Analysis of Video Segmentation for Spatial Resolution Reduction Video Transcoding

Fareed Jokhio, Tewodros Deneke, Sébastien Lafond, Johan Lilius
Åbo Akademi University
Department of Information Technologies
Joukahainengatan 3-5, 20520 Turku, Finland
Email: {fjokhio, tdeneke, slafond, jolilius}@abo.fi

***Abstract*—In this paper different methods of video segmentation are analyzed to perform spatial resolution reduction video transcoding with multiple processing units. A distributed video transcoder is built in which different processing units perform the transcoding operation. To fully utilize the computational power of different processing units the distribution of computational load should be equal. In video transcoding different frames require different computational power hence inefficient video segmentation will lead towards lower performance. We have analyzed three possible methods of video segmentation: (1) each segment has equal size, (2) each segment has equal number of frames, and (3) each segment has equal number of group of pictures. The performance of the system, the relationship between the processing units used and speed in computation is measured in terms of standard deviation of transcoding time of different processing units.**

***Keywords*-Video transcoding, video downscaling, message passing interface (MPI), distributed computing, multiprocessing**

I. Introduction

The number of users using video contents on Internet is growing rapidly. Several peoples watch online video everyday on video services such as YouTube. Every well known TV channel is able to deliver video contents to their audiences through the web. The users access the web contents with a variety of devices, and with various communication channels. The devices at the end-user side may have various processing power, display size and supported video codec. Some users have their desktop computers connected with high speed network and they demand high definition video and some users watch low resolution video on their mobile phones. Currently there are several video formats available and with the passage of time their number is increasing. It is not efficient to store a video in all possible formats with different resolutions and frame rates. The best possible way to provide better service to the end user is to store the video in a suitable format and then transcode it on demand. The videos are stored and transmitted in compressed form for better utilization of storage medium and communication channel. The transcoder will have a compressed video as input and will provide a compressed video as output. In the broader view, the video transcoder can perform a:

- bit rate reduction
- spatial resolution reduction
- temporal resolution reduction

With bit rate reduction the frame size and frame rate remain the same. The bit rate reduction is possible by reducing the video quality.

The spatial resolution reduction transcoding will produce the output video with a smaller resolution than the original video but the frame rate will remain the same. With spatial resolution reduction less number of bits will be required due to the smaller size of frames.

The temporal resolution reduction transcoding produces the output video with a lower frame rate. The bit rate will also be less but the frame size remains unchanged.

It is also possible to combine both spatial resolution reduction and temporal resolution reduction video transcoding.

The main contribution of this paper is the analysis of video segmentation to perform spatial resolution reduction video transcoding on different processing units.

II. Background and related work

Researchers have done significant work on video transcoding in the past few years and there are several video transcoding architectures available. In [1] an approach was proposed to compute downscaled motion vectors from the original motion vectors for a video sequence. In [2] motion vector generation for spatial resolution reduction and impact of motion estimation refinement on it is discussed. In [3] the causes of drift errors were analyzed for spatial resolution reduction transcoding.

In order to get more speed up for the transcoding process, cluster based distributed transcoding approaches were also developed in the past few years. Multimedia web server in a cluster is discussed in [4]. The video contents were generated dynamically according to the bit rate and the bandwidth requirements. In [5] distributed video transcoder was proposed in which the compressed video was fully decoded and then re-encoded according to new bit rate and format. In this work distributed transcoding of MPEG-2 was performed.

Currently, multi-core systems are available but to utilize their processing power existing sequential processing applications needs to be modified for better speed up. In this work, instead of processing operations in parallel on a single video frame we selected to use the distributed processing approach.

In a distributed system each processing element has its own memory and communication among them is done with message passing. We have chosen the MPI (message passing interface) for our transcoder. The MPI standard is open source. This standard is flexible, efficient, portable, practical and easy for application development. The MPI is used for MIMD (multiple instructions multiple data) systems [6]. In the MPI model the computation can be done on a set of machines. Every machine will get its part of the work and communication among different machines is possible through sending and receiving messages. The data transfer among two machines will be done through send and receive messages [6] [7].

III. Spatial resolution reduction video transcoding

There are two possible ways to downscale the compressed video resolution. In the first way the entire video is decoded and then re-encoded according to the new resolution and quality. Because the motion estimation process in video encoding takes more than 60% of the transcoding time [8], this is an inefficient approach to downscale a compressed video resolution. The second way is to down scale the macroblocks and their motion vectors. Figure 1 shows the spatial resolution down conversion of the macroblocks.

