Joint Disparity and Variable Size-Block Optimization Algorithm for Stereoscopic Image Compression

Aysha Kadaikar, Gabriel Dauphin, Anissa Mokraoui

L2TI, Institut Galilée, Université Paris 13 Sorbonne Paris Cité
99, Avenue Jean-Baptiste Clément 93430 Villetaneuse, France
{kadaikar, gabriel.dauphin, anissa.mokraoui}@univ-paris13.fr

Abstract

This paper adresses the disparity map estimation problem in the context of stereoscopic image coding. It is undeniably the fact that the use of variable size blocks offers the possibility to describe more precisely the predicted view but at the expense of a high bitrate if no particular consideration is taken into account by the estimation algorithm. Indeed more information related to the block layout, considered here as a block-length map, is required at the prediction step. This paper presents an algorithm which jointly optimizes the block-length map as well as the disparity map so as to ensure a good reconstruction of the predicted view while minimizing the bitrate. This is done thanks to a joint metric taking into account the quality of the reconstruction as well as the bitrate needed to encode the maps. Moreover the developed algorithm iteratively improves its performance by refining the estimated maps. Simulation results conducted on several stereoscopic images from the CMU-VASC dataset confirm the benefits of this approach as compared to competitive block matching algorithms.

Keywords: Stereoscopic image, Disparity, Optimization, Distortion, Entropy, Stereo-matching, Block matching, Variable size-block.

1. Introduction

Applications using stereoscopic videos allow the viewer to have a greater immersion in the scene as it is the case with stereo-visioconferencing or 3D-TV [1, 2]. Stereoscopic images are composed of two views of the same scene acquired from two cameras capturing the scene from two slightly different viewpoints. The viewer has an impression of immersion in the scene by making his left eye look at the left view and the right eye look at the right view. This kind of images, composed of two views, require twice the amount of information needed to encode it when stored or transmitted, as compared to traditional 2D-images. One simple method would consist in encoding the two views separately, but more efficient methods have been developed, they use the high redundancies between the two views. Indeed in stereoscopic images, objects in the scene are shifted from one view to another. This special displacement is called disparity. Such images are generally encoded according to the following scheme: (a) first, one of the two views is taken as the reference one. Let's say the left one. It is encoded independently; (b) then the second view (the right one) is predicted from the reference view by estimating a disparity map; (c) a residual image is computed between the original right view and its prediction. The disparity map is usually encoded using an entropy coder while the left view and the residual image are encoded using transforms such as DCT (Discrete Cosine Transform) [3, 4, 5]. They are all sent to the decoder which reconstructs the right view using the left view and the disparity map, compensated with the residual image. This coding scheme is known as the disparity compensated scheme because of its ressemblance with the motion compensated scheme developed in video codec [6]. One can also observe in the literature that a large number of disparity estimation algorithms are inspired by motion estimation algorithms.

The disparity map estimation is therefore of primary importance in the coding process. A good map estimation in terms of distortion improves the quality of the predicted view but can be very expensive in terms of bitrate. In the context of stereoscopic image coding, the disparity map estimation should lead to a map reducing the distortion of the predicted view while reducing also the bitrate needed to encode it. Estimating the disparity for each pixel is highly costly in terms of bitrate. Usually one disparity is estimated for each block, meaning that pixels in any given block are predicted using the same disparity. This reduces considerably the bitrate needed to encode the disparity map. Using blocks of different sizes can further reduce the bitrate of the disparity map [7, 8]. It enables to transmit only one disparity value for a big block and many disparity values for another block divided into many subblocks. The former could for example be relevant for a uniform region or an object being located at roughly the same depth of the true 3D scene. The latter could be appropriate when predicting textured objects of small sizes. Using blocks of variable sizes comes at a cost, that of transmitting to the decoder sufficient information as to how the blocks of variable sizes are displayed.

This paper presents a new algorithm jointly optimizing the disparity and the block-length maps. The developed method relies on an entropy-distortion metric taking into account the reduction of the distortion of the predicted view but also an approximation of the bitrate needed to encode both the disparity and the block length maps. At each step of the algorithm, a decision is taken to decide if a given block should be predicted using a single disparity value or if that block should be divided into four subblocks and predicted using four different disparity values. This choice is coupled with a refinement of the disparity map.

The rest of the paper is organized as follows. Section 2 introduces the stereo-matching optimization problem. Section 3 proposes the suboptimal optimization algorithm. Section 4 provides simulation results evaluating the developed algorithm's performance. Finally section 5 concludes this paper.

2. Stereo-matching optimization problem

This section deals with the problem of estimating the disparity map achieving the best compromise between yielding a good prediction and encoding the disparity map at a low bitrate.

The reference view and the view to be predicted are assumed to be rectified so that the disparities are searched within the same scan lines. The stereo-matching optimization algorithm is intended to yield a good estimate of the predicted view from a given reference view while requiring a low bitrate to encode the disparity map. To further improve the performance, blocks of variable sizes are considered. All considered block layouts follow a traditional quad-tree structure, that is square subblocks can only be divided into four identical non-overlapping square blocks. Some notations and assumptions are introduced before describing the rate-distortion optimization problem.

2.1. Notations and assumptions

In what follows, the left view, denoted $\mathbf{I}_L$, is taken as the reference view and the right view, denoted $\mathbf{I}_R$, as the one to be predicted using $\mathbf{I}_L$. The predicted view is denoted $\widehat{\mathbf{I}}_R$. All these views are of size $I \times J$ pixels.

