NON-REDUNDANT SEARCH PATTERNS IN LOG-SEARCH MOTION ESTIMATION

Astrid Lundmark

Image Coding Group
Department of Electrical Engineering
Linköping University

ABSTRACT

Motion estimation is still the step in a video coding scheme that requires the largest part of the available computational resources. Many paths have been tried to reduce the computational complexity of the motion estimation step. Here we will describe how it can be reduced for log-search motion estimation by a simple change of search strategy. We will also examine how multiple search paths influence the performance of the motion estimation algorithm. Search patterns for log-search on a hexagonal grid are also suggested.

1. INTRODUCTION

To avoid the high complexity of exhaustive search over the search area, logarithmic search[1] and the closely related 3-step search [2], which will both hereafter be referred to as log-search, were introduced. They first make a coarse estimate of the motion vector and then successively refine this by searches in a smaller search area around the best match(es) found so far. Fig. 1 (a) shows how the search pattern looks like if a 3 by 3 search area is used in each step, and the size of the refinement vector is reduced by a factor of 2 between the iterations. From Fig. 1 (a) it is clear that many of the arrows meet, i. e. the search areas overlap. This can at first glance seem to be an advantage, since it allows some error recovery. However, the overlapping search areas imply a strange a priori probability distribution assumption for the motion vectors leading to sub-optimal decisions. Overlapping search areas also make the computational load unnecessarily high. Error recovery is handled well by using multiple search paths [3, 4].

Multiple search paths [3, 4] means keeping not only the best candidate motion vector at each iteration, but a pre-defined number of motion vectors. This improves performance at the expense of increased computational complexity.

Instead of the up to now prevailing partly-overlapping search pattern shown in Fig. 1 (a), we propose the non-overlapping search pattern of Fig. 1 (b).

Using this search pattern, larger motion vectors will be available within the search area for a fixed number of iterations, compared to the traditional search pattern of Fig. 1 (a). The maximum size of motion vectors for different number of iterations can be seen in Table 1.

The number of match criterion computations that have to be performed to find vectors of length l is roughly proportional to $\log_2(l)$ for the conventional scheme and to $log_3(l)$ for the proposed non-overlapping scheme.

The different vectors in the search area will be given uniform a priori probabilities, instead of the non-uniform probability distribution that is an effect of the partly-overlapping scheme of Fig. 1 (a).

i	l_2	l_3
1	1	1
2	3	4
3	7	13
4	15	40
5	31	121

Table 1. The longest motion vectors in full pixel resolution possible to reach by i iterations of log-search motion estimation using the search pattern in Fig. 1 (a) (l_2) and the proposed search pattern in Fig. 1 (b) (l_3).

2. SEARCH PATTERNS FOR NON-OVERLAPPING LOG-SEARCH

In the previous description in Section 1, and the experiments that will be described in Section 4, the size of the local search area, which will be denoted SSF, was 9 (3∗3). It would also be possible to use e.g. an SSF of $5 * 5$, or non-separable tilings [5].

Let us analyse what effect SSF has on the computational complexity of non-overlapping log-search matching. The number of match criterion evaluations, N, per estimated motion vector depends on the number of iterations of the algorithm, i, and SSF

