Optimizing QPs for Multiview Image Coding for Free Viewpoint Video

Vanessa Testoni, Dinei Florêncio and Max Costa

***Resumo*— Teleconferência e outras aplicações de vídeo em 3D têm se tornado de grande interesse. Além de displays estéreo, para obter um efeito de vídeo 3D realístico com ponto de vista livre (FVV) é necessário prover paralaxe de movimento. Uma das formas mais eficientes de sintetizar FVV de imagens naturais é aplicar renderização baseada em imagens, isto é, basear a imagem sintética em vídeos tomados de diferentes posições em relação à cena. Uma nova abordagem para codificação de vídeo *multiview* foi proposta recentemente e mostrou que codificando cada macrobloco de cada imagem com um passo de quantização (QP) apropriado pode-se melhorar a taxa de compressão em até 2X. Os QPs utilizados para obter tal melhoramento são escolhidos baseados em uma regra experimental, que leva em consideração os pesos da contribuição de cada pixel na imagem sintética final. Nesse artigo, deriva-se o mapeamento ótimo dos QPs para um codec semelhante. Mostra-se então que o mapeamento anteriormente proposto é otimo para uma fonte Gaussiana, e indica-se o mapeamento ótimo para uma fonte genérica.**

***Palavras-Chave*— videoconferência, 3D, codificação de vídeo**

***Abstract*— Teleconferencing and other applications of 3D video have become of increasing interest. Besides stereo display, realistic 3D requires also free viewpoint video (FVV), in order to provide adequate motion parallax. One of the most efficient ways of synthesizing FVV of natural scenes is by image based rendering, based on multiview video. A new approach for multiview coding has been recently proposed, and shows that encoding each macroblock of each frame with appropriate quantization parameters (QPs) improves compression by around 2X. The QPs used to acheive such improvement are tuned according to an ad-hoc relation to the weights used in the synthetic view. In this paper we show that the ad-hoc relation is actually optimum for a Gaussian source, and derive the optimum QPs for the generic case.**

***Keywords*— videoconferencing, telepresence, multiview video coding, 3D video.**

I. Introduction

Three dimensional (3D) video has received increased attention lately [1]. In particular, in the entertainment world, 3D movies are, again, the latest fad, 3D games (and stereo monitors) are now widely available, and TV manufacturers are expected to introduce 3D TV as the new consumer electronics revolution. 3D video, clearly, is at an inflection point that will lower costs significantly, and bring stereo display technology to the masses. Most of this revolution is due to new technologies in stereo display, which have introduced reasonable cost, high quality stereo display using shutter glasses, and barrier-based autostereoscopic displays. Of course, improved technology and reduced costs in capture, transmission, and rendering of computer graphics have all also contributed to the recent surge of 3D.

We point out, however, that realistic 3D requires motion parallax, in addition to the stereo parallax current provided by most displays and applications. Providing motion parallax for computer generated images is nearly trivial. Nevertheless, doing the same for natural images is a much more complex problem. When dealing with natural images, each step of the process is harder: capture, rendering, and transmission are all challenging, significant research is still needed in many of these aspects, before commercial systems can be widely deployed. In particular, to provide motion parallax to natural scenes, the most widely used technology is image-based rendering [2], [3], [4], [5]. In this technique, multiple cameras capture the scene, each from a different viewpoint, which are in turn used to synthesize a new image from the desired viewpoint. The larger the number of views, the less artifacts can be expected. Typical systems vary from as low as four cameras to as high as 64 cameras. The high number of cameras implies a high requirement for bandwidth, during acquisition, transmission, storage, and every step of the processing chain.

Providing motion parallax, means adapting the displayed image to the current position of the viewer. This can be done, for example, by head tracking as proposed in [6]. However, for realistic perception, the time lag between the head movement and the display of the new image has to be kept to a minimum. Recent research shows that above a 30-50ms delay, most people can notice the delay, severely impairing the depth perception, and causing discomfort, and possibly nausea or other side effects. To achieve such low delays, the rendering operation needs to be local. That is, the receiver/renderer needs instant access to all the pixels required to synthesize the new view. Most packet networks have a delay which is already above that limit. Thus, if the images have to be transmitted over a network with a delay above just a few milliseconds, this implies encoding several video streams, and sending them to the viewer site for decoding, and generating the synthetic view locally. This leads to the need of multiview video coding.

