Scalable Video Transmission over Wireless Networks

by

Siyuan Xiang
B.Eng., Hangzhou Dianzi University, 2004
M.Eng., Tongji University, 2008

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Electrical and Computer Engineering

Scalable Video Transmission over Wireless Networks

by

Siyuan Xiang
B.Eng., Hangzhou Dianzi University, 2004
M.Eng., Tongji University, 2008

Supervisory Committee

Dr. Lin Cai, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Wu-Sheng Lu, Departmental Member
(Department of Electrical and Computer Engineering)

Dr. Alex Thomo, Outside Member
(Department of Computer Science)

Supervisory Committee

Dr. Lin Cai, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Wu-Sheng Lu, Departmental Member
(Department of Electrical and Computer Engineering)

Dr. Alex Thomo, Outside Member
(Department of Computer Science)

ABSTRACT

With the increasing demand of video applications in wireless networks, how to better support video transmission over wireless networks has drawn much attention to the research community. Time-varying and error-prone nature of wireless channel makes video transmission in wireless networks a challenging task to provide the users with satisfactory watching experience. For different video applications, we choose different video coding techniques accordingly. E.g., for Internet video streaming, we choose standardized H.264 video codec; for video transmission in sensor networks or multicast, we choose simple and energy-conserving video coding technique based on compressive sensing. Thus, the challenges for different video transmission applications are different. Therefore, This dissertation tackles video transmission problem in three different applications.

First, for dynamic adaptive streaming over HTTP (DASH), we investigate the streaming strategy. Specifically, we focus on the rate adaptation algorithm for streaming scalable video (H.264/SVC) in wireless networks. We model the rate adaptation problem as a Markov Decision Process (MDP), aiming to find an optimal streaming strategy in terms of user-perceived quality of experience (QoE) such as playback interruption, average playback quality and playback smoothness. We then obtain the optimal MDP solution using dynamic programming. However, the optimal solution

requires the knowledge of the available bandwidth statistics and has a large number of states, which makes it difficult to obtain the optimal solution in real time. Therefore, we further propose an online algorithm which integrates the learning and planning process. The proposed online algorithm collects bandwidth statistics and makes streaming decisions in real time. A reward parameter has been defined in our proposed streaming strategy, which can be adjusted to make a good trade-off between the average playback quality and playback smoothness.We also use a simple testbed to validate our proposed algorithm.

Second, for video transmission in wireless sensor networks, we consider a wireless sensor node monitoring the environment and it is equipped with a compressive-sensing based, single-pixel image camera and other sensors such as temperature and humidity sensors. The wireless node needs to send the data out in a timely and energy efficient way. This transmission control problem is challenging in that we need to jointly consider perceived video quality, quality variation, power consumption and transmission delay requirements, and the wireless channel uncertainty. We address the above issues by first building a rate-distortion model for compressive sensing video. Then we formulate the deterministic and stochastic optimization problems and design the transmission control algorithm which jointly performs rate control, scheduling and power control.

Third, we propose a low-complex, scalable video coding architecture based on compressive sensing (SVCCS) for wireless unicast and multicast transmissions. SVCCS achieves good scalability, error resilience and coding efficiency. SVCCS encoded bitstream is divided into base and enhancement layers. The layered structure provides quality and temporal scalability. While in the enhancement layer, the CS measurements provide fine granular quality scalability. We also investigate the rate allocation problem for multicasting SVCCS encoded bitstream to a group of receivers with heterogeneous channel conditions. Specifically, we study how to allocate rate between the base and enhancement layer to improve the overall perceived video quality for all the receivers.

Contents

List of Tables

List of Figures

List of Abbreviations

APQ	Average Playback Quality
AVC	Advanced Video Coding
CBR	Constant Bit-Rate
CS	Compressive Sensing
DASH	Dynamic Adaptive Streaming over HTTP
DCT	Discrete Cosine Transform
DPI	Dots Per Inch
GOP	Group of Pictures
IR	Interruption Ratio
HTTP	Hyper-Text Transport Protocol
MDP	Markov Decision Process
NAL	Network Abstraction Layer
NAT	Network Address Translation
OSMF	Open Source Media Framework
P2P	Peer-to-Peer
PS	Playback Smoothness
PSNR	Peak Signal to Noise Ratio
QoE	Quality of Experience
QoS	Quality of Service
RS	Reed-Solomon
SVC	Scalable Video Coding
SVCCS	Scalable Video Coding with Compressive Sensing
TV	Total Variation
UWT	Undecimated Wavelet Transform
VBR	Variable Bit-Rate

ACKNOWLEDGEMENTS

I would like to express my appreciation to my supervisor Dr. Lin Cai for her guidance, support and encouragement throughout my Ph.D. study. This dissertation would have not been possible without her help.

I would like to thank Dr. Wu-Sheng Lu for detailed explanation of fundamentals of Compressive Sensing, which is an important basis of this dissertation. I would like to thank Dr. Jianping Pan for valuable ideas and comments in our cooperation works.

I gratefully acknowledge my supervisory committee, Dr. Wu-Sheng Lu, Deparment of Electrical and Computer Engineering and Dr. Alex Thomo, Department of Computer Science for their valuable advice on my research work. I would like to thank my external examiner, Dr. Jie Liang, School of Engineering Science, Simon Fraser University for making my dissertation complete.

Thanks to many of my colleagues and friends at University of Victoria, it is a pleasure to study and live in Victoria. Especially, I would like to thank Ruonan Zhang, Haoling Ma, Yuanqian Luo, Zhe Yang, Le Chang, Bojiang Ma, Yan Jie, Vivek Tiwari, Min Xing, Lei Zheng, Xuan Wang, Kan Zhou, Yi Chen, Haoyuan Zhang and Zhe Wei.

I would like to thank my parents and parents-in-law for their encouragement and support during my Ph.D. study. I would also like to thank my wife Jiaping for her love and patience. I am grateful for having Jiaping in my life.

Siyuan Xiang, Victoria, BC, Canada

DEDICATION

To my dearest wife and parents

Chapter 1

Introduction

1.1 Background

During the past two decades, with the advancement of video compression and wireless communication technology, video applications, e.g., Internet video, video conference and surveillance are becoming prevalent. Recent statistics show that video applications account for the highest percentage of the traffic mix in the Internet. Cisco forecasts that the sum of all forms of video, including TV, video on demand, Internet video, and peer-to-peer (P2P) video, etc., will exceed 90% of the global consumer traffic by 2015, while mobile video will account for more than 50% of the total mobile traffic [12]. Video transmission over wireless network presents many challenges. For example, to ensure Quality of Experience (QoE) of the users, efficient rate adaptive transmission is required to fully utilize the bandwidth while avoiding video playback interruptions to deal with the bandwidth fluctuations of the wireless cannel; error resillient video bitstream is desirable to deal with error-prone nature of the wireless channel.

For different video applications, we choose different video coding techniques accordingly. E.g., for Internet video streaming, we choose standardized H.264 video codec; for video transmission in sensor networks or multicast, we choose simple and energy-conserving video coding technique based on compressive sensing. Thus, the challenges for different video transmission applications are different.

For video streaming in wireless Internet, the increasingly popular video websites such as YouTube and Vimeo will be the major providers of mobile videos. Progressive download is currently the dominant video delivery techniques of these video websites.

It has several advantages over the traditional streaming techniques using RTP/UDP. First, it is simple to deploy. At the server side, any web server can host videos and serve as a streaming server; at the client side, the user only needs a flash player or web browser supporting HTML5 for video playback. Second, the HTTP/TCP protocols used in progressive download are more firewall and network address translation (NAT) friendly, and the congestion control mechanism in TCP simplifies the design of the application layer. Third, for progressive download, a server can store several versions of a video to meet the requirements of heterogeneous users, so a user can select the right version of the video according to the device decoding capability, display size and available network bandwidth.

However, selecting the appropriate version of a video to match the available bandwidth may not be easy for users and their decisions might be error-prone. In addition, with progressive download, the client always downloads as much video data as possible. [32] reports that only half of the videos are fully downloaded and this number drops dramatically when the users are not satisfied with the video quality. It is likely that when a user turns off the video player or switches to another video, a large amount of un-watched video has been buffered unnecessarily, which wastes the resources of both the network and the end-systems. It is particularly undesirable for mobile devices with limited energy supply.

Dynamic adaptive streaming over HTTP (DASH) [41] is a promising technique to overcome the aforementioned disadvantages of progressive download. Videos encoded in different versions are chopped into small segments. After the client receives one segment, it has a chance to decide which version of the video to request for the next segment, based on the current network condition. Thus, rate adaptation can be performed at the client side naturally and flexibly. Also, the client has a chance to control the client-side queue length to avoid streaming buffer overflow, e.g., when the download rate is much higher than the playback rate.

Currently, commercial adaptive streaming products such as Microsoft Smooth Streaming and Apple Live Streaming support single-layer H.264 advanced video coding (AVC) encoded videos. Multiple versions of a video with different resolution, frame rate and quality are obtained by encoding the source video multiple times with different configurations, and the different versions of the video are completely independent to each other. Thus, not only more server storage space is needed, but also the web caching hit-ratio is reduced.

Recently, scalable video coding (H.264/SVC) has been introduced to the DASH

framework to improve the system performance [37]. With SVC, a video is encoded once only, and it can be decoded many times with different resolution, frame rate and quality. However, how to improve the rate adaptation algorithm to provide users with a satisfactory quality of experience (QoE) is still a challenging and open question. The problem is even more challenging when a user uses a handheld device via a wireless access link for video streaming, as the handheld devices typically have limited energy supply and computation capacity, and the wireless links are highly dynamic due to the time-varying fading, shadowing, interference and hand-off, all of which motivate this work.

On the other hand, for video transmission in wireless networks, compressive sensing (CS) has been an active research area in signal processing and communication societies recently. Unlike traditional transform domain compression method which acquires the complete image signal then compress the signal by removing the redundancy, CS unifies these two operations by making random linear projection of the source signals. We have seen the first application of CS in image acquisition single-pixel image camera [17]. Thanks to its simplicity and less power requirement, it has a good potential to be widely deployed in wireless sensor nodes. In addition, the acquired measurements have special property referred as "democracy", i.e., the measurements are equally important and the more measurements generated, the better quality of the recovered image signal. This characteristic is promising to make the image and video bitstream more error resilient and scalable. Therefore, we study how to take the advantage of the salient feature of compressive sensing in video transmission in wireless network.

In order to reduce the error and erasure of wireless channels, error correction coding such as Reed-Solomon (RS) code and convolutional code has been widely used. However, this type of channel coding is not flexible. It can correct the bit errors only if the error rate is smaller than a given threshold. Therefore, it is hard to find a single channel code suitable for unknown or varying wireless channels. For unicast applications, retransmission in the link layer or the transport layer can help recover the errors at the cost of delay. When we utilize the broadcast nature of wireless medium to multicast video, due to the independence of different receivers' channels, the data needed to retransmit are different for different receivers, which makes retransmission difficult and expensive.

Can we find a flexible channel coding for wireless unicast and multicast? That is, for a wide range of channel error rate, the effectiveness of channel coding degrades

gracefully when the channel condition becomes worse. In addition, for multicast applications, without the feedback from individual receivers, the sender can transmit more data that are helpful to all the receivers. These requirements are indeed difficult and challenging for traditional channel coding design. Fortunately, compressive sensing technologies can help to achieve the above goals.

If we only rely on compressive sensing as an image compression method, there is a huge gap in terms of coding efficiency between compressive sensing and conventional coding methods [19]. Although compressive sensing has the advantage of being a joint source and channel coding, its coding efficiency needs to be improved, since minimizing bandwidth consumption is one of the most important goals in codec design, particularly for wireless transmissions.

This dissertation focuses on 1) efficient DASH rate adaptation algorithm; 2) transmission control algorithm for CS video in wireless sensor networks; 3) scalable video codec design and transmission algorithm for wireless videocast.

1.2 Problem Statement

This dissertation includes three thrusts, motivated by the following three important issues.

- **Dynamic Rate Adaptation for Adaptive Video Streaming in Wireless Networks**

  In a DASH system, compressed video is stored at the web server with different quality, resolution and bitrate. For each version, the video is splitted into small segments. The client-side rate adaptation algorithm at the video player can request appropriate video version to adapt the bitrate of the video to the available bandwidth. The rate adaptation algorithm can adjust the requested video version based on the varying available bandwidth to fully utilize the network bandwidth and avoid possible playback interruption. However, frequent quality switching may lead to inferior user watching experience. Therefore the rate adaptation algorithm has an important impact on user watching experience.

  Recently, Scalable Video Coding (SVC) has been considered in a DASH system. It is promising to improve the DASH system performance and user QoE. With SVC, the source video can be compressed once with different quality, resolution and frame rate. It also provides the rate adaptation algorithm the capability to

"upgrade" the already received video to better quality, which can improve user QoE. How to design an efficient rate adaptation algorithm for streaming SVC over HTTP is still an open issue.

- **Transmission Control for Compressive Sensing Video over Wireless Channel**

  As mentioned in the previous section, the compressive sensing video coding technology has a good potential to be deployed in wireless sensor nodes. In this dissertation, we consider a wireless sensor node monitoring the environment and it is equipped with a single-pixel camera and some other sensors such as temperature and humidity sensors. The sensor node needs to send the data out in a timely and energy efficient way. This is not a trivial problem which involves the following issues and requirements. 1) The first issue is rate allocation. The single-pixel image camera can capture the image and obtain a fixed number of measurements for each video frame. We need to determine the number of measurements transmitted for each image while considering the transmission power consumption, the wireless channel condition and the recovered image quality. To achieve a better perceived video quality, we need to avoid large image quality fluctuations. 2) The second issue is scheduling. The CS acquired measurements are scalable and the traffic is elastic, while the packets from the other sensors may be non-elastic. Due to the heterogeneity, it is necessary to differentiate these two types of traffic. Therefore, we need to optimize the scheduling of packet transmissions from different traffic flows on each time slot. 3) The third issue is power allocation. In general, we can minimize the transmission power by sending less measurements. However, we need to take both the wireless channel conditions and the recovered image quality into consideration. 4) The last issue is the delay constraint. In order to deliver the packets within a reasonable delay, we need to keep the transmission queues stable. How to design an efficient transmission control algorithm which addresses the above issues is a challenging problem.

- **Scalable Video Coding with Compressive Sensing for Wireless Videocast**

  CS has been used as a joint source and channel coding for video transmission over wireless channel. Due to the "democracy" property of the measurements,

it has a good potential to outperform the traditional error correction coding such as Reed-Solomon (RS) code and convolutional code. Because this type of channel coding is not flexible. It can correct the bit errors only if the error rate is smaller than a given threshold.

If we only treat compressive sensing as an image compression method, there is a huge gap in terms of coding efficiency between compressive sensing and conventional coding methods [19]. Although compressive sensing has the advantage of being a joint source and channel coding, its coding efficiency needs to be improved, since minimizing bandwidth consumption is one of the most important goals in codec design, particularly for wireless transmissions. How to design a low-complex, scalable video coding architecture based on compressive sensing and efficient transmission algorithm is still an open and challenging research issue.

1.3 Contributions

This dissertation makes following three contributions.

- **Dynamic Rate Adaptation for Adaptive Video Streaming in Wireless Networks**

  In this dissertation, we investigate the streaming strategy for dynamic adaptive streaming over HTTP (DASH). Specifically, we focus on the rate adaptation algorithm for streaming scalable video (H.264/SVC) in wireless networks. We model the rate adaptation problem as a Markov Decision Process (MDP), aiming to find an optimal streaming strategy in terms of user-perceived quality of experience (QoE) such as playback interruption, average playback quality and playback smoothness. We then obtain the optimal MDP solution using dynamic programming. However, the optimal solution requires the knowledge of the available bandwidth statistics and has a large number of states, which makes it difficult to obtain the optimal solution in real time. Therefore, we further propose an online algorithm which integrates the learning and planning process. The proposed online algorithm collects bandwidth statistics and makes streaming decisions in real time. A reward parameter has been defined in our proposed streaming strategy, which can be adjusted to make a good trade-off between the average playback quality and playback smoothness. We also use a

simple testbed to validate our proposed algorithm. Experimental results show the feasibility of the proposed algorithm and its advantage over the existing work.

- **Transmission Control for Compressive Sensing Video over Wireless Channel**

  This dissertation considers a wireless sensor node monitoring the environment and it is equipped with a compressive-sensing based, single-pixel image camera and other sensors such as temperature and humidity sensors. The wireless node needs to send the data out in a timely and energy efficient way. This transmission control problem is challenging in that we need to jointly consider perceived video quality, quality variation, power consumption and transmission delay requirements, and the wireless channel uncertainty. The above issues are addressed by first building a rate-distortion model for compressive sensing video. Then we formulate the deterministic and stochastic optimization problems and design the transmission control algorithm which jointly performs rate allocation, scheduling and power allocation. Extensive simulations have been conducted to demonstrate the effectiveness of the proposed transmission control algorithm.

- **Scalable Video Coding with Compressive Sensing for Wireless Videocast**

  In this dissertation, we propose a low-complex, scalable video coding architecture with compressive sensing (SVCCS) for wireless unicast and multicast transmissions. SVCCS achieves good scalability, error resilience and coding efficiency. SVCCS encoded bitstream is divided into the base and enhancement layers. The layered structure provides quality and temporal scalability. In the enhancement layer, the CS measurements provide fine granular quality scalability. The state-of-the-art technologies including analysis-based ℓ_1 optimization are incorporated to improve the compressive sensing coding efficiency. In addition, we investigate the rate allocation problem for multicasting SVCCS encoded bitstream to a group of receivers with heterogeneous channel conditions. Specifically, we study how to allocate rate between the base and enhancement layer to improve the overall perceived video quality for all the receivers while satisfying the real-time video transmission delay requirement. We first build a rate distortion model to capture the rate distortion charactersitics of the SVCCS

encoded bitstream. Then we propose a rate allocation algorithm using this model. Simulation results show that SVCCS is more effective and efficient for wireless videocast than the existing solutions. We also demonstrate the accuracy of the proposed rate distortion model and the effectiveness the proposed rate allocation algorithm.

1.4 Dissertation Organization

The rest of the dissertation is organized as follows.

Chapter 2 presents the the rate adaptation algorithm for streaming SVC in a DASH system. First we describe the motivation of the work, followed by a review of the related work and background. Then we formulate the rate adaptation problem as an MDP problem. Based on this framework, we propose offline and online rate adaptation algorithms. To evaluate the proposed algorithm, we conduct simulation and experiment to verify these algorithms.

Chapter 3 presents the transmission control problem for CS video over wireless channel. First, we give a brief description of the background of CS. Then we build a rate-distortion model for CS video. Based on this model, we formulate the deterministic and stochastic optimization problems and solve them respectively. Finally, we conduct extensive simulations to evaluate the effectiveness of the proposed algorithms.

Chapter 4 presents the low-complex scalable video coding architecture design and transmission algorithm for wireless video multicast. First we describe the coding/decoding architecture design. Then we formulate and solve the rate allocation algorithm. Finally, simulations are conducted to verify the proposed coding architecture and rate allocation algorithms.

Chapter 5 concludes the dissertation and suggest the future research directions.

1.5 Bibliographic Notes

Most of the work in this dissertation have appeared in research papers. The work in Chapter 2 has been published in [51, 50]. The work in Chapter 3 has appeared in [49]. The work in Chapter 4 is based on research paper [48].

Chapter 2

Dynamic Rate Adaptation for Adaptive Video Streaming in Wireless Networks

In this chapter, we design the rate adaptation algorithm for streaming scalable video over HTTP in wireless networks. The main contributions of this chapter are three-fold. First, we formulate the rate adaptation problem as a finite MDP, aiming to find an optimal streaming strategy in terms of user-perceived QoE such as playback interruption, average playback quality and playback smoothness. We obtain the optimal streaming strategy by dynamic programming under the reinforcement learning framework [44]. A reward parameter is defined in our proposed strategy, which can be adjusted to make a trade-off between average playback quality and smoothness. Second, since the optimal solution requires the knowledge of available bandwidth statistics and has a high computational complexity, which makes it difficult to obtain the optimal solution in real time. We propose an online algorithm which integrates the learning and planning process, i.e., the proposed algorithm collects bandwidth statistics and makes streaming decisions in real time. Third, we have prototyped a scalable video streaming framework including the server-side video pre-processing and client-side SVC video player. A real sample video encoded in SVC is used to evaluate the proposed streaming strategies and compare them with the existing work using both wireless testbed experiments and simulations. The experimental and simulation results show the advantage of the proposed algorithms.

The rest of the chapter is organized as follows. Section 2.1 summarizes the related

work. Section 2.2 formulates the optimal streaming problem as an MDP. Section 2.3 presents the proposed optimal streaming policy and the online algorithm. The evaluation framework, testbed configurations and experimental results are described and given in Section 2.4, followed by concluding remarks and further research issues in Section 2.5.

2.1 Background and Related Work

Different from the application-layer multicasting [21], in a DASH system, rate adaptation is conducted at the client side, which is also called pull-based rate adaptation [5]. At the server side, a source video is encoded into different versions with different resolution, frame rate and quality. For each version, the video is divided into small segments. A web server can host these segments and send them to the clients upon HTTP requests. At the client side, after a user clicks the play button, the streaming starts. The video player first obtains the general information of the video, such as the number of versions and the corresponding resolution, frame rate and quality of each version. Then, the video player will decide the right version according to its own display size, decoding capability and network condition. Usually, the playback does not start until a sufficient number of segments are received. After the client receives a segment completely, the rate adaptation algorithm will decide which version to request for the next segment based on the current network condition and the client-side state such as the number of buffered segments. In this way, the workload of the server is reduced dramatically. Figure 2.1 shows the general workflow of the video player.

There are extensive research efforts on adaptive video streaming over HTTP [41, 24, 2, 20, 27, 13, 26, 23]. [41] introduced the 3GPP specification of dynamic adaptive streaming over HTTP, which describes the framework of the adaptive streaming system. In [2], the commercial adaptive streaming products including Microsoft Smooth Streaming, Netflix player and open source media framework (OSMF) player were evaluated and compared. The results show that the performance of these products still needs to be improved substantially.

