II. BLOCK MATCHING ALGORITHM

A. Three Step Search

Three Step Search (TSS) [1] is one of the first non full search algorithm which use three steps. It is mainly used for real time video compression with low bit rate video application such as video conferencing and videophone. Fig. 1 illustrates an example of TSS algorithm. The search starts with a step size equal to or slightly larger than half of the maximum search range. In each step, nine search points are compared. They consist of the central point of the square search and eight search points located on the search area boundaries as shown in Fig. 1. The step size is reduced by half after each step, and the search ends with the step size of one pel. This search proceeds by moving the search area center to the best matching point in the previous step. We need three search steps for a maximum search range between 8 and 15 pels.

B. Diamond Search

The DS adopts two diamond-shaped search patterns [5], illustrated in Fig. 2: large diamond search pattern (LDSP) with nine search points and small diamond search pattern (SDSP) with five search points. The LDSP is repeated until that it reaches the edge of the search window, or a new minimum matching distortion point occurs at the center of LDSP. The search pattern is then switched to SDSP, which is used to refine the search algorithm [6].

C. Small Diamond Search

Small diamond search (SDS) is fast block matching motion estimation. Compared with FS algorithm, SDS algorithm requires very less computation while achieving acceptable performance. The SDS algorithm employs as search pattern a small diamond search pattern (SDSP) of 5 points as shown in Fig. 2 (b). The main improvement of this algorithm is the speed performances or the number of searching point. SDS reduces the number of searching point significantly if there is stationary or quasi stationary block.
D. Cross Diamond Search

Like the SDS, the Cross Diamond Search (CDS) [7, 8] is a fast motion search algorithm which uses one search pattern illustrated in Fig. 3.

This form is adapted for the fasts motions, but for the slow sequences SDS can give betters results. This algorithm can also be considered as derived from DS, the only difference it allows as shown in the Fig. 3 to improve research on the horizontal and vertical component of the motion. Fig. 4 illustrates an example of CDS path.

E. Hexagon-based search

HEXBS applies the same DS search strategy by replacing the diamond shaped search pattern with a hexagon-shaped search pattern which saves computational energy with slightly decreased performance [9]. This algorithm uses two search patterns: large hexagonal search pattern (LHSP) and small hexagonal search pattern (SHSP) illustrated in Fig. 5.

F. Nearest-Neighbors Search

The Nearest-Neighbors Search (NNS) algorithm [10] employs a novel motion vector prediction technique, a highly localized search pattern, and a computational constraint explicitly incorporated into the cost measure. When compared to the FS algorithm, the NNS algorithm can significantly reduce the number of computations. When compared to other fast search algorithms, the NNS algorithm provides better rate-distortion performance, while still requiring comparable or less computations. As illustrated in Fig. 6, the algorithm employs eight equal size layers, with consecutive layers having different centers and containing at most four untested candidate motion vectors.

III. THE PROPOSED HORIZONTAL DIAMOND SEARCH

Like the CDS, the HDS is a fast motion search algorithm with search pattern illustrated in Fig. 7. The HDS exploits the centre-biased characteristics of the real world video sequences by using a smaller initial step size. The objective of HDS is to reduce the number of operations needed for motion estimation. The aim is to achieve an acceptable image quality while independently targeting the reduction of the computational complexity.

As shown in the Fig. 7, the main advantage of this algorithm is to improve search on the horizontal component of the motion. Thus, in motion estimation, we assume that the objects move in a translational movement for, at least, a few frames.

IV. SIMULATION RESULTS

H.264 UBLive software developed by Ulvideo Inc. [11] is an encoder developed on C++ language, which is highly optimized algorithmically, enabling it to achieve objective and subjective performance levels close to the public JM encoder with significantly reduced time complexity, is used to implement these different algorithm (TSS, DS, CDS, HEXBS, and HDS). We use Foreman, Tb420, and Mobile in CIF (352x288) as test video sequences which consist of different motion contents. We use different quantification (QP = 30, 32, 36 and 38). SAD is used to evaluate the distortions for block matching with the block size of 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, and 4x4. The precision of interpolation is fixed at the quarter pixel.