The right view is first divided into non-overlapping squared blocks of fixed size denoted $L_{max} \times L_{max}$. Each of these blocks may be further partitioned into a quadtree structure. An example of such a layout is shown on Fig. 1 with blocks denoted as B_k. As it is difficult to keep track of indexes in a quadtree structure, blocks are denoted as B and the map containing all blocks is denoted $\mathbf{B}$.

Each block B is located at position (i_B, j_B) in a coordinates system starting from the top left of the image, with i belonging to $\{0, \dots, I-1\}$ and j to $\{0, \dots, J-1\}$. It has a size of $l_B \times l_B$ pixels where the length l_B belongs to $L = \{L_{min}, 2L_{min}, \dots, L_{max}\}$. Each block is also associated to a unique disparity d_B chosen amongst a search area $D = \{D_{min}, \dots, D_{max}\}$. Hence, B is identified by (d_B, l_B, i_B, j_B).

The disparity map, the block-length map and the block map are represented by $\mathbf{d}, \mathbf{l}, \mathbf{B}$. All these maps are of the same size denoted S.

2.2. Coding of the block layout

The pixel values of predicted right view $\widehat{\mathbf{I}}_R$ on the block B are related to that of the left view $\mathbf{I}_L$ using the disparity d_B as follows:

$$\widehat{\mathbf{I}}_R(i_B + u, j_B + v) = \mathbf{I}_L(i_B + u, j_B + v + d_B) \quad (1)$$
$$\text{where } \; u, v \in \{0, \dots, l_B - 1\}.$$

It may seem from (1) that in order to compute the predicted right view on block B, the decoder needs, not only the disparity d_B and the length l_B, but also the location i_B, j_B. In terms of bitrate this would be quite excessive. However this is avoided since a specific tree structure is adopted where an example is shown in Fig. 2. Its first layer has as many nodes as the number of non-overlapping square blocks that can fit into an image of size $I \times J$. The other layers are subsequent quadtree decompositions. Each block is a leaf in this tree structure, it is located at the first layer if the block is of size $L_{max} \times L_{max}$, at the second layer if it is of size $\frac{L_{max}}{2} \times \frac{L_{max}}{2}$, and so on for the blocks of smaller size. As a result, by assigning to the tree an exploring order, the block-length map $\mathbf{l}$ and the image size I, J are sufficient to recover the block layout and the locations of each block.

The proposed exploring order corresponds to the breadth-first search in the tree. Indeed all neighbor nodes are explored before the children nodes. This order has been chosen as opposed to the depth-first search because breadth-first search is expected to better balance the optimization process on the whole image rather than focusing on the first blocks right at the beginning of the algorithm. The considered exploring order takes into account the raster scanning order when ordering neighboring nodes. This is illustrated in Fig. 1 by showing a block-length map and the corresponding image of size 64×96 pixels with a specific block layout whose size is ranging from 4×4 to 32×32

As a consequence in terms of notations, it makes sense to claim that all quantities depending on a block B, depend only of $\mathbf{d}, \mathbf{l}$ when summed up along $B \in \mathbf{B}$.

2.3. Formulation of the rate-distortion problem

In this section, the quality of the predicted right view is measured with the global distortion, $E_G(\mathbf{B})$, defined as the sum of the square differences. As only one disparity is assigned to each block, this global distortion is the sum of all local distortions accounting for the difference between the predicted right view and the true right view on a given block B:

$$E_G(\mathbf{d}, \mathbf{l}) = \sum_{B \in \mathbf{B}} E_B(B) \qquad (2)$$

$$E_B(B) = \sum_{u=0}^{l_B-1} \sum_{v=0}^{l_B-1} \left(\mathbf{I}_L(i_B + u, j_B + v + d_B) - \mathbf{I}_R(i_B + u, j_B + v) \right)^2.$$

Note that applying the remark ending subsection 2.2, $E_G(\mathbf{B})$ is being replaced by $E_G(\mathbf{d}, \mathbf{l})$.

The disparity map is encoded using a lossless coder (entropy encoder). It approximates the bitrate needed to encode the disparity map by S times the entropy of a specific random variable $\mathcal{D}$. $\mathcal{D}$'s categorical distribution $p_E(\mathcal{D} = d|\mathbf{d})$ is computed from the disparity map $\mathbf{d}$ as the ratio of the number of blocks having the disparity d to the total number of blocks S. For the sake of simplicity, the entropy of $\mathcal{D}$ (i.e. $H(\mathcal{D})$) is denoted $H(\mathbf{d})$ as it can be computed using only $\mathbf{d}$.

$$H(\mathbf{d}) = -\sum_{d \in D} p_E(\mathcal{D} = d|\mathbf{d}) \log_2 \left(p_E(\mathcal{D} = d|\mathbf{d})\right). \tag{3}$$

The block-length map is also encoded using a lossless coder. The same notation choices have been made. $SH(\mathbf{l})$ approximates the bitrate needed by the entropy coder and more specifically denotes S times the entropy of a random variable $\mathcal{L}$ whose distribution $p_E(\mathcal{L} = l|\mathbf{l})$ is computed from the block-length map $\mathbf{l}$ as the ratio between the number of blocks having the length l and the total number of blocks S.

$$H(\mathbf{l}) = -\sum_{l \in L} p_E(\mathcal{L} = l|\mathbf{l}) \log_2 \left(p_E(\mathcal{L} = l|\mathbf{l})\right). \tag{4}$$

