Hash-Aided Motion Estimation and Rate Control for Distributed Video Coding

Shantanu Rane {ID# 4893673, srane@stanford.edu}

EE392J Project Report, Winter 2004.

Abstract

In this project we apply robust visual hash codes to low-complexity video encoding. The visual hash code consists of a short bit-string extracted from an image-block by projecting that block onto a number of randomly generated low-frequency patterns. It is robust in the sense that minor degradations in the image-block cause minor changes in the hash bit-string. We use this hash in conjunction with a distributed video codec that encodes video frames independently but decodes them conditionally, given the previously decoded frames as side-information. We show that, by transmitting the hash code of an image-block in the current frame as helper information, the decoder can coarsely determine the motion between the current and previous frames. The decoder can then motion-compensate the side-information frames such that they are good estimates of the current frame. We show experimentally that such a hash-aided motion estimation algorithm significantly improves the quality of the side-information with very low bitrate overhead. This directly reduces the bit-rate required per frame at the encoder. Further, since the motion compensation using the visual hash involves trivial hamming distance calculations, they can be replicated at the encoder, with only a small increase in complexity. The encoder then has a local copy of the motion-compensated side information, and can use it to control the encoding rate for each video frame.

1 Introduction

A video sequence is typically encoded a few times and decoded many times. Therefore, it is advantageous to shift complex operations like motion estimation to the encoder and provide a simple and fast decoder to the end user. However, there exist dual scenarios such as camera-phones and sensor networks, where the video needs to be encoded by a low-complexity, low-power device and transmitted to a powerful decoder, such as a central control station, a server, or a personal computer. In these cases, it is necessary to move the complex task of motion estimation to the decoder, and to keep the encoder as simple as possible. Recently, distributed video coding schemes have been proposed for such applications in which video frames are encoded independently (simple intraframe encoding) and decoded conditionally (complex interframe decoding), as shown in Fig. 1. These schemes have been called Wyner-Ziv video coding schemes, in honor of Wyner and Ziv's information theoretic work on distributed source coding. Using the previously decoded frame(s) as side information, the decoder estimates the motion that has occurred, and forms a motion-compensated prediction of the current video frame. Finally, it corrects the errors in this prediction using the bits transmitted by the encoder, to generate an accurate estimate of the current frame. This radical departure from conventional video compression gives rise to many new and interesting problems, two of which we shall consider in this project:

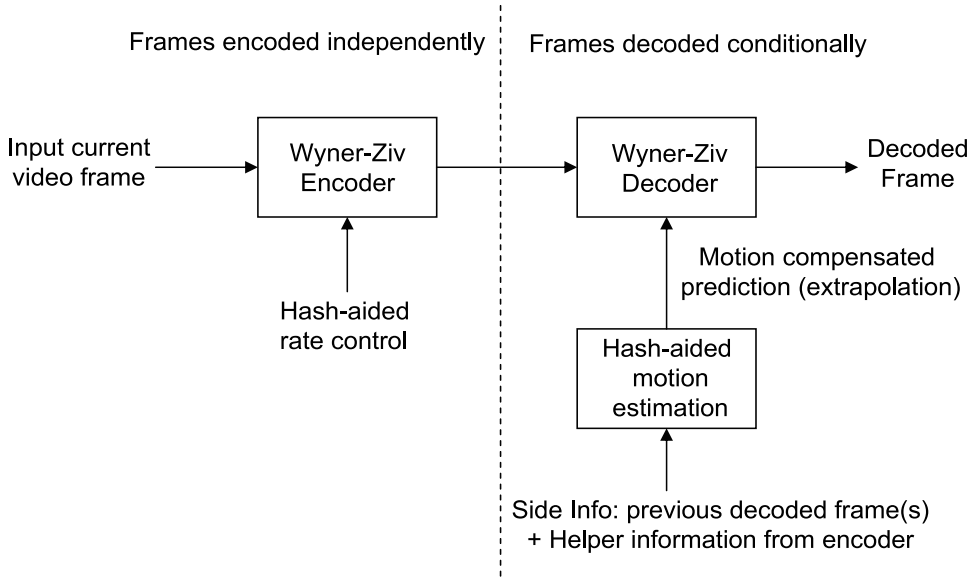


Figure 1: Hash-aided distributed video coding scheme.

