ACM/SPIE Multimedia Computing and Networking 2005, January 2005.

An experimental analysis of DCT-based approaches for fine-grain multi-resolution video

Jie Huang[*a], Wu-chi Feng[a], Jonathan Walpole[a], and Wilfried Jouve[b]

[a]Dept. of Computer Science, Portland State University, OR, USA 97207
[b]ENSEIRB, Talence, France 33400

ABSTRACT

This paper presents the architectural trade-offs to support *fine-grain* multi-resolution video over a *wide range* of resolutions. In the future, video streaming systems will have to support video adaptation over an extremely large range of display requirements (e.g. 90x60 to 1920x1080). While several techniques have been proposed for multi-resolution video adaptation, which is also known as spatial scalability, they have focused mainly on limited spatial resolutions. In this paper, we examine the ability of current techniques to support wide-range spatial scalability. Based upon experiments with real video, we propose an architecture that can support wide-range adaptation more effectively. Our results indicate that multiple encodings with limited spatial adaptation from each encoding provides the best trade-off between efficient coding and the ability to adapt the stream to various resolutions.

Keywords: Multi-resolution video, DCT

1. INTRODUCTION

With the development of a vast number of heterogeneous devices able to display multimedia data such as cell phones, palm-based computers, laptops, and plasma displays, the ability to seamlessly deliver high quality video tailored to a specific device's display characteristics will become increasingly important. To support streaming to such devices, fine-grain multi-resolution video can be used to adapt to the underlying network constraints as well as the client's computational and display constraints. Meanwhile, coupled with this divergence of device characteristics, the ability to generate extremely large video images has become possible. These include high-definition TV signals (1920x1080) and a video stream stitched together from multiple camera views into a single consolidated view. The result of these two trends means that multi-resolution approaches will have to adapt video to a potentially large number of resolutions over a wide range and provide for region-of-interest cropping at the same time.

In order to support video adaptation over a large number of resolutions, a spectrum of solutions is possible. At the one end of the spectrum, a single compressed high-resolution stream can be used to derive all required resolutions. The stream can be compressed into a multi-resolution or spatially scalable format. To adapt the stream down to smaller resolutions the server can send only partial information by dropping the data that is not needed. The stream can also be compressed into a non-scalable format. The server can transcode the stream to create the desired lower resolution. The main characteristic is that a full encoding of the stream is done once and all other resolutions are created from that stream. At the other end of the spectrum, it is conceivable to directly encode the uncompressed video into each resolution that is required. While requiring a large amount of computation to compress and a lot of space to store, there is little overhead in streaming these streams to clients. In between these approaches are a range of solutions that can be used to support multi-resolution video. How far spatially scalable encodings or transcodings can be pushed, however, is not well understood.

In this paper, we present an in-depth analysis of the architectural trade-offs in providing fine-grain multi-resolution video. We compare two extreme approaches for supporting multi-resolution video adaptation: the one-stream-for-all-resolutions approach and the one-stream-per-resolution approach. We study the coding efficiency for a single resolution, the coding efficiency for all resolutions, and storage costs for these two approaches. We believe that there are hybrid systems between these two ends with multiple streams coupled with spatial adaptation that can provide good trade-offs under particular situations. Our work will provide guidelines to find such configurations.

[*] jiehuang@cs.pdx.edu

This paper is organized as follows. Section 2 presents related work in scalable video encoding and adaptive video streaming. In Section 3, we discuss the architectures available and the common mechanisms for DCT-based encoding and transcoding techniques for multi-resolution video. Our experiments and analysis are based on these common mechanisms not on any particular technique. We describe the experiment set up and analyze the results in Section 4. We summarize the paper and provide directions for future work in Section 5.

Contributions of this work: The main contribution of this paper is an understanding of effective architectures for fine-grain multi-resolution video from available techniques. We answers questions like: how many resolutions can a single stream support? How can coding parameters of each layer for scalable encoding be determined? For multiple streams, how can the resolutions for each stream be determined and how should video bit-rates and resolution ranges be allocated among streams? It is not our goal to improve video encoding or transcoding techniques nor to study the performance of a particular algorithm. We construct our experiments based on a non-scalable MPEG-1 encoder; the experiment and our analysis are based on general DCT-based video compression algorithms and are applicable to MPEG video and H.26x video. Many discussions are useful for wavelet-based video too.

2. RELATED WORK

Video encoding schemes with spatial scalability have been well studied[1,2,4,5,8,17] and fast transcoding schemes for spatial size conversion have been proposed to support multiple resolutions in one video stream[13,10,14,20]. However, these techniques have focused on downscaling each dimension by a factor of 2 or 4. While many of these techniques suggest that they can be generalized to larger scales, it is unclear how such approaches work in practice when the scaling factor needed is beyond 4. Part of our work is to draw common mechanisms for spatial scalability and transcoding from these techniques and to apply these common mechanisms to a large number of scaling factors to find out their working ranges.

Our work studies the adaptation range and granularity of multi-resolution video and architectures to support it. To stream the multi-resolution video effectively, adaptive streaming mechanisms are needed including stream-switching schemes to switch among streams[18], packet scheduling algorithms to send out packets at different layers in a scalable stream[10,1112,16], and feedback control mechanisms to change transcoding parameters[3].

3. SUPPORTING FINE-GRAIN, WIDE-SCALE, MULTI-RESOLUTION VIDEO

Despite rapid progress in storage capacity and transmission bandwidth, video compression is still a key technology for video applications simply because of the high resource requirement of raw video and the large compression ratios achievable. Currently, most compression techniques are either DCT-based or wavelet-based. Wavelet-based algorithms perform a wavelet transform on the entire image, which results in a hierarchical representation of an image. In the hierarchy, each layer represents a frequency band, which corresponds to a resolution. Thus, wavelet-based compression supports multi-resolution video inherently. One of the main drawbacks of wavelet compression is its limited ability to perform region-of-interest adaptation due to the wavelet transform. While somewhat challenging to support multi-resolution video for DCT-based compression, its adoption into standards such as the MPEG series and the H.26x series coupled with the ability to support region-of-interest adaptation makes it an interesting research topic.

There are a variety of ways to provide wide-range fine-grained video adaptation. These include techniques such as scalable encoding, transcoding, multiple-encodings, or mixtures of these techniques. In order to provide a clear discussion, we structure our presentation of algorithms based on the number of "full" encodings that are used to represent the video on the server. As mentioned in the introduction, there are two extremes in this spectrum. Scaling or transcoding the video from a *single* encoding can be done, or encoding as many streams as required by the varying display requirements of the clients. In between, there are also several hybrid approaches. In the remainder of this section, we describe these approaches including their advantages and disadvantages for wide-range fine-grained adaptation.