Liu et al. proposed a rate adaptation algorithm for adaptive video streaming [24]. The decision of switching to a video version of a higher or lower bit-rate is made based on the measured segment fetch time, which can be converted to the average segment throughput and buffer state. The algorithm is evaluated using constant bit-rate (CBR), single-layer video traffic only, and the queue length may sometimes

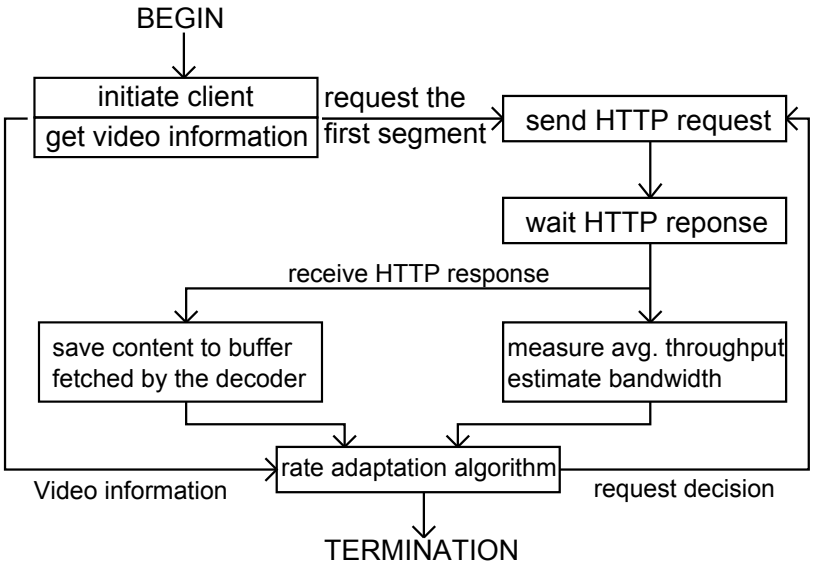


Figure 2.1: Video Player State Diagram

exceed the maximum buffer size. In [14], a quality-adaptation controller based on the feedback control theory was proposed. The controller tries to maintain the buffer level as stable as possible to match the video bit-rate with the available bandwidth. As the server needs to maintain the information for each user to perform rate adaptation, the complexity of the server is increased.

Recently, SVC has been introduced to adaptive video streaming. With SVC, we can encode video once and decode the bitstream multiple times with different resolution, frame rate and video quality [7, 55], so the server storage space and encoding time can be saved. In addition, thanks to the layered structure of SVC, we may even upgrade an already received segment to a higher quality [38]. [37] showed the advantage of using SVC in adaptive HTTP streaming over the single-layer AVC in terms of caching efficiency. In [38], the authors proposed a priority-based media delivery strategy using SVC with RTP and HTTP streaming. In the pre-buffering phase, the most important base layer is transmitted first, so there are more base-layer frames than enhancement-layer frames in the buffer. This scheme was designed assuming that the temporary bandwidth reduction is the only possible bandwidth variation, and the bandwidth will restore to a normal level after the temporary reduction. Thus, it cannot fully handle the random variation of the available bandwidth at the wired or wireless bottleneck.

Different from the existing approaches, in this chapter, we focus on the rate adaptation algorithm for streaming SVC video in wireless networks, considering the random and less predictable variation of the available bandwidth. We also consider the more general case where the layered video is encoded in variable bit-rate (VBR).

2.2 Problem Formulation

Considering the limited computation capacity of handheld devices and the high variation of wireless access links, we formulate the optimal video rate adaptation problem as a finite Markov Decision Process, which can deal with the random network condition with a relatively simple approach. For each video segment, the client uses MDP to make a decision on which action to conduct given the current client state. There are four components for MDP, i.e., action, state, transition probability and reward. In the following, we define them one by one. The symbols used in this section are listed in Table 2.1.

As shown in Fig. 2.1, after a segment is obtained completely, the rate adaptation

Table 2.1: Symbol list

Symbol	Description
T_s	constant playback time of a segment
N_s	number of frames per segment
B_T	target buffer size in terms of the number of segments
F	target buffer size in terms of the number of frames
N_T	total number of segments
f	video playback frame rate
a_t	action made at time step t
A_i	request the next segment with i layer higher (for $i \geq 0$) or lower (for $i < 0$) than the current one
A_u	"upgrade" the last received segment to a higher version
A_w	wait for a time duration of T_s
q_t	queue length in terms of the number of buffered frames at time step t
Δq_t	queue length variation after a new segment has been retrieved at time step t
v_t	version index of the last received segment at time step t
Δv_t	difference of video versions requested in consecutive steps.
bw_t	available bandwidth at step t
d_t	number of received segments at time step t
s_t	system state at time step t
m_d^v	size of version v of segment d
$R(s)$	reward function mapping a state to a reward
α	weight parameter which makes a trade-off between the average playback quality and playback smoothness.

algorithm has a chance to decide the video version of the next segment to request and whether the client should be idle for a while to avoid buffer overflow. We define the sequential actions as $\{a_t\}, t = 0, 1, \cdots$. a_t is the decision made at step t, where the step duration equals the time to retrieve one segment. Note that the step duration is not a constant, since the segment download time varies according to the segment size and the available bandwidth. L is the number of versions. The action set for a given state is $\mathcal{A}(s) = \{A_i, A_u, A_w\}$, where A_i $(i = -L+1, \cdots, L-1)$ means to request the next segment with i layer higher (for $i \geq 0$) or lower (for $i < 0$) than the current one, A_u means to "upgrade" the last received segment to a higher version, and A_w means to wait for a time duration of T_s which is the constant playback time of a segment.

We define a state at step t as $s_t = (q_t, \Delta q_t, v_t, \Delta v_t, bw_t, d_t)$. Here, q_t is the queue length in terms of the number of buffered frames. Obviously, q_t is in the range of $(0, F)$, where $F = B_T \times N_s$, B_T is the target buffer size in terms of the number of segments, and N_s is the number of frames per segment. Δq_t is the queue length variation after a new segment has been retrieved, i.e., $\Delta q_t = q_t - q_{t-1}$, which indicates whether the requested video's bit-rate matches the available bandwidth. Δq_t is in the range of $[-F, N_s]$. v_t is the version index of the last received segment. Δv_t indicates the difference of video versions requested in consecutive steps. bw_t is the available bandwidth at step t. d_t is the number of received segments, which is in the range of $[0, N_T]$, where N_T is the total number of segments the client needs to request.

From the definition of the states, we can observe that the Markov property exists, since all of these states depend on their immediately previous state only, i.e.,

$$\Pr\{s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, \cdots, s_0, a_0\} = \Pr\{s_{t+1}|s_t, a_t\}.$$

To obtain the state transition probability, the most challenging issue is to obtain the model for bw_t. For wireless streaming scenarios, the bottleneck is often in the wireless access link, and the finite-state Markov chain has been widely used to model the variation of wireless channels [47, 53]. Thus, we use a discrete-time finite-state Markov model to capture the variation of the bandwidth, and the state transition probabilities can be obtained from the measurement or derived from the wireless channel model [47]. Given the time duration for downloading the current segment, we can estimate the probability distribution of the bandwidth for the next segment using the state transition probability matrix of the Markov model.

For the problem of our interest, we can derive the state transition probability for

the MDP by

$$\mathcal{P}^a_{ss'} = \Pr\{s_{t+1} = s' | s_t = s, a_t = a\}. \tag{2.1}$$

The state at step t is $s = (q, \Delta q, v, \Delta v, bw, d)$. If action $a_t = A_i$ is selected, with probability $\mathcal{P}^a_{ss'} = \Pr\{bw' | bw\}$, the new state will be $s' = (q', \Delta q', v', \Delta v', bw', d')$, i.e.,

$$\begin{aligned} v' &= v + i, \quad \Delta v' = i, \\ q' &= q - \left\lceil (m^{v'}_{d+1} \times f)/bw' \right\rceil + N_s, \\ \Delta q' &= q' - q, \quad d' = d + 1, \end{aligned} \tag{2.2}$$

where $m^{v'}_{d+1}$ is the size of version v' of segment $d+1$ and f is the playback frame rate (since we are dealing with the stored video streaming, the client can have the knowledge of the size of every segment). The second line in (2.2) means that the queue length of the next time slot is equal to the current queue length minus the number of frames consumed when downloading a new segment and plus the number of frames contained in the new segment. If $a_t = A_u$, the new state is

$$\begin{aligned} v' &= v + 1, \quad \Delta v' = \Delta v + 1, \\ q' &= q - \left\lceil [(m^{v'}_{d} - m^{v}_{d}) \times f]/bw' \right\rceil, \\ \Delta q' &= q' - q, \quad d' = d. \end{aligned} \tag{2.3}$$

Similarly, we can derive other state transition probabilities.

The reward in MDP is the payoff obtained when a particular action is taken at a state,

$$r_{t+1} = R(s_t = s), \tag{2.4}$$

where R maps the state to a reward. Table 2.2 lists the rewards defined for different states. $*$ can be any value for the state, F^+ represents that the number of buffered frames is larger than $B_T \times N_s$. The reward of a state can be looked up in the table from the top to the bottom, using the reward of the first entry in the table matching the current state. The values of rewards need to be carefully designed, since it is closely related to the control objective. The stored video has a finite length, and when the state reaches $d = N_T$, i.e., all the segments have been downloaded, the streaming task completes, which is called an episodic task. Therefore, we give state $(*, *, *, *, *, N_T)$ a reward of 0. Besides, any action taken in this state will not change

Table 2.2: Rewards Associated with States

$s_t = s$	$R(s)$
$(*, *, *, *, *, N_T)$	0
$(0, *, *, *, *, *)$	$-F + \Delta q$
$(F^+, *, *, *, *, *)$	$-F - \Delta q$
$(*, \Delta q, *, \Delta v, *, *)$	$\min(-\alpha\lvert\Delta v\rvert, -\lvert\Delta q\rvert)$

the state, i.e., the terminal state will not affect the decision process. By giving the minimum reward when the buffer is empty, we can minimize playback interruption; by giving a negative reward to the state when the number of buffered frames is larger than the desired value, we can avoid buffer overflow. When both Δq and Δv are 0, the maximum reward (0) is given, since in these states, the playback will be smooth and the selected video version matches the available bandwidth well.

In addition, we can associate a weight parameter α with the reward to make a trade-off between the average playback quality and playback smoothness. When α is smaller, the video streaming can be more adaptive to the available bandwidth to achieve a higher average playback quality; when α is larger, a higher priority is given to the playback smoothness. Note that the reward is independent of the bandwidth, since we are unable to control the varying bandwidth.

2.3 Algorithm Design

2.3.1 Optimal Solution

We formulate the rate adaptation problem as an optimization problem. The objective is to find a strategy $\pi(s)$ for the action taken at a state s to maximize the reward received in the long run. Given a deterministic strategy, the state-value function is thus

$$V^{\pi}(s) = \sum_{s'} \mathcal{P}^{a}_{ss'} \left[R(s) + \gamma V^{\pi}(s') \right], \tag{2.5}$$

where γ is the discounting rate $0 \leq \gamma \leq 1$. Note that in our case, we can set γ to 1, since we are dealing with an episodic streaming task. An optimal strategy $\pi^*(s)$ should maximize the state-value function in the long run, i.e.,

$$\pi^*(s) = \arg\max_{\pi} \sum_{s'} \mathcal{P}^{a}_{ss'} \left[R(s) + \gamma V^{*}(s') \right], \tag{2.6}$$

where $V^*(s)$ is the optimal value function. Then, we can obtain the optimal streaming strategy using a value iteration algorithm [44]. The solution is a table that maps each state to an optimal action. Furthermore, to reduce the number of states for MDP and the input size for value iteration, we divide the buffer size (in frames) into a number of bins and index them as q_b starting from 0 to $\lfloor B_T \times N_s/BS \rfloor$, where BS is the number of frames in each bin. Then we use $q_b \times BS$ to represent the number of buffered frames for each bin.

2.3.2 Online Real-Time Algorithm

The optimal streaming algorithm can provide us important insights for dynamic rate adaptation, but it has several limitations which make it less practical. First, the optimal streaming algorithm requires the knowledge of the available bandwidth statistics, i.e., the bandwidth state and transition probability between states. This information is difficult to obtain or estimate accurately beforehand. Using finite state Markov models for dedicated wireless channels is one way to obtain the channel statistics, but the available bandwidth statistics for shared wireless access links or backbone links should depend on not only the physical channel dynamics but also the less-predictable competition from other users. Second, and more importantly, we rely on the value iteration algorithm to solve (2.6), and the complexity of these algorithms are proportional to the number of states. For the optimal video rate adaptation problem, the number of states can be so large that the computational complexity makes it difficult, if not impossible, to be used in real time.

One category of online algorithms, such as Q-Learning and Saras [44], do not require the complete knowledge of the system dynamics but need to repeat the streaming process many times, and take a long time to improve the policy. Thus, they are not practical for our problem as well.

In the following, we propose an online algorithm integrating the learning and planning process, which learns bandwidth statistics and makes decisions in real time and the reasoning of the proposed algorithm still comes from reinforcement learning. The rate adaptation algorithm is decomposed into two modules, bandwidth statistics estimation and real-time search. After one segment is received, the bandwidth statistics estimation module will update the average bandwidth and transition probability, and then the real-time search module will determine the best action based on the bandwidth statistics and the current state. We will describe these two modules in

the following subsections.

2.3.2.1 Bandwidth Statistics Estimation

In order to obtain the bandwidth statistics, we divide the bandwidth into $L+1$ regions (states) based on the average bitrate of video layers. $\bar{r}_i$ is the average bitrate of layer i. The regions are $[0, \bar{r}_1]$, $(\bar{r}_1, \bar{r}_2]$, $(\bar{r}_2, \bar{r}_3]$, $\cdots$, and $(\bar{r}_L, \infty]$. When the d-th segment is received completely, we can calculate the effective throughput l_d for downloading this segment and determine which bandwidth region (state) it falls in. We define a transition count matrix C whose element c_{ij} denotes the number of bandwidth transitions from state i to j. We can calculate the transition probability as

$$P_{ij} = \frac{c_{ij} + k}{\sum_{k=1}^{L+1} c_{ik} + k(L+1)}, \tag{2.7}$$

where k is the Laplacian smoothing parameter. Laplacian smoothing can avoid overfitting and naturally initialize the transition probability as equal. The average bandwidth $\bar{b}_i$ of state i is

$$\bar{b}_i = \frac{\sum_d I(l_d, i)}{\sum_{k=1}^{L+1} c_{ki}}, \tag{2.8}$$

where $I(l_d, i)$ is defined as

$$I(l_d, i) = \begin{cases} l_d, & \text{if } l_d \in (\bar{r}_{i-1}, \bar{r}_i], \\ 0, & \text{otherwise.} \end{cases} \tag{2.9}$$

In this way, as the number of received segments increases, the bandwidth statistics will be more accurate.

2.3.2.2 Real-Time Search

The proposed real-time search algorithm is listed in Algorithm 1, where D is the search depth threshold to define how many steps we look forward into the future.

$$\pi^*(s) = \arg\max_a Q^*(s, a), \tag{2.10}$$

$$V^*(s) = \max_a Q^*(s, a), \tag{2.11}$$

$$Q^*(s,a) = \sum_{s'} \mathcal{P}^a_{ss'} \left[R(s) + \gamma V^*(s')\right]$$
$$= R(s) + \gamma \sum_{s'} \mathcal{P}^a_{ss'} V^*(s'). \tag{2.12}$$

The reasoning behind the real-time search algorithm is from (2.10)–(2.12). We can

Algorithm 1 Real-Time Search Algorithm

```
1: procedure OptStateValue(s, k)
2:     if k ≥ D then
3:         return R(s)
4:     end if
5:     max ← −∞
6:     for all a ∈ A(s) do
7:         q ← OptActionValue(s, a, k)
8:         if q > max then
9:             best ← a
10:            max ← q
11:        end if
12:    end for
13:    return (best, max)
14: end procedure

15: procedure OptActionValue(s, a, k)
16:    q ← R(s)
17:    for all s' from s, a do
18:        (best, v) ← OptStateValue(s', k + 1)
19:        q ← q + γ P_ss' v
20:    end for
21:    return q
22: end procedure
```

see the recursive relationship between the optimal state-value function $V^*(s)$ and the optimal action-value function $Q^*(s,a)$, where $Q^*(s,a)$ is the long-term return when the state is s and action a is taken. (2.10) and (2.11) are equivalent since they both indicate that action a which maxmizes $Q^*(s,a)$ is the optimal action for state s. While in (2.12), $Q^*(s,a)$ is dependent on the expected optimal value function of the next state s'. In the online algorithm, procedure OptStateValue corresponds to (2.11) and procedure OptActionValue corresponds to (2.12). In line 13, the procedure OptStateValue returns the best action for state s and the long-term return corresponding to the search depth D. In line 17, the state s' can be obtained from s and a based on the transition models defined in (2.2) and (2.3).

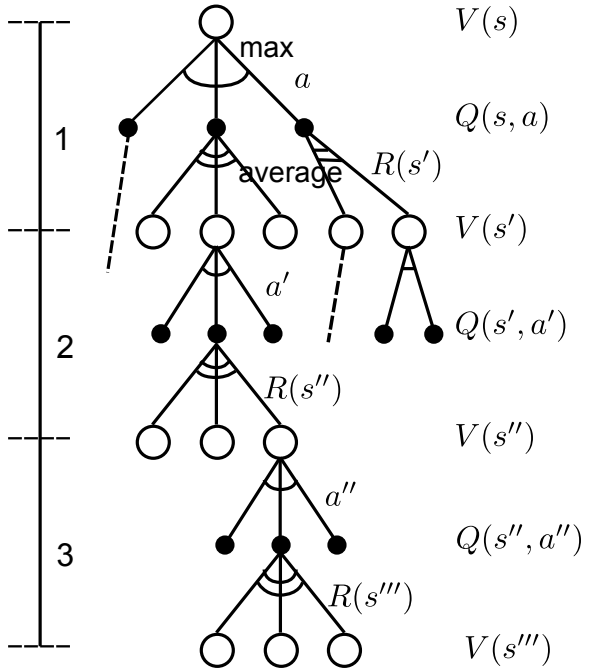


Figure 2.2: Search Tree

Essentially, the states can be viewed as the nodes of a tree and the possible combinations of actions and bandwidth transitions are the edges of the tree as depicted in Fig. 2.2. Each circle denotes a value state and each dot denotes an action-value state. D is set to 3. The online algorithm is traversing the tree, and the time complexity of the online algorithm is $O(b^D)$, where b is the number of branches of the tree (the product of the number of actions and possible bandwidth transitions). If the search depth is equal to the length of a video, then we can obtain an optimal solution. But the computation is too high to obtain the result in real-time, so the search depth is set to a small value in the proposed real-time search algorithm. Thus, the algorithm gives a suboptimal solution. We use D to make a trade-off between the optimality and computational complexity. In Section 2.4, we demonstrate that the performance of the proposed RT algorithm is close to the optimal when D is 3, and the computation cost is low enough for real-time decision making.

2.4 Performance Evaluation

In this section, we first define the objective QoE metrics in terms of playback interruption, average playback quality and playback smoothness. Then we describe the

evaluation framework including the layered video storage structure, SVC video player implementation details and the experiment settings. We evaluate the proposed online algorithm and compare it with the optimal solution and the existing state-of-the-art rate adaptation algorithm [24] by experiments and simulations.

2.4.1 QoE Metrics

1) *Interruption Ratio*: Every $1/f$ second (f is the video frame rate), the video player displays one frame, which is defined as one display event. If there is no decoded frame available to display, a playback interruption occurs. Let n_0 be the number of occurrences that a frame to be displayed is not available. Denote by n_t the total number of display events, the interruption ratio (IR) is defined as $\text{IR} = n_0/n_t$. Among the performance metrics, we give the IR the highest priority because interruptions during playback are most unpleasant for users. It means that if the IR of an algorithm is higher than the other, then this algorithm is inferior no matter how good the other performance metrics are.

2) *Average Playback Quality*: We define a continuous playback of layer i video as one run and its length in terms of the number of display events as n_r for the r-th run. There are totally N runs. The layer index 0 denotes that a playback interruption occurs. The weighted sum of the layer index is used to measure the average playback quality (APQ), which is defined as $\text{APQ} = \sum_{r=1}^{N}(n_r \times i)/\sum_{r=1}^{N}(n_r)$. We also use the PSNR metric to measure the average playback quality. The experimental results show that the APQ has a positive correlation with PSNR.

3) *Playback Smoothness* [30]: Intuitively, a longer run length leads to a smoother watching experience. The mean square root of run length is used to measure the playback smoothness (PS), and we have $\text{PS} = \sqrt{\sum_{r=1}^{N}(n_r)^2/N}$. It also gives a more fair evaluation when the length of one run is much larger than the others, compared with the arithmetic average.

2.4.2 Evaluation Framework and Testbed Settings

In this section, we describe the layered video storage structure at the server side, video player implementation details, experiment and testbed settings.

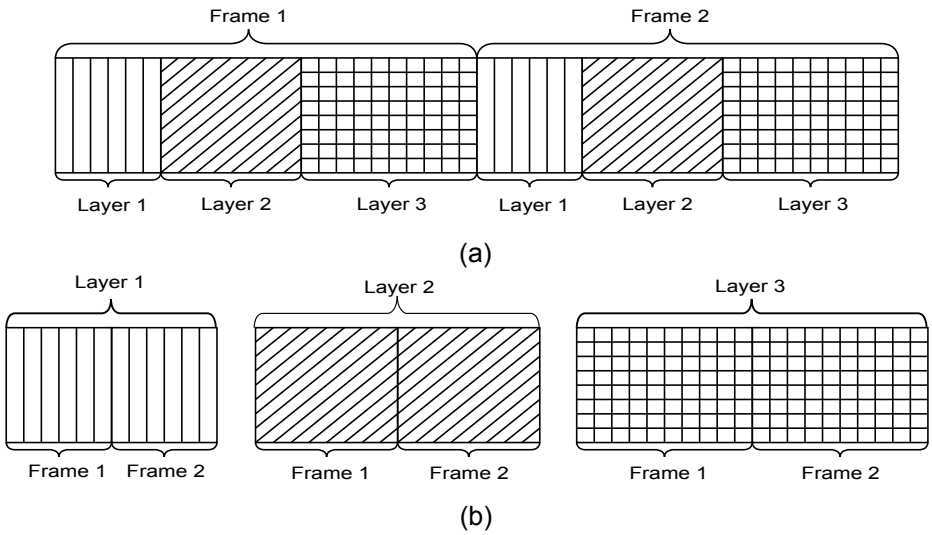


Figure 2.3: Layered Video Storage Structure.

2.4.2.1 Layered Video Storage Structure

At the server side, the video is segmented and encoded into multiple layers. To minimize the server load, the server stores the SVC video in a layered segment structure. Fig. 2.3(a) shows the structure of the original bitstream generated by an SVC encoder (supposing that each segment has two frames and there are three layers). In the server, the Network Abstraction Layer (NAL) units of all the frames with the same layer index are stored together as shown in Fig. 2.3(b). When the client receives the layer segments, it can reorganize them into the original bitstreams for decoding and playback.