1. **Motion estimation:** In the above explanation, the decoder has to estimate the motion between the previous frame and the current frame, without having access to the current frame! Therefore, the process of forming the motion compensated prediction, will be greatly simplified if we transmit helper information about the current frame to the decoder at a very low bitrate. For this, we propose to extract a *visual hash* or *signature* of a portion of the current frame and transmit the hash to the decoder as helper information. We show how this hash can be used to perform motion estimation at the decoder. Using the motion vectors, the decoder then calculates the motion compensated side information, which is hopefully, a good estimate of the current frame. A better estimate of the current frame is advantageous because it directly reduces the *Wyner-Ziv bit-rate* required to decode the current frame from its estimate.

2. **Rate Control:** The Wyner-Ziv bit-rate can be viewed as the bit-rate necessary to correct errors in a hypothetical dependence channel between the current frame (source) and the motion compensated previous frame (side information). Since the encoder cannot access the side information, there is no established method to determine how many errors the dependence channel has introduced, and consequently, how many parity bits are required to correct these errors. To estimate and control the Wyner-Ziv bit-rate, the encoder would need to access, partially or wholly, the motion compensated prediction. The above hash-aided motion estimation algorithm can be performed at far lower complexity than conventional block-matching motion estimation algorithms. Therefore we propose to replicate this algorithm at the encoder to achieve a coarse motion compensated prediction for the current frame. Based on the mean squared error between the prediction and the current frame, the Wyner-Ziv encoder can then fashion a rate-control algorithm.

2 Related Work

The emerging field of distributed video coding leverages Wyner and Ziv's information theoretic treatment of the rate-distortion function with side information available only to the decoder [Wyn76]. For the mean squared error distortion measure, it was shown that, when the source and the side information have jointly gaussian statistics, there is no loss in rate distortion performance, even when the encoder does not have access to the side information. Even without jointly gaussian statistics, significant bitrate savings are obtainable owing to the correlation between the source and the side information. Therefore, it is expected that distributed video coding schemes can approach the highly efficient performance of conventional video compression algorithms. A number of such schemes have been proposed recently [Aar04a, Aar04b, Pur02], and for this project, we will focus on the system used in [Aar04a], which is similar to that shown in Fig. 1. The hash used in [Aar04a] consists of a subsampled version of the block being coded. This hash is transmitted to the decoder and serves as an initial guess of the current block, thus helping motion estimation. The overhead incurred by this helper information is up to 80 bits per 8×8 block. Rate control was not considered because the system assumed that a feedback channel was available.

In this project, we propose to use a different hash which is robust in the sense that correlated images produce "nearly similar" hashes, while uncorrelated images produce dissimilar hashes. The visual hash codes proposed in [Fri00, Fri99] for digital watermarking, satisfy this requirement[1]. In brief, generating the visual hash involves projecting the image on to a number of randomly generated low-frequency patterns, and thresholding the projections, resulting in a 0 or 1 for each projection.

The remainder of this report is organized as follows: We explain the method of generating the visual hash in Section 3. This is directly adapted from [Fri99]. We then describe the motion estimation and rate control applications in detail in Sections 4 and 5.

