramesh, C., and Johansson, K. H. (2016). Separated design of encoder and controller for networked linear quadratic optimal control. *SIAM Journal on Control and Optimization*, 54(2):662–689. [7](#)
- [49] Rasool, F., Huang, D., and Nguang, S. K. (2012). Robust h output feedback control of networked control systems with multiple quantizers. *Journal of the Franklin Institute*, 349(3):1153–1173. [7](#)
- [50] Sha, L., Gopalakrishnan, S., Liu, X., and Wang, Q. (2009). Cyber-physical systems: A new frontier. In *Machine Learning in Cyber Trust*, pages 3–13. Springer. [1](#)
- [51] Slepian, D. and Wolf, J. K. (1973). A coding theorem for multiple access channels with correlated sources. *Bell System Technical Journal*, 52(7):1037–1076. [8](#)
- [52] Stankovic, J. A. (2014). Research directions for the internet of things. *IEEE Internet of Things Journal*, 1(1):3–9. [1](#)

- [53] Stankovic, V., Liveris, A. D., Xiong, Z., and Georghiades, C. N. (2006). On code design for the slepian-wolf problem and lossless multiterminal networks. *IEEE transactions on Information Theory*, 52(4):1495–1507. [8](#)
- [54] Tan, M. (1993). Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, pages 330–337. [10](#)
- [55] Tatikonda, S. and Mitter, S. (2004). Control under communication constraints. *IEEE Transactions on Automatic Control*, 49(7):1056–1068. [6](#)
- [56] Tatikonda, S., Sahai, A., and Mitter, S. (2004). Stochastic linear control over a communication channel. *IEEE Transactions on Automatic Control*, 49(9):1549–1561. [6](#)
- [57] Teneketzis, D. (2006). On the structure of optimal real-time encoders and decoders in noisy communication. *IEEE Transactions on Information Theory*, 52(9):4017–4035. [6](#)
- [58] Wang, Y., Li, H., and Qian, L. (2017a). Belief propagation and quickest detection-based cooperative spectrum sensing in heterogeneous and dynamic environments. *IEEE Transactions on Wireless Communications*, 16(11):7446–7459. [2](#)
- [59] Wang, Y., Zhang, Z., and Li, H. (2017b). Universal quickest sensing of spectrum change in millimeter wave communications: A data driven approach. In *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pages 1–6. IEEE. [2](#)
- [60] Xiao, B., Starke, M., Liu, G., Ollis, B., Irminger, P., Dimitrovski, A., Prabakar, K., Dowling, K., and Xu, Y. (2015). Development of hardware-in-the-loop microgrid testbed. In *2015 IEEE Energy Conversion Congress and Exposition (ECCE)*, pages 1196–1202. IEEE. [x](#), [4](#)
- [61] Yüksel, S. (2017). On stochastic stability of a class of non-markovian processes and applications in quantization. *SIAM Journal on Control and Optimization*, 55(2):1241–1260. [8](#)

- [62] Yüksel, S. and Başar, T. (2013). *Stochastic networked control systems: Stabilization and optimization under information constraints*. Springer Science & Business Media. [7](#)
- [63] Zamir, R., Shamai, S., and Erez, U. (2002). Nested linear/lattice codes for structured multiterminal binning. *IEEE Transactions on Information Theory*, 48(6):1250–1276. [8](#), [52](#)
- [64] Zanero, S. (2017). Cyber-physical systems. *Computer*, 50(4):14–16. [1](#)
- [65] Zhao, J. and Cao, G. (2008). Vadd: Vehicle-assisted data delivery in vehicular ad hoc networks. *IEEE transactions on vehicular technology*, 57(3):1910–1922. [41](#)
- [66] Zheng, X., Julien, C., Kim, M., and Khurshid, S. (2015). Perceptions on the state of the art in verification and validation in cyber-physical systems. *IEEE Systems Journal*, pages 1–14. [1](#)

Appendix

A Solar Power Truck

The auto truck has received interests in the area. In this part, I would introduce how I built a solar power truck for my labs.