3.1. Single Encoding Video for All Resolutions

One approach to provide wide-range fine-grained video adaptation is to *encode the highest resolution once* and generate all other resolutions from the one encoding. Under this approach, resolution adaptation can be accomplished using one of two ways. First, more time can be spent encoding the stream such that it is more amenable to resolution downscaling

(e.g. scalable-encoding the stream with a base layer and enhancement layers). In this way more time is spent on the encoding but providing smaller resolutions is easier because dropping the layers can provide smaller resolutions. We note here that the use of the term *enhancement layer* usually means higher SNR quality. For our purposes, we use enhancement layer to refer to layers that provide higher resolutions. The second way to provide resolution adaptation is to compress the stream with minimal extra information (e.g. enhancement layers) and spend more time adapting the video stream to a different resolution as needed through transcoding. This can involve (i) a full transcoding where the stream is more or less decompressed and recompressed, (ii) a limited transcoding where the DCT coefficients are mathematically altered in the compressed domain, or (iii) a simple transcoding by dropping AC coefficients. While making compression relatively easy, the second way has higher computational overhead at streaming time for a smaller resolution stream.

The first two transcoding methods extract low-resolution information in the pixel domain and the third method extracts low-resolution information in the DCT domain. The difference is whether the DCT coefficients are for pixels of the reduced resolution or for pixels of the full resolution respectively. In the latter case, down-scaling is done after decoding at the receiver side.

Extracting information in the pixel domain is easy for the full transcoding. The compressed video is first decompressed then downsized in the pixel domain and re-encoded. The limited transcoding uses fast algorithms that generate DCT-coefficients for low-resolution pixels from DCT-coefficients for full-resolution pixels through matrix multiplication[5, 9, 19] without generating low-resolution images. These algorithms are fast since no DCT or IDCT is involved while the results are equivalent to DCT transforms on low-resolution pixels. Hence, the coding efficiency of the limited transcoding should be close to the full transcoding and we will take the full transcoding approach for its simplicity.

Extracting low-resolution information in the DCT domain is usually accomplished by dropping high-frequency DCT coefficients that are for full-resolution pixels. We will ignore drift errors caused by dropping coefficients.

Besides DCT coefficients, we also need to deal with motion vectors for different resolutions. For pixel domain downscaling, we use the "open-loop" approach as described by Dugad and Ahuja[6] and do motion estimation for each resolution since we are re-encoding anyway. For DCT-coefficient dropping, we use the motion vectors for the original full resolution video since the DCT is always done based on the full resolution. This difference should not influence our comparison between the two architectures.

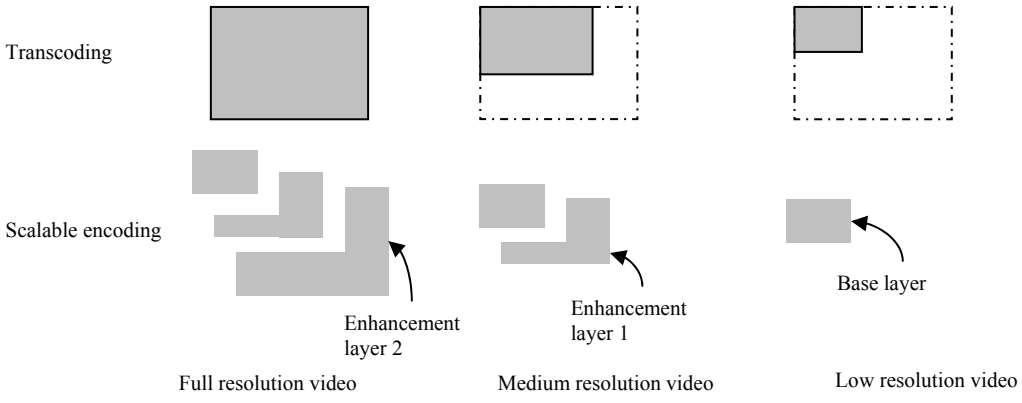


Figure 1 Transcoding vs. Scalable Encoding. This figure shows that the basic mechanisms behind scalable encoding and transcoding are the same. If there is a way for transcoding techniques to extract information from the full resolution video to form a lower resolution video, the same extraction algorithm can be used for scalable encoding. For transcoding, the extracted information or the original information is encoded. For scalable encoding, the extracted information and the left-over information are encoded.

We believe that the common mechanisms for scalable encoding and transcoding are the same, as shown in Figure 1. If a transcoding technique extracts information from the full resolution video to form a lower resolution video, the same extraction algorithm can be used for scalable encoding. For transcoding, the extracted information or the original information is encoded. For scalable encoding, the extracted information and the left-over information are encoded. Transcoding has better coding efficiency for a single resolution because it does not have the overhead of multiple layers.

Scalable encoding has better overall coding efficiency for multiple resolutions when the overhead for multiple layers does not exceed the savings from encoding left-over information. We will evaluate the overhead for multiple layers and the savings from differential encoding. We then evaluate only transcoding for the one-stream approach.

3.2. One Encoding Per Resolution

Another simple approach to provide multi-resolution video is to encode as many streams as there are resolutions required. In this way, each encoding has been optimized for a particular display (or at least one of similar resolution). The key advantage of such an approach is that the stream is optimized for each resolution. The main drawback, however, is the computational overhead involved in creating potentially many streams. For stored video systems, this is further complicated by the fact that the resolutions required may not be known *a priori*. To understand the potential limitation of *n encodings for n display devices*, we recently conducted a survey of common devices able to display multimedia data. A partial list of the results is shown in Table 1 where some sizes are not listed because they are close to a size in the table. As shown by Table 1, there are at least 34 different display types available today. Undoubtedly, this will continue to grow in the future with smaller and higher-resolution devices.

For this paper, we will use the one encoding per resolution approach in the experiments to provide a baseline for how well one could have done for a particular resolution (PSNR as well as bandwidth requirements).

Cell Phones		PDAs		Laptops		Top-of-the-line Monitors
96x36	240x160	160x160	480x160	640x480	1280x1024	1680x1050
96x65	320x208	160x240	480x320	800x600	1400x1050	1920x1200
101x80	320x240	240x100	640x240	1024x480	1440x900	2048x768
128x128	640x200	240x200	800x480	1024x768	1600x1200	2048x1536
160x128	640x320	320x240	800x600	1280x800		2560x1600
208x176		320x320				

Table 1 Available display sizes

3.3. Hybrid Approach

In between the approaches described in the previous two subsections, one can encode several candidate resolutions that cover a class of displays and then create all other resolution streams from the encoded streams. In effect, this combines the two approaches. The goal of such a system would be to provide several candidate starting streams that are relatively distant in resolution and then adapt the stream from there. The reason we believe this is important can be illustrated with the following example. Suppose we have a one encoding video stream of resolution 720x480, a standard DVD resolution. Further suppose we want to display this on a device that is 90x60 in resolution. Using the one encoding approach, the only values that would be required for the 90x60 display would be the DC values within the stream. Representing a low resolution image by DC values might not be an efficient algorithm for resolution-downscaling and sending an MPEG stream with only DC values is inefficient; so it will not make good use of the bandwidth, particularly over wireless medium to which the small devices are usually connected.