3 Generation of the Visual Hash

The visual hash used in this project is based on the principle that similar looking images have similar low frequency components. For instance, in JPEG compression, two similar looking 8×8 blocks would have approximately equal low-frequency DCT coefficients. In other words, if a low-frequency DCT coefficient is drastically altered, then the image is also drastically altered. Just as DCT coefficients of an image block are obtained by projecting the block on to a set of DCT basis images, we will obtain projections of an image block on to a set of randomly generated low frequency patterns. Upon thresholding these projections, we obtain a binary string which is our visual hash. By the above argument about low-frequency coefficients, similar looking image blocks will have a similar set of projections, and similar hashes. Altering an image block slightly may slightly alter its hash. Note that this is different from some cryptographic hashes, in which even a slight alteration of the image block would result in a completely different hash; a signal that the image has been tampered with. The process of generating the hash is presented below. The details are taken from [Fri00], where the hash was first used to watermark an image in such a way

[1] For this application, we are mainly concerned with ability of the hash to provide a robust (in the above sense) and low bitrate signature of an image. Indeed, the security aspect, which is vital for a watermarking scheme, is of no consequence here!

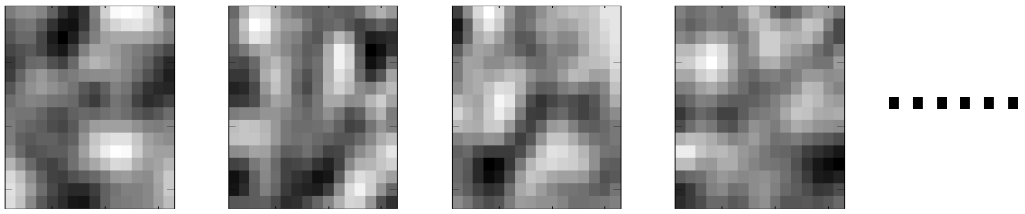


Figure 2: Low frequency patterns are generated by repeated averaging on random matrices. The hash is generated by projecting the image block on to these patterns.

that the watermark is preserved or only slightly altered when the image is subjected to moderately severe degradations. We will use the hash for motion estimation.

Consider the process of generating a N bit hash for an image block I of size $B \times B$ pixels.

1. Form matrices W_i, $i = 1, 2, ..., N$ each of size $B \times B$ pixels with elements $W_i(k, l) \sim U[0, 1]$ $\forall\ i, k, l$.

2. Using simple repeated 2×2 pixel averaging on the matrices W_i, obtain low-pass patterns L_i. These low-pass patterns are shown in Fig. 2.

3. Write out the matrices, I, L_i as vectors. Compute the inner-products $P_i = I^T L_i$, $i = 1, 2, ..., N$. P_i are thus the projections of the image block on to the low frequency patterns L_i.

4. Threshold the projections to obtain bits b_i, $i = 1, 2, ..., N$ according to the rule $b_i = 1$ if $|P_i| < Th$, and $b_i = 0$ else. The threshold is chosen such that 50% of the projections are above Th in absolute value and the remaining 50% are below Th in absolute value.

4 Hash-Aided Motion Estimation

In order to facilitate motion estimation and compensation at the decoder, the encoder transmits the N-bit hash of a block of size $B \times B$ from the current frame to the decoder. For this project, $B = 8$ or 16. The decoder thus has the decoded previous frame and the hash codes of the $B \times B$ blocks in the current frame. It then performs block matching with hashes, as shown in Fig. 3. For block matching, it finds the hash corresponding to each block in the decoded previous frame within the allowable search range. The motion compensated estimate is then decided to be a candidate block A in the previous frame such that the hamming distance between the hash code of the block A and the hash code received from the encoder is minimized over the search range. This procedure is repeated for all blocks in the current frame. The resulting motion compensated frame is a more accurate estimate of the current frame than simply using the previous frame as an estimate of the current frame. Fig. 4 plots the PSNR between the current frame and the motion compensated prediction versus the bit-rate of the helper information, i.e., the bit-rate expended in transmitting the hash to the decoder. The zero rate point corresponds to the case of no helper information, where the previous frame is not motion compensated at all. To see how hash-aided motion estimation improves the side information, consider the sample error frame of the Foreman sequence shown in Fig. 5. The left half of the figure shows the error between the current frame and the previous frame without any motion compensation. The right half shows the error between the current frame and its estimate obtained by hash-aided motion estimation.