The UBLive code uses the NNS algorithm. In this section we compare the performance of CDS, NNS, HEXBS and TSS algorithm to the HDS. To take advantage of the correlation of different block, the initial block-size and the accuracy of the MV prediction are important. We note that these algorithm are implemented using the same idea that the NNS. For example, for CDS, we found the predicted motion vector and after we execute the search CDS. The center of search is not the center point at (0, 0), but the predicted motion vector.

Rate distortion analysis

The rate-distortion (R-D) performance of BMA algorithm is shown in Fig. 8. It can be seen that the R-D curve of CDS and HDS is superior to that of NNS especially when the test sequence contains larger motion such as the Tb420 sequence. However, the performance of TSS is poorer.

Complexity analysis

In addition, we measured the computation time for H.264/AVC encoder on a PC P4 (2.66 GHz). Table I list the speed of H.264/AVC encoder, in terms of frame/second, using NNS, CDS, TSS, HEXBS and HDS algorithm. It can be noted that the complexity of HDS is very low compared to the other motion estimation
algorithm. But, NNS outperforms the HDS by at least 2%. It is significantly noticeable that The CDS, TSS, and HEXBS are very complexes compared to HDS.

**SSIM analysis**

For each sequence, we have two coded sequences one with the proposed HDS and the other with NNS, CDS, TSS, and HEXBS. We compute the Structural SIMilarity (SSIM) index [12, 13] for each macroblock (MB) in a frame, and we compare the value of SSIM obtained by HDS compared to the other algorithms. The SSIM index is a novel method for measuring the similarity between two images. It can be viewed as a quality measure of one of the images being compared provided the other image is regarded as of perfect quality. Fig. 9 shows the Foreman sequence for QP equal to 30. In this figure, the blue point shows that the SSIM using HDS is greater than the SSIM using one of the other algorithms for this macroblock. Otherwise, the macroblock is indicated with yellow point. Fig. 9 shows also, at the right corner, the number of macroblock that the BMA SSIM is better than HDS SSIM, and the number of macroblock that the HDS SSIM is better than BMA SSIM.

We can see that in all cases, there is an important number of macroblock that the HDS SSIM is greater than SSIM of other algorithms. According to these results, in most cases, the HDS produces a much better image quality compared to NNS, CDS, HEXBS, and TSS algorithm. The HDS achieves higher performance, and has a low complexity effort compared to NNS, CDS, HDS, and HEXBS.

**V. CONCLUSION**

The results illustrated in this paper lead us to conclude that if we use an appropriate search pattern that goes along with the nature of the motion, we can have a better estimation quality, and less important computing times since the block matching will be faster. Based on this strategy, we have proposed a new motion estimation algorithm entitled HDS that obey this rule. Simulation results demonstrate the fact that HDS outperforms the other BMA used in this study, such as NNS, CDS, TSS, and HEXBS. A major reduction in the computational complexity is obtained with better image quality depending on the motion characteristic of the video.

**ACKNOWLEDGMENT**

This work is supported by UBVideo Inc., Vancouver, CANADA. The authors would like to thank Pr. F. Kossentini for their contributions to this work.

**REFERENCES**


TABLE I

Complexity analysis of H.264/AVC in terms of frame/s with different search pattern: NNS, CDS, TSS, HEXBS and HDS, for QP=30.

<table>
<thead>
<tr>
<th>Sequences</th>
<th>CDS</th>
<th>HEXBS</th>
<th>TSS</th>
<th>HDS</th>
<th>NNS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foreman</td>
<td>20.98</td>
<td>21.05</td>
<td>24.74</td>
<td>29.81</td>
<td>30.72</td>
</tr>
<tr>
<td>Tb420</td>
<td>22.94</td>
<td>23.08</td>
<td>22.30</td>
<td>24.55</td>
<td>24.68</td>
</tr>
<tr>
<td>Mobile</td>
<td>27.51</td>
<td>27.08</td>
<td>24.77</td>
<td>28.53</td>
<td>28.74</td>
</tr>
</tbody>
</table>

Fig. 8. Rate-distortion curves for different motion estimation algorithms: NNS, CDS, TSS, HEXBS, and HDS, for Foreman, Tb420, and Mobile.

Fig. 9. Comparison of the SSIM performances for each macroblock for Foreman coded using HDS and CDS, TSS, HEXBS, and NNS, without loop filter (QP = 30).

if SSIM HDS > SSIM other BMA (CDS, TSS, HEXBS, and NNS) then Yellow MB, otherwise, Bleu MB.