Multiview video coding encodes several video signals as one stream. There was significant effort in standardizing some of these technologies a few years ago [7], [8]. The basic idea is to exploit the similarity/redundancy between the several views to improve compression. However, the redundancy between views turns out to be similar to the redundancy between frames, and the additional gains obtained using such multiview techniques are usually small. As a matter of fact, gains are mostly obtained by encoding the intra (I) frames in a dependent fashion. Additional compression gains for predicted

The authors are with University of Campinas, Campinas, SP, Brazil, and Microsoft Research, Redmond, WA, USA, E-mails: vtestoni@decom.fee.unicamp.br, max@fee.unicamp.br, dinei@microsoft.com.

(P) and bidirectional (B) frames are usually in the order of 10% or less.

More recently, a new approach for multiview coding was introduced, which exploits the (estimated) viewer position to improve compression [9]. Basically, the authors propose to encode the video stream based on the likelihood that each image is going to be actually used to synthesize the final frame. They also propose to measure the distortion on the synthetic view, instead of on the decompressed videos. With a simple approach, based on varying the macroblock quantization parameter according to estimated weights of each pixel in the synthesized image, the authors show gains on the order of 50% for typical cases. The authors give a heuristic mapping between the pixel weights and the macroblock quantization parameter QP.

In this paper we analyze this choice of mapping. More specifically, the mapping has two ad-hoc stages: a quadratic averaging between pixels to obtain a "mean MB weight", and a logarithmic mapping between this averaging and the QP for the macroblock. We first derive mathematically the optimal mapping between the pixels weights and QP. We show that the proposed mapping in [9] is actually optimum for a Gaussian random variable. However, video signals are not Gaussian. We thus derive optimum results for a typical signal, which is based on the rate-distortion curve for that signal. The optimum way to average the weights within a macroblock is not as treatable. For this, we analyze the results of the quadratic average proposed in [9], and compare to a few other norms.

The rest of the paper is organized as follow: Section 2 briefly details the set up and presents more details about [9]. In Section 3 we derive the optimum mapping between weights and QPs. In Section 4 we discuss the averaging step within the macroblock. In Section 5 we present some experimental results, and in Section 6 we present some conclusions.

II. Multiview Coding using QP based on predicted viewer position

Although multiview video coding may be employed in several applications, we focus here on one particular scenario: providing motion parallax on natural video scenes, captured in real time. A typical application would be 3D video conferencing. Fig. 1 show an overview of the target system. Note that the video transmission over the network has to be outside the synthesis loop, otherwise we would add the full network round trip delay to the lag between the head motion and the corresponding change in the synthetic view.

One of the contributions of [9] is to point out that, since the user only sees the synthesized video, the performance measurement has to be done in that video, instead of directly in the decompressed videos. Fig. 2 illustrates the performance criteria, showing the PSNR comparison to be made between the final, synthesized video, and synthesized video that would be obtained using the uncompressed video frames. An indirect, and undesirable, consequence is that the performance of the compression algorithm is now attached to the algorithm used in synthesizing the video. However, these two problems are, indeed, tightly coupled, so it is only natural that tuning one of them to the other will improve final results.

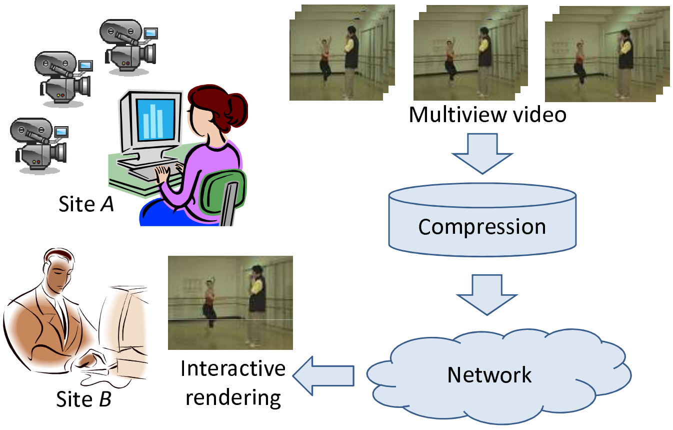

Fig. 1. Immersive tele-conferencing scenario. The system could be symmetrical, i.e. site B could also sent its multiview video to site A.

