IMAGE COMPRESSION IN A MULTI-CAMERA SYSTEM BASED ON A DISTRIBUTED SOURCE CODING APPROACH

*G. Toffetti†, M. Tagliasacchi†, M. Marcon†, S. Tubaro†, A. Sarti†, K. Ramchandran**

†Dipartimento di Elettronica e Informazione,
Politecnico di Milano,
P.zza Leonardo da Vinci, 32
20133 Milano, Italy
phone: + (39) 2399 3647, fax: + (39) 2399 3413
email: {toffetti, tagliasacchi, tubaro, sarti}@elet.polimi.it

* EECS Department.,
UC Berkeley
269 Cory Hall Berkeley, CA 94720
phone: 510-642-2353
email: kannanr@eecs.berkeley.edu

ABSTRACT

This paper illustrates an algorithm specifically designed for encoding multiple views of the same scene taken from calibrated cameras. The assumption here is that these views are strongly correlated as they represent the same content viewed from different perspectives. In order to keep the encoding complexity low, the proposed algorithm builds on PRISM (Power-efficient, Robust, hIgh compression, Syndrome-based Multimedia coding), a video coding framework based on distributed source coding principles. The encoder is constrained to perform a very cheap coarse 3D reconstruction of the scene, whereas the decoder has access to the best 3D reconstruction to be used as side information. Preliminary results on synthetic objects demonstrate that it is possible to achieve a coding efficiency gain with respect to INTRA coding at a low encoding complexity.

1. INTRODUCTION

Traditional image and video coding are usually limited to compress one source at a time, representing the scene from a single viewpoint. With the proliferation of cheap acquisition devices it is easier to take simultaneously several looks at the same scene from different angles. At the receiving end, these views can be blended together to form a 3D reconstruction of the scene. Recent results [1] show that a 3D TV might be a reality in the near future. An interesting application scenario is characterized by cameras deployed as a sensor network, consisting of a large number of sensing devices that are low-power and with wireless communication capabilities. Although the potential applications of sensor networks are manyfold, ranging from home security to environment control, traffic monitoring and more, in this paper we concentrate on camera sensor networks. Each node, equipped with a digital camera, views the scene from a different perspective and it communicates the sensed images to the central node. As we are assuming that the communication medium is wireless, without loss of generality we can state that the neighboring nodes can listen to what is being transmitted to the central node thus allowing some sort of exchange of information among distributed nodes. In order to achieve a good coding efficiency, it is mandatory to take advantage of the geometrical correlation among the multiple views. Figure 1 shows an example of how the cameras can be deployed. We want to encode the view taken from camera X exploiting the views from the neighboring cameras A, B, and C. If the encoder is not power constrained, a viable solution consists in communicating A, B and C to node X and to perform a 3D reconstruction of viewpoint X using this information to form the best predictor Y. The encoder transmits the prediction residue $N = X - Y$, thus saving bits with respect to INTRA coding. As the sensing cameras usually have limited power, we want to move the complexity of the costly 3D virtual reconstruction to the decoder side, located at the central node. In order to achieve this we exploit the ideas of PRISM [2][3], a video coding framework built on distributed source coding principles. PRISM is able to flexibly distribute the complexity by moving most or all of the motion estimation task from the encoder to the decoder. In this paper we take a similar approach, where motion estimation is replaced by 3D rendering. Briefly, the encoder at node X receives A, B and C and builds a low-cost coarse 3D reconstruction of the view, Y_c. Based on this information the encoder tries to infer the correlation with the side information that will be available at the decoder, thus deciding the bit allocation. The decoder has access to A, B and C and not being power constrained can build the best 3D reconstruction Y. Y is used as a side information to decode the view X. Section 4 elaborates on this topic giving further details on the way the correlation is estimated and how the side information can be effectively used.

The problem of coding multiple views using a distributed source coding approach has been recently explored in the literature. In [4] communication among cameras is not allowed but some prior information about the geometry of the cameras and the distance of the objects is required. This work elaborates a strategy for efficiently encoding the positions of the objects given that only the decoder will have access to the other views. The work in [5] is related to the algorithm proposed in this paper but the encoding process heavily relies on the Wyner-Ziv codec presented in [6]. Low-encoding complexity is achieved without communication among cameras but a feedback channel is needed.

The rest of this paper is organized as follows. Section 2 reviews the rendering algorithm used to build the 3D virtual reconstruction. Section 3 briefly illustrates the basic ideas of PRISM while Section 4 details the proposed algorithm. Preliminary experimental results on synthetic images are given in Section 5.