A.1 Key Components

Briefly, I was using micro-controller MSP430 by TI company. In fact, based on different electronic components to be used, a small truck may have different functions, which can make the small truck to do different jobs. Here, we will give the truck example with module CC2500 and HY-SRF05, which enable the truck with wireless communication and distance sensing abilities.

Solar Battery

In Fig. [A.1](#), we show a common solar battery with the capacity as 10000mAh, and output: DC 5V/1A. Similar solar battery from other brands are also acceptable with the output voltage as 5V. In this lab, we use this solar battery to reduce the battery cost.

Truck Chassis Kit

As shown in Fig. [A.2](#), the car chassis includes 1 x Car Chassis, 2 x Gear Motor, 2 x Car Tire, 2 x Speed Encoder, 2 x Fastener, 1 x Universal Wheel, 1 x Screwdriver, 1 x Battery Box. Usually, an instruction is attached to the chassis package to show how to build the car chassis, so we will omit the details here. Note since we will use the solar battery to power the truck, the battery box will be replaced with the solar battery with some lines being replaced and connected.

Here, the minimal input power for each motor is 3V, and when we apply the power to the motor, it will always be running. That is not what we want. In order to control the motor, we need an L298N Dual H-Bridge Motor Controller module.

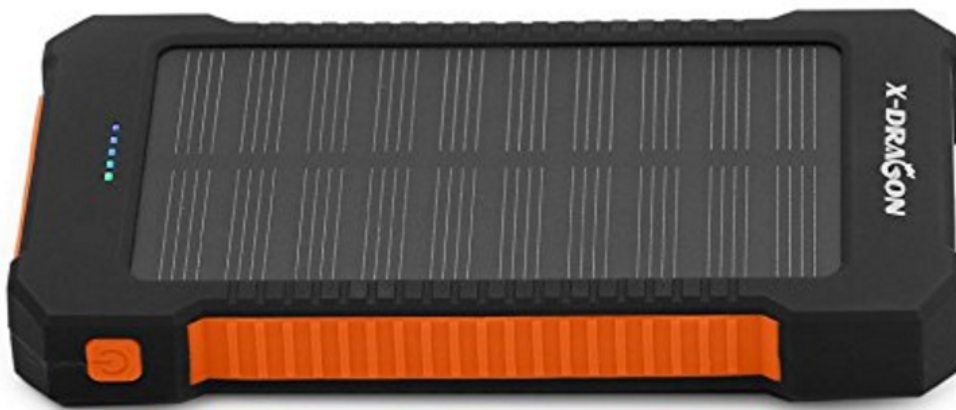


Figure A.1: Solar battery

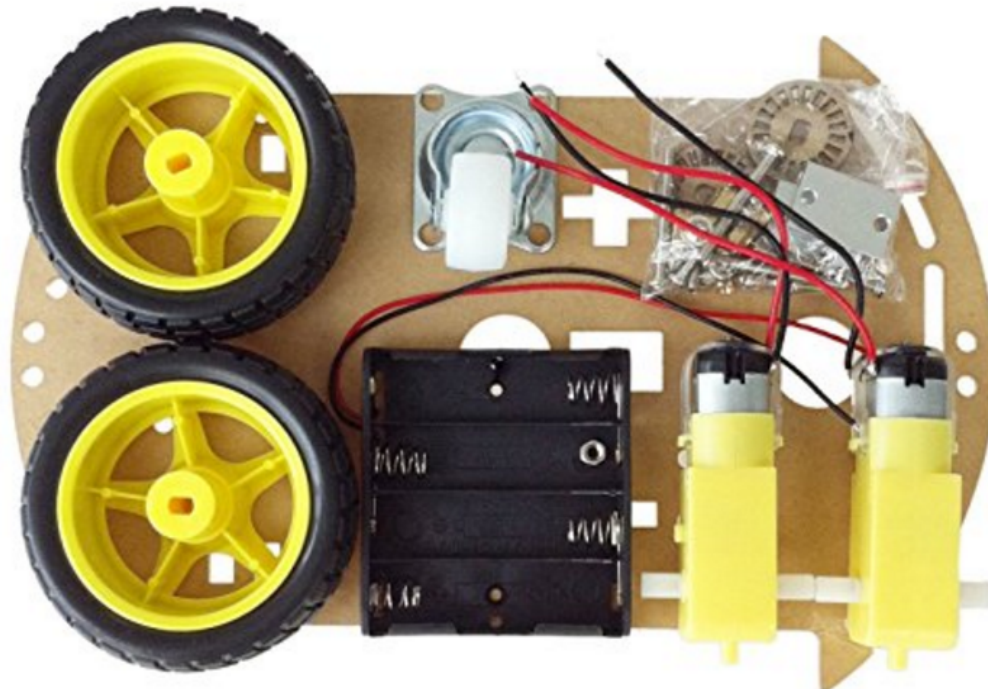


Figure A.2: Robot car chassis

L298N Dual H-Bridge Motor

