High Quality Region Based Motion Estimation with Dynamic Aspect Ratios

Aleksandar Berić  
Eindhoven University of Technology  
Dept. of Electrical Eng.  
Eindhoven, The Netherlands  
Email: a.b.beric@tue.nl

Ben van der Waal  
Delft University of Technology  
Faculty of EEMCS  
Delft, The Netherlands  
Email: b.vanderwaal@ewi.tudelft.nl

Ramanathan Sethuraman, Gerard de Haan and Jef van Meerbergen  
Philips Research Laboratories  
Eindhoven, The Netherlands  
Email: {ramanathan.sethuraman, g.de.haan, jef.van.meerbergen}@philips.com

Abstract—In the domain of motion estimation based applications, in order to keep the bandwidth requirements low, the usage of multiple levels of memory hierarchy is a necessity. Usually, the first level (L0) holds the search area of the estimator and the second (L1) holds the region wherein the estimation is performed. This work presents the region-based motion estimation wherein the aspect ratio of the region dynamically changes per video field/frame. We extend this idea to the extreme case where the aspect ratio of the region changes from horizontal stripe to vertical column. This idea enables fast convergence of the motion estimator and keeps the bandwidth requirements towards the off-chip image memory to their minimum, one access per pixel, regardless of the number of motion estimation scans. To demonstrate our idea, experiments were performed on the test set of video sequences using the state-of-the-art de-interlacing as the application. The results are encouraging regarding both, objective quality metric as well as visual perception.

I. INTRODUCTION

In the last few decades, multimedia applications have been gaining tremendous popularity and are inseparable part of today’s contemporary life. Among them, video-oriented applications take dominant part. Video processing algorithms are executing in plethora of Systems-on-Chip (SoCs), spanning from tiny hand-held devices to high-end television sets. In majority of these applications, motion estimation algorithms are utilized in number of video applications, e.g. compression/decompression, tracking algorithms, or tv-based applications. SoCs used in the tv-applications, like de-interlacing or picture-rate up-conversion (motion estimation based) typically reference few fields/frames of image data located in the off-chip memory. The bandwidth towards the off-chip image memory is usually the bottleneck in these SoCs. This is especially visible in the last few years when the high definition television (HDTV) gained a lot of attention and publicity. The trend of increasing the resolution of the television sets is clearly visible. From standard definition (SD, 0.5 Mpixel) through high definition (HD, ranging from 1 Mpixel to 2 Mpixels) to even bigger resolutions. In parallel to the increasing bandwidth requirements, the ever-lasting trend for quality and performance is also present.

Exploiting the locality of reference through multiple levels of memory hierarchy [1] by using different levels of caches, buffers or scratchpads, is the proven method to cope with the high bandwidth requirements. We focus in this work to the motion estimation based video applications like de-interlacing and picture-rate up-conversion. As our starting point, we select the model utilizing the two-level memory hierarchy as shown in figure 1 introduced in [2]. In the mentioned applications, the L0 scratchpad (marked as L0 in fig. 1) holds the so-called Search Area (SA) of the motion estimator/compensator. The L1 scratchpad is introduced in order to reduce the bandwidth requirements towards the off-chip image memory to minimum. The L1 scratchpad can hold one stripe of the image [2], [3] or one region of the image [4], [5], [6]. Both methods have their advantages and disadvantages which will be discussed in detail in the next sections of this paper.

This work presents the region-based motion estimation/compensation with dynamic aspect ratios which reduces artifacts at the borders of the regions, offers faster convergence and keeps the bandwidth requirements to minimum. We will demonstrate our results through the 2-dimensional generalized sampling theory based de-interlacing application [7]. As part of the quality analysis of our motion estimator, we measure the performance through objective error criterion and tackle the issue of visual perception. The remainder of this paper is organized as follows. Section II briefly describes the motion estimation algorithm used in this work. Section III states the prior work regarding stripe-based and region-based motion estimation. The dynamic aspect ratio region-based motion estimator is presented in Section IV. The results are presented in Section V and the conclusions are drawn in Section VI.