2. 3D RENDERING ALGORITHM

In this section we briefly review the rendering method [7] used in our coding scheme. The algorithm receives in input three (or more) images from calibrated cameras. One camera is chosen as the preferred one and a depth map is estimated, indicating for each pixel of the image the depth of the object. For a given depth map f, a cost function is defined as:

$$E(f) = E_{data}(f) + E_{smooth}(f) \quad (1)$$

The authors wish to acknowledge the support provided by the European Network of Excellence VISNET (http://www.visnetnoe.org)

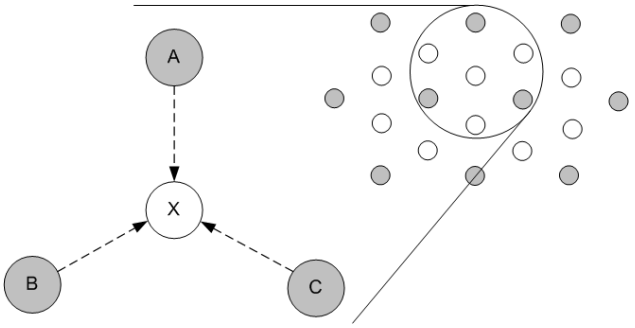

Figure 1: Camera sensor network. White circles represent images to be INTRA coded while grey circles are images encoded with the proposed algorithm.

Our data term is defined as a non-positive value which results from the differences in intensity between corresponding pixels. It is computed for every pixel p of the preferred image (we indicate this image with the index j) by these steps:

1. from p, we get the corresponding 3D-point by retro-projecting it from the preferred camera center of projection with the selected depth and then we project this 3D-point on each other calibrated images obtaining a set of $n-1$ corresponding pixels $q_1, q_2, ..., q_i, ...q_n | i \neq j$
2. on every non-preferred image we compute the SSD (Sum of Square Difference) using a square window centered on q_i and one centered on p, obtaining the set of values $d_1, d_2, ..., d_i, ...d_n | i \neq j$
3. finally

$$E_{data}(f) = min(0, \sum_{p \in i=ji \neq j} d_i - K) \qquad (2)$$

where K is a positive constant large enough to capture significant variation of the SSD function (a typical value is $K = 30$).

The smoothness term is quite similar to the one used in [8] and its goal is to make neighboring pixels in the preferred image tend to have similar depths. In order to minimize the multi-variate cost function $E(f)$ we use an approach based on graph cuts [8]. This methods are fast enough to be practical, but unlike simulated annealing, graph cuts methods cannot be applied to arbitrary functions. We use some recent results [9] that give graph constructions for a quite general class of energy functions. The optimal depth map is then filtered to remove outliers and to reduce the quantization noise. Finally, in order to create a complete model of the object we blend together the several surface patches obtained with the previous graph cuts based method. Further details can be found in [7]. Figure 2 shows as an example the views taken from the four calibrated cameras. Figure 3 depicts the rendered views using each of the three surrounding cameras as the preferred one. We can notice that depending on the chosen reference the quality of the reconstruction in each region varies. In both cases the central image is the one we are encoding.

3. BACKGROUND ON PRISM

The PRISM video coder is based on a modification of the source coding with side-information paradigm, where there is inherent uncertainty in the state of nature characterizing the side information. The Wyner-Ziv Theorem [10] deals with the problem of source coding with side-information. The encoder needs to compress a source X when the decoder has access to a source Y. X and Y are correlated sources and Y

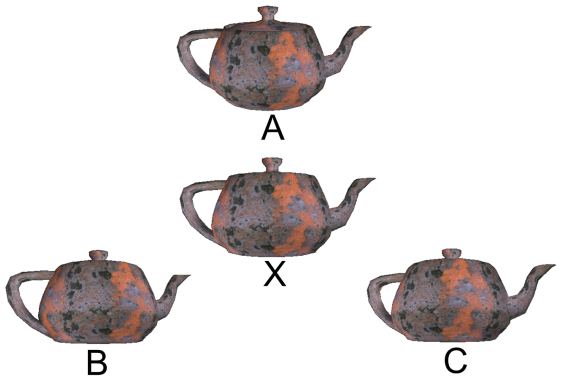

Figure 2: Original views taken from viewpoints A, B, C and X. The latter is encoded with the proposed algorithm.