$$N = i * SSF. \tag{1}$$

The number of pixels in the total search area, s, is given by

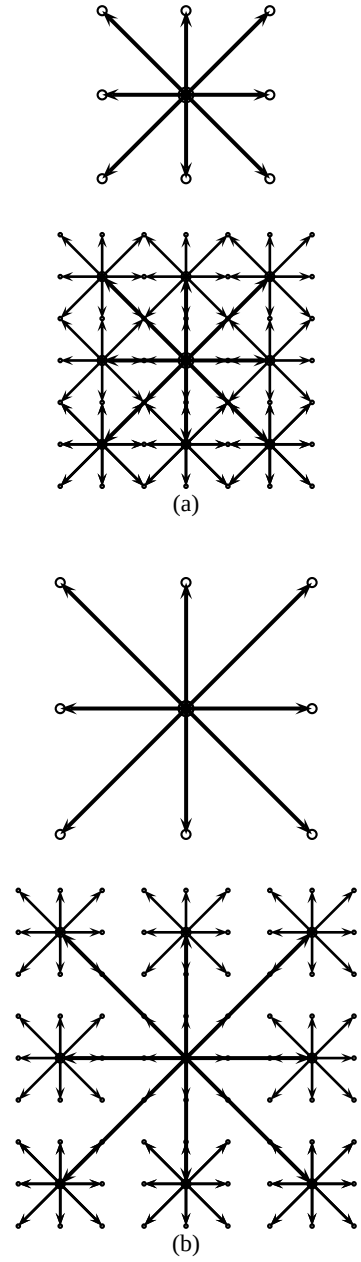
$$s = SSF^i \tag{2}$$

Combining equations 1 and 2, we obtain

$$N = SSF \log_{SSF}(s) = \frac{SSF}{\ln(SSF)} \ln(s) = C_{SSF} \ln(s). \tag{3}$$

We see that the complexity is proportional to the logarithm of s, hence the name log-search. The complexity also depends on SSF. How the complexity factor C_{SSF} varies with SSF is evaluated for some integer SSF:s in Table 2.

To try to lower the computational complexity, the number of positions evaluated in each iteration could be lowered to 5, by using the Koch-tiling pattern described in [5]. Four iterations of the search is shown in Fig. 2. This would result in search areas of the type shown in Fig. 3. For each iteration of the algorithm, the search area increases by a factor of 5, its form remains unchanged,



(a)

(b)

Fig. 1. (a): Two iterations of log-search matching with half as large refinement vectors in each step and a local search area of 3 by 3. In the first iteration (top image), 9 positions are searched. Around the best match(es) found, a refined search is made (bottom image). All possible outcomes of the algorithm are shown. (b:) The proposed logarithmic matching using refinement vectors with lengths $\frac{1}{3}$ of the sizes in the previous step, and a local search area of 3 by 3. All possible outcomes of the algorithm are indicated to show that it is possible to find all motion vectors.

SSF	C_{SSF}
2	2.8854
3	2.7307
4	2.8854
5	3.1067
6	3.3487
7	3.5973
8	3.8472
9	4.0961
10	4.3429
11	4.5874
12	4.8292
13	5.0683
14	5.3049
15	5.5390

Table 2. The computational complexity for non-overlapping log-search matching is minimized if three points are searched in each iteration.

but its orientation changes. The lengths of the vectors on the edge of the search area increase by a factor of $\sqrt{5}$. A disadvantage of this search pattern is that the search areas are quite irregular.

3. SEARCH PATTERNS ON A HEXAGONAL GRID

A hexagonal sampling of the plane is sometimes desired, since it gives sample areas that have more compact shape than the squares given by cartesian sampling. As far as we know, log-search matching on a hexagonal grid has not been presented earlier. Here we will give suggestions for search patterns for log-search on a hexagonal grid. First, we note that Equation 3 is valid also for log-search on a hexagonal grid. Using three search points in each iteration would therefore give optimal performance. Even if this is possible by using a search pattern giving so-called "fudgeflakes" [6] as search areas, these search areas would not be centered around the zero vector. To remedy this, we propose to use a search pattern consisting of 7 points. The search pattern is shown in Fig. 5. The resulting search area, shown in Fig. 6, is hexagon-like, but has a fractal borderline. By increasing the number of searches in each iteration, SSF, the fractal dimension of the border can be lowered, making the search region approach an ordinary hexagon [5]. However, since increasing SSF implies that the complexity factor C_{SSF} also increases, the suggested search pattern seams to be the most interesting one.