This layered segment storage structure can better utilize the web caching infrastructure and the client can request any layer segment flexibly. Since the frames of different layers are separately stored, the client uses HTTP pipelining to request several layer segments and construct the video. Other solutions including partial HTTP requests or letting the web server extract the requested layer segments on-the-fly not only add the complexity of the server, but also slow down the response time to the client requests.

One concern of the proposed storage structure is that the client needs to wait until all the layers in a frame are downloaded before playback. A solution to minimize the latency is that the client can establish parallel TCP connections to request the different layers simultaneously, which is left for future investigation.

2.4.2.2 Video Player Implementation

The client-side video player is implemented using an open-source SVC decoder [6]. As depicted in Fig. 2.4, the video player consists of three modules, rate adaptation, decoder and display. The rate adaptation module determines the version of the next video segment to request. The decoder fetches a segment from the segment buffer and decodes the segment as fast as possible and stores the decoded picture in the picture buffer. The decoder will not fetch a new segment until the number of pictures left to be displayed is smaller than a threshold. In this way, the buffered segments have a chance to be enhanced with higher-layer segments that may arrive later. The display module simply fetches a picture from the buffer and displays it on the screen every $1/f$ seconds. Meanwhile, the video player also collects the system state information including the buffer state and the playback index.

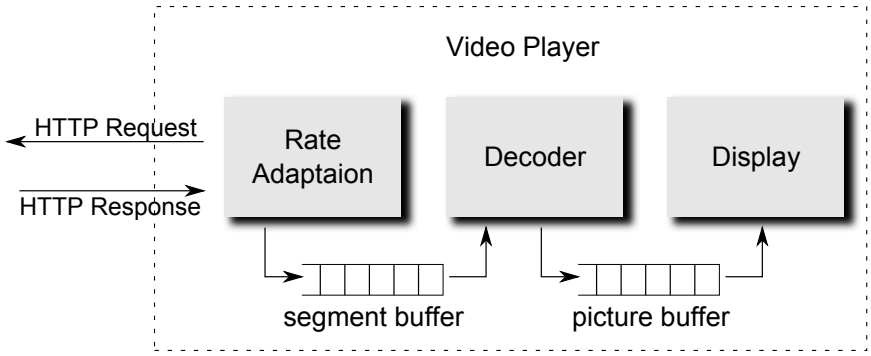


Figure 2.4: Video Player Structure

Table 2.3: Layer Configuration

Resolution	Avg. bit-rate (Kbps)	std bit-rate deviation	Y-PSNR	Layer index
320x180	112.84	39.01	35.47	1
320x180	238.94	88.84	39.44	2
640x360	363.82	140.33	35.90	3

2.4.2.3 Experiment and Testbed Settings

We use the open-source SVC codec JSVM [35] to encode the sample video ("Big Buck Bunny" [1]) into three layers, and their configurations are listed in Table 2.3. The encoding rate of layer n is the cumulative rate of all the layers up to n. Note that the Y-PSNR of Layer 3 is lower than that of Layer 2, but we still prefer Layer 3 video which has a higher resolution, as it leads to a better watching experience when displayed on a larger screen due to a higher dots per inch (DPI). Typically, PSNR reflects the mean square errors between the original video signal and the received signal; if we use the original videos with different layers, the layer-wise PSNR comparison becomes unfair as a received lower-layer video with a higher PSNR may have a worse visual quality. To make the PSNR comparison meaningful with layered video, we upscale the lower resolution 320x180 video to 640x360 video using the "bicubic" interpolation method and then calculated the PSNR accordingly. The PSNRs for the 3 layers from low to high quality are 30.99, 32.62 and 35.90 dB, respectively. In this way, the PSNR can reflect the visual quality of layered videos.

Each layer is chopped into small segments of 17 frames. The total number of segments, N_T, is 200, and the frame rate is 24 frames per second. From the experiments, we find that the segment size of 17 frames is small enough to react to the varying bandwidth, and large enough to keep the HTTP overhead low. The target buffer size is $B_T = 20$ segments. The playback starts when 4 segments are received.

Fig. 2.5 shows the testbed configuration. The testbed used `Lighttpd` as the streaming web server. The server and the wireless router are connected via wired links, and two laptops access the web server through the wireless router. The laptop C1's CPU is dual-core at 2.53 GHz with 2 GB memory, and for another laptop C2, the CPU is dual-core at 2.26 GHz with 4 GB memory. OpenWRT is installed on the wireless router, configured in the IEEE 802.11b mode and the downlink transmission rate is set to 1 or 2 Mbps to emulate a dynamic wireless environment with different

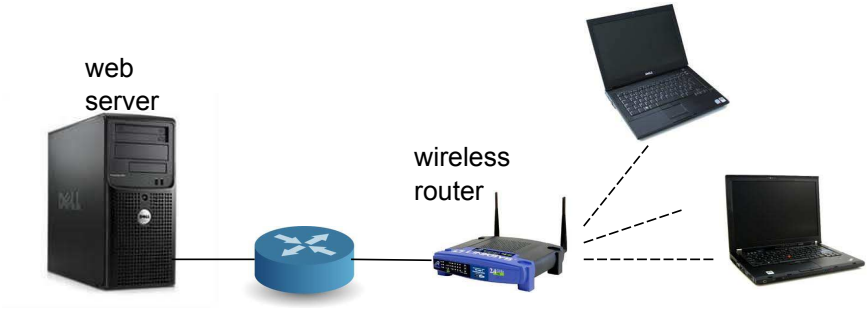


Figure 2.5: Network Topology for the Experiments

Table 2.4: Experiment 1 results

Algorithm	IR	APQ (PSNR)	PS	Max queue
RA	0	2.57 (34.73)	260.80	21.0
RT $\alpha = 10$ $D = 1$	0	2.24 (33.90)	700.96	19.0
RT $\alpha = 10$ $D = 2$	0	2.63 (34.88)	258.996	19.0
RT $\alpha = 10$ $D = 3$	0	2.72 (35.07)	643.60	19.0
RT $\alpha = 10$ $D = 4$	0	2.45 (34.36)	905.80	19.0
RT $\alpha = 12$ $D = 3$	0	2.63 (34.85)	1132.63	19.0
FL(3)	0	3 (35.90)	3400	21.0

congestion levels. In our experiment, we assume that the download time is much larger than the round-trip latency, so the downloading time is proportional to the segment size. Otherwise, we need to estimate the round-trip latency and the transmission delay separately, which is left for future research.

2.4.3 Experiments and Simulations

2.4.3.1 Experiment 1 (no background traffic)

The transmission rate of the wireless router is set to 1 Mbps, and the effective goodput is about 717.6 Kbps (measured by downloading a large file through HTTP). We conducted the experiment for 10 runs and the presented performance results are the average values.

In Table 2.4, we compare the proposed online realtime algorithm, RT, with the rate adaptation algorithm, RA, proposed in [24] and the fixed layer algorithm, FL(3), which always requests all three layers. Generally, when the search depth D is larger, the performance of RT is better. But we cannot increase it too much. For C1, when $D = 1$ or 2, it takes less than 1 ms to make the decision for a new segment. When $D = 3$, it takes about 10 ms, still much less than the duration of one frame. When $D = 4$, it takes 240 ms to make one decision, which is approximately the playback time of six frames. We believe that the decision time less than the playback time of one frame is acceptable. Therefore, in the following experiments, D is set to 3 by default. (For C2, we also set D to be 3, although its CPU cycle is slightly lower than

Table 2.5: Experiment 2 results

Algorithm	IR	APQ (PSNR)	PS	Max queue
RA	0	1.19 (31.33)	179.98	20.7
RT $\alpha = 10$	0	1.29 (31.50)	529.88	18.6
FL(1)	0	1 (30.99)	3400	20.9
FL(2)	0.27	1.46 (32.62)	34.11	5.2

C1).

For the proposed RT algorithm with $D = 3$, as we increase α to 12, when compared with $\alpha = 10$, we can see that APQ is reduced but PS is increased, which shows that α can make a trade-off between APQ and PS.

Comparing RT (D=3) and RA, we note that RT can achieve both a higher APQ and PS, while maintaining a lower queue length. Since the available bandwidth is sufficient to support Layer 3 video, FL(3) outperforms the RT and RA algorithm. However, the FL algorithm performance will degrade dramatically when there is any competing traffic and the available bandwidth is more dynamic. We can see this in the following experiments.

2.4.3.2 Experiment 2 (with background traffic)

In this experiment, the transmission rate of the wireless router is also set to 1 Mbps. Laptop C1 runs the SVC Player, and another laptop C2 downloads a large file from the web server. From Table 2.5, we can see that the available bandwidth is sufficient to support Layer 1 video but cannot support Layer 2 video, since for FL(2) algorithm, the IR is as high as 0.27. Because we give the IR the highest priority when comparing rate adaptation algorithms, FL(2) is inferior compared to RA and RT, although its APQ value is better than the others. Both RA and RT algorithm can achieve APQ between 1 and 2, and RT is better than RA in terms of both APQ and PS.

2.4.3.3 Experiment 3 (competing video flows with on-off background traffic)

In this experiment, the transmission rate of the wireless router is set to 2 Mbps. Each laptop runs the SVC player and an on-off background traffic flow. The on-off traffic flow downloads a file (1.3MB) from the server, and then sleeps for 10 seconds, and this process repeats until the end of the experiment. Since the time needed to

Table 2.6: Experiment 3 results

C	Algorithm	IR	APQ (PSNR)	PS	Max queue
C1	RA	0	1.46 (31.93)	146.33	21
	RT $\alpha = 10$	0	1.56 (32.04)	269.08	19
	FL(1)	0	1 (30.99)	3400	21.0
	FL(2)	0.05	1.89 (32.62)	545.55	19.1
C2	RA	0	1.56 (32.07)	141.29	21
	RT $\alpha = 10$	0	1.59 (32.11)	321.89	19
	FL(1)	0	1 (30.99)	3400	20.8
	FL(2)	0.02	1.95 (32.62)	1040.64	19.9

download the fixed-size file varies due to contention, there can be 0 to 2 background flows during the experiment, which makes the available bandwidth more dynamic. In order to evaluate the fairness of the rate adaptation algorithms, both laptops run the same rate adaptation algorithm. In Table 2.6, we can see from the FL algorithm that the video version can be supported is between Layer 1 and Layer 2. RT is better than RA in terms of both APQ and PS, and the two videos on different laptops have roughly the same QoE performance, which demonstrates that the proposed RT algorithm together with TCP congestion control can allow the competing videos to obtain a fair share of the bandwidth.

2.4.3.4 Simulation Results

We also evaluate the gap between the proposed online RT algorithm with the optimal streaming policy, OS, obtained by dynamic programming. Since the optimal streaming policy cannot be generated in real time, we use the trace-driven simulation to compare these algorithms.

The traces share the same bandwidth statistics with Experiment 1 and 2, denoted as S1 and S2, respectively. The statistics of the average bandwidth for different states and the transition probability are obtained off-line as the input of the dynamic programming. Similar to the experiment, the results are the average over 10 runs. Optimal solution's queue length is in the unit of segments, while that of online algorithm is the number of frames. Since the queue variation unit of OS and RA is in different unit, the α of these two algorithms are set differently. From Table 2.7, both RT and OS are better than RA, and there exists a small performance gap between RT and OS, which indicates the trade-off between the performance and computational complexity.

Table 2.7: Simulation Results

Simu	Algorithm	IR	APQ (PSNR)	PS	Max queue
S1	RA	0	2.35 (34.17)	153.48	21
	RT $\alpha = 10$	0	2.57 (34.66)	517.98	19
	OS $\alpha = 2$	0	2.82 (35.24)	1434.99	19.4
	FL(3)	0	3 (35.90)	3400	21.0
S2	RA	0	1.34 (31.56)	84.59	21
	RT $\alpha = 10$	0	1.78 (32.32)	300.49	19
	OS $\alpha = 2$	0	1.80 (32.33)	498.5	17.5
	FL(1)	0	1 (30.99)	3400	20.8
	FL(2)	0.09	1.82 (32.62)	193.13	9.1

Figure 2.6: RA

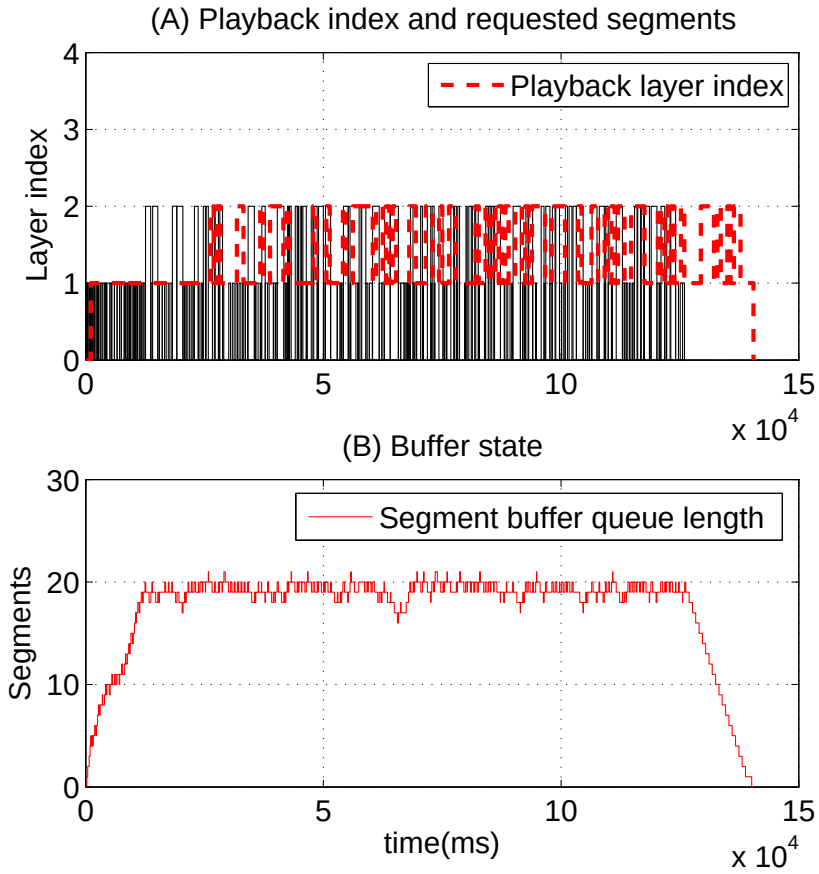


Fig. 2.6-2.8 show the playback trace of one run in simulation S2. We can see from the figure that RA suffers from frequent layer switching. The proposed online algorithm, RT, is more aggressive than the optimal OS algorithm. Although the APQ of RT and OS in this run are both around 1.78, OS achieves a higher PS by never requesting Layer 3 video, and the average queue length of OS is smaller.

Another issue is that the Markov model used for the varying bandwidth may not be accurate. It is important to test how sensitive the performance of OS is to the model accuracy. In the test, we used P_2 as the Markov model to drive the channel emulator in the experiment, and used both P_1 and P_2 in the dynamic programming to obtain the action decisions. The average bandwidth and steady state probabilities of the wireless link under different profiles are listed in Table 2.8. The two matrices,

Figure 2.7: RT

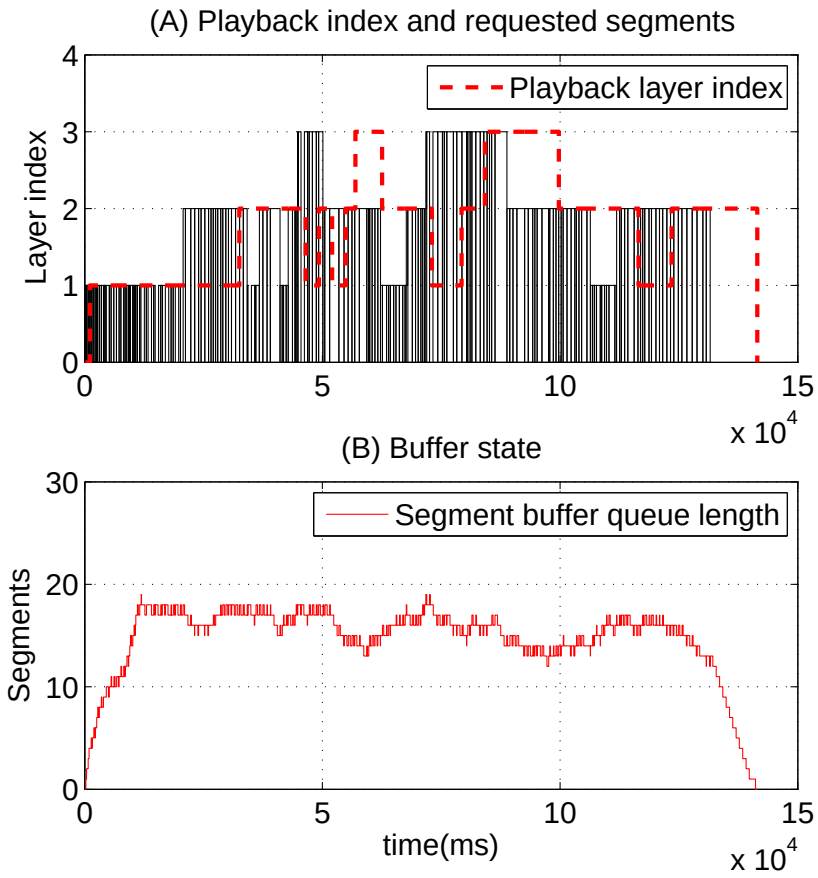
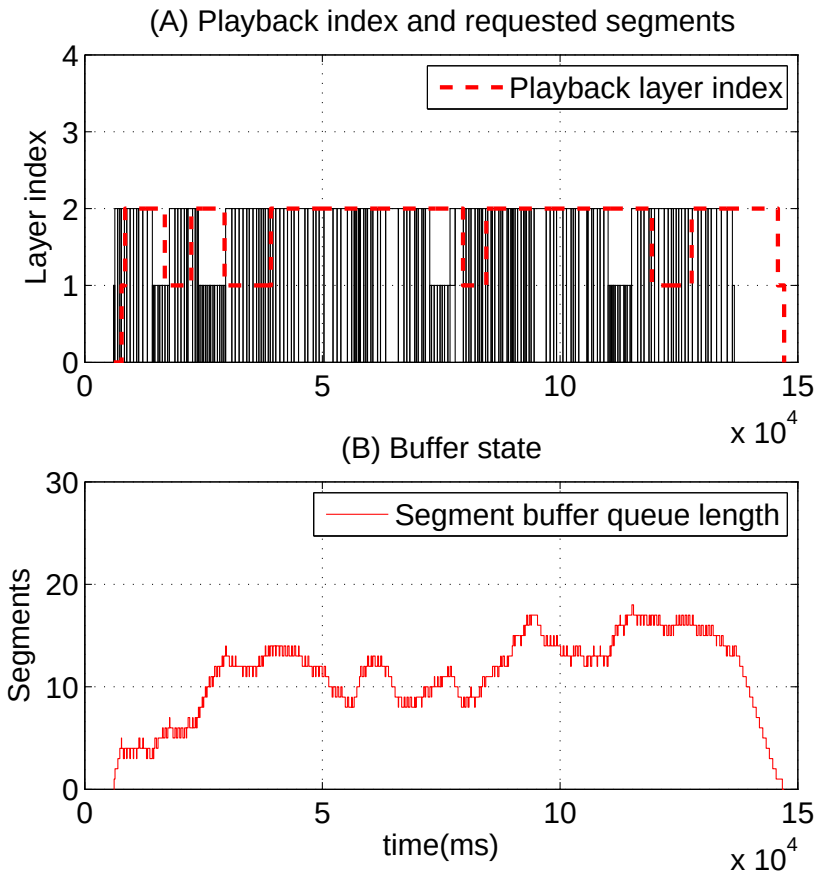

(A) Playback index and requested segments
Playback layer index
Layer index
0
1
2
3
4
0
5
10
15
x 10^4
(B) Buffer state
Segment buffer queue length
Segments
0
10
20
30
0
5
10
15
time(ms)
x 10^4

Figure 2.8: OS


(A) Playback index and requested segments
Playback layer index
Layer index
0
1
2
3
4
0
5
10
15
x 10^4
(B) Buffer state
Segment buffer queue length
Segments
0
10
20
30
0
5
10
15
time(ms)
x 10^4

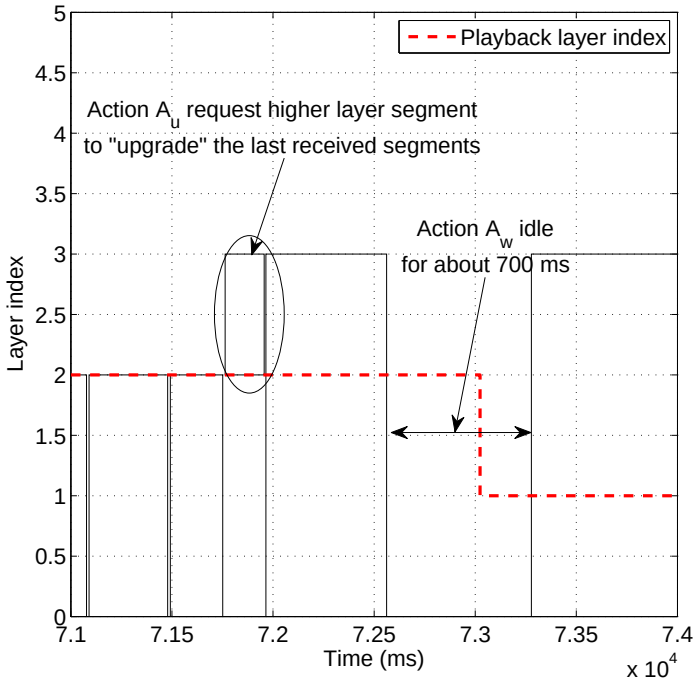


Figure 2.9: A Zoom-in of Playback Trace of RT

Table 2.8: State Prob. and Available Bandwidth

State	1	2	3	4
Bandwidth (Kbps)	50.32	180.63	260.38	550.75
Steady state prob (P_1)	0.026	0.102	0.407	0.465
Steady state prob (P_2)	0.103	0.256	0.385	0.256

P_1 and P_2 are, respectively,

$$\begin{bmatrix} 0.5 & 0.05 & 0.05 & 0.4 \\ 0.2 & 0.25 & 0.2 & 0.35 \\ 0.2 & 0.1 & 0.2 & 0.5 \\ 0.1 & 0.1 & 0.1 & 0.7 \end{bmatrix}, \begin{bmatrix} 0.25 & 0.75 & 0 & 0 \\ 0.3 & 0.4 & 0.3 & 0 \\ 0 & 0.2 & 0.6 & 0.2 \\ 0 & 0 & 0.375 & 0.625 \end{bmatrix}.$$

The results are shown in Table 2.9. OS(P_i) means that the streaming strategy is obtained using transition matrix P_i for $i = 1, 2$. From the table, when the matrix in the decision process does not match the real situation, the performance degrades slightly, but still in a tolerable range. Also, comparing the results in Tables 2.9, even with a mismatched model for the available bandwidth, the proposed OS still substantially outperforms RA in terms of both APQ and PS.