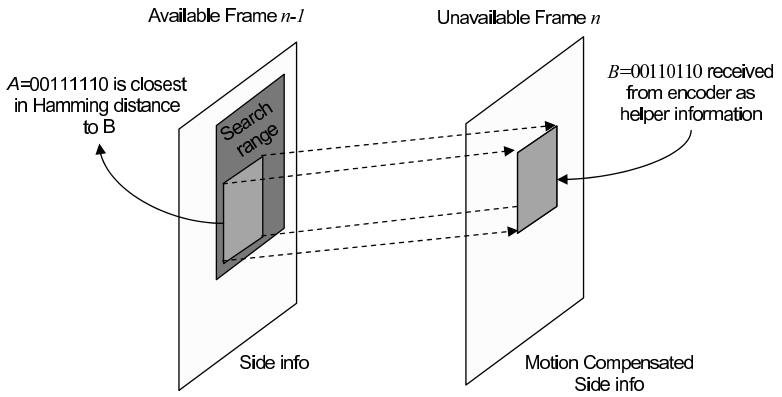


Figure 3: Hash-aided motion estimation at the decoder essentially performs block matching with hashes.

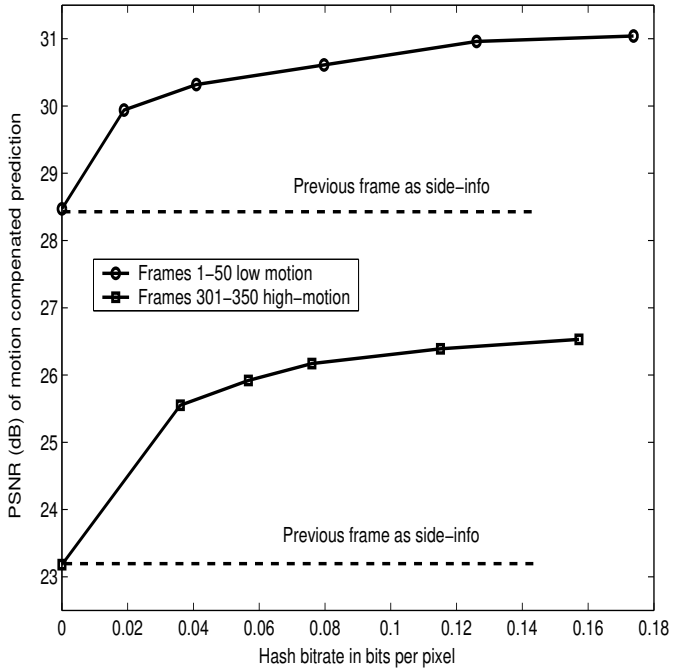


Figure 4: PSNR of the estimate of the current frame increases when helper information in the form of a visual hash code is transmitted to the decoder in the form at low bit-rate. These results are obtained with two 50-frame sub-sequences of the *Foreman QCIF* sequence. The two sub-sequences were chosen such that one (Frames 0-50) had a very little global motion while the other (Frames 300-350) had a large camera pan.