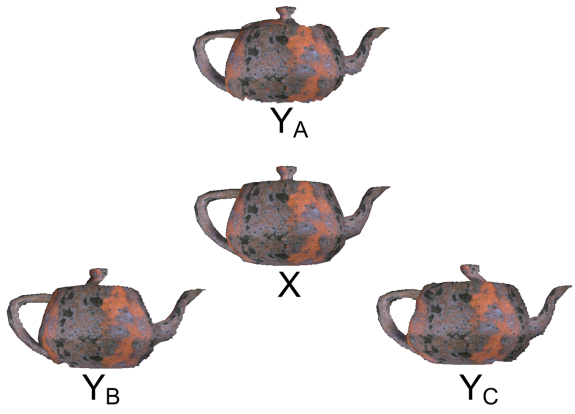

Figure 3: Rendered images Y_A, Y_B and Y_C from viewpoint X obtained using A, B and C as preferred image in the rendering process.

is available only at the decoder. From information theory we know that for the MSE distortion measure and $X = Y + N$ where N has a Gaussian distribution, the rate - distortion performance for coding X is the same whether or not the encoder has access to Y.

For the problem of source coding with side information, the encoder needs to encode the source within a distortion constraint, while the decoder needs to be able to decode the encoded codeword subject to the correlation noise (between the source and the side-information). While the results of Wyner and Ziv are non-constructive and asymptotic in nature, a number of constructive methods to solve this problem have since been proposed (such as in [11][12][6]) wherein the source codebook is partitioned into cosets of a channel code.

For the PRISM video coder [2], the video frame to be encoded is first divided into non-overlapping spatial blocks of size 8x8. The source $\mathbf{X}$ is the current block to be encoded. The side-information $\mathbf{Y}$ is the best (motion-compensated) predictor for $\mathbf{X}$ in the previous frame and let $\mathbf{X} = \mathbf{Y} + \mathbf{N}$. We first encode $\mathbf{X}$ in the intra-coding mode to come up with the quantized codeword for $\mathbf{X}$. Now, we do the syndrome encoding, i.e., we find a channel code that is matched to the "correlation noise" $\mathbf{N}$, and use that to partition the source codebook into cosets of that channel code. The encoder transmits the syndrome (indicating the coset for $\mathbf{X}$) and a CRC check of the quantized sequence. In contrast to MPEG, H.26x, etc., it is the decoder's task to do motion search, as it searches over the space of candidate predictors one-by-one

to decode a sequence from the set labeled by the syndrome. When the decoded sequence matches the CRC check, decoding is declared to be successful. For further details please refer to [3].

4. PROPOSED ALGORITHM

We refer in the following to the camera configuration of Figure 1. Cameras A, B and C as well as all the other cameras marked in gray encode images in INTRA mode, i.e. without reference to other cameras. Cameras marked in white use the proposed algorithm based on distributed source coding. We focus on camera X. At the encoder A, B and C encode the respective views and send to the central node the quantized versions $\dot{A}$, $\dot{B}$ and $\dot{C}$. Node X listens to what is being transmitted and uses these three views to build a coarse virtual reconstruction of X, Y_c, using the rendering algorithm described in Section 2. In order to reduce the computational complexity at the encoder, the rendering of the virtual view is carried out at reduced resolution. In our implementation we downsample the received images by a factor of four in both direction before feeding them to the rendering algorithm. The encoder processes the image on a block-by-block basis. For each block, the encoder checks the correlation existing with the co-located block in the coarse virtual reconstruction Y_c, computing $MSE_c = \sum_i |X^i - Y_c^i|^2$. Based on this measure, the encoder tries to infer the correlation that will be observed at the decoder $N = X - Y$, where Y represents the best predictor that can be found at the decoder side. The decoder is free from power constraints and can thus use all the information available to build the best side information. The rendering algorithm we are using gives different results based on the image that is chosen as reference. This is due to the fact that it adds a depth measure only to the locations visible from the image chosen as the reference. In order to get a better side information, we can actually perform three different 3D reconstructions, using respectively A, B and C as reference, producing in output Y_A, Y_B and Y_C as shown in Figure 3. The decoder is able to use as a predictor any of these images. It is possible to note that depending on the spatial region of interest, we would pick the one that gives the best predictor. Moreover, the rendering algorithm is not able to produce a result that has a precise reconstruction at the pixel level. This might not be an issue in computer vision applications, but it is not satisfactory when it is used to build a predictor for encoding X. For this reason we allow the decoder to perform a motion search in small range about the co-located block in each of the rendered views Y_A, Y_B and Y_C, in order to increase the correlation of the encoded image with its side information. The missing link between the encoder and the decoder is assured by an off-line module that collects correlation noise statistics defining a mapping between what is observed at the encoder and the best predictor disclosed at the decoder. The proposed algorithm inherits most of the coding tools of PRISM. While PRISM computes at the encoder the MSE at zero motion (co-located block in the reference frame), we calculate here the MSE between the block and the co-located block in the coarse rendered view Y_c. On the other hand, at the decoder PRISM searches for the best prediction in the reference frame, whereas our algorithm searches in the three high quality rendered views Y_A, Y_B and Y_C. Let us summarize the steps carried out by our algorithm:

Encoder:

- receive the images $\dot{A}$, $\dot{B}$ and $\dot{C}$
- compute a coarse low-resolution rendering of X, Y_c using $\dot{A}$, $\dot{B}$ and $\dot{C}$
- for each block
  - compute the MSE between X and the co-located block in Y_c
  - compute the DCT transform of block X
  - INTRA encode block X
  - read the statistics collected by the classifier to infer the correlation noise and to perform bit allocation
  - send syndrome bits and the CRC signature of the quantized block

Decoder:

- receive the images $\dot{A}$, $\dot{B}$ and $\dot{C}$
- compute three high-quality high-resolution reconstructions of X, Y_A, Y_B and Y_C using $\dot{A}$, $\dot{B}$ and $\dot{C}$
- for each block
  - read syndrome bits and CRC
  - perform a motion search around the co-located blocks in Y_A, Y_B and Y_C
  - when the CRC of the decoded block matches with the CRC sent by the encoder, a decoding success is declared and motion search is stopped (see Section 3)
  - recover the reconstructed block by IDCT

The decoder continues the motion search until it finds a predictor whose correlation with the block to be decoded is below the noise margin for which the channel code was designed (see Section 3). When this happens a decoding success is declared. If the motion search concludes without a match, decoding fails and a simple error concealment technique is employed by pasting the co-located block of one of the three rendered views. In order to increase the decoding speed, predictors are visited in such a way that a decoding success occurs as soon as possible. A spiral search searches for predictors in each of the images Y_A, Y_B and Y_C starting from zero motion (co-located blocks). The same candidate motion vector is tested for all the three images before proceeding to the next predictor.

5. EXPERIMENTAL RESULTS

We performed some preliminary tests on synthetic objects. Both *teapot* and *pitbull* have been modeled by a 3D software and several snapshots have been rendered from it. At the encoder the rendering algorithm is fed with images having one sixteenth of the original resolution, resulting in a speed-up of a factor of 100 with respect to the full quality reconstruction performed at the decoder side. We compared the rate-distortion performance of the proposed algorithm with INTRA coding. For the latter we used the H.263+ INTRA mode as a benchmark. Figures 4 and 5 shows the reconstructed image quality as a function of bit-rate for *teapot* and *pitbull*. The numbers indicated in the plot refer to the quantization parameter (QP) used (the actual quantization step is $2 \cdot QP$). At low bitrates the proposed algorithm outperforms INTRA coding whereas at high bitrates there is a PSNR penalty. This is due to the fact that a couple of blocks are incorrectly decoded and the error concealment technique pastes the co-located block from Y_A. This PSNR drop does not reflect the perceived quality though. At the same quantization step size, reflecting the subjective quality better than PSNR in this case, we observe a bit saving of around 20%. Figures 6 and 7 show the decoded *teapot* and *pitbull* images at $QP = 8$. Although we haven't run specific simulations in this direction, we expect to be able to exploit the robustness features of PRISM. Let us say for example that A is not available at the decoder. The decoder can pick another neighboring camera and still perform a 3D rendering of X. If the new side information is still within the noise margin the decoder will still be able to decode. This is because PRISM is not tied to a single deterministic predictor but it encodes for the statistical correlation between the block and its side information.