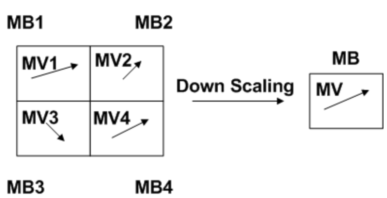

Figure 1. Spatial resolution down conversion

In spatial resolution downscaling several 8x8 DCT blocks are downscaled into one 8x8 DCT block. Averaging the DCT blocks is the most efficient way to get downscaled DCT block [9]. To calculate the new motion vector for the downscaled DCT block there are several methods available just as the averaging, the weighted mean, the median etc. Further details of spatial resolution downscaling can be found in [9].

The macroblocks in the compressed source video can have different modes. The possible macroblock modes can be INTRA, INTER and SKIP. After downscaling a macroblock, its mode must be selected based on the modes of its source video macroblocks. If the source video has at least one INTRA mode macroblock then the new macroblock will be coded as INTRA macroblock and the corresponding DCT coefficients will be coded. If there is no intra macroblock and if there is at least one macroblock with the INTER type then the new macroblock will be coded as INTER type macroblock. In this case the corresponding DCT coefficients of the predictive residues will be coded. If there are only SKIP mode macroblocks then the new macroblock type will be SKIP.

IV. System Overview

To perform the video transcoding we selected ffmpeg [10] transcoder which is open source and designed for a single machine. This transcoder supports several types of different video and audio formats. We modified the source code in such a way that it can be executed on multiple processing units using the Message Passing Interface (MPI) programming technique. In order to execute the MPI based version of the ffmpeg transcoder at least two processing units are required. The processing units could be even two cores inside a single processor. In our MPI based transcoder one processing unit will be used as the master and the others will be used as slaves or workers. Here the term worker is used to indicate one instance of the transcoder executing on a processing node. The worker indicates a process instead of a physical processing unit. It is possible to have less physical processing units and more worker processes executing on them.

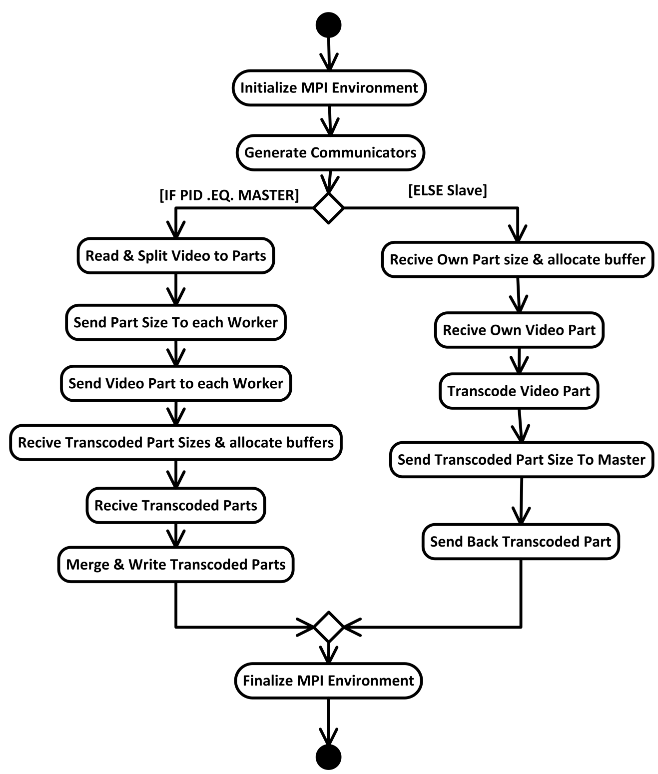

Figure 2. Video Transcoder with Message Passing Interface

Figure 2 shows the tasks performed by the master machine and worker machines for a single video sequence. In the MPI based implementation the number of worker machines will be provided to the master and then the video segmentation will be performed accordingly. In order to get faster transcoding the number of workers should be equal to the number of physical processing units or a multiple of them. In MPI based systems every master is assumed to be a single process with some ID and worker is assumed to be a process with its unique ID. The master machine performs the video segmentation of the source video and sends those segments to worker machines. Each worker will start transcoding its video segments. Once the transcoding is completed by the workers, the transcoded video is sent back to the master. Finally the master performs the mergering operation.