The L298N H-Bridge is typically used in controlling motors' speed and direction. An H-Bridge is a circuit that can drive a current in either polarity and be controlled by Pulse Width Modulation (PWM), where PWM is a means in controlling the duration of an electronic pulse. In PWM, the longer the pulses the faster the wheel will turn, the shorter the pulses, the slower the water wheel will turn. One example is shown in Fig. [A.3](#)¹.

The pins for L298N are shown in Table [A.1](#).

MSP430

We use the micro-controller as MSP430 from TI, i.e., MSP-EXP430G2553 LaunchPad as shown in Fig. [A.4](#). The details about the launchpad can be found online with MSP430G2 data sheet. Here we would just cover the utility of the PIN as shown in Fig. [A.5](#). We will list the distribution of the PINs in next part.

The MSP-EXP430G2 LaunchPad development kit is an inexpensive and simple development kit for the MSP430G2xx Value Line series of micro-controllers. It is an easy way to start developing on the MSP430 MCUs with on-board emulation for programming and debugging as well as buttons and LEDs for a simple user interface. The MSP430G2553 16-bit MCU has 16KB of flash, 512 bytes of RAM, up to 16-MHz CPU speed, a 10-bit ADC, capacitive-touch enabled I/Os, universal serial communication interface, and more.

CC2500

Fig. [A.6](#) shows the communication module we use here. If no communication will be considered in the robot design, this part can be ignored. The CC2500 is a low-cost 2.4 GHz transceiver designed for very low-power wireless applications. The RF transceiver is integrated with a highly configurable baseband modem. The modem supports various modulation formats and has a configurable data rate up to 500 kBaud. The pin configuration is shown in Fig. [A.7](#).

¹Picture from <http://www.instructables.com/id/Arduino-Modules-L298N-Dual-H-Bridge-Motor-Controll/>