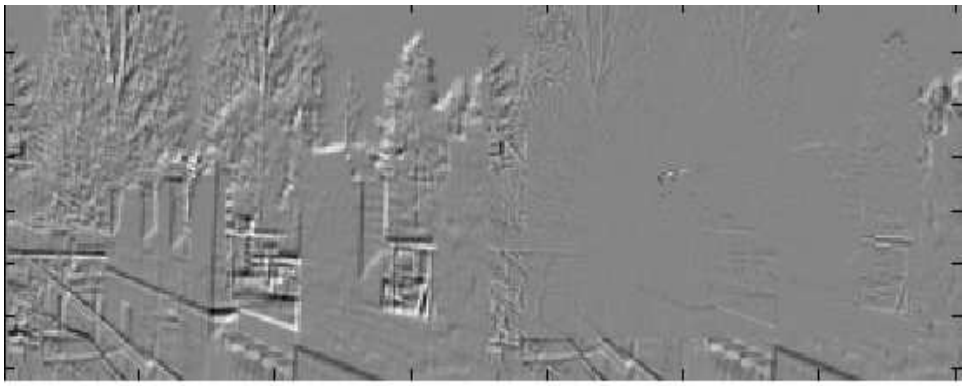


Figure 5: Sample error frame from the *Foreman QCIF* sequence. Hash-aided motion estimation provides a more accurate estimate of the current frame than that obtained by just using the previous frame as side information. This result is obtained by transmitting a 16 bit hash applied to each 16×16 pixel block of the video frame, resulting in a hash bit-rate of about 0.12 bits per pixel. Note that if a blocks hash is the same as that of its co-located block in the previous frame, then it is not transmitted. Instead, a single bit indicating "No hash necessary" is transmitted.

We make the following observations from Fig 4.

1. Hash codes transmitted at a very low bit-rate significantly improve the quality of the side-information. It is expected that the slight increase in transmission rate due to the helper information will be more than offset by the reduction in the Wyner-Ziv encoding bit-rate needed to correct errors in the hypothetical dependence channel between the current frame at the encoder and its estimate at the decoder. Measurement of the Wyner-Ziv encoding rates is outside the scope of the current project and hence is not reported here, but is part of the author's current research.

2. The improvement in the PSNR of the motion compensated prediction is more pronounced for video sequences which have more motion.

3. Most of the PSNR improvement is obtained at a small hash bit-rate, i.e., with a small number of random low frequency patterns. Eventually, when the number of bits in the hash, N is increased, the new low frequency patterns are very similar to one or more already existing patterns, and provide no new information about the image block. Therefore the PSNR improvement tapers off at higher bit-rates.

5 Hash-Aided Rate Control

The bit-rate of the Wyner-Ziv encoder depends upon the accuracy with which the motion compensated side information approximates the current frame. The better the approximation, the smaller is the Wyner-Ziv bit-rate required to correct the errors in the hypothetical dependence channel between the current frame and its estimate. Determining this required rate at the encoder is a problem because the encoder does not have access to the decoder's estimate of the current frame. We propose to use the motion estimation algorithm discussed above, for this purpose. Note that, in performing block matching with hashes, the decoder performs a number of computationally

trivial hamming distance operations. Thus hash-aided motion estimation has far smaller complexity than conventional block matching motion estimation which needs to calculate a large number of S.A.Ds (sum of absolute differences) or squared errors. Therefore, we propose to replicate the hash-aided motion estimation at the encoder to obtain an motion compensated estimate of the current frame. This can be used to control the Wyner-Ziv encoder's bit-rate as follows:

1. Keep the hashes corresponding to all possible blocks in the previous frame in a hash store. Alternatively, just keep a memory store comprising of one previous frame.
2. Find the hash of the current macroblock and perform block matching with the hashes of all blocks in a limited search range in the previous frame.
3. Record the MSE between the actual current frame and its motion compensated prediction at the encoder. To reduce the complexity, the motion estimation at the encoder can have a reduced search range, or lower accuracy (integer pel as opposed to fractional pel) compared with that at the decoder.
4. Use the trend of MSE to determine the Wyner-Ziv encoder bit-rate. Alternatively, one can also record the number of hash-bits transmitted for a given frame and use the trend in the hash bit-rate to determine the Wyner-Ziv encoder bit-rate. For frames having little or no motion, the hamming distance between the hashes of co-located macroblocks is zero. In this case, the hash is not sent at all. Clearly, the larger the number of blocks for which no hash is sent, the more similar the previous frame is to the current frame, and the smaller the required Wyner-Ziv encoding bit-rate. (Note: In this project we merely suggest that this trend information can be used for rate control. But the distributed video encoding portion is outside the scope of this project, and we have not implemented any rate control algorithm at this stage.).