V. Experimental Setup

The experimental system consists of 2 quad-core Intel(R) Xeon(R) processors (E5430) at 2.66 GHz. The configurations of the proceesing units is as shown in table I.

Cache size	6144 KB
Address sizes	38 bits physical, 48 bits virtual

Table I
Configuration of the procesing units

The total number of proceesing elements is 8 in our systems. The figure 3 shows the architecture of the hardware platform.

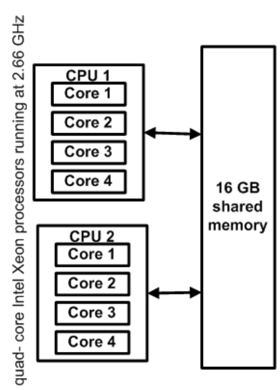

Figure 3. Workstation with eight processing units

The big buck bunny video sequence [11] was used as a source video to perform transcoding operations. The format of the source video is H263 16CIF (1408 x 1152) with 24 fps. The size of the source video is 220 MB and its play time is 09 minutes and 56 seconds. The total number of frames in this video sequence is 14315.

VI. Video segmentation

To get more speed up in distributed video transcoding, the method of video segmentation plays a very important role. In ideal conditions the source video should have segments in such a way so that every segment should require the same computational processing power. The video segments should be independent units such that they can be sent to different workers to perform independent transcoding operations. The compressed source video can contain different types of frames just as I (intra) frames, P (predicted) frames and B (bi-directional predicted) frames. The stream structure for H.263 down to Block level is shown in figure 4.

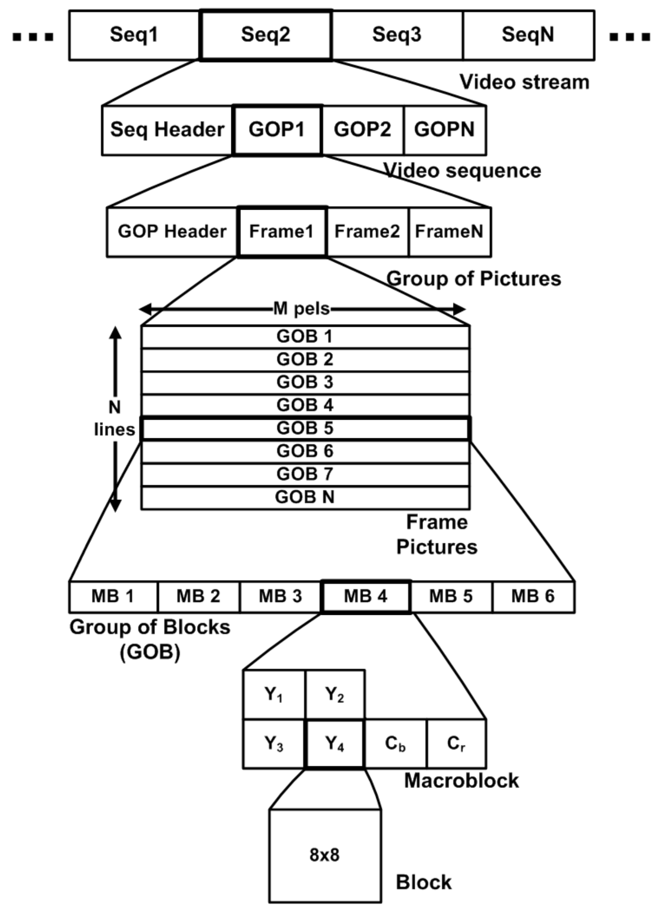

Figure 4. H.263 stream structure down to Block level

The video sequence consists of several groups of pictures, one group of pictures consists of an intra frame at the beginning and a number of P and B frames. Due to interdependencies among different types of frames the video segmentation can be done at group of pictures (GOP) level only. The group of pictures can be of two types, either they will be open-GOPs or closed-GOPs. The segmentation of both types of group of pictures will be performed differently. The closed-GOPs segmentation is very easy and simple. It will consist of an Intra frame at the beginning followed by a number of P and B frames. The closed-GOPs can be transcoded independently. In the case of open-GOPs one reference frame (I or P) will be required to decode the B type frame from another GOP. This extra frame will be discarded after decoding. Segmentation of open-GOP is further discussed in [5]. The video segmentation can be performed in many different ways just like:

- equal size segmentation
- equal number of group of pictures (GOP) in each segment
- equal number of frames in each segment