4. EXPERIMENTAL RESULTS

We used the complex-motion scene flowergarden to compare the performance of motion-compensated prediction using full search, traditional logarithmic search, and the logarithmic search proposed in Fig. 1 (b). Four iterations were made in both the logarithmic estimators. For the traditional search scheme, this gives a natural limitation in the length of motion vectors to 15. For the proposed scheme, we limited the search area to the same size, in order to be able to make fair comparisons. For the logarithmic search methods, the number of match value calculations was $36m$, where m stands for number of search paths. This number could be lowered if using the fact that some of the matches have already been calculated. That procedure, however, requires some overhead, and it is

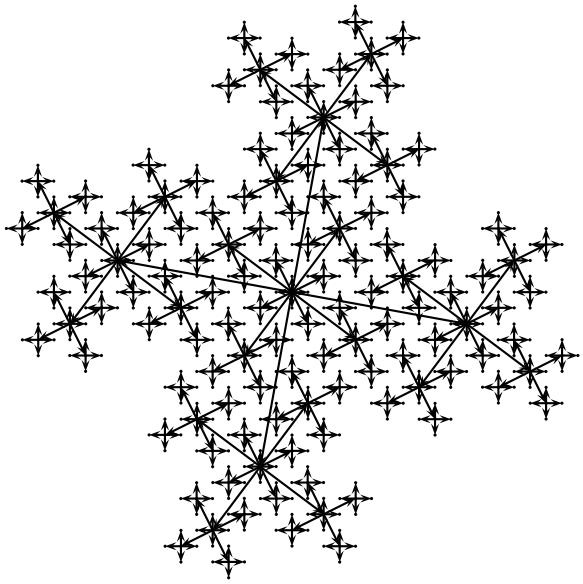


Fig. 2. Five positions, one of which representing zero motion, are searched in each iteration. Four iterations of the search are depicted.

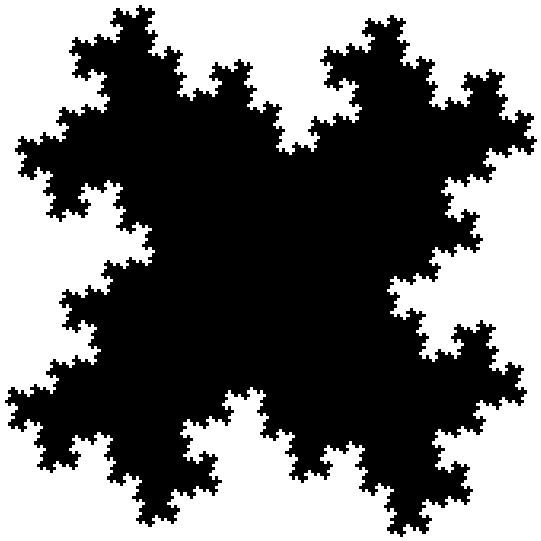


Fig. 3. After a few iterations the search regions obtain a fractal appearance.

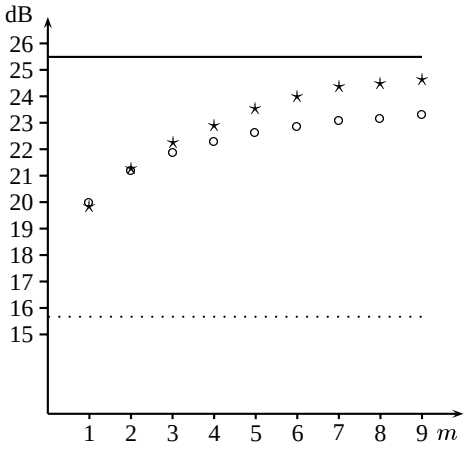


Fig. 4. Results of motion compensated prediction using no motion estimation (dotted line), full search motion estimation (solid line), log-search with traditional search pattern ($\circ$), and log-search with the proposed search pattern ($\star$). Test sequence was "flower garden" in 352 by 240 pixels resolution. One motion vector in full-pixel resolution was estimated for each 8 by 8 block in the image. The PSNR for the motion compensated prediction is given as a function of m, the number of search paths used in the log-search.

doubtful if it is meaningful when the match criterion is relatively simple. For full search, the search area was also 31 by 31 pixels, which gives 961 match value calculations. The outcome in terms of PSNR for the motion compensated prediction is shown in Fig. 4.