II. 3DRS MOTION ESTIMATION ALGORITHM

For keeping the computational cost relatively low, the three-dimensional recursive search (3DRS) block matching algorithm [8] is utilized in this paper. The algorithm is based
on the full-search block matcher (FSBM), which divides the image into blocks of pixels, $B(\vec{X})$. To each center $\vec{X}$, a displacement vector at time instant $n$ is assigned, $\vec{D}(\vec{X}, n)$. The displacement vectors are selected from a candidate set, $CS^{max}$, defined by

$$CS^{max} = \{ \vec{C} \mid -N \leq C_x \leq +N, -M \leq C_y \leq +M \}$$

where $N$ and $M$ are constants that limit the possible motion vector to a search area, $SA(\vec{X})$. The displacement vector with the minimal sum-of-absolute-differences (SAD) is selected for the motion vector field. The SAD criterion is chosen because it offers a good compromise between computational complexity and quality.

The candidate set of the FSBM is reduced for the 3DRS block matcher based on two assumptions:

1) Objects within a frame are larger than blocks

2) Objects have inertia

The first assumption implies that the evaluation of all possible vectors within $CS^{max}$ is not necessary. It should suffice to have a candidate vector set that contains motion vectors taken from the neighbors.

$$CS(\vec{X}, n) = \{ \vec{C} \in CS^{max} \mid \vec{C} = \vec{D}(\vec{X} + \begin{pmatrix} iX \\ jY \end{pmatrix}, n) \}$$

where $X$ is the block width and $Y$ is the block height. However, not all spatial neighbors are available due to the causality problem. In addition, a problem arises because the motion vectors are initialized to zero. The second assumption solves the problem of causality by using temporal candidates where the spatial candidates are not available. The temporal candidates are taken from the previous calculated motion vector field. To reduce the amount of evaluations, only the spatial and temporal candidates as depicted in Figure 2 are utilized for a scanning direction from left to right and top to bottom. The problem of initialization is solved by a (pseudo) random update candidate. Spatial candidate $S1$ is utilized as an offset to which a (pseudo) random vector is added. This vector is chosen cyclically from a predefined update set, such as

$$US_i = \{ \vec{y}_u, -\vec{y}_u, \vec{x}_u, -\vec{x}_u, 2\vec{y}_u, -2\vec{y}_u, 3\vec{x}_u, -3\vec{x}_u \}$$

where $\vec{x}_u = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $\vec{y}_u = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$.

To improve the quality of the motion vector field, sub-pixel accuracy is utilized. Quarter pel resolution is achieved by the following update set

$$US_f(1/4) = \{ \frac{1}{4}\vec{y}_u, -\frac{1}{2}\vec{y}_u, \frac{1}{4}\vec{x}_u, -\frac{1}{2}\vec{x}_u \}$$

The total candidate set, $CS(\vec{X}, n)$, includes a null vector and is defined for a scanning direction from left to right and top to bottom by

$$CS(\vec{X}, n) = \begin{cases} \vec{D}(\vec{X} + \begin{pmatrix} 0 \\ -Y \end{pmatrix}, n), \\ \vec{D}(\vec{X} + \begin{pmatrix} X \\ 0 \end{pmatrix}, n), \\ \vec{D}(\vec{X} + \begin{pmatrix} 2X \\ 2Y \end{pmatrix}, n - 1), \\ \vec{D}(\vec{X} + \begin{pmatrix} 0 \\ -Y \end{pmatrix}, n) + \vec{U}, \end{cases}$$

where the update vector $\vec{U}$ is chosen from the update set

$$US = US_i \cup US_f(1/4)$$

Not all motion vector candidates are equally reliable. For example, the temporal candidate taken from the previously calculated motion vector field is less reliable than the spatial candidates taken from the current motion vector field [3, 9]. To solve this, penalties are added to the calculated SAD values, except for the spatial candidates. For the temporal candidate, a penalty of 0.33% of the maximal pixel value multiplied by the number of pixels within the SAD window is added. While the (pseudo) random update and the null vector candidates have a penalty of 1%.

III. Stripe-based and Region-based Scanning

In [3] it was shown that performing two or three motion estimation scans gives a significant improvement of image quality. In addition, alternation of scanning direction helps in faster convergence of the motion vector field and meandering style [10] of scanning proved to be superior to the classical style (top-to-bottom-and-left-to-right).