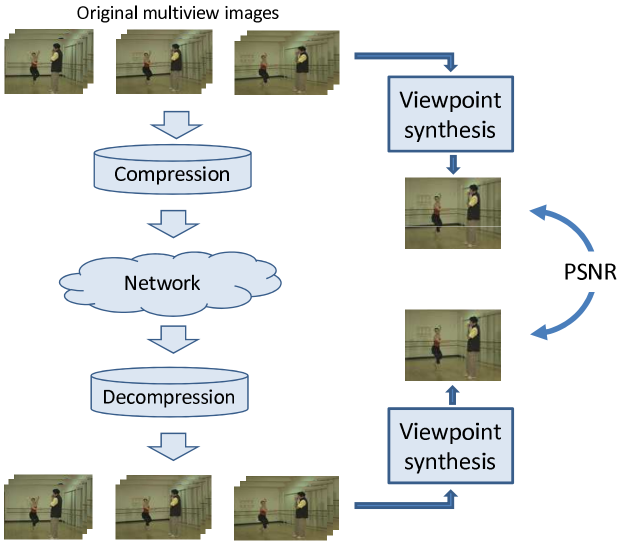

Fig. 2. A more appropriate error criteria: instead of measuring PSNR directly on the decompressed frames, it is more appropriate to measure between the synthetic frames obtained using the decompressed images and the original ones.

A. Video rendering

We use the same view synthesis algorithm employed in [9]. It assumes we have views from a number of cameras, as well as a single depth map to facilitate the virtual view rendering. Fig. 3 shows an example of a dataset for a particular frame. As shown in Fig. 4, given a virtual viewpoint, we first split it to be rendered view into light rays. For each light ray, we trace the light ray to the surface of the depth map, obtain the intersection, and reproject the intersection into nearby cameras (Cam 3 and 4 as shown in Fig. 4). The intensity of the light ray is thus the weighted average of the projected light rays in Cam 3 and Cam 4. The weight can be determined by many factors [4]. In the simplest form, we can use the angular difference between the light ray to be rendered and the light ray being projected, assuming the capturing cameras are at roughly the same distance to the scene objects [8].

Fig. 5 shows one of the images rendered from a virtual view point nearby Cam 2 (slightly rotated view direction). The weight maps clearly demonstrate whether a pixel in a captured video frame will be useful for rendering this particular virtual

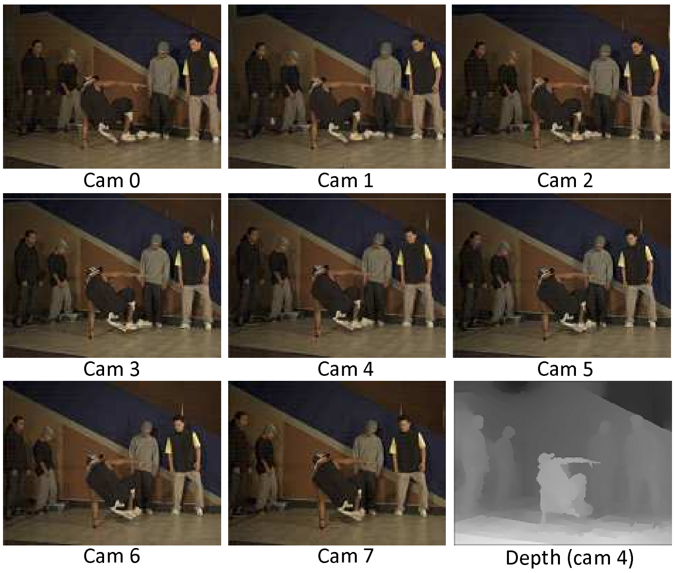

Fig. 3. An example multiview data set.