3.4. Approach Summary

There are clearly a number of techniques and approaches that can be employed to support multi-resolution video. The question that remains unanswered, however, is how should one structure the video to support such adaptation to a large number of devices. Furthermore, will this structure support region-of-interest cropping that may ultimately be required for manipulating large images on devices with small displays? Intuitively, we believe that supporting such video will fall into the hybrid approach category because it allows the efficient trade-off between computation for encoding and computation for display-dependent decoding. How far such encodings can be pushed, however, is a question left unanswered.

In the remainder of this paper, we will present a number of experiments to highlight the various trade-offs one can make in supporting multi-resolution video. We will show the limitations of extending some of the traditional techniques for resolution adaptation to a large range of resolutions. At the end we will have a framework in which to think about multi-resolution video adaptation and how systems should structure such video encodings.

4. EXPERIMENTS AND ANALYSIS

4.1. Experiment setup

We used 300 frames from the movie *The Italian Job* and 300 frames from the sitcom *Friends* as our test sequences. The sequence from *The Italian Job* is motion-intensive; the sequence from *Friends* is of slow motion with a relatively fixed background. The results for the two sequences are similar and we present more analysis for the *The Italian Job* sequence.

We extracted the content of video sequences from DVDs. The original resolution of the DVD content is 720x480. We decompressed the 720x480 sequence into a sequence of uncompressed images. To create smaller reference frames, we downsized the image sequence to 540x360, 360x240, 270x180, 180x120, and 90x60. The downsized image sequences are deemed as the reference for that resolution in our experiments. These resolutions can be used for cell phones, PDAs, laptops, and desktops. These resolutions also provide a wide range of bit-rates that could fit varying network conditions.

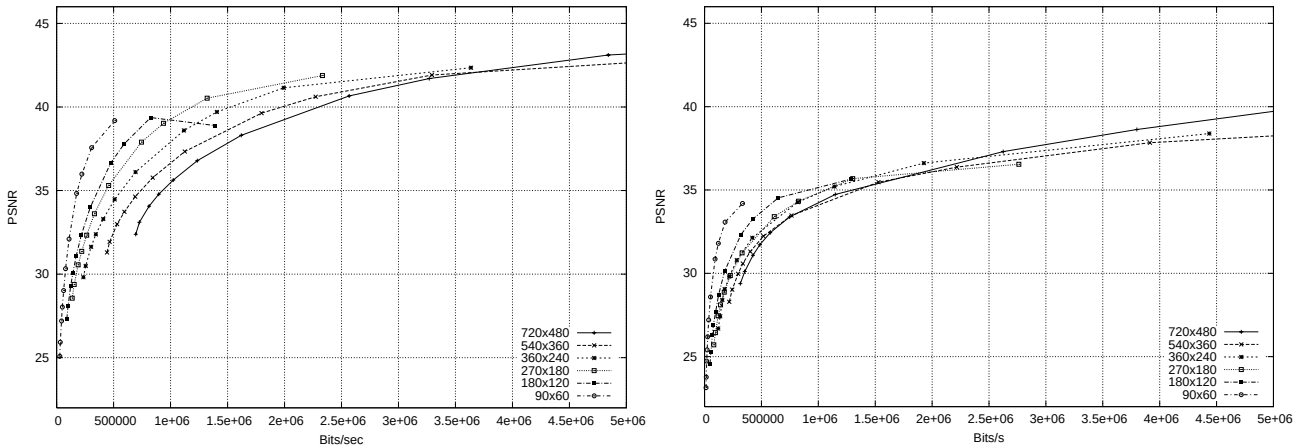
Because the creation of the smaller reference frames may depend upon the actual downscaling tool used, we decided to use two different tools for converting spatial resolutions. The first is *pnmscale* from the Netpbm image manipulation libraries; the other is *mogrify* from ImageMagick. We used an option in *pnmscale* that does pixel mixing; that is, the color of a pixel in the target image is a weighed average of the colors of neighbor pixels in an original image. We used re-sampling-based scaling in *mogrify* and we chose the filter function *sinc*, which is the Fourier transform of a low pass filter that throws away high frequency signals in the original image. Using both of these programs, we calculated two sets of reference images for the sequence. This ensures that the results were not biased by the downsampling algorithm. The results turned out to be very similar for these two scaling approaches. In order to save space, we present only those obtained by using *mogrify* in this paper.

To compress the video streams into their respective formats, we used the MPEG-1 codec in *ffmpeg* [7] in our experiments.

4.2. Multi-resolution adaptation

In the first set of experiments, we are interested in determining the range of bit-rates at which a stream can be encoded. Figure 2 shows the bit-rates of the video encoded at different resolutions with different quantization scales. We have recorded the average PSNR for each quantization scale and resolution. For the experiments, the video is encoded with the GOP size 12, two B frames between any I/P frames, and quantization scales (31, 28, 24, 20, 16, 12, 8, 6, 4, 3, 2, 1).

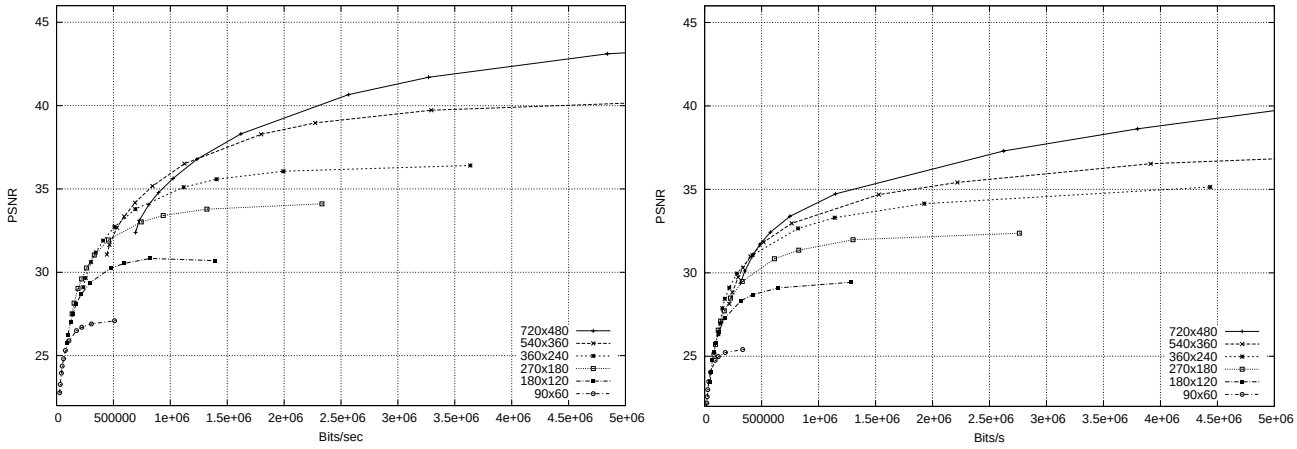
From Figure 2, we can see that multi-resolution video offers many bit-rate options for both sequences especially at low bit rates. SNR adaptation and multi-resolution video combined provide a large adaptation space. For some target rates, there is more than one option. For example, in the *The Italian Job* sequence, if the available bandwidth is 760KB, video at several resolutions are streamable with PSNR from 39.2db for 90x60 to 32.7db for 540x480.