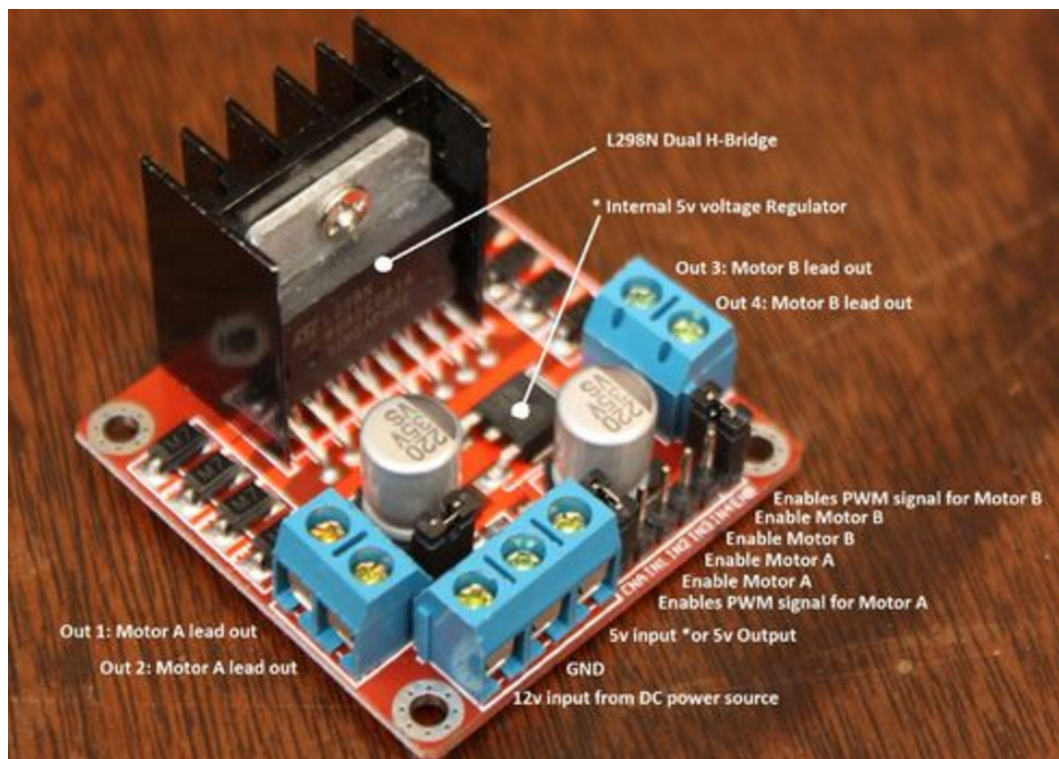


Figure A.3: L298N

Table A.1: The pins for L298N

<i>Pin</i> Functions
<i>Out1</i> : Motor A lead out
<i>Out2</i> : Motor A lead out
<i>Out3</i> : Motor B lead out
<i>Out4</i> : Motor B lead out
<i>GND</i> : Ground
<i>5v</i> : 5v input ²
<i>EnA</i> : Enables PWM signal for Motor A
<i>In1</i> : Enable Motor A
<i>In2</i> : Enable Motor A
<i>In3</i> : Enable Motor B
<i>In4</i> : Enable Motor B
<i>EnB</i> : Enables PWM signal for Motor B