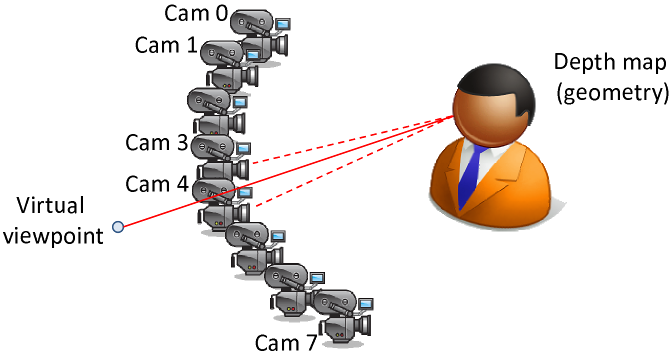

Fig. 4. The rendering process from multiview video.

view. In Fig. 5, brighter pixels are the ones with larger weights. Note that even for Cam 7, which is the farthest from the virtual viewpoint, there are still pixels being used due to occlusions. Naturally, during compression of the multiview video, the pixels with high weights shall be encoded with high quality, while the remaining pixels can be encoded with low quality.

B. Mapping weights to QP

The next step is to map the weights estimated for each pixel into the quantization parameters. The simplest form is to use a macroblock coder, similar to H.264. In [9] the authors propose an ad-hoc mapping. More specifically they use:

$$QP_{mb} = BaseQp - 6log_2 \sqrt{1/256 \sum_{mb} w_i^2} \tag{1}$$

where w_i is the predicted weight for each pixel in the macroblock, and $BaseQp$ is a parameter that controls the overall compression ratio.

Note that the above ad-hoc relation between w_i and QP_{mb} has two main components: the quadratic averaging of the pixel weights within the macroblock, and the logarithmic mapping between that average and QP. In next section we analyze the optimum choice for the mapping between the block weights and QP, and in Section 4 we make some analysis and considerations about the method of averaging the weights within a macroblock.

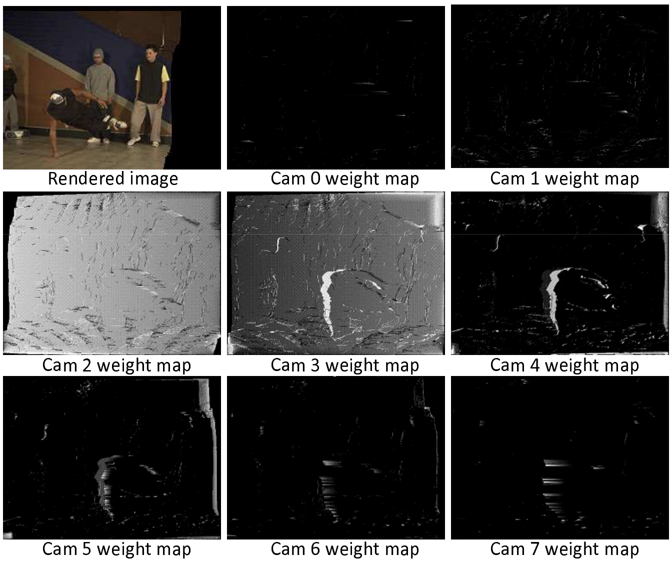

Fig. 5. The weight maps generated by the rendering process.

III. Optimum mapping between weights and QP

We now derive the optimum expression for the optimum QP_{mb}, assuming every pixel in each macroblock of a given frame v has the same weight $w_i = w_{mb}$. Thus, our estimated value $\hat{p_s}$ of a pixel of the desired image can be expressed as a linear combination of the pixels in the captured views, i.e.:

$$\hat{p_s} = \sum_{p_i \in P} w_i p_i \tag{2}$$

where P is the set of pixels p_i's in each of the captured views that contribute to the formation of the estimated pixel. The set P and the weights w_i's are obtained by using the view synthesis algorithm mentioned in Section II-A, and is taken as a black-box for effects of this paper.