Table 2.9: Model Sensitivity Test

Env	B_T	ALG	IR	APQ	PS	Max queue
P_2	20	OS(P_1)	0	1.92	183.79	20
		OS(P_2)	0	1.88	246.54	20.4
	30	OS(P_1)	0	1.88	229.58	29.9
		OS(P_2)	0	1.87	268.32	30.1

We further zoom in the playback trace for RT. In Fig. 2.9, the rectangle represents a segment and the width of it denotes the download time duration (from the time instant of sending out the HTTP request to that of receiving the segment completely). The horizontal gap between the edges of the rectangles is due to the waiting action to avoid buffer overflow. Fig. 2.9 shows the advantage of the proposed video streaming framework using SVC: the rectangles rise from a non-zero layer index (circled and annotated by arrows) are the layer segments to "upgrade" the already buffered segments to improve both APQ and PS, which is not possible when using the traditional AVC streaming techniques.

2.5 Summary

In this chapter, for DASH-based adaptive video streaming in wireless networks, we have formulated the rate adaptation problem as an MDP and used dynamic programming to obtain the optimal streaming policy. To reduce the complexity of the optimal streaming policy, we have proposed an online algorithm which learns the bandwidth statistics and determines the request decisions for the future. The trade-off between the average video quality and playback smoothness can be made by adjusting the parameter in the reward function. Experimental results have shown that the proposed solution is feasible and can substantially outperform the existing one.

Chapter 3

Transmission Control for Compressive Sensing Video over Wireless Channel

In this chapter, we consider a wireless sensor node monitoring the environment and it is equipped with a compressive-sensing based, single-pixel image camera and other sensors such as temperature and humidity sensors. The wireless node needs to send the data out in a timely and energy efficient way. This transmission control problem is challenging in that we need to jointly consider perceived video quality, quality variation, power consumption and transmission delay requirements, and the wireless channel uncertainty.

Our main contributions in this chapter are three-fold. First, to perform rate control, we propose a rate-distortion model to capture the relationship between the number of measurements and the recovered signal distortion based on the CS theory. We have shown the accuracy and effectiveness of the proposed model. Second, we formulate a deterministic optimization algorithm as a benchmark, and a more practical stochastic optimization problem for optimizing the rate control and power control. In addition, we propose a supplementary stochastic optimization algorithm which put different priority in conserving power and video quality requirements. For the deterministic optimization problem, Lagrangian optimization and dynamic programming are used to obtain the optimal solution. For the stochastic optimization problem, the Lyapunov optimization technique is used to design a joint rate control, power control and scheduling algorithm. Third, we conduct extensive simulation to

validate the proposed rate-distortion model and the control algorithms and show their effectiveness.

The rest of the chapter is organized as follows. Section 3.1 gives a brief introduction of compressive sensing and the related work. Section 3.2 describes the system model and the formulation of the optimization problems. Section 3.3 details the derivation of the joint rate control, power control and scheduling algorithm. The performance of the proposed rate-distortion model and control algorithm is studied in section 3.4, followed by concluding remarks in section 3.5.

3.1 Background and Related Work

3.1.1 Background of Compressive Sensing

Compressive sensing or sampling[16][8] was proposed as a new acquisition framework which can sample and compress sparse or compressible signals in a single operation.

Suppose that a signal $x \in \mathbb{R}^n$ can be transformed to a coefficient vector θ with some basis Ψ, i.e., $x = \Psi\theta$. Ψ can be any representing basis such as DCT or wavelet. If S elements of vector θ is nonzero, then x is called S-sparse signal. If we sort the magnitude of θ in a descending order and it follows the following power-law

$$|\theta|_n \leq Rn^{-1/p}, \tag{3.1}$$

where R and p are positive constants, the signal x is compressible. p determines how fast the amplitude of the coefficients decays. Image signals are usually compressible. When we use the largest S elements of θ, denoted as θ_S, with all other elements set to zero to approximate θ, the approximation error follows

$$\|\theta - \theta_S\|_{\ell_2} \leq CRS^{-r}, \tag{3.2}$$

where C is a constant and $r = 1/p - 1/2$. If we increase S, according to (3.2), the approximation error decays quickly.

The m measurements of compressive sensing, y, are obtained by multiplying signal x with a measurement matrix $\Phi \in \mathbb{R}^{m \times n}$, i.e., $y = \Phi x$. Using the reverse operation to recover x is infeasible, since, given $m < n$, $y = \Phi\tilde{x}$ is an underdetermined system with infinite number of solutions. However, [16] and [8] have shown that ℓ_1 minimization

can recover the original signal with high probability, which can be formulated as

$$\begin{aligned} \min \quad & \|\tilde{\theta}\|_{\ell_1} \\ \text{subject to} \quad & \|y - A\tilde{\theta}\|_{\ell_2} \leq \epsilon, \end{aligned} \tag{3.3}$$

where $A = \Phi\Psi$ and ϵ is the noise energy in the measurements. The recovered signal $x = \Psi^*\hat{\theta}$, where Ψ^* is the adjoint of Ψ, and $\hat{\theta}$ is the solution to (3.3). For compressible signal, [8] provided the following theorem to bound the recovery performance.

Theorem 1. *If x obeys (3.1), with high probability*

$$\|x - \hat{x}\|_{\ell_2} \leq CR(m/\log n)^{-r}, \tag{3.4}$$

where m is the number of measurements and $r = 1/p - 1/2$.

We can see the relationship between (3.4) and (3.2). When $S = O(m/\log n)$, the right hand side of both equations are equivalent. It means $\|x - \hat{x}\|_{\ell_2} = \|\hat{\theta} - \theta\|_{\ell_2}$. Therefore, this theorem states that if there are $O(S \log n)$ measurements, the recovery is as good as knowing x and selecting the largest S coefficients from θ.

3.1.2 Related Work

How to deliver traditional video over wireless networks have been heavily investigated [52]. There are a few work investigating transmitting compressive sensing measurements over wireless channels. [11] used compressive sensing as a joint source and channel coding to protect sparse signals over erasure channels. The proposed method compensates the lost measurements during transmission by sending more measurements. This property is also referred as democracy of compressive sensed measurements [22].

For compressive sensing image or video rate control algorithm, the channel error/erasure rate is not the only factor to consider. [34] proposed a system for joint video compression, rate control and error correction over wireless sensor network using compressive sensing. The proposed rate control algorithm considers network congestion and received video quality. The rate control algorithm uses a rate-distortion model from a conventional video compression method [42], which may not fully capture the rate-distortion characteristic of compressive sensing measurements. In addition, it did not consider transmission power control to maximize the lifetime of wireless sensor nodes.

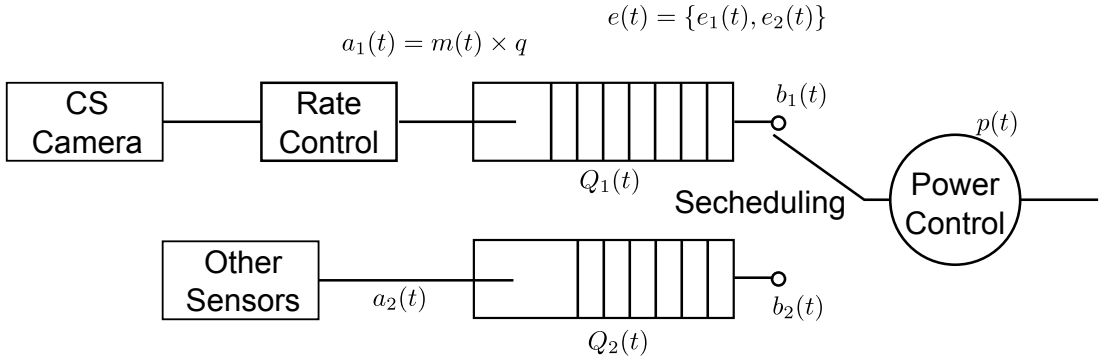


Figure 3.1: System Architecture

[29] proposed a joint compression and transmission power control algorithm for packets processing in wireless nodes. Stochastic network optimization is used to obtain the control decisions to minimize the power consumption while making the transmission queues stable. However, how to consider rate control and the perceived quality for compressive sensing video transmission is still an open issue, which motivates this work.

3.2 System Model and Problem Formulation

3.2.1 System Model

Fig. 3.1 shows the internal structure of the sensor node. The symbols of the system model are listed in Table 3.1. The single-pixel image camera acquires F images per second and generate $A_{\max}$ compressive sensing measurements for each image. Naturally, we divide the time into slots with a duration of $1/F$ seconds, so the camera generates $A_{\max}$ measurements per time slot. The rate control module decides how many measurements $m(t)$ should be put into Q_1 to transmit. Each measurement is represented by q bits. Therefore, the arrival rate of Q_1 is $a_1(t) = m(t) \times q$ and $a_1(t)$ is in the range of $(0, A_{\max} \times q)$ at time slot t. The queue backlog of Q_1 at time slot t is $Q_1(t)$. Denote $d(t)$ as the distortion incurred when admitting $m(t)$ measurements into the queue for each image. We assume that the number of bits per measurement is fixed as q, so the number of measurements is the only way to adjust the video rate. Other sensed (non-video) data will be put into another queue Q_2. The queue backlog at time slot t is $Q_2(t)$ and its arrival rate is $a_2(t)$ bits/slot.

We have two queues and need to select at most one queue per time slot for trans-

Table 3.1: Symbol list

Symbol	Description
f	frame rate per second
A_{max}	number of Measurements generated by image camera per image
Q_1	The Queue storing CS measurements
Q_2	The Queue storing other sensor data
$Q_1(t)$	backlog of Q_1 at time slot t
$Q_2(t)$	backlog of Q_2 at time slot t
$m(t)$	number of measurements admitted to Q_1 at time slot t
q	number of bits for each CS measurement
$a_1(t)$	arrival rate of Q_1 at time step t
$d(t)$	distortion incurred when admitting $m(t)$ measurements into the queue for each image
$a_2(t)$	arrival rate of Q_2 at time step t
$e_k(t)$	indicating if Q_k is selected for transmission
$b_k(t)$	service rate for queue k
$p(t)$	power level for wireless transmission for time slot t
p_{max}	maximum power level for each time slot
B	wireless channel bandwidth
η	transceiver efficiency
d	distance between the transmitter and the receiver
α_l	path-loss exponent
$h(t)$	fading coefficient at time slot t
α, β, γ	rate distortion model parameters

mission. We denote the decision as $e(t) = \{e_1(t), e_2(t)\}$, where $e_k(t) = 1$ means selecting queue k to transmit and otherwise $e_k(t) = 0$. When both $e_1(t)$ and $e_2(t)$ are 0, nothing is transmitted, and we have $e_1(t) + e_2(t) \leq 1$. Define $b_k(t)$ as the service rate for queue k. It is a function of $e(t)$ and the power level $p(t)$, where $0 \leq p(t) \leq p_{\max}$ and $p_{\max}$ is the maximum transmission power. $p(t) = 0$ indicates that $e_1(t) + e_2(t) = 0$ and vice versa. The service rate according to channel capacity is approximately

$$b_k(t) = \frac{\eta B}{f} e_k(t) \log\left(1 + p(t)\frac{d^{-\alpha}G}{N_0 B}|h(t)|^2\right) \text{(bits/slot)}, \tag{3.5}$$

where B is the channel bandwidth (Hz), η is a communication system coefficient representing the transceiver efficiency, d is the distance between the transmitter and receiver, α is the path-loss exponent and $h(t)$ is the fading coefficient at time slot t. We assume that $h(t)$ is known at the beginning of each slot. We need to determine the power level $p(t)$ for each time slot. The queue dynamics is

$$Q_k(t+1) = \max\left[Q_k(t) - b_k(t), 0\right] + a_k(t). \tag{3.6}$$

3.2.2 Rate-Distortion Model

Before we formulate the optimization problem, we need to build a rate-distortion model for compressive sensed video. According to (3.4), ℓ_1-minimization is as good as recovering the largest S non-zero coefficients. We define the rate distortion model as follows

$$D(m) = \alpha m^{\beta}, \tag{3.7}$$

where D(m) denotes the mean square distortion function, m is the number of measurements, α and β are the parameters of this model. Note that for notation simplicity, $m = a(t)$ and $D(m) = d(t)$. To better approximate the rate-distortion relation, we use following model with one more parameter,

$$D(m) = \alpha m^{\beta} + \gamma. \tag{3.8}$$

In practical, we can determine the parameters of the model online by decoding a few sets of measurements and calculating the distortion, and then use these measurement-

distortion data to fit the model and obtain the parameters that can be used for a number of consecutive frames.

3.2.3 Deterministic Optimization Problem

We first formulate the optimal scheduling, power and rate control as a deterministic optimization problem assuming the channel condition $h(t)$ and arrival rate $a_2(t)$ are known. The knowledge about the channel and arrival rate are difficult to obtain in reality. But it can serve as a benchmark and its performance can be used as an upper bound for comparison purpose. The deterministic optimization problem can be decomposed to a power allocation problem (P1) and a rate allocation problem (P2). If we can transmit the maximum amount of data within time period T, then we have a chance to maximize the perceived video quality, since the distortion decreases as the number of transmitted measurements increases. Therefore, we first formulate and solve problem (P1), which finds an optimal power allocation scheme to maximize the amount of data can be transmitted during T. Then, we formulate and solve rate allocation problem (P2) to minimize the weighted sum of distortion and distortion variation subject to the maximum amount of data constraint. The details of these two deterministic optimization problems are as follows.

The objective of the power allocation problem is to allocate total power budget $\bar{p} \times T$ among slots 0 to $T-1$ to maximize the total transmission rate, where $\bar{p}$ is the maximum average power consumption and T is the number of total transmission slots. The power allocation problem is formally defined as follows.

$$\text{(P1)} \quad \text{maximize} \sum_{t=0}^{T-1} b(t), \tag{3.9}$$

$$\text{subject to} \sum_{t=0}^{T-1} p(t) \leq T \times \bar{p}, \tag{3.10}$$

$$0 \leq p(t) \leq p_{\max}, \tag{3.11}$$

where $b(t)$ is the aggregated transmission rate of the two queues. By solving the above optimization problem, we can obtain the power allocation $p(t)$ for every slot and the maximum amount of data for T slots $b^* = \sum_{t=0}^{T-1} b^*(t)$.

The objective of the rate allocation problem is to decide the number of measurements per slot, $m(t)$, to minimize the weighted sum of distortion and distortion

variation. The problem is formulated as follows.

$$\text{(P2)} \quad \text{minimize} \sum_{t=0}^{T-1} d(t) + \mu(d(t) - d(t-1))^2, \tag{3.12}$$

$$\text{subject to} \sum_{t=0}^{T-1} (a_2(t) + m(t) \times q) \leq b^*, \tag{3.13}$$

$$0 \leq m(t) \leq A_{\max}, \tag{3.14}$$

where μ is the weighting parameter of distortion and distortion variation. The inequality (3.13) assures that all data arrived during T time slots can be transmitted. Thus the queues are stable.

3.2.4 Stochastic Optimization Problem

We next formulate two more practical stochastic optimization problems. First, we formulate the stochastic problem $(P3)$ (power constraint) corresponding to the above deterministic problem. We aim to minimize the time average expected weighted sum of distortion and distortion variation, while keeping all the queues mean-rate stable, and maintaining the time average expected power consumption to be lower than a threshold. The optimization problem is formally defined as follows.

$$\text{(P3)} \quad \text{minimize} \lim_{t\to\infty} \sup \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\left\{ d(\tau) + \mu(d(\tau) - d(\tau-1))^2 \right\}, \tag{3.15}$$

$$\text{subject to} \lim_{t\to\infty} \frac{\mathbb{E}\{Q_k(t)\}}{t} = 0 \text{ for all } k, k = 1, 2, \tag{3.16}$$

$$\lim_{t\to\infty} \sup \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\{p(\tau)\} - \bar{p} \leq 0, \tag{3.17}$$

$$e_k(t) = \{0, 1\}, \tag{3.18}$$

$$e_1(t) + e_2(t) \leq 1, \tag{3.19}$$

$$0 \leq m(t) \leq A_{\max}, \tag{3.20}$$

$$0 \leq p(t) \leq p_{\max}, \tag{3.21}$$

where $\bar{p}$ is the maximum average power consumption and μ is the parameter to set the weights of distortion and distortion variation. Note that $h(t)$, $a_2(t)$, $p(t)$ and $d(t)$ are time dependent random variables. (3.16) requires that all queues are

mean-rate stable. The time average expectation (3.15) and (3.17) can be viewed as the objective function (3.12) and inequality (3.10) divided by T and take lim sup of T to infinity, respectively. Since for deterministic optimization algorithm, $b(t)$ and $d(t)$ are functions of deterministic values of $h(t)$ and $a_2(t)$ and the expectation of a deterministic value is itself. In addition, these two optimization problems both require that the queues should be stable. Thus, these two problems are equivalent.

Alternatively, if the requirement of the received video quality is more stringent, we can define the second optimization problem $(P4)$ (distortion constraint) as follows.

$$\text{(P4) minimize } \lim_{t\to\infty} \sup \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\left\{p(\tau)\right\}, \tag{3.22}$$

$$\text{subject to } \lim_{t\to\infty} \frac{\mathbb{E}\left\{Q_k(t)\right\}}{t} = 0 \text{ for all } k, k = 1, 2, \tag{3.23}$$

$$\lim_{t\to\infty} \sup \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\left\{d(\tau)\right\} - \bar{d} \leq 0, \tag{3.24}$$

$$\lim_{t\to\infty} \sup \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\left\{(d(\tau) - d(\tau-1))^2\right\} - \bar{d_v} \leq 0, \tag{3.25}$$

$$e_k(t) = \{0, 1\}, \tag{3.26}$$

$$e_1(t) + e_2(t) \leq 1, \tag{3.27}$$

$$0 \leq m(t) \leq A_{\max}, \tag{3.28}$$

$$0 \leq p(t) \leq p_{\max}, \tag{3.29}$$

where $\bar{d}$ is the maximum expected video distortion and $\bar{d_v}$ is the maximum expected distortion variation. In this problem, we aim to minimize the power consumption while keeping all the queues stable and satisfying the minimum video quality requirement. In the above two problems, we need to perform rate control which determines $m(t)$, scheduling which determines $e_k(t)$, and power control which determines $p(t)$ for each time slot.

3.3 Transmission Control Algorithms

3.3.1 Deterministic Optimization Algorithm

For the deterministic optimization problem, we first solve the power allocation problem followed by the rate allocation. We can use the Lagrangian optimization technique to rewrite the problem as

$$\text{minimize} \sum_{t=0}^{T-1} -b(t) + \lambda \sum_{t=0}^{T-1} p(t). \tag{3.30}$$

By [18], we can minimize $b(t)$ for each time slot independently with common λ. The algorithm is detailed in procedure PowerAllocation in Alogrithm 2. Basically, the algorithm increases λ from 0 until the total power reduces to the total power budget. After each iteration, λ is increased by a constant value $a > 0$. In line 6, $p(t)$ is set to the value which makes the gradient of $-b(t) + \lambda p(t)$ to 0. When the algorithm terminates, we can obtain optimal $p^*(t)$ and $b^*(t)$ which in turn can be used to obtain the maximum transmission rate and solve the rate allocation problem. There are other faster solutions for the power allocation problem, since the objective function is concave and the feasible region is a convex set. Convex optimization technique [3] can be applied to solve the problem, which is left for future exploration. Dynamic programming has been used in rate allocation in conventional video coding [43]. We use a dynamic programming algorithm to obtain the optimal solution. The total amount of transmission rate is known, so the total amount of data left to the measurements is $M_b = b^* - \sum_{t=0}^{T-1} a_2(t)$. The algorithm is detailed in procedure RateAllocation in Algorithm 2. The algorithm maintains a list of possible rate allocation nodes for each time slot. Each node of time slot t represents a rate allocation decision from time slot 0 to t. A node has a few attributes to describe the rate allocation decision: obj is the accumulated weighted sum of distortion and distortion variation; r is the accumulated sum of rates; d is the distortion of the current time slot. Lines 19-21 initialize the nodes of time slot 0. Lines 23-37 iterate all the time slots from 1 to $T-1$ and maintains a node list for each time slot. Lines 28-34 keep the nodes with the smaller objective value when two nodes has the same accumulated rate, and remove nodes whose rates exceeding the target rate budget M_b. When the algorithm terminates, we can find an optimal path of length T which minimizes the objective function.

We designed a simple transmission strategy to make sure that the optimal trans-

Algorithm 2 Deterministic Control Algorithm

procedure PowerAllocation($T, \bar{p}$)
  TotalPowerBudget $\leftarrow T \times \bar{p}$
  $\lambda \leftarrow 0$
  repeat
    for $t \leftarrow 0, T-1$ **do**
      $p(t) \leftarrow \frac{B\eta}{f\lambda \ln 2} - \frac{d^{\alpha} N_0 B}{G|h(t)|^2}$
      if $p(t) > p_{max}$ **then**
        $p(t) \leftarrow p_{max}$
      else if $p(t) < 0$ **then**
        $p(t) \leftarrow 0$
      end if
    end for
    TotalPower $\leftarrow \sum_0^{T-1} p(t)$
    $\lambda \leftarrow \lambda + a$
  until TotalPower $\leq$ TotalPowerBudget
end procedure

procedure RateAllocation(M_b, T)
  $t \leftarrow 0$
  for $m \leftarrow 1, A_{max}$ **do**
    $s.d \leftarrow d(m)$, $s.m \leftarrow m$, $s.obj \leftarrow d(m)$
    Add s to list(t)
  end for
  for $t \leftarrow 1, T-1$ **do**
    for all $s \in$ list($t-1$) **do**
      for $m \leftarrow 1, A_{max}$ **do**
        $n.d \leftarrow d(m)$, $n.r \leftarrow m \times q + s.r$
        $n.obj \leftarrow s.obj + d(m) + \mu(d(m) - n.d)^2$
        if $\exists o \in$ list(t) with $o.r = n.r$ **then**
          if $o.obj > n.obj$ **then**
            $o \leftarrow n$
          end if
        else if $n.r < M_b$ **then**
          Add n to list(t)
        end if
      end for
    end for
  end for
end procedure

mission rate can be fully utilized. Specifically, we delay the transmission of all the data by T slots, thus before transmission, all the data generated during T slots are buffered in the queues. Then it takes another T slots to transmit all the buffered data, since the total amount of transmission rate during the second T-slots duration and the total arrival in the first T-slots duration satisfy inequality (3.13). In this case, the scheduling problem becomes simple, and we can arbitrarily select any queue to transmit so long as the queue is non-empty and the transmission rate can be fully utilized. The transmission delay of this simple transmission strategy is $2T$. When T approaches infinity, the delay goes to infinity as well. The queues are stable, since the average arrival rate is less than the average transmission rate.