Hence an approximation of the required bitrate is deduced from:

$$S\left(H(\mathbf{d}) + H(\mathbf{l})\right). \tag{5}$$

To achieve the paper goal, the optimization problem can be posed in several ways resulting in a low distortion of the predicted view requiring a low bitrate. One might expect the lowest global distortion for a given bitrate or the smallest bitrate for a given global distortion. Solving one issue or the other for every bitrate or every global distortion value leads to the same rate-distortion curve made from all the optimal points. This paper proposes the Lagrangian formulation to solve the optimization problem. The Lagrangian cost function $J(\lambda, \mathbf{d}, \mathbf{l}, S)$ is then defined as follows:

$$J(\lambda, \mathbf{d}, \mathbf{l}, S) = E_G(\mathbf{d}, \mathbf{l}) + \lambda S\left(H(\mathbf{d}) + H(\mathbf{l})\right). \tag{6}$$

Minimizing the Lagrangian cost function $J(\lambda, \mathbf{d}, \mathbf{l}, S)$ for every value of the Lagrange multiplier λ leads to the same optimal curve. As pointed in [9], finding the optimal solution is not an easy task. Indeed any modification of a given disparity at a given block may modify the appropriateness of a block division or the choice of a disparity at another block.

3. Joint disparity and block-length maps optimization algorithm (JDBLMO)

This section focuses on the minimization of the Lagrangian cost function, $J(\lambda, \mathbf{d}, \mathbf{l}, S)$ introduced in the previous section, for a given value of λ. Given the complexity of this optimization problem, a sub-optimal optimization algorithm is developed. In the rest of the paper, this algorithm is called JDBLMO for for Joint Optimization of the Disparity and Block-Length Map . It is briefly described in Fig. 3.

The proposed sub-optimization algorithm is performed according to three main stages. The first stage, described in Subsection 3.1, is related to an initialization process where the initial disparity and block-length maps are computed using the traditional Block-Matching Algorithm (BMA). The second stage, described in Subsection 3.2, is an adaption of the R-Algorithm developed in [10] since the R-algorithm has been proposed only for same size blocks. This algorithm is named ARA (Adaptation of the R-Algorithm). The third stage, described in Subsection 3.3, reduces the function cost $J(\lambda, \mathbf{d}, \mathbf{l}, S)$ by means of some Block Divisions and is called Block Divisions Algorithm (BDA). The BDA and ARA are repeated as long as an improvement is being observed.

3.1. Initialization

In this initialization step, the right view is first partitioned into blocks of size $L_{max} \times L_{max}$, setting the block-length map as a constant map. The traditional BMA is then applied on each block. For a given block B, the disparity d_B is selected by minimizing the Sum of Squared Differences (SSD) computed between the pixel-values of the right view and the predicted view considering only pixels in B. Using E_B defined in (2), d_B is expressed as:

$$d_B = \arg \min_{d \in D} E_B(d). \tag{7}$$

The collection of all d_B forms an initial disparity map $\mathbf{d}$. When associated with the constant map $\mathbf{l} = [L_{max} \dots L_{max}]$, this forms an initial block map $\mathbf{B}$. This block map $\mathbf{B}$ is then converted into a queue whose processing is consistent with the tree-structure exploring order defined in section 2.2.

Introduce some notations to handle $\mathbf{B}$ as a queue. B_f is the front block and $\mathbf{B}_t$ is the tail of the queue:

$$\mathbf{B} = [B_f, \mathbf{B}_t]\,. \tag{8}$$

Removing the first block and placing it at the queue's end is expressed as:

$$\mathbf{B} := [\mathbf{B}_t, B_f]\,. \tag{9}$$

This action is used in steps 2.b.,2.d. and 3.f. in Fig. 3. For the sake of simplicity each block $B \in \mathbf{B}$ is now considered as a couple (d_B, l_B) and the front block is denoted as (d_f, l_f).

A decreasing counter denoted c is introduced, both in the ARA and the BDA. It ensures that each block B be processed exactly once, regardless of the evolving size of the queue. This guarantees that at the end of the processing the queue's first block is located at the top left of the image. This is how it works: c is first set as the initial size of the block map, it is then decremented at each iteration and repetition terminates when c reaches 0. Control of the repetition is achieved using the decreasing counter c (steps 2.a., 2.e. and 2.f. and steps 3.b., 3.g. and 3.h. in Fig. 3).

3.2. Adapation of the R-Algorithm (ARA)

The ARA is an adaptation of the R-algorithm developed in [10]. This latter is briefly introduced to extend its strategy to blocks of non-equal sizes.

The R-algorithm finds a suboptimal solution of an optimization problem slightly different from (6). Indeed the modified cost function J_r does not take into account $H(\mathbf{l})$ the size of the blocks since it is assumed to be of equal-size $l \times l$:

$$J_r(\lambda, \mathbf{d}) = E_{Gr}(\mathbf{d}) + \lambda_r H(\mathbf{d}), \tag{10}$$

where $E_{Gr}(\mathbf{d}) = E_G(\mathbf{d}, l \ldots l)$ and $\lambda_r = \lambda S$.