The work [2] analyses the impact of different levels of memory hierarchy to the bandwidth requirements towards the off-chip image memory. In this model, the L0 scratchpad holds the search area of the motion estimator/compensator (the architecture is as depicted in fig. 1). The introduction of stripe-based L1 scratchpad reduced the number of pixel accesses from image memory to only one access per pixel.

This is enabled since the stripe-based L1 scratchpad holds $SL1_Y$ (usually equivalent to the height of the SA, $L0_Y$)
complete block-lines (hence, \( SL_1 \) is equal to the width of the field/frame) and slides downwards or upwards through the image. During sliding, the oldest data which is not used any more by the algorithm gets replaced by the new block-line. The drawback of this approach is that the multiple motion estimation scans are usually not utilized within the stripe due to the limited height of the stripe stored in the scratchpad.

On the other hand, in region-based approach, the image is divided into number of regions with fixed aspect ratio and each region is processed independently. Note that the region can take the same area like in the stripe-based approach \( SL_1 \) \( \times \) \( SL_1 \) but reconfigured differently. Namely, \( RL_1 \gg L0_X \) and \( RL_1 \gg L0_Y \). By doing that, motion estimation can be performed within each region multiple times. However, the bandwidth requirements exceed one access per pixel since vertically neighboring regions must overlap. The next paragraph clarifies this issue.

For blocks to be processed \( X \) located close to the border of a region, the search area \( SA(X) \) lies partly outside the region. This is depicted in Figure 3. To avoid an ambiguity between the need to have these blocks available in the L0 scratchpad (search area), while they are not present in the L1 scratchpad (region), the L1 scratchpad stores these blocks in addition to the region. The result of this is that some blocks have to be present in the L1 scratchpad for multiple regions. Regions are processed left to right which leads to the possibility to reuse some blocks when moving from a region to its right neighbor. This solves the issue only partially, namely, vertical border between two horizontally neighboring regions.

However, when all regions located at the same horizontal stripe are processed (regions 1, 2 and 3 in figure 4, the next region to be processed is located in the different stripe (region 4 in fig. 4). To process the first few top block-lines of the region 4, due to the fact that the part of the SA partly lies within the region 1, few block-lines from region 1 would be needed. Hence, apart from being present within the region 1, these blocks need to be part of the region 4. This analysis could be extended for regions 5 and 6 and leads to the conclusion that the blocks located in the stripe marked with darkest gray in fig. 4 have to be fetched from the off-chip image memory two times. The height of the mentioned stripe is determined by the height of the search area and is equivalent to \( OVERLAP_Y = 2 \ast (SA_Y/2) \) where the integer division is used in the formula. The number of such stripes is defined with: \( CEIL((DIM_Y - RL_1Y)/(RL_1Y - OVERLAP_Y)) \) where \( DIM_Y \) denotes the height of the field/frame and integer division is used. The increase of the bandwidth towards the off-chip memory is directly proportional to this number.

**IV. Dynamic Aspect Ratios for Regions**

For blocks located close to the border of a region, motion vector candidates lie outside the region. In some cases, that particular region might not be processed, and hence the motion vector value is not up-to-date. For example, during a meandering from right to left, the right-most blocks have spatial candidate S2 (figure 2 is horizontally mirrored in this case) that lie in the neighboring region which is not yet processed. Similar situation occurs with the S1 candidate as well during the reverse scan within a region (scanning direction is from bottom-to-top). Therefore, artifacts arise at the border between the regions when a fixed aspect ratio...
is utilized. To reduce these artifacts, we propose to use the region-based motion estimation wherein the aspect ratio of a region dynamically changes from image to image. When performing motion estimation on one image pair we utilize one aspect ratio. For the next image pair, a different aspect ratio is utilized still maintaining the same area of the region. These two aspect ratios are utilized cyclically.