3.3.2 Lyapunov Optimization Algorithm

The deterministic control can provide an upper-bound for the transmission control performance, but it is not realistic as it requires the complete knowledge of future channel condition $h(t)$ and the arrival rate $a_2(t)$, and the computation complexity of the rate allocation algorithm is high. Therefore, we mainly focus on the stochastic optimization problems (P3) and (P4) and use Lyapunov optimization technique to solve them. In the stochastic optimization problem formulation, $h(t)$ and $a_2(t)$ for different time slots are considered as i.i.d. random variables and so are their functions $p(t)$ and $d(t)$.

Lyapunov optimization technique is simple to implement (no curse of dimensionality) and does not require the complete knowledge of $h(t)$ and $a_2(t)$ or their statistics but only the $h(t)$ of the current time slot which can be estimated using the feedback from the receiver. In addition, the performance-delay tradeoff has been proved to be $O(1/V)$ and $O(V)$ while ensuring the queue stability [28], which shows explicit convergence of the algorithm.

To solve problem (P3), we define $y(t) = p(t) - \bar{p}$ and a virtual queue $Z(t)$ as follows

$$Z(t+1) = \max\left[Z(t) + y(t), 0\right]. \tag{3.31}$$

If we can make the virtual queue stable, then the inequality constraint (3.17) can be satisfied [28]. For completeness, we present the proof here. By sample path property,

we have

$$\frac{Z(t)}{t} - \frac{Z(0)}{t} \geq \frac{1}{t}\sum_{\tau=0}^{t-1} y(\tau).$$

Then, taking expectation and limit sup of t to infinity, we can obtain

$$\limsup_{t\to\infty} \frac{\mathbb{E}\{Z(t)\}}{t} \geq \limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{y(\tau)\}. \tag{3.32}$$

If the virtual queue $Z(t)$ is mean rate stable, then the left-hand-side of (3.32) is 0 and the right-hand-side of (3.32) is less than or equal to 0. Thus, we have

$$\limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{p(\tau)\} - \bar{p} \leq 0.$$

Let $\Theta(t) = [Q_1(t), Q_2(t), Z(t)]$ be the concatenated vectors of actual queues and virtual queue. The Lyapunov function and one-slot conditional Lyapunov drift are defined as

$$L(\Theta(t)) = \frac{1}{2}\sum_{k=1}^{2} Q_k(t)^2 + \frac{1}{2}Z(t)^2, \tag{3.33}$$

$$\Delta(\Theta(t)) = \mathbb{E}\{L(\Theta(t+1)) - L(\Theta(t))|\Theta(t)\}, \tag{3.34}$$

respectively. According to [28, Lemma 4.6], the drift-plus-penalty is bounded as follows,

$$\begin{aligned}\Delta(\Theta(t)) + V\mathbb{E}\left\{d(t) + \mu((d(t) - d(t-1))^2)|\Theta(t)\right\} \\ \leq B + V\mathbb{E}\{d(t)|\Theta(t)\} + Z(t)\mathbb{E}\{y(t)|\Theta(t)\} \\ + \sum_{k=1}^{2} Q_k(t)\mathbb{E}\{a_k(t) - b_k(t)|\Theta(t)\},\end{aligned} \tag{3.35}$$

where $V > 0$ is a weighting parameter and B is defined as

$$B \geq \frac{1}{2}\sum_{k=1}^{2}\mathbb{E}\left\{a_k(t)^2 + b_k(t)^2|\Theta(t)\right\} + \mathbb{E}\left\{y(t)|\Theta(t)\right\} - \sum_{k=1}^{2}\mathbb{E}\left\{\tilde{b}_k(t)a_k(t)|\Theta(t)\right\}, \tag{3.36}$$

where $\tilde{b}_k(t) = \min[b_k(t), Q_k(t)]$. Instead of minimizing the left-hand-side of (3.35), we minimize the right-hand-side bound opportunistically. The optimization problem (3.15) is casted to the following deterministic optimization problem.

$$\begin{aligned}
\text{(P5)} \quad \text{minimize} \quad & V(d(t) + \mu(d(t) - d(t-1))^2) \\
& + \sum_{k=1}^{2} Q_k(t)(a_k(t) - b_k(t)) + Z(t)y(t), && (3.37) \\
\text{subject to} \quad & e_k(t) = \{0, 1\}, e_1(t) + e_2(t) \leq 1, && (3.38) \\
& 0 \leq m(t) \leq A_{\max}, \quad 0 \leq p(t) \leq p_{\max}. && (3.39)
\end{aligned}$$

We can decompose the above optimization problem into two separate optimization problems. The first one is a rate control problem.

$$\begin{aligned}
\text{(P6)} \quad \text{minimize} \quad & V(d(t) + \mu(d(t) - d(t-1))^2) \\
& + Q_1(t)m(t)q, && (3.40) \\
\text{subject to} \quad & 0 \leq m(t) \leq A_{\max}.
\end{aligned}$$

Since the distortion of the previous time slot, $d(t-1)$, is known, the objective function of problem (3.40) is solely dependent on $m(t)$. We can set the derivative of the objective function to 0, i.e.,

$$\begin{aligned}
& 2\alpha\beta V(2\mu\gamma + (1-2\mu)d(t-1))m^{\beta-1} \\
& + 2\alpha^2\beta V\mu m^{2\beta-1} + Q_1(t)q = 0, && (3.41)
\end{aligned}$$

to obtain the optimal rate value.

The second problem is for scheduling and power control.

$$\begin{aligned} \text{(P7)} \quad \text{minimize} \quad & H = -\sum_{k=1}^{2} Q_k(t) b_k(t) + Z(t) y(t), \\ \text{subject to} \quad & e_k(t) = \{0,1\}, e_1(t) + e_2(t) \leq 1, \\ & 0 \leq p(t) \leq p_{\max}. \end{aligned} \tag{3.42}$$

Rearranging the objective function and removing the fixed term $-Z(t)\bar{p}$, we have

$$\begin{aligned} H = & -\frac{\eta B}{f} \sum_{k=1}^{2} Q_k(t) e_k(t) \log \left(1 + p(t) \frac{d^{-\alpha} G}{N_0 B} |h(t)|^2\right) \\ & + \frac{1}{f} Z(t)(e_1(t) + e_2(t)) p(t). \end{aligned} \tag{3.43}$$

To solve problem (3.42), we consider the three possible scheduling strategies separately, i.e., $e(t) = \{0,1\}$, $\{1,0\}$ and $\{0,0\}$. When $e(t) = \{0,0\}$, $H = 0$ and $p(t) = 0$. For the other strategies, we have

$$H = -\frac{\eta B}{f} Q_k(t) \log \left(1 + p(t) \frac{d^{-\alpha} G}{N_0 B} |h(t)|^2\right) + \frac{1}{f} Z(t) p(t). \tag{3.44}$$

We can set the gradient of (3.44) to zero to obtain the minimizer

$$p(t) = \frac{\eta B Q_k(t)}{Z(t) \ln 2} - \frac{d^{\alpha} N_0 B}{G |h(t)|^2}, \tag{3.45}$$

then we can choose the scheduling strategy and $p(t)$ which minimize H. The algorithm for each time slot is summarized as follows.

Step 1: Solve equation (3.41) to obtain $m(t)$. If $m(t)$ is out of the range of $[0, A_{max}]$, set $m(t)$ to the nearest boundary.

Step 2: For all 3 possible scheduling strategies, solve (3.45) to obtain $p(t)$. If $p(t)$ is out of the range of $[0, p_{max}]$, set $p(t)$ to the nearest boundary. Then use $p(t)$ to calculate H based on (3.44). Choose the scheduling strategy and $p(t)$ which minimize H.

For problem (P4), similarly, two virtual queues can be built to satisfy the distortion and distortion variation constraints. Specifically, we define the following two

virtual queues

$$Z_1(t+1) = \max\left[Z_1(t) + y_1(t), 0\right], \tag{3.46}$$

$$Z_2(t+1) = \max\left[Z_2(t) + y_2(t), 0\right], \tag{3.47}$$

where $y_1(t) = d(t) - \bar{d}$ and $y_2(t) = (d(t) - d(t-1))^2 - \bar{d}_v$. Similar to problem (P3), we minimize the right-hand-side bound of (3.35), then decompose the problem into a power control problem and a rate control problem and solve them separately. As the approach is similar to that for problem (P3), we do not include the full details.

3.4 Evaluation

In this section, we evaluate the performance of the proposed rate-distortion model and the joint rate control, scheduling and power control algorithm. We adopted compressive sensing components similar as those in [48]. The structurally random matrix [15] is chosen as the measurement matrix, Undecimated wavelet transform (UWT) is chosen as the basis, and analysis-based ℓ_1 minimization is used to recover the signal from the measurements.

3.4.1 Rate-Distortion Model

Two video sequences "Bus" and "Foreman" [45] are used to test how well the model capture the rate-distortion characteristic. The resolution of the two videos are 176×144 and the frame rate is 15 frames per second. For each video sequence, we randomly selected 3 frames and obtain the rate-distortion curve for each frame using model (3.7) and (3.8), respectively. Specifically, we generate 10 sets of different number of measurements for each frame: $200, 500, 1000, 2000, 3500,$ $5000, 7500, 10000, 15000,$ and 20000. We set the number of bits for each measurement to 6. We recover the images from these measurements and calculate the distortion, so we can obtain 10 rate-distortion points for each frame. We then use these rate-distortion points to fit the models. Figs. 3.2 (a) and (b) show the rate-distortion relationship captured by (3.7) for the two video sequences; (c) and (d) show the results using model (3.8) for the two video sequences. We can see that model (3.8) can capture the rate-distortion characteristic better, because the extra parameter γ brings one more degree of freedom. Other videos are tested and also show that model (3.8) is better than

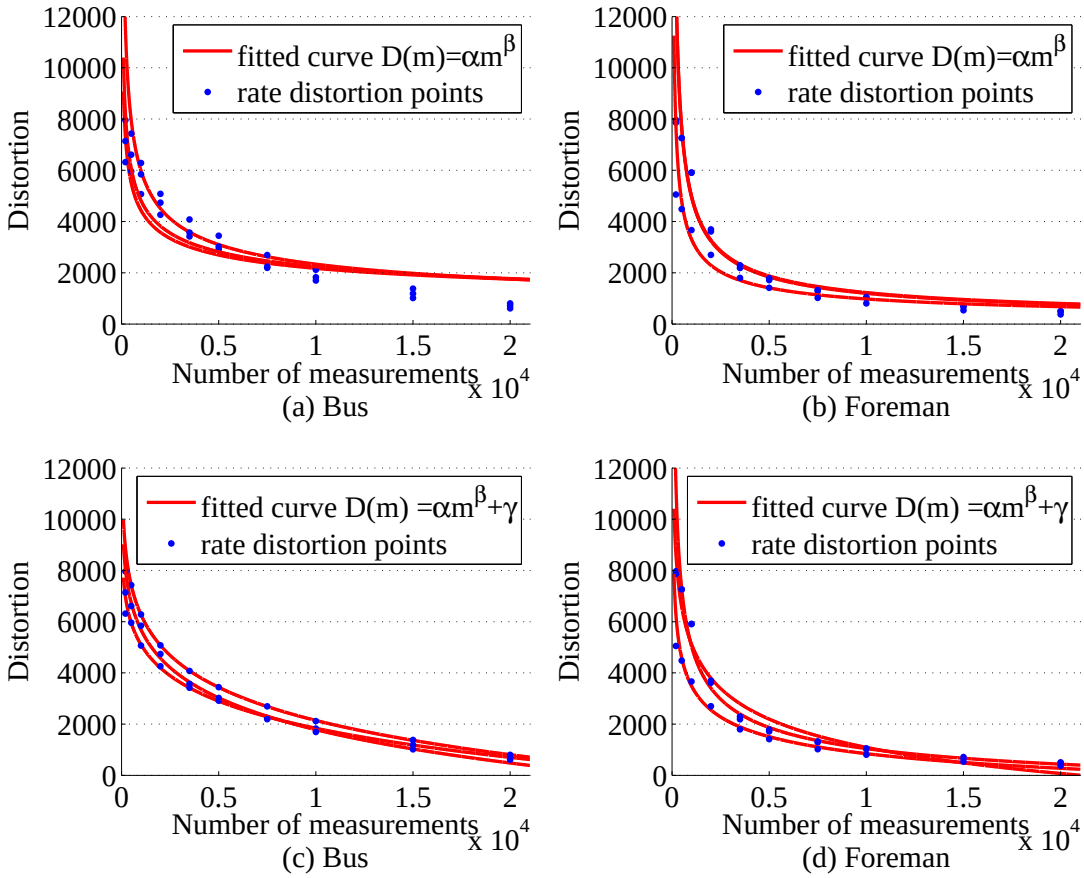


Figure 3.2: Model comparison

model (3.7). In the following, we adopt model (3.8) for the rate-distortion estimation. The proposed model can capture the rate-distortion relationship of all the frames of various types of videos. However, the parameters of the model may vary from frame to frame depending on the characteristic of the video. We can obtain the model parameter for each frame, but it suffers from high computational cost. Therefore, to reduce the computational cost in obtaining the parameters of the model, we used the same set of parameters for 15 consecutive frames to explore the temporal redundancy of these frames. This value can be larger for video surveillance applications where the videos are less dynamic. The distortion is calculated using $\|x - \hat{x}\|_{\ell_2}$, where x is the original image signal and $\hat{x}$ is the reconstructed image signal. It is converted to PSNR by PSNR $= 10 \times \log_{10}(\frac{255\times255\times H\times W}{\|x-\hat{x}\|_{\ell_2}})$, where H and W are the height and width of the image, respectively.

V	Avg. distortion (PSNR dB)	Avg. distortion variation (dB)	Avg. Q1 backlog (packets)	Avg. Q2 backlog (packets)	Avg. Q1 arrival rate (kbits)	Avg. power (Watt)
1	7209.67 (15.01)	308.32 (0.42)	0.58	0.31	2.64	0.65
10	5486.04 (17.38)	167.60 (0.29)	1.64	0.75	8.25	0.74
10^2	3767.88 (20.65)	98.99 (0.23)	4.55	2.01	23.26	0.79
10^3	2209.03 (25.29)	57.97 (0.23)	12.44	4.57	55.09	0.80
10^4	1529.25 (28.48)	42.02 (0.23)	63.04	17.92	78.75	0.80

Table 3.2: Problem P3, power constraint, $\bar{p} = 0.8$ Watt

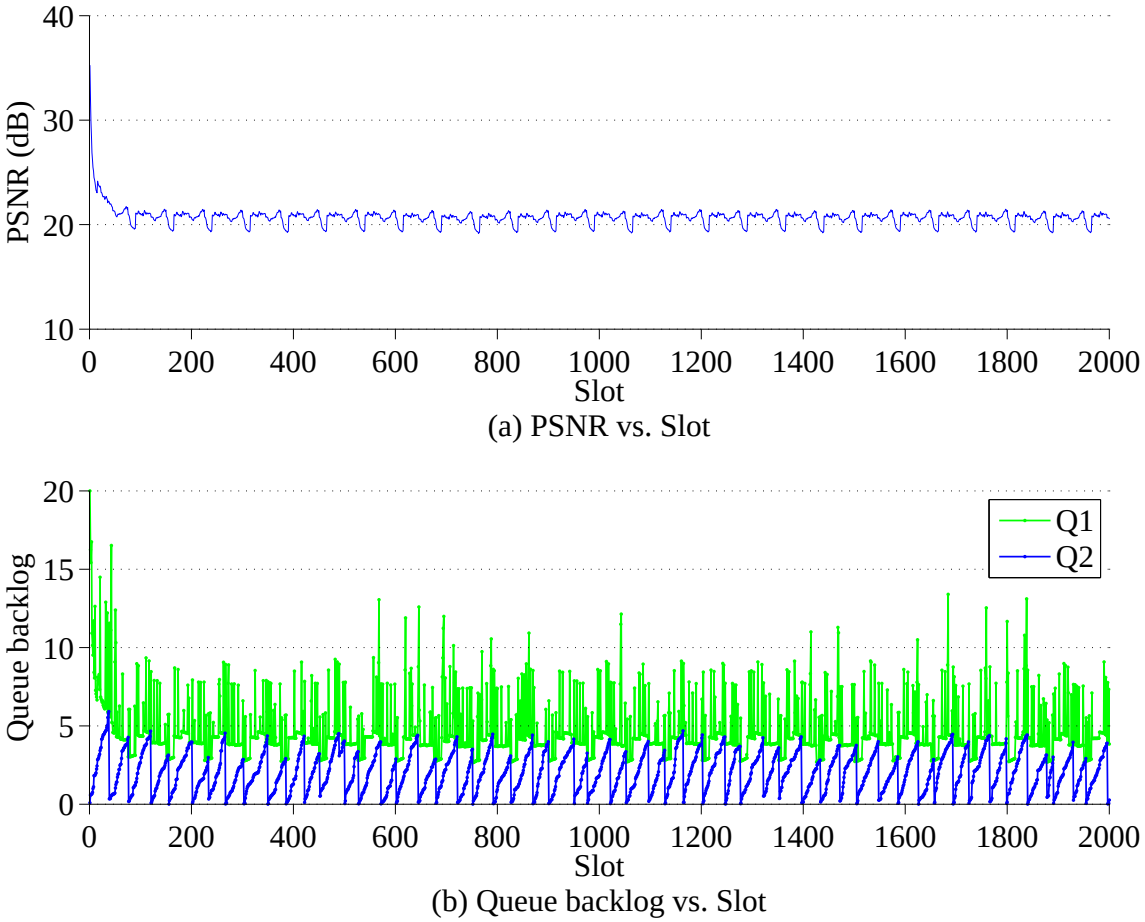
3.4.2 Algorithm Evaluation

In this section, we evaluate the performance of the joint rate control, scheduling and power control algorithm. We consider a wireless transmission system with $B =$ 600 KHz and transmission power $0 \leq p \leq 1$ Watt. We set the reference SNR to 4.75 dB when the transmission power is 0.5 Watt, and $\gamma_0 = \frac{d^{-\alpha}G}{N_0 B} = 5.97$. The transceiver efficiency η is set to 0.9. We assume that the arrival rate of the aggregated traffic from the non-video sensing data $a_2(t)$ confirms to the Poisson distribution with the arrival rate of 0.1 packet per time slot and the packet size is 6 kbits.

For the deterministic control algorithm, the average power consumption is set to 0.8 Watt. The total number of time slots is 5000. In order to reduce the complexity of the rate control algorithm, we perform rate allocation for every group of 50 slots and the parameter M_b is the summation of the transmission rate of the corresponding slots. Meanwhile, the number of all possible measurements per frame is reduced from 20000 to 40 with an interval of 500 measurements. The results of the algorithm are as follows. The average distortion is 1502.86 (28.63 dB), the average distortion variation is 30.84 (0.18 dB), the average power consumption is 0.8 Watt.

For problem (P3), we set the time average power $\bar{p}$ to 0.8 Watt. On each time slot, we perform the control decisions obtained using Lyapounov optimization.

Table 3.2 shows the simulation results. When V is larger, the flow control algorithm will admit more measurements into Q_1, as the average number of bits increases from 2.64 to 78.75 kbits. When V is sufficiently large, the algorithm fully utilize



(a) PSNR vs. Slot

(b) Queue backlog vs. Slot

Figure 3.3: $P3$ trace $V = 100$

the channel transmission rate and achieve minimum distortion. Since the number of measurements per frame is increased, the average distortion is reduced. Because the trade-off between queueing delay and performance of min drift-plus-penalty algorithm, the average queue backlog of Q_1 and Q_2 increased from 0.58 to 63.04 and from 0.31 to 17.92 packets, respectively. The virtual queue $Z(t)$ is stable, so the average power consumption is less than or equal to the desired value 0.8 W. Compared with the results of the upper bound (with the deterministic control algorithm), as V is increased, the distortion and distortion variation are approaching to the optimal value of the deterministic algorithm, which demonstrate the effectiveness of the stochastic algorithm.

Fig. 3.3 shows the simulation trace of the power constraint problem (P3) for $V = 100$. Figs. 3.3 (a) and (b) show the PSNR vs. slot and queue backlog vs. slot, respectively. The total number of slots of the simulation is 5000, and the figures

V	Avg. distortion (PSNR dB)	Avg. distortion variation (dB)	Avg. Q1 backlog (packets)	Avg. Q2 backlog (packets)	Avg. Q1 arrival rate (kbits)	Avg. power (Watt)
10^4	1998.43 (26.16)	65.95 (0.3)	14.30	5.61	61.98	1.00
5×10^6	1998.87 (26.15)	65.80 (0.29)	15.39	5.60	62.02	0.88
10^8	1999.26 (26.15)	65.28 (0.29)	17.70	5.82	61.74	0.69

Table 3.3: Problem P4, distortion constraint, $\bar{d} = 2000$, $\bar{d_v} = 38.72$

show the first 2000 slots as the remaining slots have the similar pattern. The average number of measurements is 3876.7 and the average PSNR is 20.65 dB. From the figure, the actual queues are stable and their average queue backlog are listed in Table 3.2. At the beginning of the simulation, backlog of Q_2 is increasing, because the scheduling algorithm always choose Q_1 to transmit or transmit nothing until the queue backlog of Q_1 and Q_2 are similar. During this time interval, the backlog of Q_1 is much larger than that of Q_2, so it is more urgent to reduce the queue backlog of Q_1. Then this pattern continues. The trace figures for other values of V are similar.

For problem (P4), the maximum time average expected distortion $\bar{d}$ is set to 2000 (26.15 dB), and $\bar{d_v}$ is set to 1500. The objective is to minimize power consumption.