The observed trends of the MSE, the hash bitrate and the Wyner-Ziv bitrate are very similar, as shown in Fig. 6 for the first 50 frames of the Foreman QCIF sequence. The Wyner-Ziv bit-rate readings were obtained from Anne Aaron with the codec described in [Aar04b], using the decoder side-information generated by hash-aided motion estimation described in Section 4. i.e., These readings show the Wyner-Ziv bit-rate that is necessary to correctly decode the current frame, when the previous frame has been motion compensated using hash-aided motion estimation. A clearer picture of the correlation between the MSE of motion compensated prediction and the Wyner-Ziv bitrate is obtained from the scatter plots in Fig. 7. The graphs clearly suggest that the Wyner-Ziv bit-rate can be reliably determined by observing the MSE of the motion compensated prediction.

6 Conclusions

In this project, we applied robust visual hash codes for motion estimation at the decoder and rate control at the encoder, to improve the overall performance of a distributed video codec. By sending the hash of each image-block to the decoder at very low bit-rate, we improved the decoder's estimate of the current frame by about 2.5-3.3 dB. This directly translates to a large saving in the Wyner-Ziv encoding rate needed to reconstruct the current video frame from the decoder's estimate. As expected, the improvements in PSNR are larger for sequences with higher motion than

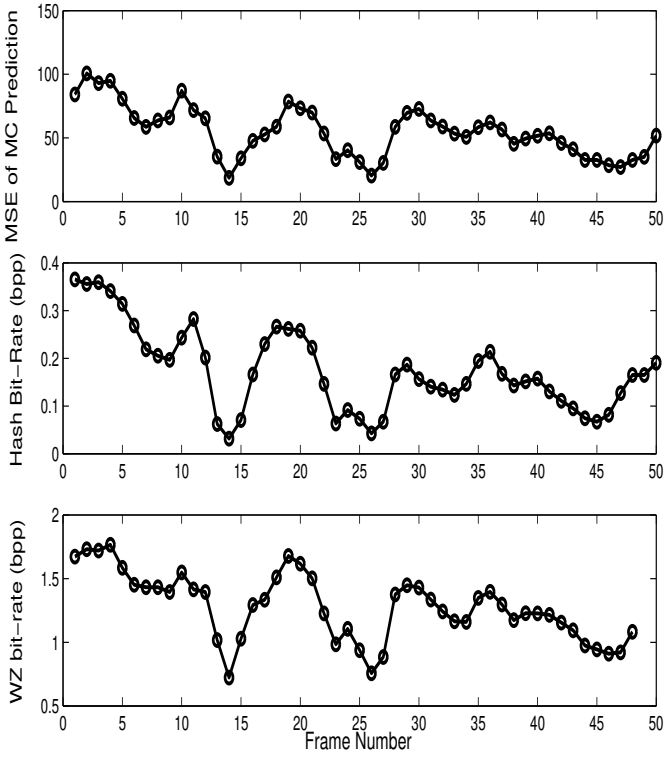
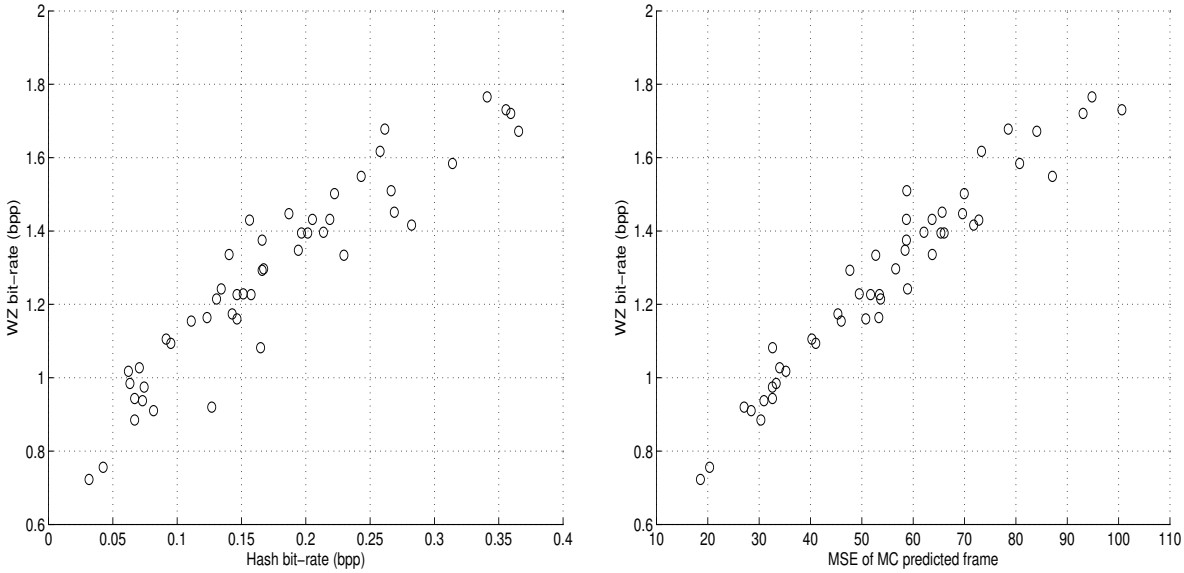