To be able to find the best aspect ratios for regions, we categorize the video sequences in four different categories regarding motion present in foreground and background:

1) Still images, resulting in non-moving foreground and background
2) The foreground is kept still (for example, the camera follows the main object in the screen) while the background is moving
3) The foreground moves through the frame while the background is kept at a fixed position in the frame
4) The most generic case, foreground and background are not kept at a fixed position in the frame

The first possibility results in a motion vector field with only null vectors, which is already produced at initialization. The motion is accurately estimated and therefore this possibility will not be taken into account any longer. The third case is viewed as the subset of the case 4. The last three cases are jointly analyzed in the next paragraphs.

In the last three cases, the foreground will consist of a couple of objects, which are most of the time covered by a few regions. Once the correct motion vector is found for an object during region-based scanning, it will be propagated into the neighboring regions during the next image-pair scan. Most likely, in just a few iterations, the complete object will then be covered by the correct motion vector. In contrast to the foreground, the background is covered by considerably more regions. In case the correct motion vector is found at the bottom of the image (in one of the lower regions), it will take considerably more time before the correct vector is propagated to the upper regions (see figure 5). To clarify this, we compare it with the situation where the image pair is scanned two times, first time from top-to-bottom and the second time from bottom-to-top. In this case, even if the correct value of the motion is caught at the bottom of the image, in the next scan it will propagate to the top. As figures 5b and 5c illustrate for the region-based case, the correct value will propagate only to the top and left region.

For regions exploiting dynamic aspect ratios, this convergence process also holds. Therefore, the choice of the aspect ratio is utmost important. If they are chosen improperly, an even slower convergence of the motion vector field, compared to a fixed aspect ratio, can be obtained as depicted in figure 6. In this figure, the correct motion vector is converged after three frames into four regions. Compared to the case with fixed aspect ratio, the correct motion vector is converged into five regions. We solve this problem by introducing extreme dynamic aspect ratios. As one aspect ratio, we select the shape of the regions as horizontal stripe covering the complete width of the screen. For the second aspect ratio, we select the shape of the regions as vertical columns covering the complete height of the screen.

In the case of extreme aspect ratios, the convergence process can be divided into three steps. In the first step, the correct motion vector is found in one of the regions. In the second step, this value propagates throughout the complete region with the reverse scan. In the third step, the estimation process starts with the next image pair and the regions are reconfigured. Taking into account their new shape, the correct value of the motion vector propagates throughout the complete screen. This process is depicted in figure 7. Fast convergence is important for the fact that the first estimate of a motion vector field is non-deterministic due to the pseudo-random motion vector update.

Another advantage of this approach is the bandwidth towards the off-chip image memory. Since it reduces to the case of stripe or column, it is possible to apply the sliding of the region through the image, as discussed in Section III. Thereby, the bandwidth requirements towards the image memory are 1 fetch per pixel.

V. Results

In order to test our method, we use the state-of-the-art de-interlacing algorithm, namely 2-dimensional generalized sampling theorem based de-interlacing as introduced in [7]. It is a three field motion estimation based algorithm where
the motion estimation is performed on the interlaced grid. We select this de-interlacing algorithm as the test application since it is very sensitive to the quality of motion estimation. We evaluated our scanning strategies on the test set containing six test sequences that are depicted in Figure 8. Since we did not utilize any sequence from the first category, these test sequences can be classified in one of the last three categories mentioned in Section IV. The *Girlsquare* and *Body* sequences belong to the second category, the *Bicycle* and *Renata* sequences to category 3, and the *Bond* and *Kiel* sequences to the last one. Zooming is taken into account via the *Kiel* sequence since this sequence does not have a uniform motion vector field for the background. In the *Bond* sequence, the foreground moves in the same direction as the background, only with a larger motion vector.

To denote the (dynamic) aspect ratio of the regions, will be indicated by the following notation: $\{(X_0, Y_0); (X_1, Y_1)\}$ where $X_0$ and $X_1$ denote the number of regions in the X-direction during two motion estimation scans on adjacent image pairs. Similar notation holds for the Y direction, $Y_0$ and $Y_1$ indicate the number of regions in the Y-direction. Note that for a fixed aspect ratio, the following equalities hold: $X_0 = X_1$ and $Y_0 = Y_1$. During the experiments, the extreme dynamic aspect ratios $\{(1,10); (12, 1)\}$ are utilized as well.