(a) The Italian Job (b) Friends

Figure 2 Streams at different resolutions with different quantiations. This figure shows average bit-rates and average PSNRs of video streams at different resolutions. Each line represents a resolution and dots on that line represent different quantization scales. Each dot in ths figure represents a feasible bit rate.

To understand what this means from a viewing perspective, we need to compare the various resolutions using the same display requirement. To accomplish this, we took the various quantization and resolution videos and upsampled them to 720x480 and PSNRs are calculated based on the original 720x480 images (in contrast to comparing it to the actual resolution it was compressed in) as shown in Figure 3. If the bandwidth is relatively small, it might be better to transmit a smaller resolution stream and upsample it to 720x480. As an example of this, at approximately 1 megabit per second in Figure 3 (a), we see that transmitting and upsampling the 540x360 image for the 720x480 display achieves a higher PSNR. Thus, sometimes reducing resolution is a more effective way to use bandwidth than lowering the SNR level.



(a) The Italian Job (b) Friends

Figure 3 Streams at different resolutions then upsampled to 720x480. This figure shows two different ways to tailor the 720x480 video to reduce the video bit-rate: changing the quantization scale or changing the resolution. Each line represents a resolution and dots on that line represent different quantization scales.

4.3. One stream per resolution

To represent the baseline cases for the various resolutions, we compressed one stream at each resolution and compared it with the reference sequences. This provides us with approximately the best compression efficiency for a given resolution. The results shown in Table 2 are for the *The Italian Job* sequence with a fixed quantization level of 4. They are used in comparison with other approaches unless otherwise specified. The simulcast case contains the stream for a resolution and all streams with lower resolutions than that resolution. For the PSNRs shown in Table 2, bit-rates for a single resolution provide the best-case baseline for per-resolution coding efficiency; bit-rates for simulcast streams provide the worst-case baseline for overall coding efficiency.

Resolution	90x60	180x120	270x180	360x240	540x360	720x480
Average Bit Rate	175.6	474.7	744.4	1,117.4	1,798.2	2,567.4
Average PSNR	34.8	36.7	37.9	38.6	39.6	40.7
Accumulated bit rate for simulcast	175.6	650.1	1,394.5	2,511.9	4310.1	6877.5

Table 2 Bit rates and PSNRs for non-scalable streams (The Italian Job). Bit-rates are in bits per seconds. Bit-rates for a single resolution provide the best-base baseline for per-resolution coding efficiency; bit-rates for simulcast streams provide the worst-case baseline for overall coding efficiency.

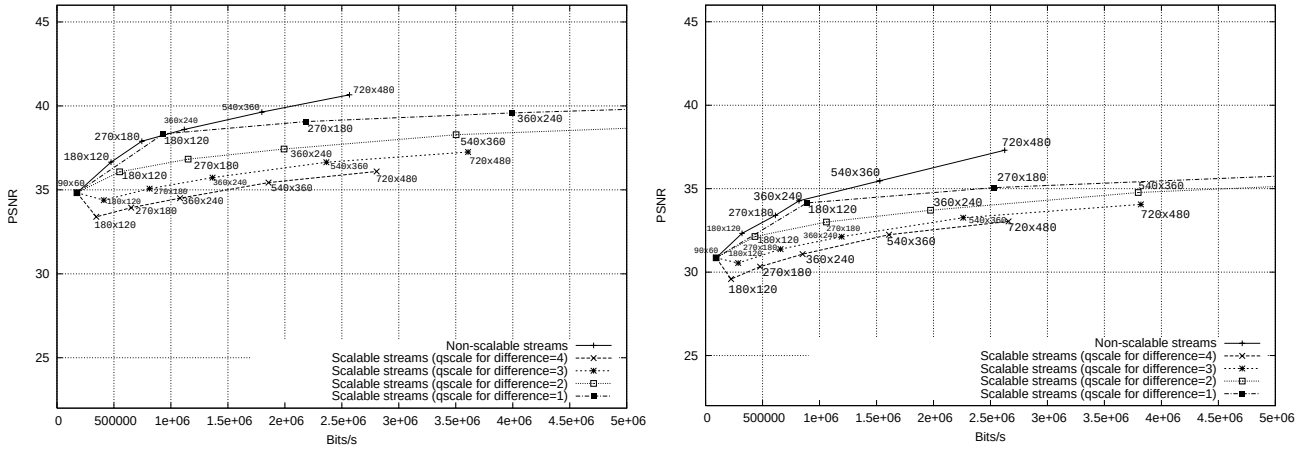
4.4. One stream for all resolutions

4.4.1. Pixel-domain-based transcoding and scalable encoding

Scalable encoding is accomplished by running the non-scalable *ffmpeg* encoder several times as described by Dugad and Ahuia[6]. The lowest resolution 90x60 is encoded as usual; then it is decoded and then upsampled to 180x120 pixels. The upsampled images are compared to the reference 180x120 images and the differential images are encoded. The 180x120 images are reconstructed by decoding the differential video and combining the decoded differential images with the upsampled images. The encoded stream for 180x120 images consists of encoded 90x60 video and the encoded differential video. The process is repeated until it reaches the full resolution 720x480.

The per-resolution coding efficiency for pixel-domain-based scalable encoding is shown in Figure 4. Different quantization scales generate very different results. When the same quantization scale used in non-scalable encoding is used for encoding the base layer and the differential layers, we have similar video bit-rates as non-scalable streams but much lower PSNRs. Since differential images consist of high frequency signals, a large quantization scale has caused significant information loss for differential images thus lowering the quality of the reconstructed images. To reconstruct images with acceptable quality, we could change the quantization scale to 1 for differential images. In this case, we get better quality than non-scalable encoding with the quantization scale 4 and quality comparable to non-scalable encoding with quantization 3. The video bit-rates are very high though compared to non-scalable streams. In fact, they are even higher than the sum of the multiple non-scalable streams (simulcast) that contain all of the corresponding resolutions in most cases as shown in Table 3.

Table 3 shows for the *The Italian Job* sequence the overall coding efficiency of non-scalable streams and scalable streams. Results of different encoding parameters are compared because it is hard to align the PSNRs or the bit-rates to do an accurate comparison. But at least we can conclude that the overall coding efficiency for scalable encoding is no better than that for non-scalable encoding. The scalable encoding approach in our experiments is quite naïve; but it nevertheless shows the inefficiency of implementing fine-grain spatial scalability for DCT-based video compression.