The disparity map $\mathbf{d}$ is first set using the BMA as in (7). The R-algorithm processes the disparity map by successively re-evaluating each selected disparity: each of them is being replaced by all the other disparities of the search area D, resulting in as many disparity maps as disparities in the search area. A rate-distortion cost is computed for each disparity map and the one minimizing the cost is set as the new disparity of the current block. The process is then iterated until re-evaluating all the disparities in the map. After that, considering the new updated disparity map, the whole process is re-iterated again until no improvements are observed in terms of rate-distortion.

This R-algorithm is now adapted to cope to not only the variable number of blocks but also the variable sizes of these blocks. Processing is achieved through the use of $\mathbf{B}$ as a queue. The disparity of the front block, d_f is replaced by all disparities $d \in D$, resulting in as many block maps $[(d, l_f), \mathbf{B}_t]$. The selected disparity $\widehat{d}$ minimizes J is given by:

$$\widehat{d} = \arg\min_{d \in D} J\left([(d, l_f), \mathbf{B}_t]\right). \tag{11}$$

This modified block is removed from the front and placed at the queue's end:

$$\mathbf{B} := \left[\mathbf{B}_t, (\widehat{d}, l_f)\right]. \tag{12}$$

Using the decreasing counter c, this process is iterated in such a way as to process all blocks once and to ensure that the front block is the block located at the image's top left corner. It is further iterated as long improvements are made, this is checked by comparing the current value $J(\mathbf{B})$ of the cost function with a stored value denoted J_{old}.

3.3. Block Division Algorithm (BDA)

This section deals with the third stage, Block Division Algorithm (BDA), of the optimization algorithm. BDA consists in processing all blocks. For each block, in deciding, whether dividing the block into four subblocks helps reducing the function cost $J(\lambda, \mathbf{d}, \mathbf{l}, S)$. This decision is taken by comparing the cost function when applied to two block maps, one where the front block is not divided and one where the front block is divided and disparities of each subblock are adjusted so as to minimize $J(\lambda, \mathbf{d}, \mathbf{l}, S)$. Note that the BDA modifies the tree-structure only within a specific layer consisting of all blocks of a given size. When it is first applied, it attempts to divide blocks of size $L_{max} \times L_{max}$. When applied a second time, it considers blocks of size $\frac{L_{max}}{2} \times \frac{L_{max}}{2}$ and so on up until reaching the size $L_{min} \times L_{min}$. The parameter l_{min} and the step 3.c. in Fig. 3 control the size of blocks that might be divided in the BDA.

Let us first describe how to transform a block map $\mathbf{B}$ into a new block map, denoted $\mathbf{B}_s$, containing four subblocks, (step 3.d. in Fig. 3):

- The front block (d_f, l_f) is divided into four subblocks denoted as $(d_f, \frac{l_f}{2})$,$(d_f, \frac{l_f}{2})$, $(d_f, \frac{l_f}{2})$, $(d_f, \frac{l_f}{2})$,

- A new block map is obtained by removing the front block and placing at the queue's end these four blocks:

$$\left[\mathbf{B}_t, (d_f, \frac{l_f}{2}), (d_f, \frac{l_f}{2}), (d_f, \frac{l_f}{2}), (d_f, \frac{l_f}{2})\right],$$

- The first subblock disparity d_f is replaced by all disparities d in the search area, and the disparity d_1 is selected so as to minimize $J(\lambda, \mathbf{d}, \mathbf{l}, S)$:

$$d_1 = \arg\min_{d \in D} J\left(\left[\mathbf{B}_t, (d, \frac{l_f}{2}), (d_f, \frac{l_f}{2}), (d_f, \frac{l_f}{2}), (d_f, \frac{l_f}{2})\right]\right),$$

- The selection of the second subblock disparity d_2 takes into account the choice of d_1. It is also computed by testing the cost function on all disparities within the search area:

$$d_2 = \arg\min_{d \in D} J\left(\left[\mathbf{B}_t, (d_1, \frac{l_f}{2}), (d, \frac{l_f}{2}), (d_f, \frac{l_f}{2}), (d_f, \frac{l_f}{2})\right]\right),$$

- The selection of the third and fourth subblock disparities d_3, d_4 are obtained in the same way:

$$d_3 = \arg\min_{d \in D} J\left(\left[\mathbf{B}_t, (d_1, \frac{l_f}{2}), (d_2, \frac{l_f}{2}), (d, \frac{l_f}{2}), (d_f, \frac{l_f}{2})\right]\right),$$
$$d_4 = \arg\min_{d \in D} J\left(\left[\mathbf{B}_t, (d_1, \frac{l_f}{2}), (d_2, \frac{l_f}{2}), (d_3, \frac{l_f}{2}), (d, \frac{l_f}{2})\right]\right),$$

- The proposed block map is then expressed as:

$$\mathbf{B}_s = \left[\mathbf{B}_t, (d_1, \frac{l_f}{2}), (d_2, \frac{l_f}{2}), (d_3, \frac{l_f}{2}), (d_4, \frac{l_f}{2})\right]. \tag{13}$$

The decision consists in comparing $J(\mathbf{B})$ with $J(\mathbf{B}_s)$. If $J(\mathbf{B}_s)$ is smaller, then the new block map considered is $\mathbf{B}_s$. If, on the contrary, $J(\mathbf{B})$ is smaller, then the front block is placed at the queue's end indicating that it has been processed and the new block map considered is

$$\mathbf{B} := [\mathbf{B}_t, (d_f, l_f)]$$

The block-division algorithm consists in repeating these two steps, computing $\mathbf{B}_s$ and making the comparison. Repetition goes on until the first block is back at the front position of the queue. This repetition is further iterated as long as at least one more block division is being decided, this is checked with a flag f that is initially switched on. It is switched off as soon as a block is divided. Control of this flag is done by steps 3.b and 3.i in Fig. 3.