Figure 6: Similar trends are observed for the MSE of motion compensated prediction, the hash bit-rate and the Wyner-Ziv bit-rate required to reconstruct the current frame from the decoder's estimate of it. These results are for the frames 1-50 of the *Foreman QCIF* sequence.



(a) Hash bit-rate vs. WZ bit-rate

(b) MSE vs. WZ bit-rate

Figure 7: Scatter plots show that the Wyner-Ziv bit-rate is highly correlated with the MSE between the current frame and its motion compensated prediction. It is also correlated with the Hash bit-rate. The 50 points on the scatter plots correspond to the first 50 frames of the *Foreman QCIF* sequence.

for videos of nearly stationary scenes. Since hash-aided motion estimation can be implemented at low complexity, we proposed to use a coarse hash-aided motion estimation algorithm at the encoder. This gives the encoder access to an approximate copy of the estimate of the current frame which will be produced at the decoder, and hence allows the encoder to control the Wyner-Ziv bit-rate. In other words, the encoder can now coarsely "sense" the dependence channel between the current frame and its decoder-based estimate. Our initial experiments demonstrate that there is a strong correlation between the mean squared error between the current frame and its motion compensated estimate and the Wyner-Ziv bit-rate required to correctly reconstruct the current frame from the decoder's estimate of the current frame.

7 Acknowledgment

The author would like to thank Prof. Bernd Girod for suggesting the problem, and Prof. Apostolopoulos and Anne Aaron for project discussions during the course. Anne also provided the bit-rate per frame data included in Fig. 6.

References

[Wyn76] A. Wyner, and J. Ziv, "The Rate-Distortion Function for Source Coding With Side Information at the Decoder, *IEEE Trans. Info. Theory*, vol. IT-22, pp.1-10, Jan 1976.

[Aar04a] A. Aaron, S. Rane, and B. Girod, "Wyner-Ziv video coding with hash-aided motion compensation at the receiver", *Submitted to Proc. IEEE Intl. Conf. Image Proc., (ICIP 2004),* Singapore.

[Aar04b] A. Aaron, S. Rane, and B. Girod, "Transform-domain Wyner-Ziv codec for video", *Proc. SPIE Visual Comm. and Image Proc. (VCIP 2004)*, San Jose, CA, Jan 2004.

[Pur02] R. Puri, and K. Ramchandran, "PRISM - A robust video coding architecture based on distributed coding principles", *40th Allerton Conf. Comm*, Allerton, IL, Oct 2002.

[Fri99] J. Fridrich, "Robust bit extraction from images", *Proc. IEEE ICMCS'99* , Florence, Italy, Vol. 2, pp. 536-540, June 7-11, 1999, pp. 536-540.

[Fri00] J. Fridrich, "Visual hash for oblivious watermarking", *Proc. SPIE, Electronic Imaging-Security and Watermarking of Multimedia Contents*, San Jose, California, January 24-26, 2000, pp. 286-294.