(a) The Italian Job (b) Friends

Figure 4 Open loop scalable encoding vs. non-scalable encoding. Transcoding is equavilent to non-scalable encoding in this case so it is not shown in this figure. Scalable encoding is done using the non-scalable ffmpeg encoder as described in 6. Different quantization scales are used for the enhancement layers.

	90x60	180x120 and lower	270x180 and lower	360x240 and lower	540x360 and lower	720x480 and lower
Accumulated bit rate for simulcast (qscale=4) / PSNR	175.6 / 34.8	650.1 / 36.7	1,394.5 / 37.9	2,511.9 / 38.6	4310.1 / 39.6	6877.5 / 40.7
Accumulated bit rate for simulcast (qscale=3) / PSNR	222.0 / 36.0	816.0 / 37.8	1,753.6 / 39.0	3159.7 / 39.7	5432.8 / 40.6	8702.7 / 41.7
Bit rate for scalable streams (base qscale=4 enhance qscale=1) / PSNR	175.6 / 34.8	932.4 / 38.3	2,182.0 / 39.1	3,995.0 / 39.6	7,305.5 / 40.3	12,092.4 / 40.7
Bit rate for scalable streams (base qscale=4 enhance qscale=2) / PSNR	175.6 / 34.8	549.8 / 36.1	1,151.3 / 36.8	1,993.2 / 37.4	3,502.8 / 38.3	5,476.3 / 38.8

Table 3 Bit-rates and PSNRs of scalable streams vs. simulcast non-scalable streams (The Italian Job). Bit rates are in bits per second. In most cases, the overall coding efficiency of scalable encoding is worse than simulcast, which is supposed to be the worst-case baseline for overall coding efficiency.

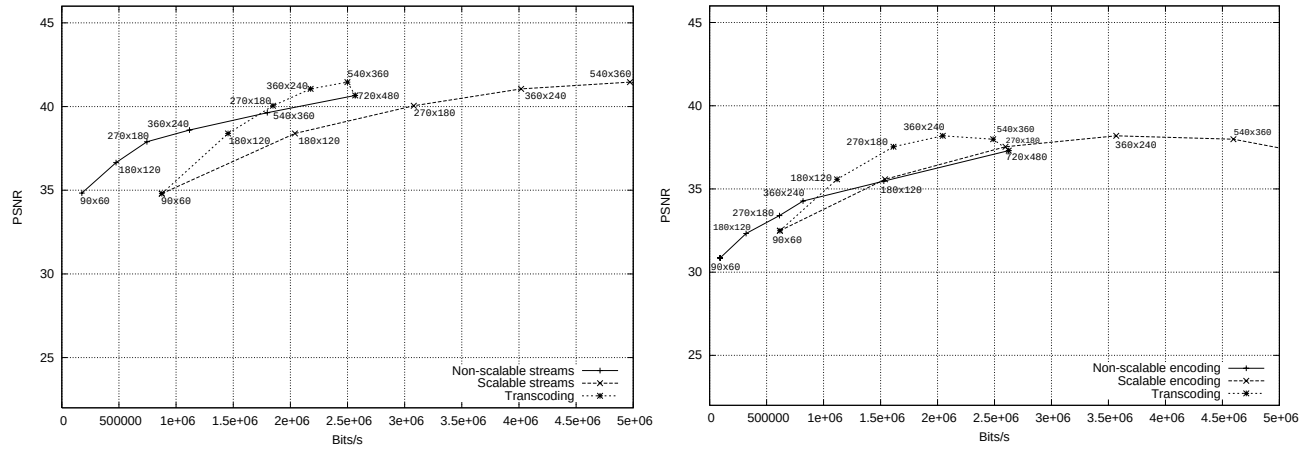
4.4.2. DCT-domain-based transcoding and scalable encoding

For transcoding, a DCT transform is performed on the set of 720x480 images. To obtain 90x60 images, only the DC component of each 8x8 coefficient matrix is kept and all AC coefficients are zeroed out before run-length coding. After decoding the transcoded stream, the images are downsampled by 8. Similarly, to obtain 180x120 images, only 2x2 coefficients are kept and the decoded images are downsampled by 4.

Scalable encoding is done based on the same scheme except that each coefficient is included only once. For example, the DC components of all matrices are encoded into the base layer and they are zeroed out in the first enhancement layer which contains the 2x2 coefficients. At the receiver side, DCT coefficients at different positions are combined back into one matrix; thus for a given resolution, the data used in decoding a scalable encoded stream are the same as those used in decoding a transcoded stream. Thus, both transcoding and scalable encoding achieve the same PSNR.

As shown in Figure 5 the PSNRs are higher than those for non-scalable streams; but the video bit-rates are much higher. So it is hard to say which is better for a given resolution (however, we can see that DCT-based algorithms perform better for the *Friends* sequence than for the *The Italian Job* sequence). We will study more about per-resolution coding efficiency in Subsection 4.4.4. Scalable encoding incurs more overhead because the zig-zag scan order of MPEG is inefficient for matrices of zeroed-out low-frequency coefficients. But it supports multiple resolutions in one stream.

The overall coding efficiency is shown in Table 4. Transcoding is the best at the price of streaming-time computation. Scalable encoding is worse than non-scalable encoding when there are a small number of resolutions and is better than non-scalable encoding only when all the six resolutions are included.



(a) The Italian Job (b) Friends

Figure 5 Effects of Dropping High-frequency Coefficients on Video Resolution. Transcoding and scalable encoding achieve same PSNRs that are higher than those of non-scalable encoding. The price is higher bit-rates. For scalable encoding, the increased bit-rates are for both higher quality and flexibility.