The assumption is that each macroblock will be quantized with a quantization step given by the H.264's parameter QP_{mb}. We want to find the set of QP for each MB that minimizes the distortion of the synthetic pixel, p_s. We perform a Lagrangian rate distortion optimization by introducing the Lagrangian variable λ:

$$J(D) = \sum_{mb} R_{mb}(QP_{mb}) + \lambda \sum_{mb} (\hat{p_s} - p_s)^2. \tag{3}$$

Note that $\hat{p_s}$ takes contribution from many pixels, from different cameras. However, if we assume the quantization and residual errors are independent from block to block, the cross terms disappear. We thus take the derivative of the above equation and make it equal to zero, which gives us the optimality condition:

$$\frac{\partial w_{mb}^2 D_{mb}(QP_{mb})}{\partial R_{mb}(QP_{mb})} = constant. \tag{4}$$

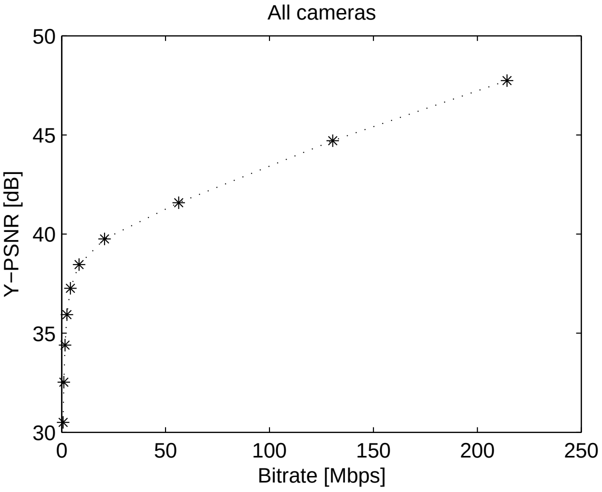

Fig. 6. Distortion x rate curve for a H.264 codec.

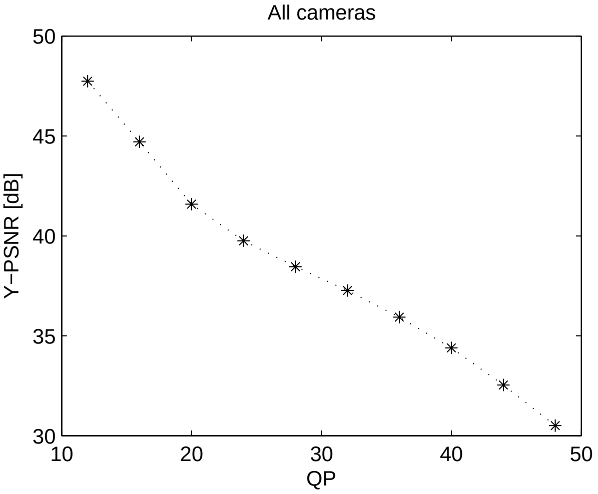

Fig. 7. Distortion x Quantization Parameter curve for a H.264 codec.

In this equation, D_{mb} is the mean square distortion of the macrobolock under consideration, and w_{mb} is the weight that is given for the pixels in the MB when composing the final synthetic view. It indicates that the derivative of the rate distortion curve should be constant across all MBs, which is what one would expect. However, since each MB is multiplied by a different weight, the corresponding RxD curve will be adjusted accordingly.

Equation (4) gives us the intuition behind the optimization, but gives us little clue on how to choose QP. To obtain the optimum QPs, we need to assume something about the rate distortion curve of the signal. Let's assume for now the variable being quantized is a Gaussian variable. For Gaussian distributions, the distortion reduces by 6dB for each increment of 1bit in the rate, or equivalently, each halving of quantization step. Moreover, the quantization step is halved for each reduction of 6 in QP. Therefore,

$$D(QP - 6) = (1/2)D(QP). \tag{5}$$

If we assume every block has the same variance, we can apply the above equation across blocks, and we get:

$$QP_{mb} + 6log_2 w_i^2 = constant. \tag{6}$$

We note that this is exactly the equation used in [9]. Thus, that equation represents the optimum QP attribution if the image was a Gaussian variable. However, since typical images do not fit well a Gaussian distribution, we propose to use a more appropriate RxD curve. Indeed, we computed the RxD curve for the sequence under study, which is plotted in Fig. 6. We then, instead of applying (6), use the rate distortion curve of the particular sequence being transmitted.