The simulation results are shown in Table 3.3. When V is increased from 10^4 to 10^8, the average distortion is approaching the desired average distortion and the average power is reduced from 1 to 0.69 Watt. Due to the trade-off between the queueing delay and the performance of min drift-plus-penalty algorithm, the average queue backlog of Q_1 and Q_2 increased from 14.30 to 17.7 and from 5.61 to 5.82 packets, respectively. The virtual queues are stable, so the time average expected distortion $\bar{d}$ is smaller than 2000. The distortion variation is larger than the target 38.72. It is because we update the rate-distortion every 15 frames, which leads to the inaccuracy. However, the distortion variation is still acceptable.

3.4.3 Incremental-V algorithm

From the previous section, we can see that when increasing V, the objective function is pushed to the optimal value but the average queue length and average queueing

delay is increased as well. For video transmission applications, we need to control the queueing delay. In this section, we propose a variant algorithm to achieve the above goal. We increase the value of V every T time slots according to

$$V(k) = V_0(1 + kT)^{\sigma}, \tag{3.48}$$

where V_0 is the initial value of V and σ controls the growing speed of V. [28] has shown that the Lyapunov optimization algorithm can achieve optimality when increasing V according to the above equation every time slot. However, for the purpose of controlling the average delay, we increase V until the average transmission delay during the last T time slots reaches the target delay t_d.

To evaluate the algorithm, we set T to 5000 time slots, σ to 0.45, and t_d to 400 ms. Fig. 3.4 shows the trace of the incremental-V algorithm, where (a) and (b) show the PSNR vs. slot and queue backlog vs. slot, respectively. From the figure, the PSNR and queue length increase in a staircase manner until the average transmission delay reaches t_d. V is increased and stopped at 135.9, the average video distortion is 3728.08 (20.74 dB); average queue length of Q_1 and Q_2 are 4.86 and 2.1 kbits, respectively; and the average power consumption is 0.79 W. In this way, we can push the objective function to the optimal value while maintaining the average transmission delay within an acceptable range.

The incremental-V algorithm shows the same convergence property and performance bound in each time period T as the previous algorithms, since the value of V is fixed for each time period. The number of time periods needed before the convergence of the algorithm to the target transmission delay depends on parameter V_0 and σ in (3.48). If we choose a larger value of V and σ, the step size of V is larger and the value of V is increased quickly, and thus it may increase the convergence speed. However, it may also lead to a longer transmission delay than expected. How to choose V_0 and σ to reduce convergence time is left for future exploration.

3.5 Summary

In this chapter, we have proposed a simple and effective rate-distortion model for compressive sensed video. Using this model, we can estimate the video distortion given the number of measurements used in signal recovery. We have formulated a deterministic optimization problem as a bench mark, and a more practical stochastic optimization

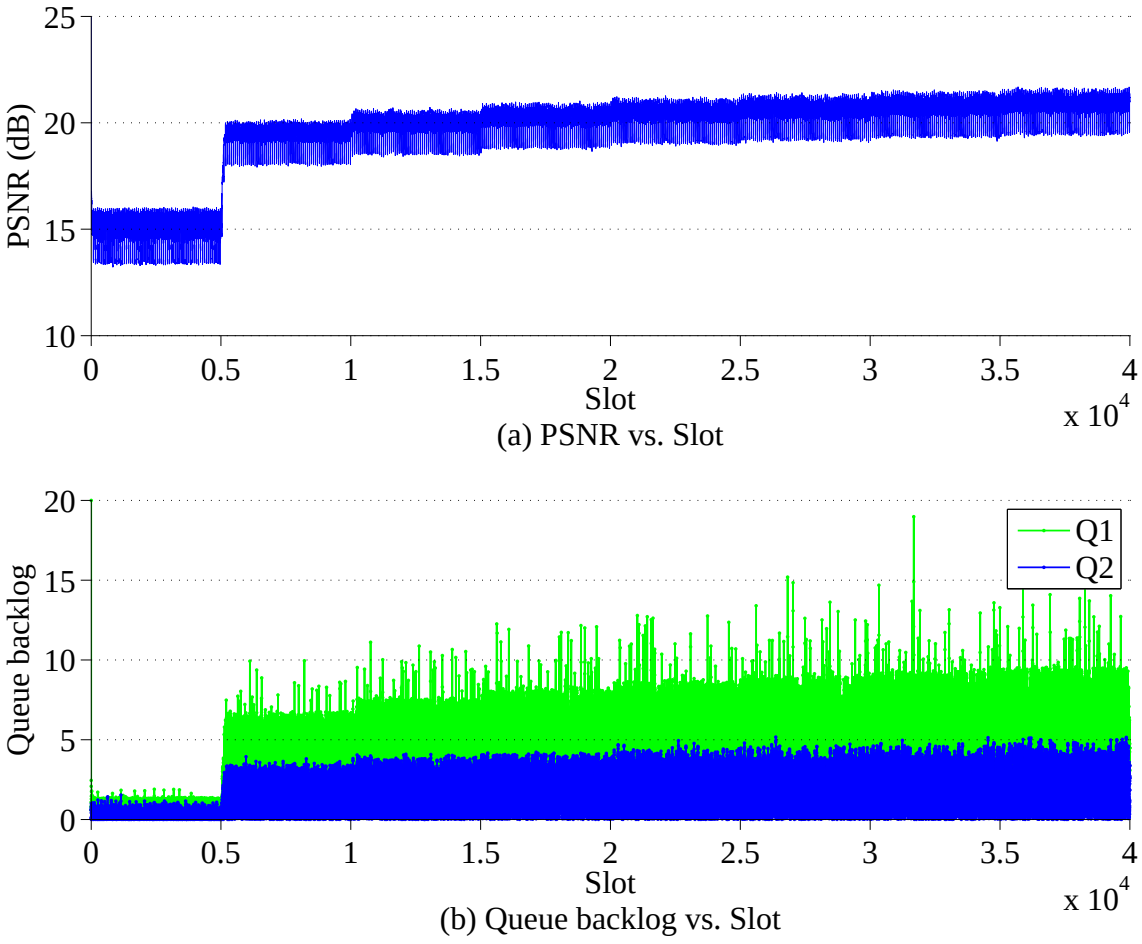


Figure 3.4: Incremental-V algorithm trace , $V = 135.89$

problem for transmitting compressive sensing video over wireless channels. A supplementary stochastic optimization problem has been formulated to consider distortion and power consumption with different priority. The proposed stochastic algorithm does not need to know the complete future channel condition and arrival rate and its computational complexity is low.

We have further proposed an incremental-V algorithm, which helps to choose the value of V while keeping the average transmission delay in a reasonable range. The simulation results have shown the effectiveness of the proposed transmission control algorithms.

Chapter 4

Scalable Video Coding with Compressive Sensing for Wireless Videocast

The main contributions of this chapter are three-fold. First, we propose a low-complex, scalable video coding architecture based on compressive sensing (SVCCS) for wireless unicast and multicast transmissions. SVCCS achieves good scalability, error resilience and coding efficiency. An SVCCS encoded bitstream is divided into a base layer and an enhancement layer. The layered structure provides quality and temporal scalability. The base layer is composed of a small portion of discrete cosine transform (DCT) coefficients. The enhancement layer consists of compressive sensed measurements. While in the enhancement layer, the CS measurements provide fine granular quality scalability. In addition, we incorporate the state-of-the-art technologies of compressive sensing (e.g., analysis-based ℓ_1 minimization and dictionary) to improve the coding efficiency. Second, we study the performance of SVCCS and the contribution of each component of the codec. We also compare the performance of SVCCS and MJPEG in wireless video multicast. Third, we investigate the rate allocation problem for multicasting SVCCS encoded bitstream to a group of receivers with heterogeneous channel conditions. Specifically, we study how to allocate rate between the base and enhancement layer to improve the overall perceived video quality for all the receivers while satisfying the real-time video transmission delay requirement. We first build a rate distortion model to capture the rate distortion charactersitics of the SVCCS encoded bitstream. Then we propose a rate allocation algorithm using this

model. Simulation results show that SVCCS is more effective and efficient for wireless videocast than the existing solutions. We also demonstrate the accuracy of the proposed rate distortion model and the effectiveness of the proposed rate allocation algorithm.

The rest of the chapter is organized as follows. Section 4.1 gives a brief introduction of compressive sensing and the related work. Section 4.2 describes the architecture of SVCCS. Section 4.3 describes the rate allocation problem and its solution. The performance of SVCCS, the proposed rate distortion model and rate allocation algorithm are studied in Section 4.4, followed by concluding remarks in Section 4.5.

4.1 Related Work

The application of compressive sensing in video and image has attracted a lot of research efforts [31, 25, 54, 10, 40]. However, there are not many research focusing on joint image/video coding based compressive sensing and transmission. There are a few related work using compressive sensing as joint source and channel coding framework. [11] used compressive sensing to protect sparse signals over erasure channels. The proposed method compensates the lost measurements during transmission by sending more measurements.

With respect to image and video transmission, [33] designed a video encoder based on CS and a streaming protocol for wireless video transmission. To exploit temporal redundancy, the difference frame of the I frame and the target frame is compressive sensed in [33]. It means that the original video frame is used as the reference frame in the encoder side, while the decoder uses the recovered image as the reference frame. The discrepancy will degrade decoded video quality. When there is a transmission error in the reference frame, the error will affect the frames depending on it.

In [46], the hardware and algorithm to acquire image and video signals were proposed, which exploited the correlation of adjacent frames. They used the joint measurement matrix and 3D wavelets as the representing basis, encoding several frames or even the whole video sequence and then recovering the frames together. However, this method increases computational complexity and also increases both the encoding and decoding delay.

Therefore, in this chapter, we propose a low-complex, scalable video coding architecture based on compressive sensing, which utilizes temporal redundancy and the state-of-the-art technologies to improve the compressive sensing coding efficiency and

makes it resilient to transmission errors.

4.2 Proposed Video Coding Architecture

Previously, coding efficiency of compressive sensing cannot compete with conventional codec such as JPEG or MPEG4 when dealing with already acquired image or video signals with high resolution and quality. Compressive sensing only shows its advantage when it acquires and compresses image at the same time. Our goals of designing SVCCS are two-fold: 1) improve its coding efficiency, and 2) make it error resilient. In this section, we describe how these goals are achieved.

4.2.1 Layered Structure Design

Figs. 4.1 and 4.2 illustrate the proposed video encoder and decoder architecture, respectively. As shown in Fig. 4.1, video frames are divided into two categories, i.e., I frames and P frames. I frames are DCT transformed and d coefficients are extracted in a zigzag order, then uniformly quantized and entropy encoded. Although the number of these coefficients is small, they contain the majority energy of the image. Therefore, after these coefficients are inversely quantized and inversely DCT transformed, the resultant image provides moderate image quality and can be used as a reference frame. Then the difference between the reference frame and the I or P frame, called the difference frame, is fed into the compressive sensing block.

The concept of the reference frame comes from conventional video codecs, where temporal redundancy of adjacent frames is exploited to improve coding efficiency. For conventional video codecs, the difference of adjacent images contains less energy thus needs less bits to represent. For compressive sensing, the difference of adjacent images is more sparse and compressible.

Motion compensation is not adopted although it can improve the coding efficiency greatly, since it is the most computational complex operation in the traditional encoder. In addition, motion compensation requires to transmit the motion vectors error-free. If motion vectors are corrupted during transmission, degradation of received video quality is inevitable.

In order to minimize transmission errors, a small portion of DCT coefficients are chosen as the reference frame instead of the whole I frame, since it is easier and less costly to protect the small amount of the DCT coefficients than the whole I frame.

As any part in the reference frame is corrupted, error will propagate among the entire group of pictures (GOP). We can see the similarity between the proposed video codec and the scalable video coding (SVC) [39]. The reference frame constitutes the base layer and the compressive sensed difference frame composes the enhancement layer. The layered structure of the proposed video coding architecture thus preserves the quality and temporal scalability.

Fig. 4.3(a) shows one possible GOP structure. The GOP size is four. The arrowed dashed lines indicate the dependence between frames. From the figure, we can see that P frames are dependent on the closest I frame(s)' reference frame(s). The P frame in the middle is dependent on the average of the two reference frames to better exploit temporal redundancy. Fig. 4.3(b) shows an alternative GOP structure. Every frame consists of the base layer and enhancement layer and there is no dependency between frames. Since this type of GOP structure does not exploit the temporal redundancy, the compression ratio might be lower compared with the GOP structure in Fig. 4.3(a). But there is no GOP structure delay and it is more suitable for time-sensitive video applications.

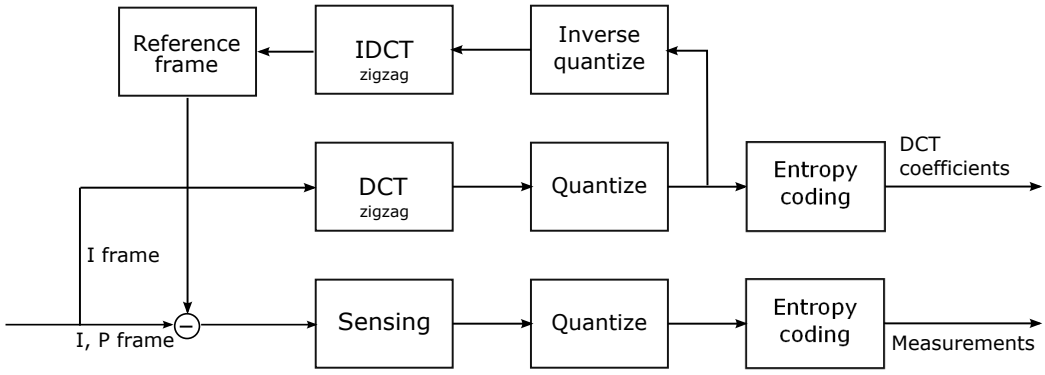


Figure 4.1: encoder

4.2.2 Components of Compressive Sensing

The structurally random matrix [15] is chosen as the measurement matrix which can be described as

$$\Phi = QWP, \tag{4.1}$$

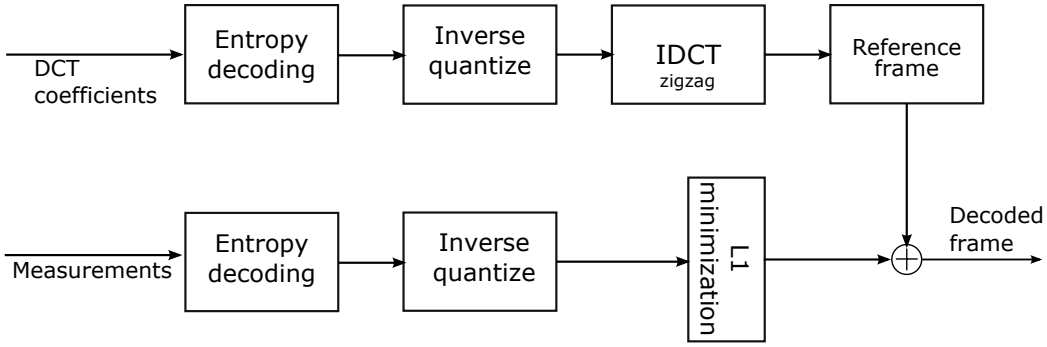


Figure 4.2: decoder

where matrix $P \in R^{n\times n}$ is a global randomizer which randomly permutes matrix W. Matrix $Q \in R^{m\times n}$ randomly selects m rows of WP. $W \in R^{n\times n}$ is a block diagonal matrix.

In addition to the synthesis-based ℓ_1 minimization, an alternative analysis-based ℓ_1 minimization can be formulated as

$$\begin{aligned} \min \quad & \|\Psi^* \tilde{x}\|_{\ell_1}, \\ \text{subject to} \quad & \|y - \Phi\tilde{x}\|_{\ell_2} \leq \epsilon. \end{aligned} \tag{4.2}$$

According to [9], when Ψ is an orthonormal basis, the above two problems are equivalent. But when Ψ is a redundant dictionary, analysis-based ℓ_1 minimization involves less unknowns so the recovery performance is superior.

Undecimated wavelet transform (UWT) is chosen as the basis Ψ. UWT has been found to outperform the orthogonal wavelet transform in image denoising. It is expected to enhance the sparsity of image.

4.2.3 Quantization

DCT coefficients and measurements are uniformly quantized. Let l be the number of bits to represent a measurement. $V_{\max} = \max(|y_i|)$. Range $[-V_{\max}, V_{\max}]$ is divided into 2^l bins. Then the measurement can be represented by the sequence number of the bin which contains it. The quantization step size $q = 2V_{\max}/2^l$. DCT coefficients quantization follows the same way. The variance of quantization error can be

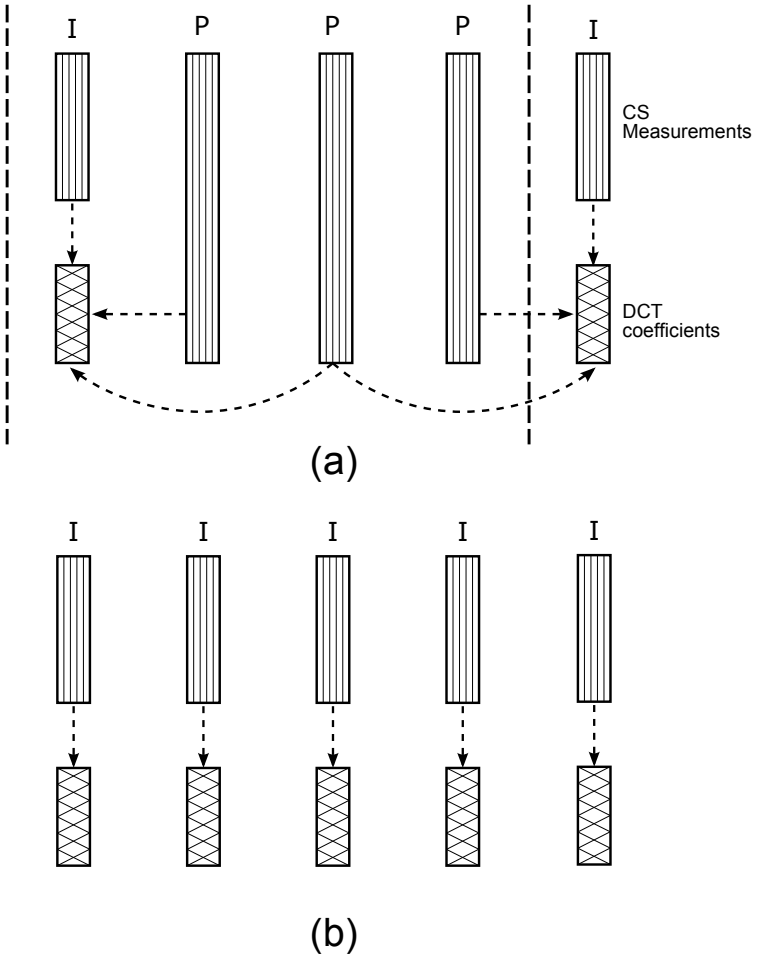


Figure 4.3: GOP Structure. Vertical dashed lines contain a group of pictures of size four.

estimated by

$$\Delta = \sqrt{\frac{q^2 m}{12}} = \frac{V_{\max}\sqrt{m}}{\sqrt{3}\cdot 2^l}. \tag{4.3}$$

When there is no other noise except quantization error, $\epsilon = \Delta$.

Last, we adopt the simple entropy coding technology, the Huffman code, to further improve the code efficiency.

Table 4.1: Symbol list

Symbol	Description
f	frame rate per second
r_c	code rate of the convolutional code
C_i	SNR of user i
p_i	measurement loss rate for user i
u	number of DCT coefficients for each frame
$u_{\max}$	maximum number of DCT coefficients for each frame
m	number of CS measurements for each frame
$m_{\max}$	maximum number of CS measurements for each frame
a, b, c, d, e, g	distortion model of SVCCS
c_u	average compression ratio of DCT coefficients
c_m	average compression ratio of CS measurements
b_u	number of bits for each DCT coefficient
b_m	number of bits for each CS measurement
$d(u, m)$	distortion model of u and m
$r_m(m)$	rate model of m
$r_u(u)$	rate model of u

4.3 Rate Allocation Problem

4.3.1 System Model

We consider a base station using the proposed SVCCS to encode the real-time videos and multicasting to a group of users with heterogeneous channel conditions. The symbols of the system model are listed in Table 4.1. The frame rate of the video is f frames per second. Every $1/f$ seconds, the base station encodes one frame and multicasts it to the receivers. Since the base layer is more important than the enhancement layer, it is protected using a convolutional code with code rate r_c and with DBPSK modulation and link layer retransmission to ensure error-free transmission; while for the enhancement layer, we exploit the joint source and channel coding feature, so the enhancement layer is transmitted without additional channel coding and with a higher order modulation such as 16QAM. Each receiver, USER_i, sends its SNR C_i to the base station. From the SNR, the base station can estimate the BER and thus the measurement loss rate p_i of the measurements for each user. Based on the SNR feedback and the rate distortion characteristics of the SVCCS bitstream, the base station needs to determine the number of DCT coefficients u for the base layer and the number of measurements m for the enhancement layer for each frame.

4.3.2 Rate Distortion Model

We build a rate distortion model to capture the rate distortion relationship. Specifically, we build a rate model and distortion model of the number of DCT coefficients u and the number of measurements m. We assume that the number of bits for the DCT coefficients and measurements is fixed. The optimal number of bits to minimize distortion is left for future exploration. We build the models for the GOP structure shown in Fig. 4.3(b), which is more suitable for time-sensitive video applications. The models can be extended easily for GOP structure in Fig. 4.3(a). The distortion model is

$$d(u, m) = (au^b + c)m^{(du+e)} + g, \tag{4.4}$$

where a, b, c, d, e and g are the model parameters. For each fixed u, the distortion solely depends on m and it exhibits power law decaying tendency, $au^b + c$ can map to α in model (3.8) and $du + e$ can map to β in model (3.8). So it is similar to the rate distortion model (3.8). The rate models are as follows.

$$r_u(u) = c_u b_u u, \tag{4.5}$$

$$r_m(m) = c_m b_m m, \tag{4.6}$$

where c_u and c_m are model parameters, which denote the average compression ratio of DCT coefficents and measurements, respectively; b_u and b_m are the number of bits for each DCT coefficient and measurement, respectively. We choose a linear model since the compression ratio is approximately constant and independent of the number of DCT coefficients and measurements according to our simulation results.