A. Equal size segmentation

	1	2	3	4	5	6	7
Size in MB # of GOPs # of Frames	220 1276 14315						
Size in MB # of GOPs # of Frames	110 765 8478	110 511 5837					
Size in MB # of GOPs # of Frames	74 532 5868	73 459 5118	73 285 3329				
Size in MB # of GOPs # of Frames	55 390 4278	55 375 4200	55 350 3961	55 161 1876			
Size in MB # of GOPs # of Frames	44 292 3153	44 349 3932	44 273 3052	44 241 2782	44 121 1396		
Size in MB # of GOPs # of Frames	37 235 2494	37 297 3374	37 233 2610	36 226 2508	37 174 2053	36 111 1276	
Size in MB # of GOPs # of Frames	32 183 1897	32 270 3082	31 214 2394	31 203 2319	32 197 2171	31 104 1248	31 105 1204

Table II
EQUAL SIZE SEGMENTATION

To perform the equal size segmentation, the total size of video is determined in bytes. This size is divided by the required number of segments. This will give the size in bytes in each segment. The segmentation is possible at the beginning of the group of pictures so some parts may have more number of bytes and some may have less number of bytes but the difference will be small in the sizes of different video segments. In this way almost equal size segments are obtained. Table II shows the segment attributes for equal size segmentation. In each segment the number of frames and number of group of pictures may be different. After performing the segmentation the video sequence header is inserted at the beginning of every segment so that it becomes an independent video sequence and can be sent to any processing unit for transcoding.

B. Equal number of GOP segmentation

In this type of segmentation the total number of GOP is counted and the video is segmented in equal number of group of pictures. The sequence header is attached at the beginning of every segment. Table III shows the segment attributes for equal number of group of pictures segmentation. In this segmentation the size of every segment will be different and the number of frames in each segment will also be different.

No of seg	1	2	3	4	5	6	7
Size in MB # of GOPs # of Frames	220 1276 14315						
Size in MB # of GOPs # of Frames	87 638 7049	133 638 7266					
Size in MB # of GOPs # of Frames	60 425 4684	64 426 4796	96 425 4835				
Size in MB # of GOPs # of Frames	47 319 3477	41 320 3584	54 320 3552	78 317 3702			
Size in MB # of GOPs # of Frames	38 255 2728	32 256 2899	41 256 2875	42 256 2847	67 253 2966		
Size in MB # of GOPs # of Frames	35 212 2230	26 213 2454	28 213 2365	36 213 2431	33 213 2347	62 212 2488	
Size in MB # of GOPs # of Frames	31 182 1885	22 183 2128	24 183 2041	28 183 2031	28 183 2052	29 183 2086	58 179 2092

Table III
EQUAL NUMBER OF GROUP OF PICTURE SEGMENTATION

C. Equal number of frames segmentation

In this type of segmentation the total number of frames is computed to generate segments having an equal number of frames. Hence we will get a set of positions indicating where the segmentation needs to be performed. The segmentation is not possible just before a P or B type frame due to interdependencies among the frames. These frames require a reference frame for decoding. Therefore the segmentation is performed at the beginning of the next group of pictures. Table IV shows for each segment its size, total number of group of pictures and total number of frames. In this type of segmentation the segments will have different size and different number of GOP. The number of frames will also not be exactly equal, but all segments will have a very small difference in the total number of frames.

The time required to perform the video segmentation is very small compared to the video transcoding time. For 220 MB video the segmentation time is 0.75 seconds in our experimental setup.

VII. RESULTS

The main objective of this paper is to analyze different video segmentation methods to get the shortest transcoding time for spatial resolution reduction video transcoding.

A. Transcoding time

The transcoding time for spatial resolution reduction from 6CIF (1408 x 1152) resolution to 4CIF (704 x 576) resolution is given in Figure 5. The figure shows that equal size segmentation is not efficient as compared to the equal

No of seg	1	2	3	4	5	6	7
Size in MB # of GOPs # of Frames	220 1276 14315						
Size in MB # of GOPs # of Frames	89 649 7167	131 627 7148					
Size in MB # of GOPs # of Frames	62 434 4774	63 424 4781	95 418 4760				
Size in MB # of GOPs # of Frames	49 329 3590	41 320 3577	55 321 3578	75 306 3570			
Size in MB # of GOPs # of Frames	41 267 2867	31 253 2868	41 255 2863	41 257 2859	66 244 2858		
Size in MB # of GOPs # of Frames	35 225 2386	26 209 2388	28 215 2393	35 209 2388	34 215 2380	62 203 2380	
Size in MB # of GOPs # of Frames	33 196 2051	21 177 2042	24 183 2047	27 185 2050	30 181 2042	28 180 2051	57 174 2032

Table IV
EQUAL NUMBER OF FRAMES SEGMENTATION

number of group of pictures and equal number of frames segmentation. Here every four macroblocks are downscaled to a single macroblock.