In the following calculations, the size of the search area is set to 9*5 blocks of 8*8 pixels. The sequences are standard definition (SD, 720*576 pixels). The amount of regions is not identical for motion estimation iterations on both adjacent image pairs, because the search area is not a square-shaped. For $(1, y)$ regions, in addition to the region, four extra block-lines have to be stored in the L1 scratchpad while for $(x, 1)$ regions eight extra block-columns need to be stored in the L1 scratchpad. The L1 scratchpad size, $S$, is set to:

$$S (\{(X_0, Y_0); (X_1, Y_1)\}) = \max (A(X_0, Y_0), A(X_1, Y_1))$$

where $A(X, Y)$ is the size needed to store $(X, Y)$ regions in the L1 scratchpad.

Table I. The Bandwidth of Data Bus 2 (DB2 BW) and Scratchpad (SP) Sizes of the Utilized Configurations. The Standard Definition (SD, 720*576 pixels) Frame Size is Assumed.

<table>
<thead>
<tr>
<th>regions</th>
<th>DB2 BW (fetch/pix)</th>
<th>SP size (kB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>${(11,10);(12,1)}$</td>
<td>1</td>
<td>202.5</td>
</tr>
<tr>
<td>(2, 4)</td>
<td>1.15</td>
<td>192.9</td>
</tr>
<tr>
<td>(4, 2)</td>
<td>1.06</td>
<td>206.6</td>
</tr>
</tbody>
</table>
dynamic aspect ratios have to be compared to results of a fixed aspect ratio that needs an L1 scratchpad size that is closest to the extreme dynamic aspect ratios. The bandwidth and scratchpad sizes for the test configurations are listed in Table I.

To measure the quality of the sequences, the mean square error (MSE) is utilized. The MSE of frame \( n \) is calculated by

\[
MSE(n) = \frac{1}{P} \sum_{\vec{x} \in MW} (F(\vec{x}, n) - F_{mc}(\vec{x}, n))^2
\]

where \( F(\vec{x}, n) \) is the luminance value of pixel \( \vec{x} \) at time instance \( n \) in the original (progressive) video sequence. \( F_{mc}(\vec{x}, n) \) is the luminance value of pixel \( \vec{x} \) at time instance \( n \) in the motion compensated de-interlaced video sequence. The measurement window (MW) consists of \( P \) pixels by \( L \) lines. The two outer block-lines and columns were excluded from the error calculation, which resulted in an \( MW \) of 688*544 pixels for an SD frame.

For the Bicycle and Girlsquare sequences, the MSE values of the first five frames are plotted in Figure 9. In the figure, it can be seen that the region-based motion estimation featuring dynamic aspect ratios converges the fastest. Especially in the case of the Bicycle sequence, the first estimate (determined by luck, i.e., pseudo-random update) is much worse for the extreme dynamic aspect ratios. However, in the second estimate, we see dramatic improvement as a result of dynamic aspect ratios applied.

The process of fast convergence is especially noticeable in the case of Girlsquare. It is also visually perceivable. To illustrate that, in figures 10 and 11 we provide the first three de-interlaced frames with motion vector overlay for the case of fixed and dynamic aspect ratios, respectively. Unlike in the case of fixed aspect ratios, in the case of dynamic aspect ratios, the convergence process is finished in the frame number 4. The complete convergence was prevented in the frame number 3 since there was the object that vertically spans across the screen (the girl and her shadow). Further, there is a clear perceivable difference in quality in fig 11 compared to fig 10. Starting from frame number 3, we see that the floor (tiles) looks much clearer in the case of extreme dynamic ratios, since the correct velocity is estimated.

VI. CONCLUSION

In this paper, we introduced the idea of dynamic aspect ratio of regions. We extended that idea to the extreme case where the region takes the shape of either vertical or horizontal stripe extending throughout the complete screen. We demonstrated that this idea offers fast convergence (especially noticeable in the specific class of sequences) independent of the results of the motion estimation performed on first few fields. Furthermore, this idea is more attractive in terms of bandwidth requirements towards the image memory compared to the classical region-based approach. Namely, unlike the classical region-based scanning, the application of this idea requires only one access per pixel in the off-chip memory.

REFERENCES