4.3.3 Rate Allocation Problem Formulation

The objective of the problem is to minimize the summation of the distortion of all receivers while satisfying the delay constraint of real-time video transmission. The

optimization problem is defined as follows.

$$\text{minimize} \quad J(u,m) = \sum_i \left((au^b + c)((1-p_i)m)^{(du+e)} + g\right) \tag{4.7}$$

$$\text{subject to} \quad \frac{c_u b_u u/r_c}{r_b} + \frac{c_m b_m m}{r_e} \leq \frac{1}{f}, \tag{4.8}$$

$$0 \leq u \leq u_{max}, \tag{4.9}$$

$$0 \leq m \leq m_{max}. \tag{4.10}$$

In the above, r_b and r_e are the data rate for the base layer and the enhancement layer, respectively. The constraint requires every frame being delivered in time, which indicates that the video bitrate should be less than the total transmission rate of the base and enhancement layers.

Although the rate allocation problem is formulated for multiple users, the solution can be used for video unicast transmission simply by reducing the number of receivers to one. In the problem formulation, we assume that the channel condition is stable, but it can be easily extended to the situation where the channel condition is varying. In the latter case, we can assume that the channel condition is stable in a short time interval and update p_i periodically.

4.3.4 Rate Allocation Algorithm

We apply Lagrangian technique to solve the optimization problem. First, we rewrite the optimization problem as follows.

$$\begin{aligned}\text{minimize} \sum_i \left((au^b + c)((1-p_i)m)^{(du+e)} + g\right) \\ + \tau \left(\frac{c_u b_u u/r_c}{r_b} + \frac{c_m b_m m}{r_e}\right),\end{aligned} \tag{4.11}$$

where $\tau > 0$ is the Lagrangian multiplier. Then we can find a τ to satisfy the constraint. To simplify the solution to the above unconstraint optimization problem, we first fix the value of u. Specifically, for a given value of u, we can find the minimum value of m which minimizes the objective function. We set the partial derivative of

the objective function on m to 0, i.e.,

$$\frac{\partial J(u,m)}{\partial m} = (au^b + c)(du + e)m^{du+e} \sum_i (1 - p_i)^{du+e} + \tau c_m b_m / r_e = 0. \tag{4.12}$$

By rearranging the terms, we can obtain the closed-form solution,

$$m = \left\lfloor \left(\frac{-\tau c_m b_m}{B r_e} \right)^{\frac{1}{du+e-1}} \right\rfloor, \tag{4.13}$$

where $B = (au^b + c)(du + e) \sum_i (1 - p_i)^{du+e}$. Then we can find the minimizer u and m which minimize the objective function. The algorithm is summarized in algorithm 3. Lines 4-22 search the Lagrangian multiplier τ which satisfies the transmission

Algorithm 3 Rate Allocation Algorithm

```
1: procedure RateAllocation
2:     TransmissionTime ← +∞
3:     τ ← 0
4:     repeat
5:         Obj ← +∞
6:         for u ← 1, u_max do
7:             m_u = ⌊ ( −τ c_m b_m / r_e ) / B )^{1/(du+e−1)} ⌋
8:             if m_u > m_max then
9:                 m_u ← m_max
10:            else if m_u < 0 then
11:                m_u ← 0
12:            end if
13:            Obj_u ← J(u, m_u)
14:            if Obj_u < Obj then
15:                u* ← u
16:                m* ← m_u
17:                Obj ← Obj_u
18:            end if
19:        end for
20:        TransmissionTime ← u* b_u c_u / r_u + m* b_m c_m / r_m
21:        τ ← τ + δ
22:    until TransmissionTime < 1/f
23: end procedure
```

delay constraint. Lines 6-19 iterate all the possible value of u, and calculate the corresponding value of m_u which minimizes the objective function and choose u^* and m^* which minimize the objective function from all the possible pairs of u and m_u. $u_{\max}$ and $m_{\max}$ are the maximum values of u and m, respectively. Line 21 increments τ by a positive constant δ.

4.4 Performance study

In this section, we first investigate the reasoning behind the codec design and its performance and compare the performance of the proposed SVCCS and the traditional solutions in supporting wireless multicast applications. Then we evaluate the proposed rate-distortion model and rate allocation algorithm.

4.4.1 Performance of SVCCS

We use "Foreman" [45] as the test video sequence to investigate the SVCCS performance. The resolution is QCIF(176×144), and frame rate is 15 frames per second. We use NESTA [4] which is a routine that can solve analysis-based ℓ_1 minimization.

Fig. 4.4 shows the compressibility of an original frame and a difference frame. A difference frame is obtained by subtracting two consecutive original frames. We use the db4 UWT with four levels. Since UWT is redundant, the number of resultant transform coefficients is 16 times larger than the frame size. Then, we sort the magnitude of the coefficients in descending order. From the figure, we can see that not only the difference frame's energy is reduced, but also the coefficients of the difference frame decay much faster and the portion of small coefficients is lager than that of the original frame.

Fig. 4.5 shows the distribution of DCT coefficients and measurements. Since the DCT coefficients contain the majority of the energy of an image and are more important, we use more bits (9 bits) to represent a DCT coefficient than that for a CS measurement (4 bits). There are 500 DCT coefficients and 12000 measurements for each frame. From the figure, the distribution is close to a Gaussian distribution. Therefore, the coding efficiency can get benefit from the Huffman coding. With Huffman coding, the average number of bits of each DCT coefficient and measuremen are reduced to 5.5 and 2.9 bits, respectively.

In order to study the contribution of each component of the video codec, we turn

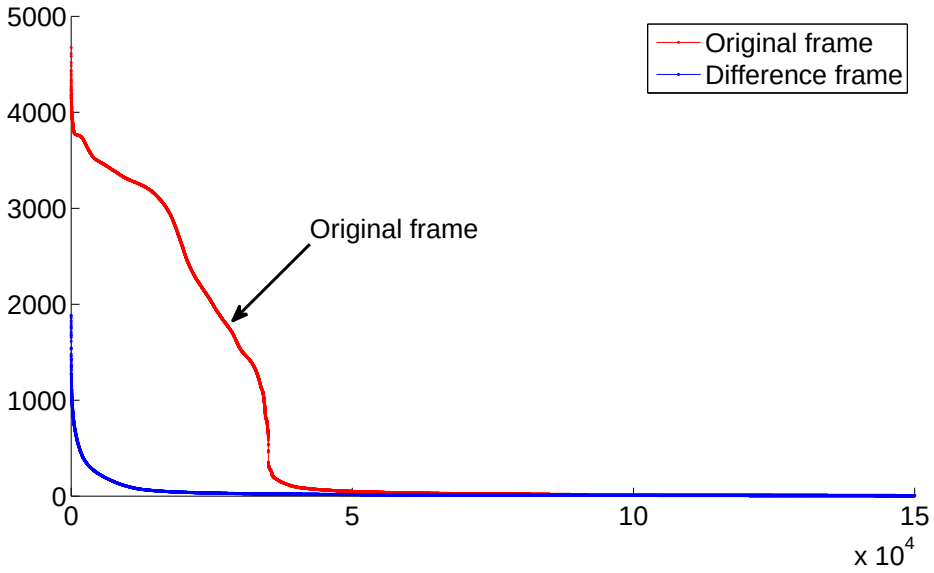


Figure 4.4: Compressibility of an original and difference frame.

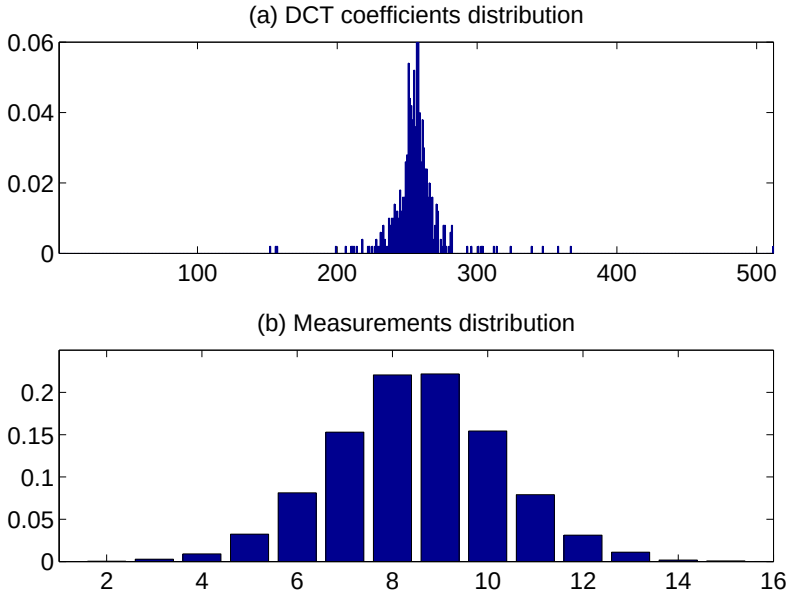


Figure 4.5: Distribution of DCT coefficients and measurements

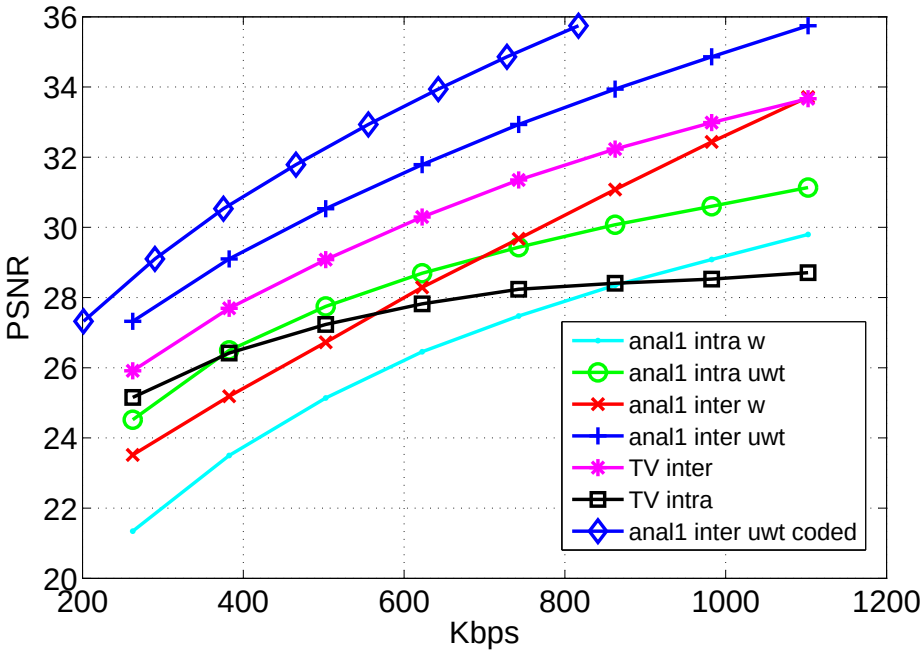


Figure 4.6: Comparison of components. anal1 denotes analysis-based ℓ_1 minimization; inter denotes inter-coding with GOP structure IPPP; intra denotes all frames are intra-coded; u denotes biorthorgonal 9/7 wavelet transform and uwt denotes UWT.

off the component or use an alternative to see the effect of the tagged component. We compare analysis-based ℓ_1 minimization with total variation(TV) [36], 9/7 wavelet transform with UWT, intra-coding with inter-coding, and entropy coding enabled with entropy coding disabled. In our study, we set the number of DCT coefficients to be 500 with 9 bits for each coefficient before Huffman coding. We change the coding rate by changing the number of measurements but with fixed 4 bits for each measurement. We use the same GOP structure depicted in Fig. 4.3 for inter-coding.

From Fig. 4.6, we can see that inter-coding is better than intra-coding for both analysis-based ℓ_1 and TV minimization. PSNR is increased by 12% (3.43 dB) on average for analysis-based ℓ_1 with UWT case and inter-coding. If analysis-based ℓ_1 and inter-coding are enabled, the UWT is 11% (3.2 dB) better than biorthorgonal 9/7 wavelet transform on average. Analysis-based ℓ_1 with inter-coding is 5% (1.6 dB) better than TV with inter-coding. When we use entropy coding, the bitrate is further reduced by 25% on average.

Next, we study the scalability of SVCCS. The effect of measurement loss is the same as changing the size of the measurement matrix. When the measurements are lost, we just eliminate the corresponding rows in the measurement matrix and

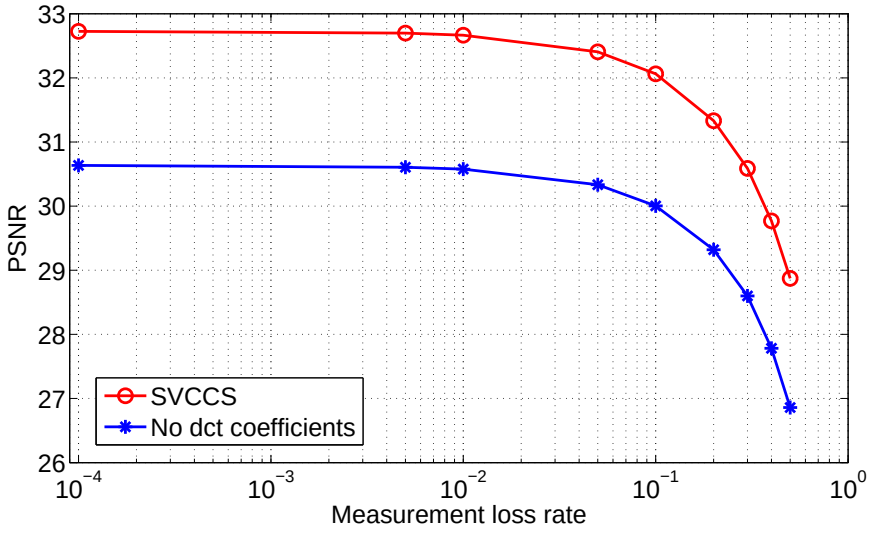


Figure 4.7: PSNR vs. measurement loss rate

estimated the quantization error Δ again in the decoder side. For SVCCS, we compare our proposed GOP structure with that described in [33], i.e., the whole compressive sensed I frame is served as a reference frame. Assume that the DCT coefficients are received correctly. Fig. 4.7 shows the comparison. Since the reference frame in [33] cannot be recovered exactly the same as that in the encoder, the decoded video quality is thus degraded. If the measurements of the reference frame are lost, the degradation is more apparent. Therefore, as shown in Fig. 4.7, the proposed GOP structure of SVCCS is more desirable.

4.4.2 Comparison with MJPEG

We study the performance of wireless multicast with SVCCS. We compare the convolutional code protected MJPEG bitstream and SVCCS. 50 frames are encoded with MJPEG and SVCCS, respectively. Table 4.2 lists the average frame size and average PSNR of all frames. The SVCCS encoded bitstream is obtained from the joint source and channel coding approach, so we do not apply channel coding. For MJPEG coded bitstream, we apply convolutional code (code rate is 1/2). After channel coding, the doubled average frame size of MJPEG is even larger than that of SVCCS; thus, SVCCS can take less channel bandwidth.

We assume that the communication channel is AWGN and modulation scheme is DBPSK. Assume that the base layer of SVCCS can be correctly received. This assumption is acceptable as the base layer only counts for 2% of the coded bitstream,

	Avg. frame size (Kbits)	Avg. PSNR (dB)
SVCCS	36.99	32.74
MJPEG	20.93	31.85

Table 4.2: Measurement and bits allocation

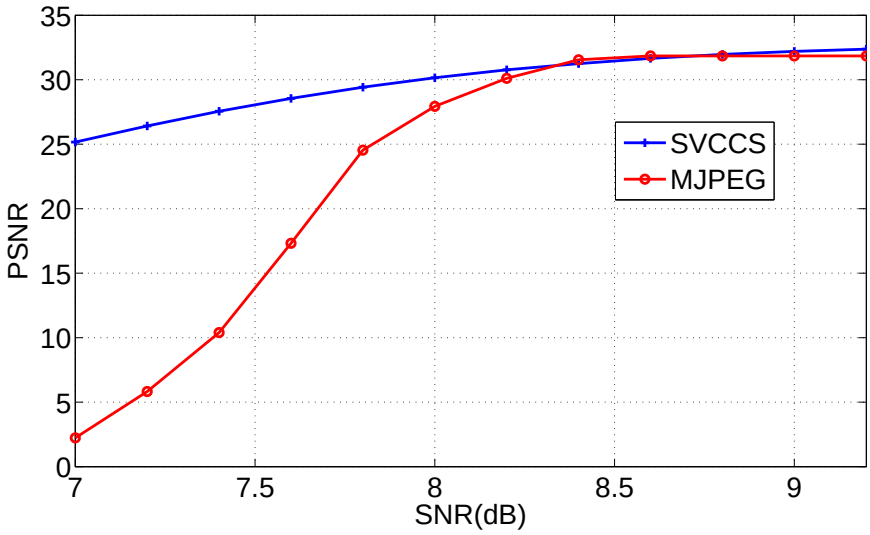
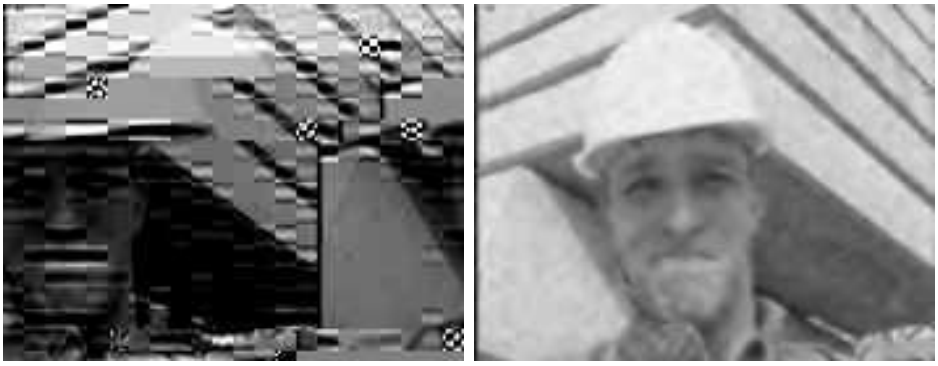


Figure 4.8: PSNR vs. SNR

which can be protected with very low cost. The average PSNR of the base layer is 21.45 dB.

Fig. 4.8 shows the advantage of SVCCS which is strongly adaptive to channel conditions. Suppose that the channel quality of users varies from 7 to 9.2 dB. In the simulation, we set the packet size of the SVCCS coded bitstream to 250 bytes. If one bit is transmitted in error, the whole packet and the contained measurements are lost. For MJPEG coded bitstream, we do not drop any packet even if there is any error, as dropping the whole packet makes the decoded quality even worse. We rely on the error resilience capability of JPEG decoder. Admittedly, this is not the best way of error concealment, but the figure is sufficient to show the better scalability of SVCCS than FEC protected MJPEG.

When we evaluate the video quality at a particular receiver whose SNR is 7.6 dB, some of the frames of MJPEG cannot be decoded and the decoded one shows worse quality than SVCCS, as shown in Fig. 4.9.



(a) MJPEG (b) SVCCS

Figure 4.9: SVCCS vs. MJPEG

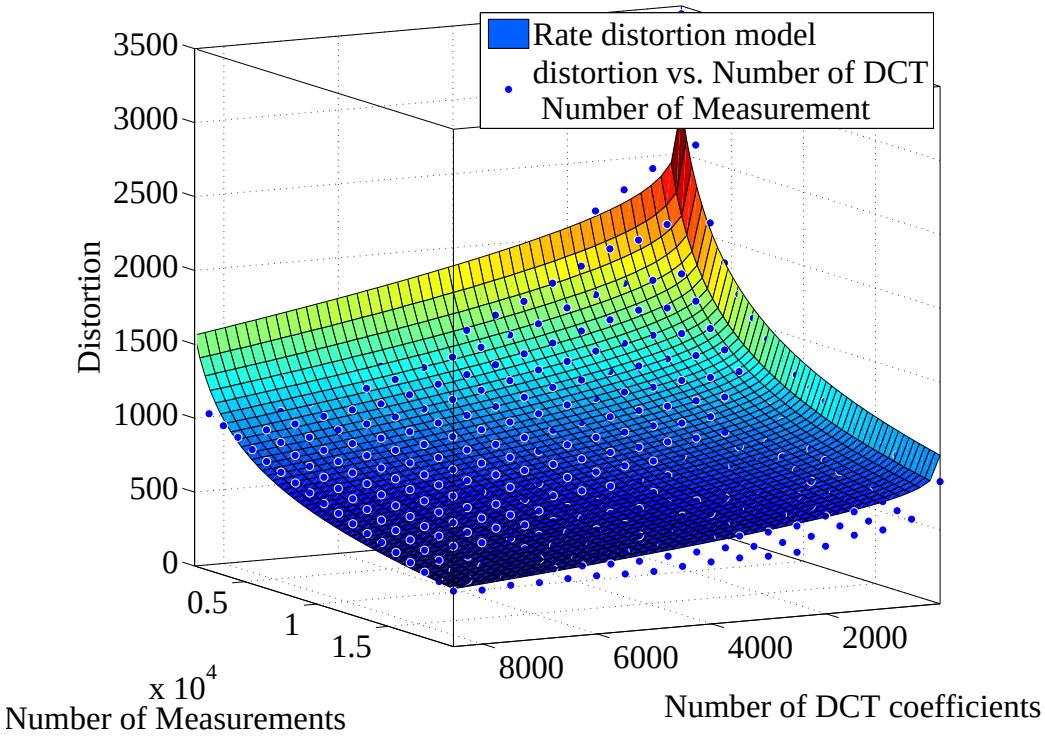


Figure 4.10: Distortion model

4.4.3 Rate Distortion Model

In this section, we evaluate the effectiveness of the proposed rate distortion model. We use the SVCCS to encode and decode images to obtain the rate-distortion data. Then the data is used to fit the model and obtain the model parameters. Fig. 4.10 shows the distortion model and the actual rate distortion data. The model fitness metric R-square value is 0.90, which demonstrates the accuracy of the proposed model. From the figure, for a fixed number of DCT coefficients, with an increasing number of measurements, the distortion decreases with a power law decaying tendency. We have fitted the model for other videos, the results are similar. Fig. 4.11 shows the rate model for DCT coefficients and measurements. The R-sqaure value for model r_u and r_m are 0.95 and 0.99, respectively. The rate model shows similar accuracy for other frames.