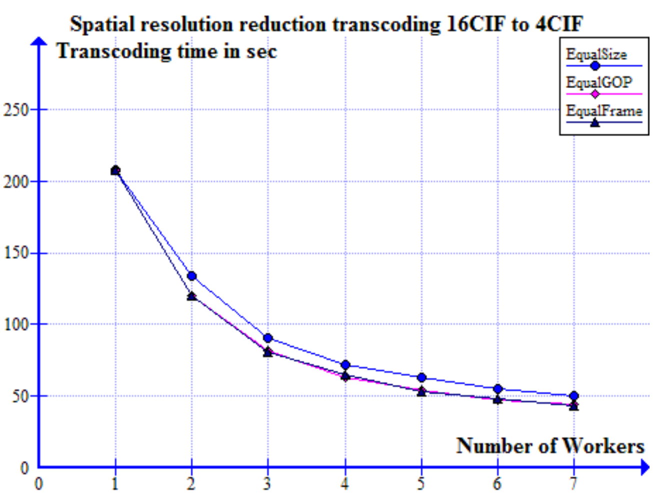

Figure 5. Spatial resolution reduction video transcodign 16CIF to 4CIF

The transcoding time for the spatial resolution reduction from the 16CIF to CIF (352 x 288) is given in the figure 6. Here every 16 macroblocks are downscaled into a single macroblock. The overall transcoding time is less as compared with the 16CIF to 4CIF downscaling. The results indicate that with different scales of downscaling the equal size segmentation is less efficient as compared with the other two methods of video segmentation.

In order to determine which method is better among these three methods, further analysis is performed based on the standards deviation of the results.

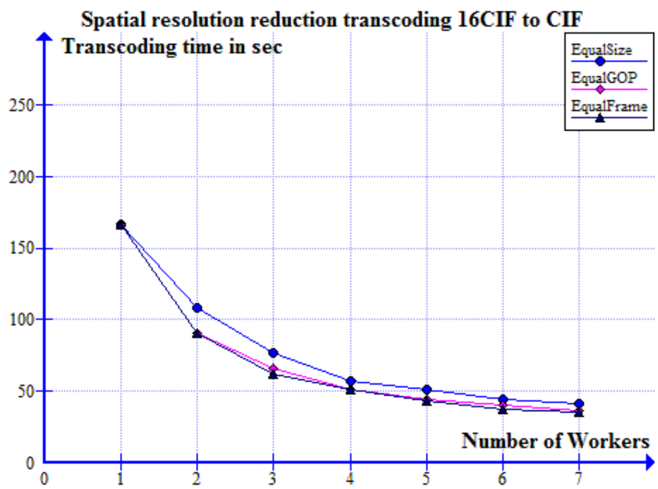

Figure 6. Spatial resolution reduction video transcodign 16CIF to CIF

B. Analysis

In order to perform further analysis the transcoding time taken by each worker for different number of segments is discussed here. The transcoding time in this analysis is for 16CIF to 4CIF downscaling.

No of seg	W1	W2	W3	W4	W5	W6	W7
1	208						
2	134	96					
3	90	81	52				
4	72	67	65	35			
5	53	63	52	48	24		
6	55	55	46	46	38	24	
7	37	50	45	45	44	28	21

Table V
VIDEO TRANSCODING TIME IN SECONDS WITH EQUAL SIZE SEGMENTATION

Table V shows the transcoding time for different workers when all segment have the same size. Because of unbalanced computational load the transcoding time of different workers are different.

No of seg	W1	W2	W3	W4	W5	W6	W7
1	208						
2	116	120					
3	74	79	81				
4	54	62	62	65			
5	53	46	52	46	48		
6	40	45	37	46	47	48	
7	43	35	41	38	42	43	38

Table VI
VIDEO TRANSCODING TIME IN SECONDS WITH EQUAL NO OF FRAMES SEGMENTATION

Table VI shows the transcoding time for different workers when segments have an equal number of frames. The results indicate that there is less disparity in the transcoding time of the different workers compared to the previous case.