4. Performance evaluation and discussions

This section analyzes the performance of the proposed algorithm JDBLMO. The ability of this algorithm to achieve a good prediction of the right view knowing the exact left view while requiring the least bitrate is discussed. The rationale is that such an algorithm should also yield good performance when it is integrated in a disparity compensated coding scheme.

The Peak Signal-to-Noise Ratio (PSNR) is adopted to measure the prediction quality computed between the original and the predicted right view. The bitrate is expressed in bits per pixel (bpp) and computed according to the entropy approximation applied to both the block-length and the disparity maps.

Simulations are conducted on stereoscopic images taken from the CMU-VASC dataset [11]. The algorithm's performance is compared with the traditional BMA using same size-block for stereoscopic images but also with the modified version of the block-matching algorithm Intra/Inter-frame Block Segmentation Coding (IIBSC) using variable size-blocks developed in [12]. The predicted view is first divided into fixed-size blocks. Each block is encoded either using an intra-frame or an inter-frame coding scheme. The latter scheme makes use of a metric assessing the quality of the predicted block as compared to the corresponding block on the original image using the Sum of Squared Differences (SSD). This scheme consists in two steps: blocks are repeatedly divided so as to lower the SSD below a threshold, then these divided blocks are repeatedly merged as long as the SSD remains below this threshold. This algorithm has been adapted to our context and is called MIIBSC (Modified IIBSC). The modifications made to this algorithm consist in allowing only the inter-frame coding and in adapting this prediction mode to the context of stereoscopic image coding. The inter-frame coding becomes the inter-view coding taking into account the two adjacent views. Furthermore only the first step is retained and is performed as follows. The fixed-size blocks are repeatedly and recursively divided into four subblocks of equal size until the mean squared error of each divided subblocks gets below a threshold.

The performance discussion is organized as follows. JDBLMO, BMA and MIIBSC algorithms are compared first in a more detailed fashion in subsection 4.1 and then in terms of average rate-distortion performance in subsection 4.2.

4.1. JDBLMO performance evaluation compared to BMA and MIIBSC

This section compares the performance of the proposed algorithms to the test algorithmes described above using the stereoscopic image ”rubik” shown at the top of Fig. 5.

In the presented simulations, the right view is first divided into blocks of size 32×32 pixels. The candidate disparities are selected within the set $[-30, -29, \ldots, 30]$. The MIIBSC is tested according to a large number of thresholds ranging from 0 to 1000 which are applied to the mean SSD. For a given threshold, a predicted view, a PSNR-value, a bit-rate and a global block layout are obtained. While the JDBLMO is performed using a large number of λ-values. The BMA is performed with four fixed set of size-blocks: 4×4, 8×8, 16×16 and 32×32. For each parameter value, each algorithm yields a predicted view, a global block layout, and a rate-distortion point composed of a PSNR value and a bitrate value.

Fig. 4 illustrates the rate-distortion performance of the algorithms by joining with lines the experimental rate-distortion points. Note that the JDBLMO (black dashed curve joining black squares) performs better than the BMA (red curve joining red pluses) and even better than the MIIBSC (blue curve joining blue circles). For a given bitrate of 0.015 bpp, the JDBLMO yields a PSNR of 32.0 dB improving by 1.8 dB the BMA's performance and by 0.9 dB the MIIBSC's performance.

Fig. 5 presents above the original right view and below three close-up views of the right part of a ”wooden egg cup” extracted from the three predicted views processed with respectively from left to right, the JDBLMO, MIIBSC and BMA. The JDBLMO yields the best visual reconstruction as the two others fail to render the triangular shape. Indeed this can be explained by the block layout selected respectively by the JDBLMO and the MIIBSC given by Fig. 6 and Fig. 7. A closer look at these figures show that the triangular shape is processed with seven subblocks by the JDBLMO and with only one block by the MIIBSC.

4.2. JDBLMO average rate-distortion performance compared to BMA and MIIBSC

This section analyzes the average rate-distortion performance of the described algorithms. Simulations are performed on ten stereoscopic images, namely ”book”, ”books”, ”sphere”, ”wdc2r”, ”whouse”, ”rubik”, ”telephone”, ”toys”, ”cdc1”, ”house2”. Eight target bitrates in bpp are defined: $[0.06\ 0.07\ 0.08\ 0.09]$ and $[0.1\ 0.25\ 0.40\ 0.55]$, the former corresponding to low bitrate and the latter corresponding to medium bitrate. The parameter values (λ for JDBLMO, threshold for MIIBSC and size-block for BMA) are set so as to comply the target bitrates. The Bjøntegaard metric [13] is then used to compute at low and at medium bitrate, an average PSNR difference between the JDBLMO and the BMA, and between the JDBLMO and the MIIBSC. These average PSNR differences are listed in Tab. 1 for each of the ten stereoscopic images, where $\Delta\texttt{BMA}$ stands for the former average difference and $\Delta\texttt{MIIBSC}$ the latter average difference. One can observe that the JDBLMO achieves better results when compared to the BMA and in most cases when it is compared to the MIIBSC. The difference is higher at low bitrate and for the BMA.