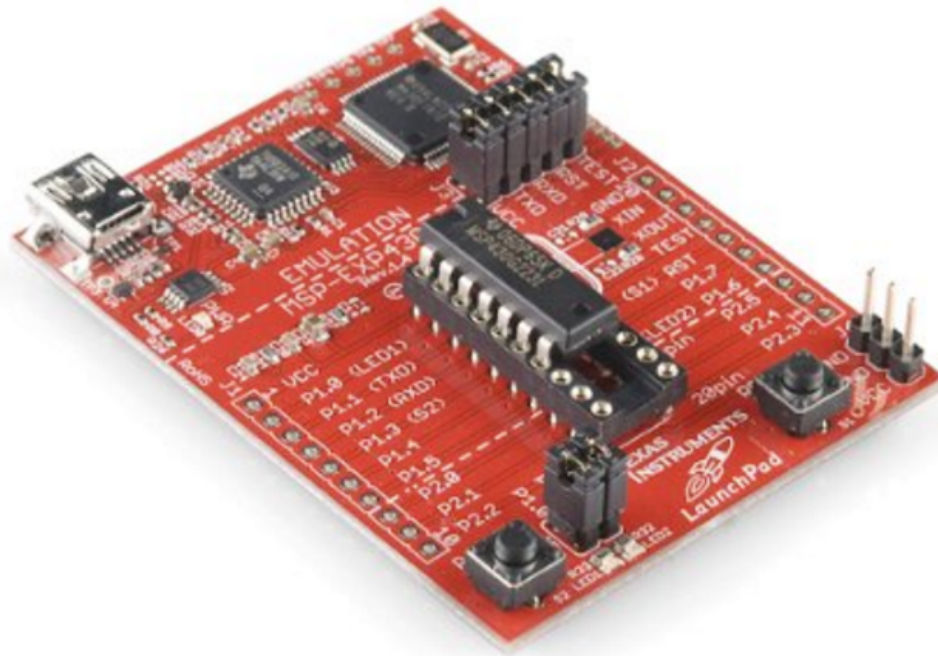
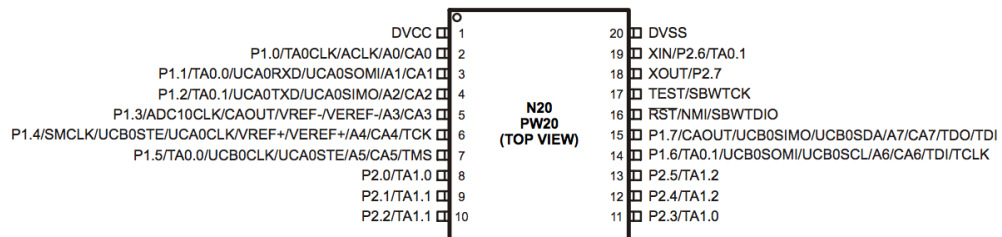


Figure A.4: MP430 by TI



NOTE: ADC10 is available on MSP430G2x53 devices only.

NOTE: The pulldown resistors of port P3 should be enabled by setting P3REN.x = 1.