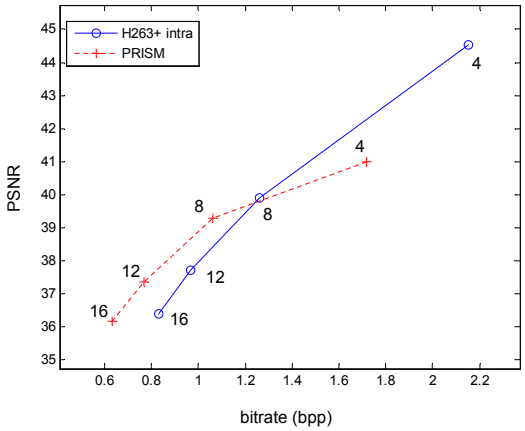

Figure 4: Teapot - PSNR vs. bitrate

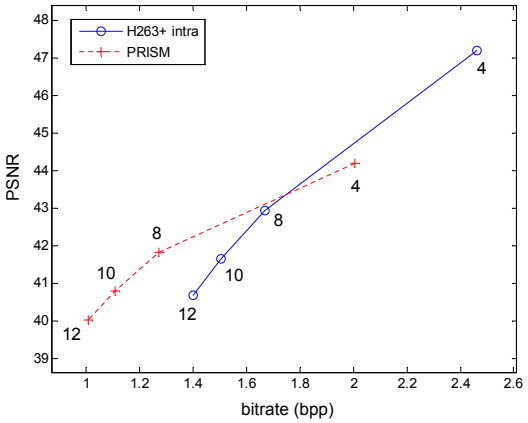

Figure 5: Pitbull - PSNR vs. bitrate

6. CONCLUSIONS

We proposed a coding algorithm for multi-camera views based on distributed source coding. Preliminary results show promising results with respect to INTRA coding. We are currently investigating the use of other parameters other than MSE_c (i.e. estimated surface angle, local smoothness, etc.) to drive the bit allocation at the encoder. Moreover we think that this approach can be extended to multi-camera video, in such a way that both temporal (from past frames) and spatial (from rendered views) predictors can be used at the decoder.

REFERENCES

[1] Anthony Vetro, Wojciech Matusik, Hanspeter Pfister, and Jun Xin, "Coding approaches for end-to-end 3D TV systems," in *Picture Coding Symposium*, San Francisco, CA, December 2004.

[2] Rohit Puri and Kannan Ramchandran, "PRISM: A New Robust Video Coding Architecture based on Distributed Compression Principles," in *Allerton Conference on Communication, Control and Computing*, Urbana-Champaign, IL, October 2002.

[3] Rohit Puri and Kannan Ramchandran, "PRISM: A video coding architecture based on distributed compression principles," Tech. Rep. No. UCB/ERL M03/6, ERL, UC Berkeley, March 2003.

Figure 6: Teapot. Decoded image (QP = 8)

Figure 7: Pitbull. Decoded image (QP = 8)

[4] Nicolas Gehrig and Pier Luigi Dragotti, "On distributed compression in dense camera sensor networks," in *Picture Coding Symposium*, San Francisco, CA, December 2004.

[5] Anne Aaron, P. Ramanathan, and Bernd Girod, "Wyner-Ziv coding of light fields for random access," in *Proceedings of the IEEE Workshop on Multimedia Signal Processing*, Siena, Italy, September 2004.

[6] Anne Aaron and Bernd Girod, "Compression with side information using turbo codes," in *Proceedings of the IEEE Data Compression Conference*, Snowbird, UT, April 2002.

[7] Gianluca Dainese, Marco Marcon, Augusto Sarti, and Stefano Tubaro, "Complete object modeling using a volumetric approach for mesh fusion," in *Workshop on Image Analysis for Multimedia Interactive Services 2005*, Montreux, Switzerland, March 2005.

[8] V. Kolmogorov and R. Zabih, "Multi-camera scene reconstruction via graph cuts," in *European Conference on Computer Vision*, Copenhagen, Danmark, May 2002.

[9] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?," in *European Conference on Computer Vision*, Copenhagen, Danmark, May 2002.

[10] Aaron D. Wyner and Jacob Ziv, "The rate distortion function for source coding with side information at the decoder," *IEEE Transactions on Information Theory*, vol. 22, pp. 1–10, January 1976.

[11] Sandeep S. Pradhan and Kannan Ramchandran, "Distributed source coding using syndromes (DISCUS): Design and construction," in *Proceedings of the IEEE Data Compression Conference*, Snowbird, UT, March 1999.

[12] Angelos D. Liveris, Zixiang Xiong, and Costas N. Georghiades, "Distributed compression of binary sources using conventional parallel and serial concatenated convolutional codes," in *Proceedings of the IEEE Data Compression Conference*, James A. Storer and Martin Cohn, Eds., Snowbird, UT, March 2003, pp. 193–202.