5. Conclusion

This paper proposed a new stero-matching image algorithm using variable size-block. This algorithm optimized jointly the disparity and block-length maps so as to ensure a good reconstruction of the predicted view while minimizing the bitrate. Simulation results shown that this sub-optimal stereo-matching algorithm achieved better prediction performance when compared to the competitive BMA algorithm using variable size-block. In future work, the proposed algorithm will be integrated in a complete disparity compensated coding scheme.

References

[1] P. Kauff, O. Schreer, An immersive 3D video-conferencing system using shared virtual team user environments, in: Proceedings of the 4th international conference on Collaborative virtual environments, ACM, 2002, pp. 105–112.

[2] W. J. Tam, F. Speranza, S. Yano, K. Shimono, H. Ono, Stereoscopic 3d-tv: visual comfort, IEEE Transactions on Broadcasting 57 (2) (2011) 335–346.

[3] S. Li, M. Yu, G. Jiang, T.-Y. Choi, Y.-D. Kim, Approaches to H. 264-based stereoscopic video coding, in: Third International Conference on Image and Graphics (ICIG'04), IEEE, 2004, pp. 365–368.

[4] W. Woo, A. Ortega, Stereo image compression with disparity compensation using the MRF model, in: Visual Communications and Image Processing, International Society for Optics and Photonics, 1996, pp. 28–41.

[5] U. Ahlvers, U. Zoelzer, S. Rechmeier, FFT-based disparity estimation for stereo image coding, in: International Conference on Image Processing, ICIP, Vol. 1, IEEE, 2003, pp. 257–264.

[6] K. Ugur, A. Alshin, E. Alshina, F. Bossen, W.-J. Han, J.-H. Park, J. Lainema, Motion compensated prediction and interpolation filter design in H. 265/HEVC, IEEE Journal of Selected Topics in Signal Processing 7 (6) (2013) 946–956.

[7] M. Gong, Y.-H. Yang, Genetic-based stereo algorithm and disparity map evaluation, International Journal of Computer Vision 47 (1-3) (2002) 63–77.

[8] D. Clewer, L. Luo, C. N. Canagarajah, D. R. Bull, M. Barton, Efficient multiview image compression using quadtree disparity estimation, in: The International Symposium on Circuits and Systems, ISCAS, Vol. 5, IEEE, 2001, pp. 295–298.

[9] A. Kadaikar, G. Dauphin, A. Mokraoui, Sequential block-based disparity map estimation algorithm for stereoscopic image coding, Signal Processing: Image Communication 39, Part A (2015) 159–172.

[10] A. Kadaikar, G. Dauphin, A. Mokraoui, Modified block matching algorithm improving rate-distortion performance for stereoscopic image coding, in: International Symposium on Signal Processing and Information Technology (ISSPIT), IEEE, 2015, pp. 478–483.

[11] CMU vision and autonomous systems center (VASC) image database. URL `http://vasc.ri.cmu.edu/idb/html/stereo/`

[12] M. Chan, Y. Yu, A. Constantinides, Variable size block matching motion compensation with applications to video coding, IEE Proceedings I-Communications, Speech and Vision 137 (4) (1990) 205–212.

[13] G. Bjontegaard, Calculation of average PSNR differences between RD, in: ITU-T SC16/Q6, 13th VCEG Meeting, Austin, Texas, USA, 2001, document VCEG-M33.

Example of a block-length map, split into six subsequences for clarity purpose:

```
8,8,4,4,4,4,8,16,16,8,8,8,8,    32,                             32,
32,                             16,16,8,8,8,4,4,4,4,16,    32,
```

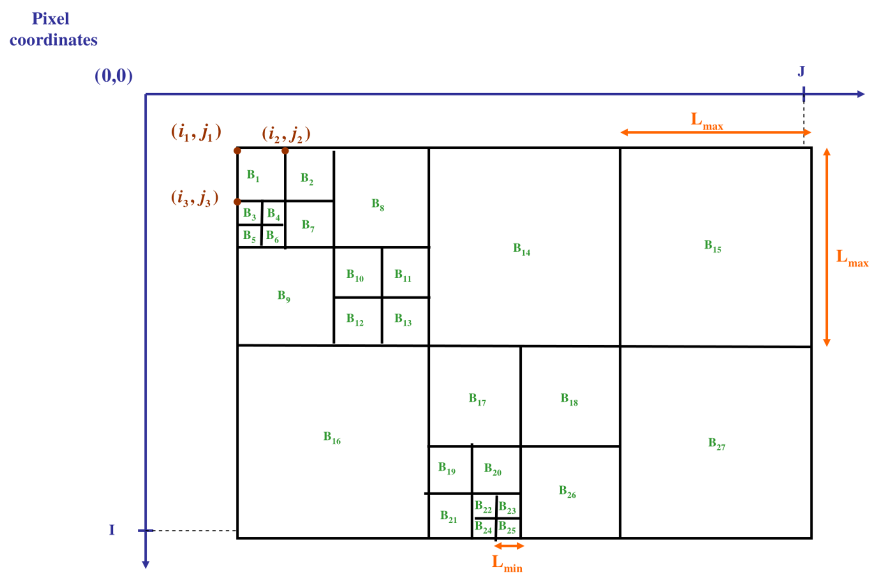

Figure 1: Example of a block-length map and its corresponding block layout.