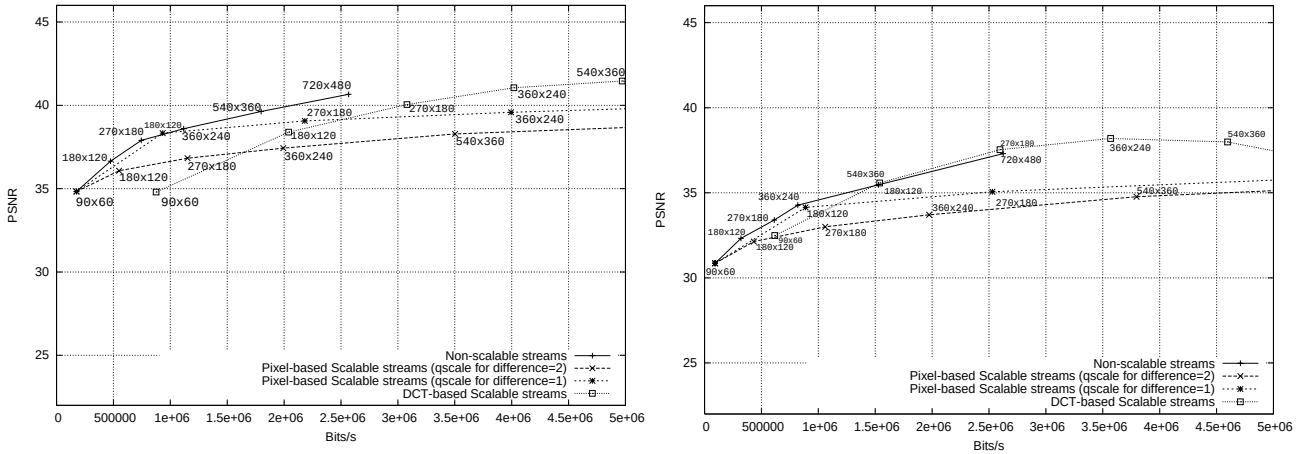
	90x60	180x120 and lower	270x180 and lower	360x240 and lower	540x360 and lower	720x480 and lower
Accumulated bit rate for simulcast non-scalable streams / PSNR	175.6 / 34.8	650.1 / 36.7	1,394.5 / 37.9	2,511.9 / 38.6	4310.1 / 39.6	6877.5 / 40.7
Accumulated bit rate for simulcast transcoded streams / PSNR	874.5 / 34.8	1,454.9 / 38.4	1,848.0 / 40.0	2,177.2 / 41.1	2,499.4 / 41.5	2,567.4 / 40.7
Bit rate for scalable streams / PSNR	874.5 / 34.8	2,039.5 / 38.4	3,079.5 / 40.0	4,018.9 / 41.1	4,971,7 / 41.5	5,489.1 / 40.7

Table 4 Bit-rates and PSNRs of scalabe encoding, simulcast of non-scalable streams, and simulcast of transcoded streams (The Italian Job). Bit rates are in bits per second.

4.4.3. Comparison of scalable encoding schemes

We compare the coding efficiency of the two scalable encoding approaches discussed in the last two subsections: pixel-domain-based and DCT-domain-based.

As shown in Figure 6, for small resolutions, DCT-domain-based scalable encoding tends to have a little higher PSNRs and much higher video bit-rates, and likely lower coding efficiency than pixel-domain-based scalable encoding. It has better coding efficiency for large resolutions. Since each scalable stream supports multiple resolutions, the overall coding efficiency for DCT-domain-based scheme is much higher than that for pixel-domain-based scheme, implying that DCT-domain-based scheme is better for supporting many resolutions.

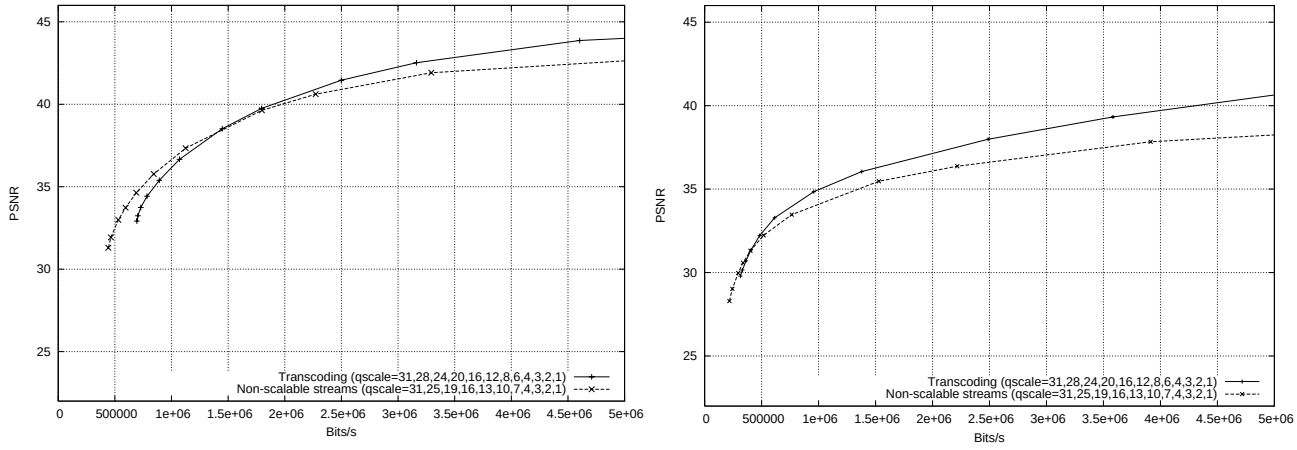


(a) The Italian Job (b) Friends

Figure 6 Two scalable encoding schemes. The pixel-domain-based scheme encodes the differential signals in the pixel domain. The DCT-domain-based scheme encodes different frequency bands for each layer.

4.4.4. More about per-resolution efficiency

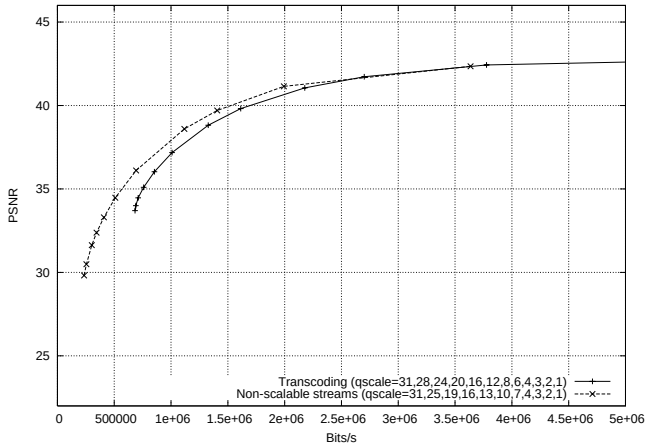
We further investigate the per-resolution coding efficiency of the one-stream-for-all-resolution architecture. Since transcoding is more efficient for a given resolution than scalable encoding, we use the results from transcoding; since transcoding by dropping high-frequency coefficients is more practical than transcoding in the pixel-domain, we consider results from transcoding through dropping high-frequency DCT coefficients.



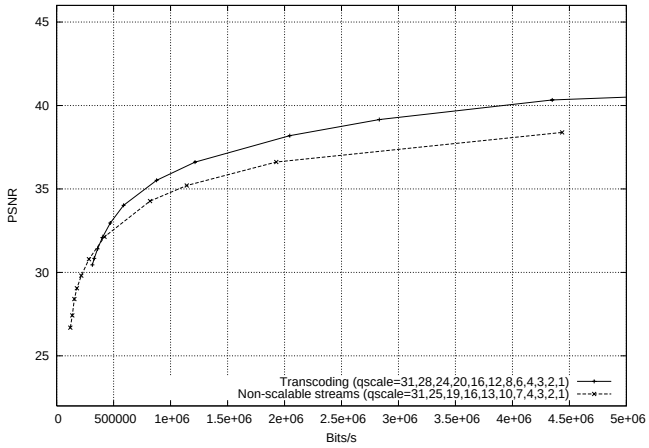
(a) The Italian Job (b) Friends

Figure 7 Non-scalable encoding and DCT-transcoding for resolution 540x360. The solid line represents transcoding and the dotted line represents non-scalable encoding. Dots on a line correspond to quantization scales.