Figure A.5: The device pin out for MSP430G2



Figure A.6: Wireless communication module CC2500

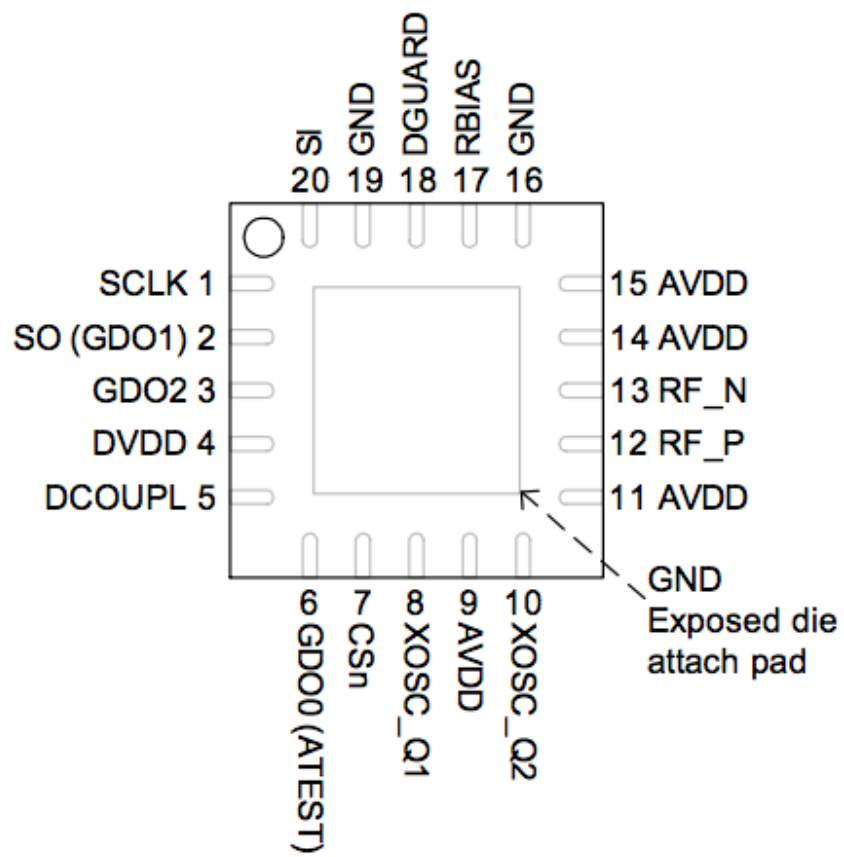


Figure A.7: CC2500 module

Ultrasonic Sensor

Fig. [A.8](#) shows the ultrasonic sensor which provides distance measurement regardless of color and lighting of obstacles. The measurement procedure can be shown in Fig. [A.9](#).

Other Tools

Fig. [A.10](#) shows the Jumpers from Female to Female, which can be used to connect the different components. The programming line works with MSP430 is shown in Fig. [A.11](#), and the circuit welding can be tested with digital multimeter as shown in Fig. [A.12](#).

A.2 Truck Circuits

The whole circuit is shown in Fig. [A.13](#), where the PINs distribution is listed in Table [A.2](#).

Conclusion

In this part, we briefly introduce a way to build a robot truck with wireless communication and distance sensing functions. Since only 20 Pins are available in our MSP430 launch board, limited functions are implemented. However, these procedures can be transplanted to build a bigger project with more functional boards.



Figure A.8: HY-SRF05

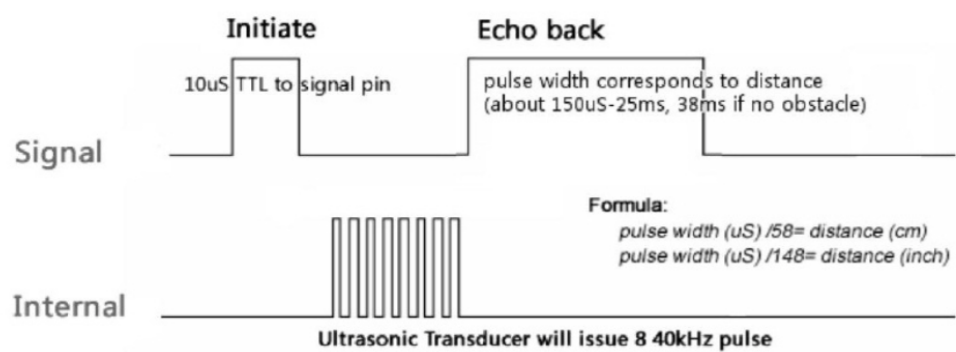


Figure A.9: HY-SRF05 measurement procedure



Figure A.10: Female to female jumper



Figure A.11: Programming line with MSP430



Figure A.12: Digital multimeter

Table A.2: PINs distribution

MSP430 PIN	Connected	Fuctions
P1.0	LED/AD	LED/AD
P1.1	TXD	UART
P1.2	RXD	UART
P1.3	KEY/AD	KEY/AD
P1.4	AD	AD
P1.5	SPI-CLK(CC2500)	Wireless module
P1.6	SPI-SO(CC2500)	Wireless module
P1.7	SPI-SI(CC2500)	Wireless module
P2.0	SPI-CS(CC2500)	Wireless module
P2.1	IN1/IN3(L298N)	Motor Driver
P2.2	IN2(L298N)	Motor Driver(Left)
P2.3	GDO0(CC2500)	Wireless module
P2.4	IN4(L298N)	Motor Driver(Right)
P2.5	GDO2(CC2500)	Wireless module
P2.6	Trig(HY-SPR05)	Ultrasonic module
P2.7	Echo(HY-SPR05)	Ultrasonic module

Vita

Liang Li received the B.S. and M.S. degrees in communication engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2010 and 2013, respectively. She is currently pursuing the Ph.D. degree in electrical engineering with the Department of Electrical Engineering and Computer Science, The University of Tennessee, Knoxville, TN, USA. Her current research interests include spectrum management, source and channel coding, machine learning, and wireless communication.