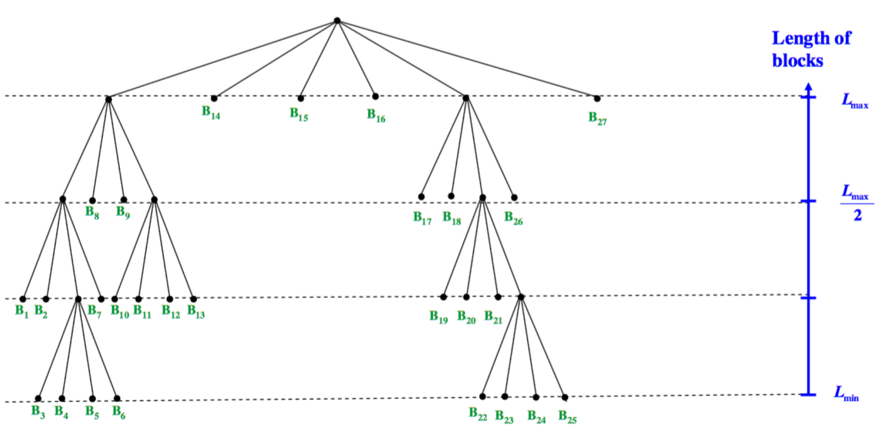

Figure 2: Tree structure associated with the block layout shown in fig. 1.

Input: Left view $\mathbf{I}_L$, right view $\mathbf{I}_R$,

minimum and maximum lengths, respectively L_{min} and L_{max} pixels.

Output: Joint estimated disparity and block-length map $(\mathbf{d}, \mathbf{l})$.

---

1. Initialization.

1.a Partition the right view into square blocks of lengths L_{max}.

1.b Compute for each block B a disparity d_B using the BMA.

1.c Consider this initial block map $\mathbf{B}$ as a queue.

1.d Set $l_{min} = L_{max}$.

2. Apply the ARA on $\mathbf{B}$:

2.a. Set $J_{old} := J(\mathbf{B})$ and $c := S$.

2.b. Remove the queue's front block denoted as (d_f, l_f).

2.c. Select the best disparity $\widehat{d}$ minimizing $J(\mathbf{B}_t, (d, l_f))$ as $d \in D$.

2.d. Update the block map: $\mathbf{B} := (\mathbf{B}_t, (\widehat{d}, l_f))$.

2.e. Decrement c.

2.f. Go back to step 2.b. if $c > 0$.

2.g. Go back to step 2. if $J(\mathbf{B}) < J_{old}$.

2.h. Go to step 3. if $l_{min} > L_{min}$.

2.i. End here the JDBLMO.

3. Apply the BDA on $\mathbf{B}$:

3.a. Set $l_{min} = \frac{l_{min}}{2}$.

3.b. Set $c := S$ and $f := 1$.

3.c. Go to step 3.f. if $l_f = l_{min}$.

3.d. Compute $\mathbf{B}_s$ as in (13)

3.e. Set $\mathbf{B} := \mathbf{B}_s$ and $f := 0$ and go to step 3.g. if $J(\mathbf{B}_s) < J(\mathbf{B})$.

3.f. Set $\mathbf{B} := [\mathbf{B}_t, B_f]$.

3.g. Decrement c.

3.h. Go to step 3.c. if $c > 0$.

3.i. Go to step 3.b. if $f = 0$

3.j. Go to step 2.

Figure 3: Description of the JDBLMO algorithm

Table 1: Average PSNR gain of the JDBLMO over the BMA and the MIIBSC

Stereo	**Low bitrate**		**Medium bitrate**	
Images	**ΔBMA**	**ΔMIIBSC**	**ΔBMA**	**ΔMIIBSC**
book	0.95	0.51	0.67	0.03
books	1.08	0.10	0.90	0.19
sphere	1.14	0.90	1.02	0.25
wdc2r	1.30	0.55	0.45	-0.74
whouse	1.45	0.55	1.46	0.44
rubik	1.64	0.22	0.74	0.44
telephone	2.45	0.81	1.89	0.34
toys	3.09	-0.27	2.51	0.13
cdc1	3.14	0.50	1.80	0.04
house2	3.24	0.12	0.51	-0.01