We can also plot the distortion as a function of the quantization parameter, which can be more directly applied to the mapping in (6). Fig. 7 shows the mapping between QP and the distortion, again for typical settings in a H.264 codec.

A few points are worth mentioning here. First, although, for simplicity, we use the whole sequence to obtain the RxD curve, performance should be essentially as good if we use only past frames. Second, better fitting of the RxD curve, should further improve results. For example, different curves could be used for I, P and B frames, or, in the extreme, the optimization could be done at the MB level, by computing several encoding possibilities, which is one of the options in many H.264 encoders.

IV. Averaging the weights

The derivation in the previous section assumed all pixels within the macroblock were used in the final image with the same weight w_{mb}. This is not usually the case, so a natural question is what is the best way to obtain a representative (i.e., average) weight for representing the whole block. A simple and intuitive option is the mean square average, but is this the best option? We now try to answer this question.

The final contribution of each pixel p_i is given by $w_i p_i$, as given in III. If we could assume that each pixel were independent, and had the same contribution to the number of bits required to encode the MB, then we could optimize rate distortion by simply writing the distortion equation for a single pixel, and deriving it in relation to the rate, as we did between MBs. However, herein the pixels are highly correlated, and we cannot set the cross terms to zero. In [9] the quadratic averaging of the weights is used as representative for the macroblock, i.e.,

$$\hat{w_{mb}} = \sqrt{1/256 \sum_{mb} w_i^2}. \tag{7}$$

So, instead of deriving a complex mathematical model, we decided to verify the choice of norm in the above averaging. More specifically, we generalize the above mean square norm averaging to the L-norm:

$$\hat{w_{mb}} = (1/256 \sum_{mb} w_i^L)^{1/L} \tag{8}$$

and experimentally compare the choices of norm $L = 2$, 4 and 8. The results are discussed in next section.