No of seg	W1	W2	W3	W4	W5	W6	W7
1	208						
2	114	120					
3	78	82	81				
4	58	58	59	63			
5	50	45	54	50	49		
6	43	40	42	45	41	47	
7	41	40	36	42	38	38	44

Table VII
VIDEO TRANSCODING TIME IN SECONDS WITH EQUAL NO OF GOP SEGMENTATION

Table VII shows the transcoding time for different workers when segments have an equal number of GOP. The results indicate that the workers have a similar computational load.

In order to determine which method of video segmentation is more efficient, the standard deviation was calculated for all methods with different number of segments. Table VIII shows the standard deviation for the different methods of segmentation. The smaller value of standard deviation indicates a better performance.

The standard deviation is calculated as the following.
$\sigma = \sqrt{\frac{1}{N}(\sum_{i=1}^{N}(x_i - \mu)^2)}$
Where σ is the standard deviation, N is the total Number of segments, x_i represents the transcoding time for the ith segment and μ is the mean of transcoding time.

No of Segments	Equal size	Equal GOPs	Equal Frames
1	0	0	0
2	19	3	2
3	16.2	1.7	2.9
4	14.5	2	4
5	13	2.8	3
6	10.7	2.5	4
7	9.7	2.5	2.8

Table VIII
STANDARD DEVIATION FOR DIFFERENT METHODS OF SEGMENTATION

The results in table VIII indicates that the segmentation with equal number of GOPs is having better performance as compared with equal number of frames segmentation and equal size segmentation.

VIII. CONCLUSION AND FUTURE WORK

In this paper, we have proposed a transcoder implementation which can provide fast transcoding and can utilize the processing power of many core systems and also distributed systems. Three different methods of video segmentation are discussed for the spatial resolution reduction video transcoding. The results indicates that the equal number of group of pictures approach for video segmentation has the lowest standard deviation of transcoding time on different processing units and is better compared to segmentation based on equal size and equal number of frames.

Further analysis can be performed to find out which method is better for other kinds of video transcoding just as bit rate reduction and temporal reduction transcoding. Other method of video transcoding may also be analyzed in which every segment will have equal number of INTRA macroblocks. This may give slightly better performance in terms of standard deviation but it will require more time in segmentation operation as compared with the existing segmentation methods hence the overall speed up of the transcoding operation may not be improved.

REFERENCES

[1] D. Shen, I. K. Sethi, and B. Vasudev, "Adaptive motion-vector resampling for compressed video downscaling," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 9, no. 6, p. 929, Sep. 1999.

[2] T. Shanableh and M. Ghanbari, "Heterogeneous video transcoding to lower spatio-temporal resolutions and different encoding formats," *Multimedia, IEEE Transactions on*, vol. 2, no. 2, pp. 101 –110, jun 2000.

[3] P. Yin, A. Vetro, B. Liu, and H. Sun, "Drift compensation for reduced spatial resolution transcoding," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 12, no. 11, pp. 1009 – 1020, nov 2002.

[4] J. Guo, F. Chen, L. Bhuyan, and R. Kumar, "A cluster-based active router architecture supporting video/audio stream transcoding service," in *Parallel and Distributed Processing Symposium, 2003. Proceedings. International*, april 2003, p. 8 pp.

[5] Y. Sambe, S. Watanabe, D. Yu, T. Nakamura, and N. Wakamiya, "High-speed distributed video transcoding for multiple rates and formats," *IEICE Transactions*, vol. 88-D, no. 8, pp. 1923–1931, 2005. [Online]. Available: http://dx.doi.org/10.1093/ietisy/e88-d.8.1923

[6] Message Passing Interface Forum, *MPI: A Message-Passing Interface Standard*. Knoxville, TN: University of Tennessee, Jun. 1995.

[7] Gropp, W., Lusk, E., and Skjellum, A., *Using MPI, Portable Parallel Programming with the Message Passing Interface*. MIT Press.

[8] P. Yin, M. Wu, and B. Liu, "Video transcoding by reducing spatial resolution," in *ICIP*, 2000.

[9] Z. Lei and N. D. Georganas, "H.263 video transcoding for spatial resolution downscaling," in *ITCC*. IEEE Computer Society, 2002, pp. 425–430. [Online]. Available: http://csdl.computer.org/comp/proceedings/itcc/2002/1506/00/15060425abs.h

[10] "Ffmpeg project." [Online]. Available: http://www.ffmpeg.org/

[11] "Big buck bunny video sequence." [Online]. Available: http://www.bigbuckbunny.org/index.php/download/