Figure 7, Figure 8, and Figure 9 compare the coding efficiency of transcoding with non-scalable encoding, the baseline for per-resolution coding efficiency, at resolutions 540x360, 360x270, and 90x60 of quantization scales form 1 to 32. Figure 7 shows that transcoding performs pretty close to non-scalable encoding for downscaling a little. For the *Friends* sequence, transcoding performs better in general than encoding at 540x360 directly; for the *The Italian Job* sequence it performs better in the high bit-rate range. Figure 8 shows that transcoding to the half size does not support some low bit-rates; and for bit-rates supported it is about 0.5 to 1db worse than non-scalable encoding for the *The Italian Job* sequence and 1 to 2db better than non-scalable encoding for the *Friends* sequence. Figure 9 shows that transcoding to 90x60 is bad for both sequences; for the *The Italian Job* sequence, the lowest bit-rate available through transcoding is higher than the highest bit-rate of non-scalable streams while the quality is about 5db worse than the best quality of non-scalable streams. While the bit-rates of transcoded streams increase, the PSNR does not change much. Transcoding performs better for the *Friends* sequence because there is less motion thus less drifting errors.

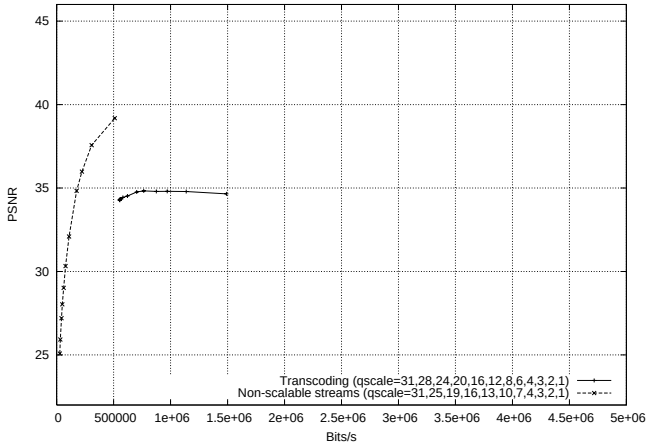


(a) The Italian Job

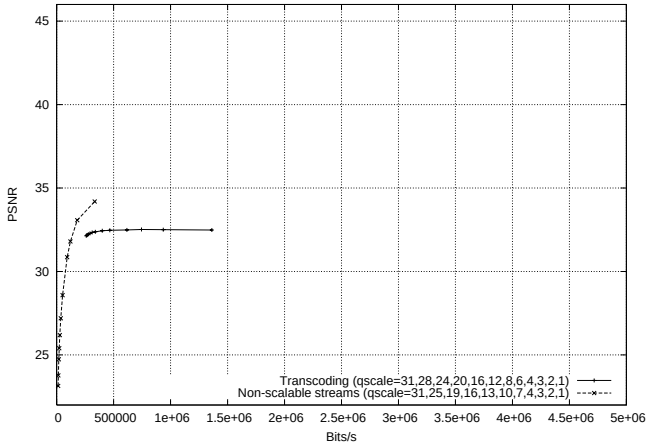


(b) Friends

Figure 8 Non-scalable encoding and DCT-transcoding for resolution 360x240. The solid line represents transcoding and the dotted line represents non-scalable encoding. Dots on a line represent quantization scales.



(a) The Italian Job



(b) Friends

Figure 9 Non-scalable encoding and DCT transcoding for resoluiton 90x60. The solid line represents transcoding and the dotted line represents non-scalable encoding. Dots on a line represent quantization scales.

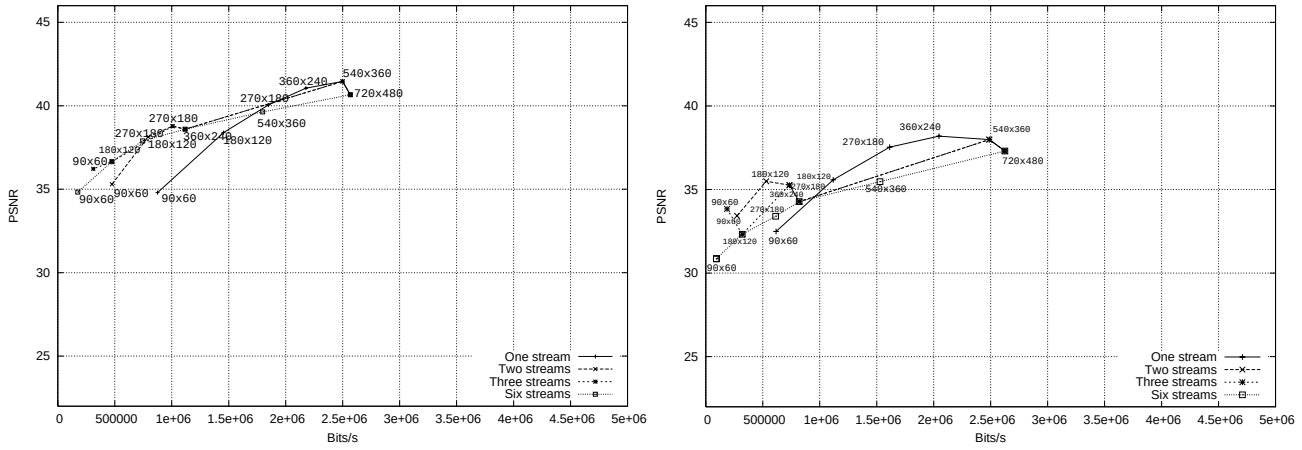
4.5. Hybrid-based Video Representation

In this subsection, we consider supporting six resolutions using two or three streams. Each stream supports two or four resolutions. Since the number of resolutions supported by a stream is small, the advantage in overall coding efficiency provided by scalable encoding is not obvious; thus we support two or four resolutions through transcoding by dropping high-frequency coefficients.

Figure 10 shows the performance of an architecture with two streams: one is at resolution 720x480, the other at 360x240. Resolution 540x360 is obtained by trancoding from the 720x480 stream and resolutions 270x180, 180x120, and 90x60 are obtained by transcoding from the 360x240 stream. Compared to the one-stream-all-resolutions approach, an additional 360x240 stream needs to be stored. The increase in storage cost is not much since the 360x240 stream is much smaller than the 720x480 stream while the efficiency for lower resolutions is greatly improved compared to the results in Figure 9. Compared to the one-stream-per-resolution, the increase in computation is not much since no DCT/IDCT or motion estimation is involved.

Figure 10 also shows the performance of a three-stream architecture: one is at resolution 720x480, one at 360x240, and the other at 180x120. Its per-resolution performance is pretty close to the one-stream-per-resolution architecture.

As shown in Table 5, the two-stream architecture and the three-stream architecture improve the coding efficiency for low resolutions at the price of the overall coding efficiency, which is still better than that of non-scalable streams.