4.4.4 Multicast with SVCCS

In this subsection, we evaluate the performance of the rate allocation algorithm. We consider a base station multicasting real-time video to a group of 5 users with hetero-

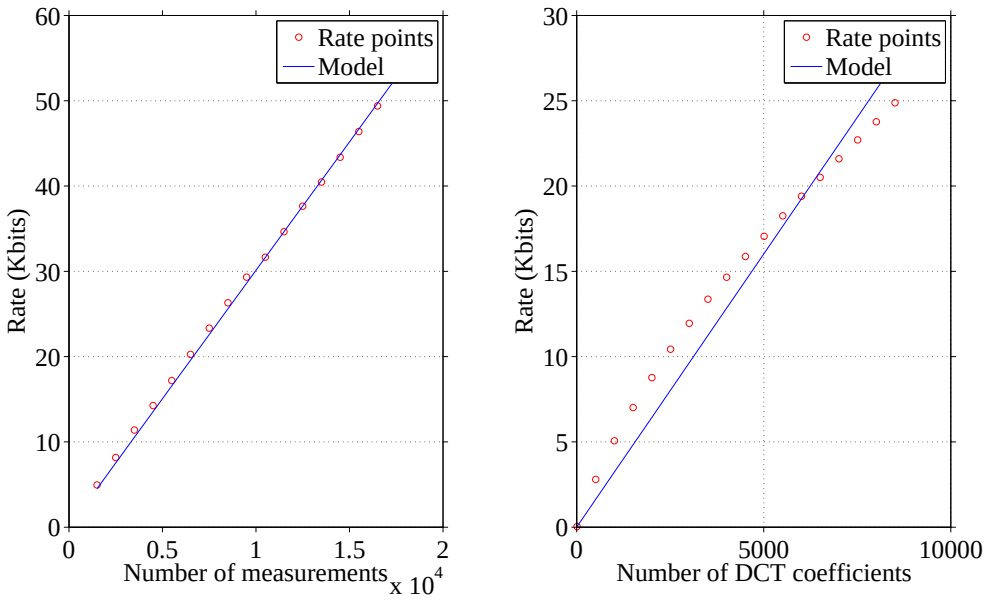


Figure 4.11: Rate model for DCT coefficients and measurements

Scenario	Avg. PSNR (dB)	Avg. bitrate (Kbps)	u	m
uni5	32.89	901.73	320	19672
bad5	26.84	903.14	310	19769
good5	36.13	897.67	340	19416

Table 4.3: Rate allocation results for different scenarios

geneous channel conditions. We assume that the wireless channel is AWGN channel. The SNR of the receivers for scenario uni5 are uniformly distributed in [10, 20] dB. The SNR of the receivers for scenario good5 and bad5 are distributed in [18, 20] and [10, 12] dB, respectively. The transmission bandwidth is set to 800 kHz. For the base layer transmission, the modulation and coding scheme are DBPSK and a convolutional code with code rate $r_c = 1/2$, respectively. The data rate r_b is 0.26 Mbps. For the enhancement layer transmission, the modulation sechme is 16QAM without channel coding. The data rate r_e is 1.04 Mbps. We use the same video as that in the previous section, and the frame rate F is 15 frames per second. For real-time video transmission, we adopted the GOP structure shown in Fig. 4.3(b). The number of bits of DCT coefficient b_u and measurements b_m are set to 9 and 4 bits, respectively. To reduce the complexity of obtaining the rate distortion model, we exploit the temporal similarity between adjacent frames. Specifically, we update the rate distortion model every 15 frames. It means that we obtain the rate distortion model of one frame and use this model as an approximation model for the following 14 frames.

Table 4.3 shows the rate allocation results for the 3 different multicast scenarios. From the table, we can see that the video quality degrades as the overall channel conditions get worse, since the PSNR value drops from 36.13 dB to 26.84 dB as the overall channel conditions degrade. We can also observe that when the overall channel conditions degrade, the rate allocation algorithm tends to allocate more measurements and reduce the number of DCT coefficients and the total bitrate is increased from 897.67 Kbps to 901.73 Kbps. The increased number of measurements can compensate the lost measurements. When the channel condition is good, the rate allocation algorithm reduces the number of measurements and increases the number of DCT coefficients to further improve the video quality. The total average bitrate for all the 3 scenarios are smaller than the total data rate, which satisfies the delay constraint. Fig. 4.12 shows the PSNR traces of 45 frames for the 5 receivers in scenario uni5. The average PSNR for the 5 receivers are 21.96, 34.20, 36.10, 36.1 and 36.10 dB, respectively. From the figure, we can see that the SVCCS bitstream is scalable in

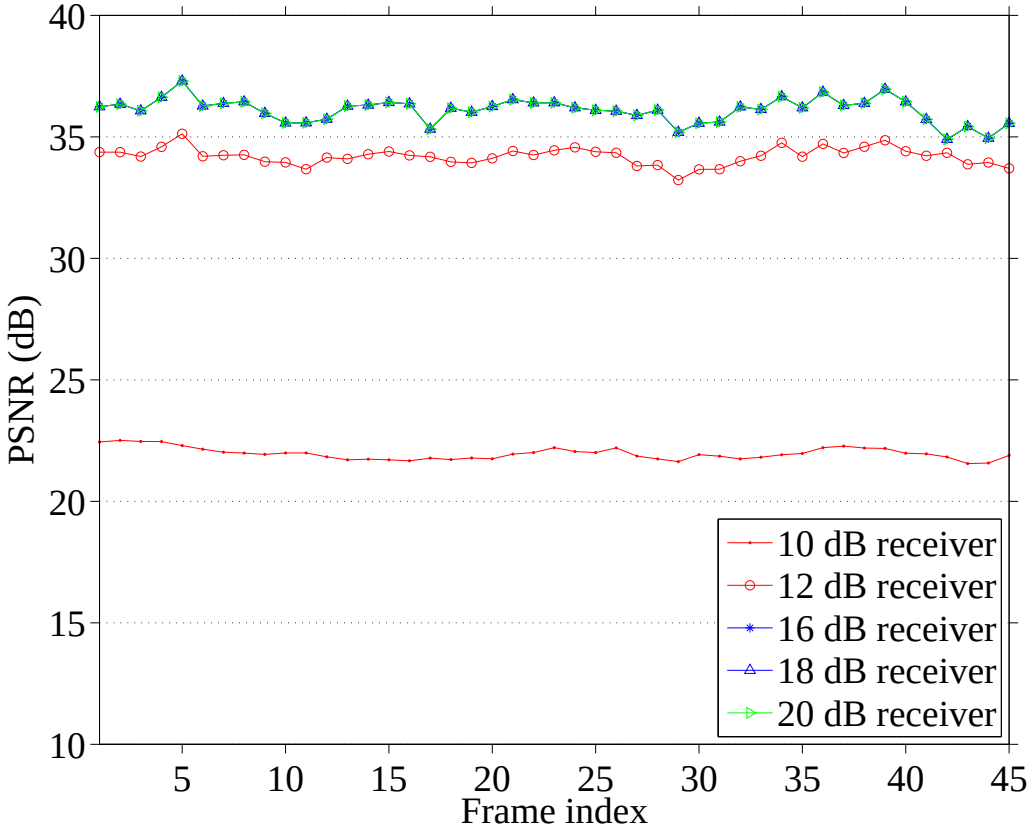


Figure 4.12: PSNR trace for the 5 users in the uni5 case

that the enhancement layers are useful for all the receivers. For each receiver, the base layer is received error-free and all the received enhancement measurements can be used in the decoding process to improve the received video quality.

4.5 Summary

In this chapter, we have proposed a low-complex, scalable video coding architecture based on compressive sensing for wireless unicast and multicast transmissions. As SVCCS can achieve good scalability, error resilience and coding efficiency, it is more effective and efficient than the traditional solution to support wireless videocast. We have built a rate distortion model for SVCCS encoded bitstream. Based on this model, we have investigated the rate allocation problem for real-time video multicast and formulated an optimization problem.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

In this dissertation, we have studied three problems of video streaming over wireless channels. The following outlines the contributions achieved.

- In Chapter 2, we have formulated the rate adaptation problem as an MDP and used dynamic programming to obtain the optimal streaming policy. To reduce the complexity of the optimal streaming policy, we have proposed an online algorithm which learns the bandwidth statistics and determines the request decisions for the future. The trade-off between the average video quality and playback smoothness can be made by adjusting the parameter in the reward function. Experimental results have shown that the proposed solution is feasible and can substantially outperform the existing one.

- In Chapter 3, we have proposed a simple and effective rate-distortion model for compressive sensed video. Using this model, we can accurately estimate the video distortion given the number of measurements used in signal recovery. We have formulated a deterministic optimization problem as a bench mark, and a more practical stochastic optimization problem for transmitting compressive sensing video over wireless channels. A supplementary stochastic optimization problem has been formulated to consider distortion and power consumption with different priority. We have further proposed an incremental-V algorithm, which helps to choose the value of V while keeping the average transmission delay in a reasonable range. The simulation results have shown the effectiveness of the proposed transmission control algorithms.

- In Chapter 4, we have proposed a low-complex, scalable video coding architecture based on compressive sensing for wireless unicast and multicast transmissions. As SVCCS can achieve good scalability, error resilience and coding efficiency, it is more effective and efficient than the traditional solution to support wireless videocast. We have built a rate distortion model for SVCCS encoded bitstream. Based on this model, we also investigated the rate allocation problem for real-time video multicast and formulated an optimization problem.

5.2 Future Work

Some of the future work are listed as follows.

- In Chapter 2, we have studied the rate adaptation problem for SVC streaming in a DASH system. There are several issues worth further investigation. For instance, to fully utilize the layered feature of SVC, we may consider other possible actions, such as to "upgrade" multiple previously received segments when possible. We may use moving window to better estimate the bandwidth state transition probability and capture the non-stationary behavior of varying bandwidth. We may also keep a history of requested segment layer index and make decisions based on these records to further improve the smoothness of video playback.

- In Chapter 3, we have investigated the transmission control problem for compressive sensing video over wireless channel. When solving the stochastic optimization problems, we assume that the channel fading coefficient is known for the current slot. In practice, channel condition can be estimated through feedback from the receiver. The impact of the delayed and inaccurate channel estimation is left for future exploration. For simplicity, we used channel capacity to calculate the transmission rate based on channel condition and power level. In reality, the transmission rate also depends on the modulation and coding schemes adopted in the system. How to extend this work to consider discrete link rates requires further investigation.

- In Chapter 4, we proposed a novel coding structure based on compressive sensing and rate allocation algorithm for wireless multicast. CS based video coding is overall a promising direction with many other open issues that worth further

investigation. When building the rate distortion model, we have fixed the number of bits for DCT coefficients and measurements. In the future work, we can explore the effect of different quantization levels of DCT coefficients and measurements on rate and distortion and extend the rate distortion model to other GOP structures; how to reduce the decoding complexity and further improve the coding efficiency is another important further research issue.

Bibliography

[1] Big Buck Bunny. `http://www.bigbuckbunny.org`.

[2] S. Akhshabi, A. C. Begen, and C. Dovrolis. An experimental evaluation of rate-adaptation algorithms in adaptive streaming over HTTP. In *ACM MMSys'11*, pages 157–168, San Jose, CA, USA, 2011.

[3] Andreas Antoniou and Wu-Sheng Lu. *Practical optimization: algorithms and engineering applications*. Springer, 2007.

[4] S. Becker, J. Bobin, and E.J. Candes. NESTA: A fast and accurate first-order method for sparse recovery. *SIAM J. on Imaging Sciences*, 4(1):1–39, 2009.

[5] A. Begen, T. Akgul, and M. Baugher. Watching video over the web: Part 1: Streaming protocols. *IEEE Internet Computing*, 15(2):54 –63, March-April 2011.

[6] M. Blestel and M. Raulet. Open SVC decoder: a flexible SVC library. ACM MM'10, pages 1463–1466, New York, NY, USA, 2010.

[7] L. Cai, S. Xiang, Y. Luo, and J. Pan. Scalable modulation for video transmission in wireless networks. *IEEE Trans. on Veh. Tech.*, 60(9):4314 –4323, Nov. 2011.

[8] E.J. Candes and T. Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, Dec. 2006.

[9] E.J. Candes, M.B. Wakin, and S.P. Boyd. Enhancing sparsity by reweighted l 1 minimization. *Journal of Fourier Analysis and Applications*, 14(5):877–905, 2008.

[10] Wai Lam Chan, Kriti Charan, Dharmpal Takhar, Kevin F Kelly, Richard G Baraniuk, and Daniel M Mittleman. A single-pixel terahertz imaging system based on compressed sensing. *Applied Physics Letters*, 93(12):121105–121105, 2008.

[11] Z. Charbiwala, S. Chakraborty, S. Zahedi, Y. Kim, M.B. Srivastava, T. He, and C. Bisdikian. Compressive oversampling for robust data transmission in sensor networks. In *IEEE INFOCOM'10*, Mar. 2010.

[12] Cisco. Cisco Visual Networking Index: Forecast and Methodology, 2010-2015.

[13] Cyril Concolato, Jean Le Feuvre, and Romain Bouqueau. Usages of dash for rich media services. In *Proceedings of the second annual ACM conference on Multimedia systems*, MMSys '11, pages 265–270, New York, NY, USA, 2011. ACM.

[14] L. De Cicco, S. Mascolo, and V. Palmisano. Feedback control for adaptive live video streaming. In *ACM MMSys'11*, pages 145–156, San Jose, CA, USA, 2011.

[15] T.T. Do, T.D. Tran, and Lu Gan. Fast compressive sampling with structurally random matrices. *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 3369–3372, Mar. 2008.

[16] D.L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, April 2006.

[17] M.F. Duarte, M.A. Davenport, D. Takhar, J.N. Laska, Ting Sun, K.F. Kelly, and R.G. Baraniuk. Single-pixel imaging via compressive sampling. *IEEE Signal Processing Magazine*, 25(2):83–91, Mar. 2008.

[18] H. Everett III. Generalized lagrange multiplier method for solving problems of optimum allocation of resources. *Operations Research*, pages 399–417, 1963.

[19] V.K. Goyal, A.K. Fletcher, and S. Rangan. Compressive sampling and lossy compression. *Signal Processing Magazine, IEEE*, 25(2):48–56, Mar. 2008.

[20] Frank Hartung, Sinan Kesici, and Daniel Catrein. Drm protected dynamic adaptive http streaming. In *Proceedings of the second annual ACM conference on Multimedia systems*, MMSys '11, pages 277–282, New York, NY, USA, 2011. ACM.

[21] M. Kobayashi, H. Nakayama, N. Ansari, and N. Kato. Robust and efficient stream delivery for application layer multicasting in heterogeneous networks. *IEEE Transactions on Multimedia*, 11(1):166–176, Jan. 2009.

[22] J.N. Laska, P. Boufounos, M.A. Davenport, and R.G. Baraniuk. Democracy in action: Quantization, saturation, and compressive sensing. *preprint*, 2009.

[23] Stefan Lederer, Christopher Müller, and Christian Timmerer. Dynamic adaptive streaming over http dataset. In *Proceedings of the 3rd Multimedia Systems Conference*, MMSys '12, pages 89–94, New York, NY, USA, 2012. ACM.

[24] C. Liu, I. Bouazizi, and M. Gabbouj. Rate adaptation for adaptive HTTP streaming. In *ACM MMSys'11*, pages 169–174, San Jose, CA, USA, 2011.

[25] Michael Lustig, David Donoho, and John M Pauly. Sparse mri: The application of compressed sensing for rapid mr imaging. *Magnetic Resonance in Medicine*, 58(6):1182–1195, 2007.

[26] Ricky K. P. Mok, Xiapu Luo, Edmond W. W. Chan, and Rocky K. C. Chang. Qdash: a qoe-aware dash system. In *Proceedings of the 3rd Multimedia Systems Conference*, MMSys '12, pages 11–22, New York, NY, USA, 2012. ACM.

[27] Christopher Müller and Christian Timmerer. A test-bed for the dynamic adaptive streaming over http featuring session mobility. In *Proceedings of the second annual ACM conference on Multimedia systems*, MMSys '11, pages 271–276, New York, NY, USA, 2011. ACM.

[28] M.J. Neely. Stochastic network optimization with application to communication and queueing systems. *Synthesis Lectures on Communication Networks*, 3(1):1–211, 2010.

[29] M.J. Neely and A. Sharma. Dynamic data compression with distortion constraints for wireless transmission over a fading channel. *Arxiv preprint arXiv:0807.3768*, 2008.

[30] S. Nelakuditi, R. Harinath, and E. Kusmierek et al. Providing smoother quality layered video stream. In *ACM NOSSDAV'00*, Jun. 2000.

[31] Jae Young Park and Michael B Wakin. A multiscale framework for compressive sensing of video. In *Picture Coding Symposium, 2009. PCS 2009*, pages 1–4. IEEE, 2009.

[32] L. Plissonneau and E. Biersack. A longitudinal view of http video streaming performance. In *ACM MMSys'12*, pages 203–214, Chapel hill, NC, USA, 2012.

[33] S. Pudlewski, T. Melodia, and A. Prasanna. C-dmrc: Compressive distortion-minimizing rate control for wireless multimedia sensor networks. In *Sensor Mesh and Ad Hoc Communications and Networks (SECON), 2010 7th Annual IEEE Communications Society Conference on*, pages 1–9, Jun. 2010.

[34] S. Pudlewski, A. Prasanna, and T. Melodia. Compressed-sensing-enabled video streaming for wireless multimedia sensor networks. *IEEE Transactions on Mobile Computing*, 11(6):1060 –1072, June 2012.

[35] J. Reichel, H. Schwarz, and M. Wien. Joint scalable video model 11 (JSVM 11). *Joint Video Team, Doc. JVT- X*, 2007.

[36] L.I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1-4):259–268, 1992.

[37] Y. Sánchez, T. Schierl, C. Hellge, T. Wiegand, D. Hong, D. De Vleeschauwer, W. Van Leekwijck, and Y. Lelouedec. iDASH: improved dynamic adaptive streaming over HTTP using scalable video coding. In *ACM MMSys'11*, pages 257–264, San Jose, CA, USA, 2011.

[38] T. Schierl, Y. Sanchez de la Fuente, R. Globisch, C. Hellge, and T. Wiegand. Priority-based media delivery using SVC with RTP and HTTP streaming. *Multimedia Tools and Applications*, 55:227–246, 2011.

[39] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the H. 264/AVC standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(9):1103–1120, 2007.

[40] Vladimir Stankovic, Lina Stankovic, and Samuel Cheng. Compressive video sampling. In *In Proc. of the European Signal Processing Conf.(EUSIPCO)*, pages 2–6, 2008.

[41] T. Stockhammer. Dynamic adaptive streaming over HTTP: standards and design principles. In *ACM MMSys'11*, pages 133–144, San Jose, CA, USA, 2011.

[42] K. Stuhlmuller, N. Farber, M. Link, and B. Girod. Analysis of video transmission over lossy channels. *IEEE Journal on Selected Areas in Communications*, 18(6):1012–1032, 2000.

[43] G.J. Sullivan and T. Wiegand. Rate-distortion optimization for video compression. *IEEE Signal Processing Magazine*, 15(6):74 –90, nov 1998.

[44] R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. Cambridge Univ Press, 1998.

[45] Test Video. `http://media.xiph.org/video/derf/`.

[46] Michael B. Wakin, Jason N. Laska, Marco F. Duarte, Dror Baron, Shriram Sarvotham, Dharmpal Takhar, Kevin F. Kelly, and Richard G. Baraniuk. Compressive imaging for video representation and coding. In *Proceedings of Picture Coding Symposium*, 2006.

[47] H. S. Wang and N. Moayeri. Finite-state Markov channel — a useful model for radio communication channels. *IEEE Trans on Veh. Tech.*, 44(1):163–171, 1995.

[48] S. Xiang and L. Cai. Scalable video coding with compressive sensing for wireless videocast. In *IEEE ICC*, June 2011.

[49] S. Xiang and L. Cai. Transmission control for compressive sensing over wireless channel. *IEEE Trans. on Wireless Communications*, Dec. 2012. to apear.

[50] S. Xiang, L. Cai, and J. Pan. Dynamic rate adaptation for adaptive video streaming in wireless networks. *IEEE Trans. on Multimedia*. submitted.

[51] S. Xiang, L. Cai, and J. Pan. Adaptive scalable video streaming in wireless networks. In *ACM MMSys '12*, pages 167–172, 2012.

[52] J. Xu, X. Shen, J.W. Mark, and J. Cai. Adaptive transmission of multi-layered video over wireless fading channels. *IEEE Trans. on Wireless Communications*, 6(6):2305 –2314, June 2007.

[53] R. Zhang and L. Cai. A packet-level model for UWB channel with people shadowing process based on angular spectrum analysis. *IEEE Trans. on Wireless Comm.*, 8(8):4048–55, Aug. 2009.

[54] Yifu Zhang, Shunliang Mei, Quqing Chen, and Zhibo Chen. A novel image/video coding method based on compressed sensing theory. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 1361–1364. IEEE, 2008.

[55] X. Zhu, R. Pan, M.S. Prabhu, and N. Dukkipati et al. Layered internet video adaptation (liva): Network-assisted bandwidth sharing and transient loss protection for video streaming. *IEEE Trans. on Multimedia*, 13(4):720 –732, Aug. 2011.

Appendix A

Publication List

1. S. Xiang, L. Cai and J. Pan. Dynamic Rate Adaptation for Adaptive Video Streaming in Wireless Networks. Submitted to *IEEE Trans. on Multimedia*, under revision.

2. S. Xiang and L. Cai. Transmission Control for Compressive Sensing Video over Wireless Channel. *IEEE Trans. on Wireless Communications*, to appear.

3. M. Xing, S. Xiang, and L. Cai. Rate adaptation for video streaming over multiple wireless access networks. In *IEEE Globecom'12*, Anaheim, CA, USA, Dec. 2012.

4. S. Xiang, L. Cai, and J. Pan. Adaptive Scalable Video Streaming in Wireless Networks. In *ACM MMSys'12*, Chapel Hill, NC, USA.

5. L. Cai, S. Xiang, Y. Luo and J. Pan. Scalable Modulation for Video Transmission in Wireless Networks. *IEEE Trans. on Vehicular Technology*, vol. 60, no. 9, pp. 4314-4323, Nov. 2011.

6. S. Xiang and L. Cai. Scalable Video Coding with Compressive Sensing for Wireless Videocast. In *IEEE ICC'11*, Kyoto, Japan, Jun. 2011.

7. S. Xiang and L. Cai. Distortion Analysis of Wyner-Ziv Distributed Video Coding. In *IEEE Globecom'10*, Dec. 2010, Miami, USA.

8. L. Cai, Y. Luo, S. Xiang, and J. Pan. Scalable modulation for scalable wireless videocast. In *IEEE Infocom'10, mini symposium*, Mar. 2010, San Diego, CA, USA.