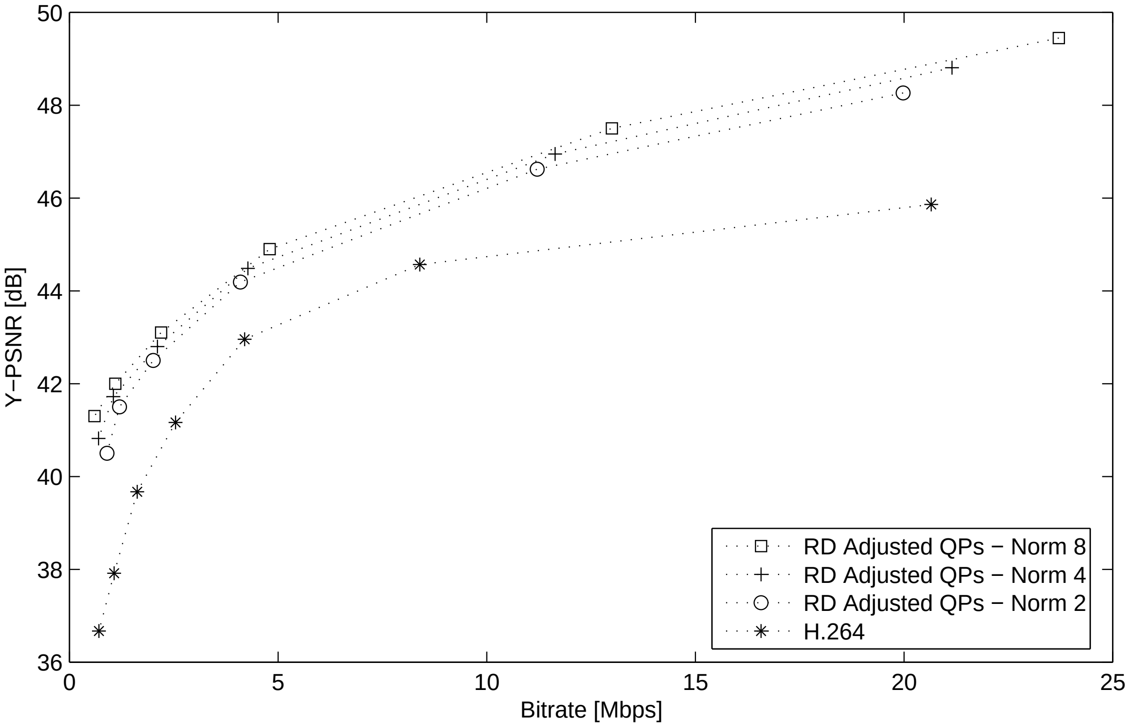

Fig. 8. PSNR results. The number refers to the PSNR between the synthetic image using uncompressed frames, and the synthetic images using each of the compression methods. The lower curve is traditional H.264. The upper curves refer to the optimized QP, when using norms of 2, 4, and 8 to average the weights withing a macroblock. The best results were obtained using L=8.

V. Results

We run experiments with the optimum mapping for the QP for the Breakdance sequence, for a few different rates. Using the same approach as in [9], we provide the results for the PSNR between the image synthesized using the compressed frames, and those synthesized using the uncompressed frames. Fig. 8 shows a plot of the results. We note that the improvement over the traditional H.264 is on the order of 2dB, or, equivalently, we can save about 50% of the rate. Early comparison with the non-optimized QP mapping do not show significant difference. One possible explanation is that the larger contribution to the rate is introduced by the few macroblocks where the weights are high, and thus, the influence of the blocks quantized with higher QP is small. Another possibility is the non-optimized averaging of weights within a macroblock, which may in turn affect the optimality of the QP mapping.

We note also in Fig. 8 that the choice of L_{norm} affect the results. By moving from L_2 norm to an L_8 norm, an improvement of around 0.3 dB can be observed. This is in line with what we expect.

VI. Conclusion

A recent paper introduced the idea of optimizing the encoding of multiview video based on the expected contribution of each macroblock on the final image [9]. In this paper we analyzed the choice of allocation of rate between macroblocks on that codec. We showed that the proposed QP allocation is optimum for Gaussian sources and when the pixels within a macroblock have all the same weight in the final image. We also derived results for the generic case of a RxD curve, and analyzed a few choices for averaging the weights within a macroblock.

Acknowledgements

The authors would like to thank Cha Zhang for many useful discussions, and Larry Zitnick for providing the video sequences.

References

[1] A. Kubota, A. Smolic, M. Magnor, M. Tanimoto, T. Chen, and C. Zhang, “Multi-view imaging and 3dtv,” *IEEE Signal Processing Magazine*, vol. 24, no. 6, pp. 10–21, 2007.

[2] H. Baker, D. Tanguay, I. Sobel, D. Gelb, M. Goss, W. Culbertson, and T. Malzbender, “The coliseum immersive teleconferencing system,” Tech. Rep., HP Labs, 2002.

[3] C.L. Zitnick, S.B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, “High-quality video view interpolation using a layered representation,” in *ACM SIGGRAPH*, 2004.

[4] C. Buehler, M. Bosse, L. McMillan, S. J. Gortler, and M. F. Cohen, “Unstructured lumigraph rendering,” in *ACM SIGGRAPH*, 2001.

[5] C. Zhang and J. Li, “Interactive browsing of 3d environment over the internet,” in *VCIP*, 2001.

[6] C. Zhang, Z. Yin, and D. Florencio, “Improving depth perception with motion parallax and its application in teleconferencing,” in *MMSP*, 2009.

[7] G. Sullivan, “Standards-based approaches to 3d and multiview video coding,” in *SPIE Applications of Digital Image Processing XXXII*, 2009.

[8] C. Zhang and T. Chen, “A survey on image-based rendering – representation, sampling and compression,” *EURASIP Signal Processing: Image Communication*, vol. 19, no. 1, pp. 1–28, 2004.

[9] D. Florencio and C. Zhang, “View-dependent multiview video compression and streaming for immersive tele-conferencing,” in *ICASSP*, 2009.