(a) The Italian Job (b) Friends

Figure 10 Multiple streams with transcoding. In this figure, six-stream is the one-stream-per-resolution approach; one-stream is the one-stream-for-all-resolution approach. Two-stream (720x480 and 360x240) and three-stream (720x480, 360x240, and 180x120) are two architectures falling in between. In the hybrid architectures, resolutions that are not stored are derived from the closest higher base resolution through dropping high-frequency DCT coefficients.

	90x60	180x120 and lower	270x180 and lower	360x240 and lower	540x360 and lower	720x480 and lower
Accumulated bit rate for simulcast non-scalable streams / PSNR	175.6 / 34.8	650.1 / 36.7	1,394.5 / 37.9	2,511.9 / 38.6	4310.1 / 39.6	6877.5 / 40.7
Accumulated bit rate for simulcast transcoded streams from 1 stream / PSNR	874.5 / 34.8	1,454.9 / 38.4	1,848.0 / 40.0	2,177.2 / 41.1	2,499.4 / 41.5	2,567.4 / 40.7
Accumulated bit rate for simulcast transcoded streams from 2 streams / PSNR	475.0 / 35.3	793.9 / 38.1	1,007.7 / 38.8	1,117.4 / 38.6	3,616.8 / 41.5	3,684.8 / 40.7
Accumulated bit rate for simulcast transcoded streams from 3 streams / PSNR	310.8 / 36.2	474.7 / 36.7	1,482.4 / 38.8	1,592.1 / 38.6	4,091.5 / 41.5	4,159.5 / 40.7

Table 5 Bit-rates and PSNRs of simulcast of non-scalable streams and simulcast of transcoded streams (The Italian Job). Bit rates are in bits per second.

5. CONCLUSION

In this paper, we have examined the various trade-offs in supporting wide-scale, fine-grained multi-resolution adaptation. We believe that in the future, video streaming algorithms for both stored and live video will need to support extremely high-resolution video mapped to a large number of display characteristics. In addition, we believe that such systems will also need to support efficient region-of-interest cropping, especially for applications such as telepresence.

Our results show that encoding *n* video streams for *n* displays results in highly compressed and optimized video streams. The main drawback of this approach is the high computational complexity required to churn out a potentially large number of streams. Scalable encodings are useful but cannot support an extremely wide-range of display characteristics. Finally, our results show that adapting a video stream between relatively close resolution requirements makes sense.

In the future, we are working towards supporting both resolution and region adaptive video streaming algorithms.

REFERENCES

1. R. Atta and M. Ghanbari, "A Drift Compensation Architecture for DCT-Pyramid Video Coding", *IEEE International Symposium on Video / Image Processing and Multimedia Communications*, Zadar, Croatia, 2002
2. Ulrich Benzler, "Spatial Scalable Video Coding Using a Combined Subband-DCT Approach", IEEE Transactions on Circuits and Systems for Video Technology, **Vol. 10, No. 7**, October 2000.
3. J.-C. Bolot and T. Turletti, "Experience with control mechanisms for packet video in the Internet", ACM SIGCOMM Computer Communication Review, **vol. 28**, pp. 4--15, January 1998.
4. Marek Domanski, Adam Luczak, and Slawomir Mackowiak, "Spatial-Temporal Scalability for MPEG Video Coding", IEEE Transactions on Circuits and Systems for Video Technology, **Vol. 10, No. 7**, October 2000.
5. Rakesh Dugad and Narendra Ahuja, "A Fast Scheme for Image Size Change in the Comrpessed Domain", IEEE Transactions on Circuits and Systems for Video Technology, **VOL. 11, NO. 4**, April 2001.
6. Rakesh Dugad and Narendra Ahuja, "A Scheme for Spatial Scalability Using Nonscalable Encoders", IEEE Transactions on Circuits and Systems for Video Technology, **Vol. 13, No. 10**, October 2003.
7. http://ffmpeg.sourceforge.net/. ffmpeg homepage.
8. Barry G. Haskell, Atul Puri, and Arun N. Netravali, *Digital Video: An Introduction to MPEG-2*. Chapman & Hall. ISBN 0-412-08411-2.
9. Q. Hu and S. Panchanathan, "Image/Video Spatial Scalability in DCT Domain", IEEE Trans. Ind. Electron, **vol. 45**, pp. 23-31, Feb. 1998.
10. W. Feng, M. Liu, B. Krishnaswami, and A. Prabhudev, "A Priority-Based Technique for the Best-Effort Delivery of Stored Video", In Proceedings of *SPIE/IS&T Multimedia Computing and Networking*, San Jose, January 1999.
11. Sang H. Kang and Avideh Zakhor, "Packet Scheduling Algorithm for Wireless Video Streaming", *12th International Packet Video Workshop (PV2002)*, Pittsburgh, PA, April 2002.
12. C. Krasic, J. Walpole, and W. Feng, "Quality-Adaptive Media Streaming by Priority Drop". In Proceedings of *NOSSDAV* 2003, Monterey, California, June 2003
13. Yuh-Reuy Lee, Chia-Wen Lin, Cheng-Chien Kao, "A DCT-Domain Video Transcoder for Spatial Resolution Downconversion", VISUAL 2002 Hsin Chu, Taiwan, P207-218, *Lecture Notes in Computer Science* 2314 Springer 2002, ISBN 3-540-43358-9. March 11-13, 2002
14. Z.Lei and N.D.Georganas, "H.263 Video Transcoding for Spatial Resolution Downscaling", Proc. *IEEE International Conference on Information Technology: Coding and Computing* (ITCC 2002), Las Vegas, Nevada, April 2002.
15. Weiping Li, "Overview of fine granularity scalability in MPEG-4 video standard", IEEE Transactions on Circuits and Systems for Video Technology, **Volume 11, Issue: 3**. March 2001
16. Zhourong Miao and Antonio Ortega, "Expected Run-time Distortion Based Scheduling for Delivery of Scalable Media", *12th International Packet Video Workshop (PV2002)*, Pittsburgh, PA, April 2002.
17. T. Naveen and John W. Woods, "Motion Compensated Multiresolution Transmission of High Definition Video", IEEE Transactions on Circuits and Systems for Video Technology, **Vol. 4, No. 1**, February 1994.
18. Xiaoyan Sun, Feng Wu, Shipeng Li, Wen Gao, and Ya-Qin Zhang, "Seamless Switching of Scalable Video Bitstreams for Efficient Streaming", IEEE Transaction on Multimedia, **Vol. Y, No. 2**, April 2004.
19. Kuan Hui Tan and Mohammad Ghanbari, "Layered Image Coding Using the DCT Pyramid", IEEE Transactions on Image Processing. **Vol. 4, No.4** April, 1995.
20. Peng Yin, Min Wu, and Bede Liu, "Video Transcoding by Reducing Spatial Resolution", ICIP 2000.