We see that motion compensation is really paying off, since using the previous picture without motion compensation gives a PSNR of only 15.7 dB, while using full search motion estimation gives a PSNR of 25.5 dB. The basic versions of logarithmic search using only one search path give PSNR around 20 dB (20.0 for the traditional search pattern and 19.9 for the proposed). Using more search paths improves PSNR substantially for the logarithmic motion estimation algorithms. The computational complexity increases linearly with the number of search paths. In this experiment, for 9 search paths, the computational complexity for logarithmic search is a third of the computational complexity for full search. The non-overlapping scheme has capacity for finding motion vectors up to a length of 40 with this number of iterations. If this capacity would be used, full search would need twenty times more match value calculations than logarithmic search with 9 search paths. For larger motion vectors, the relative difference between the required computational complexity for full search and logarithmic search increases, since the number of match value calculations rise with the square of the motion vector length, while the number of matches for logarithmic motion estimation rises with the logarithm of the motion vector length.

5. CONCLUSIONS

The proposed non-overlapping search scheme for log-search motion estimation shows improvements in computational complexity and proposes a more simple and straightforward alternative to the partly-overlapping schemes that have been presented so far.

Non-overlapping log-search is ideally suited for use with multiple search paths. The non-overlapping property also facilitates the computation of certainty measures for the motion vectors.

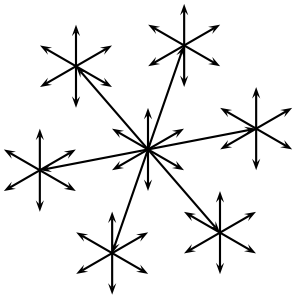


Fig. 5. Proposed search pattern for log-search on a hexagonal grid. Two iteration are shown.

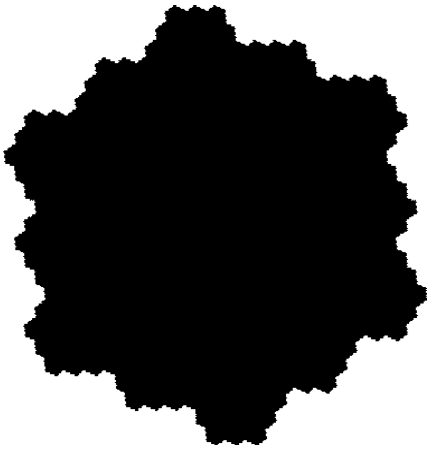


Fig. 6. Resulting search area for log-search with the proposed search pattern.

We have also suggested log-search matching on a hexagonal grid and given a search pattern for its implementation.

6. REFERENCES

[1] J.R. Jain and A.K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Transactions on Communications*, vol. COM-29, pp. 1799–1808, Dec. 1981.

[2] T. Koga, A. Iinuma, Y. Iijma, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," in *Proceedings of the National Telecommunications Conference*, 1981, pp. G5.3.1–G5.3.5.

[3] Anders Lindman, "Implementering och utvärdering av rörelseestimeringsalgoritmer för kodning av TV/HDTV," M.S. thesis, Linköping University, Sweden, Feb. 1992.

[4] K. W. Cheng and S. C. Chan, "Fast block matching algorithms for motion estimation," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 1996, pp. 2311–2314.

[5] Astrid Lundmark, Niclas Wadströmer, and Haibo Li, "Hierarchical subsampling giving fractal regions," *IEEE Transactions on Image Processing*, vol. 10, no. 1, pp. 167–173, Jan. 2001.

[6] Benoit B. Mandelbrot, *The Fractal Geometry of Nature*, W. H. Freeman and Company, 1982.