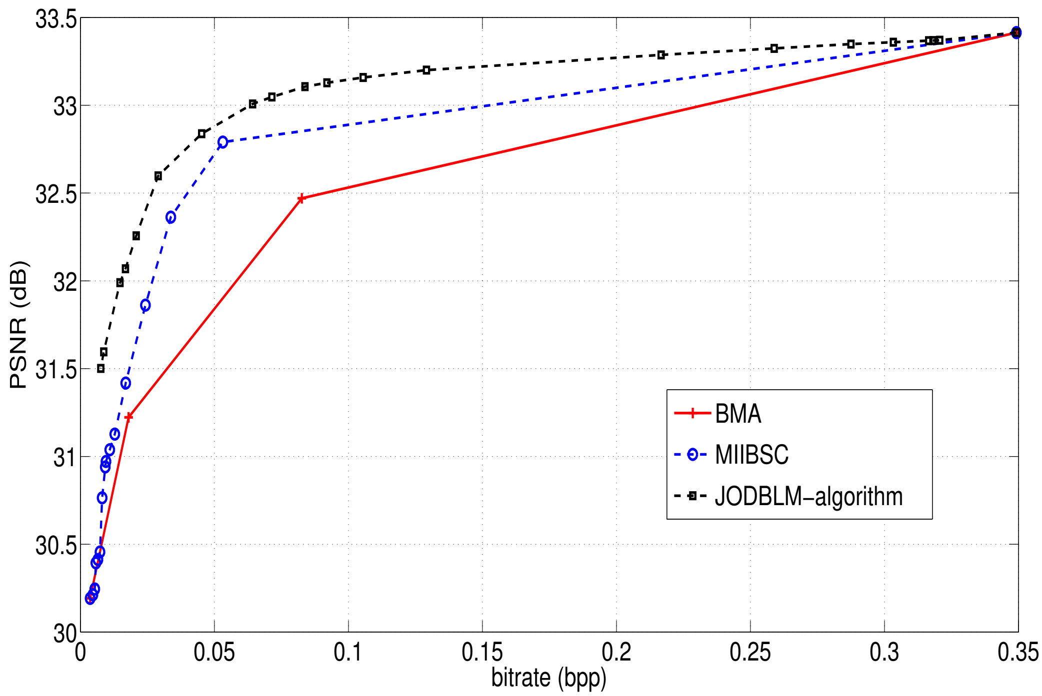

Figure 4: Rate-distortion performance of the JDBLMO, MIIBSC and BMA performed on ”rubik”.

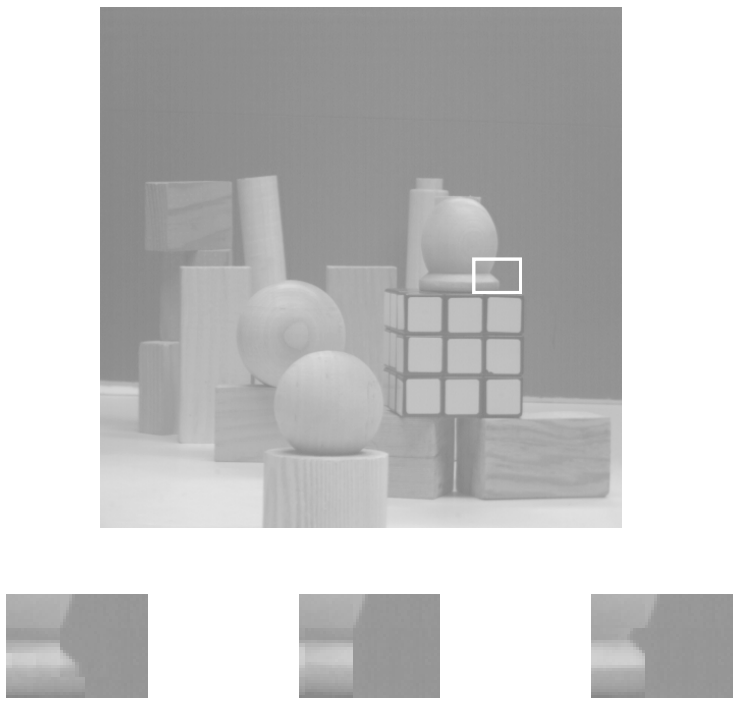

Figure 5: Above: original right view ”rubik”. Below, from left to right close-up views extracted from the three predicted views processed with the JDBLMO, MIIBSC and BMA.

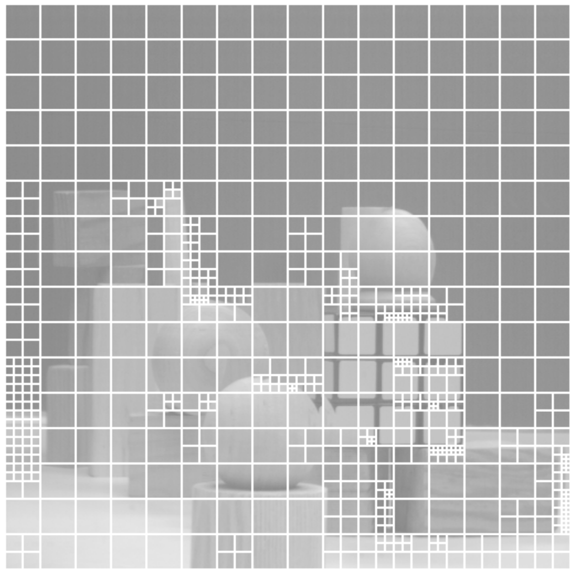

Figure 6: Block-length map resulting from the JDBLMO

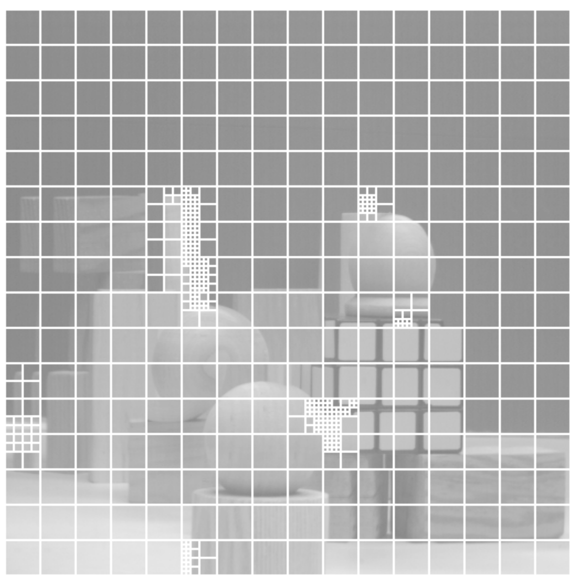

Figure 7: Block-length map resulting from the MIIBSC