Two Novel Algorithms for High Quality Motion Estimation in High Definition Video Sequences

Diego Noble*, Marcelo Porto**, Luciano Agostini**, Ricardo M. Araujo**, Luis C. Lamb*

*Institute of Informatics, Federal University Rio Grande do Sul
Porto Alegre, RS, Brazil
<dvnoble,lamb>@inf.ufrgs.br

**Technological Development Center, Federal University of Pelotas
Pelotas, RS, Brazil
<mporto,agostini,ricardo>@inf.ufpel.edu.br,

Abstract—In this paper, we propose two new algorithms for high quality motion estimation in high definition digital videos. Both algorithms are based on the use of random features that guarantee robustness to avoid dropping into a local-minimum. The first algorithm was developed from a simple two stage approach where a random stage is complemented by a greedy stage in a very simple fashion. The second algorithm is based on a more refined class of algorithms called Memetic Network Algorithms where each instance of the search may exchange information with its neighbour instances according to some rules that control the information flow. The proposed algorithms were implemented and tested exclusively with high definition sequences against well known fast algorithms like Diamond Search and Three Step Search. The results show that our algorithms can outperform other algorithms in quality yielding an increment in complexity that may be amortized if resources for a parallel execution are available. Additionally, we provide further evidence that fast algorithms do not perform well in high definition.

Keywords-Motion Estimation; Video Coding; High Definition

I. INTRODUCTION

Most modern video coding standards like H.264/AVC, for example, use block motion estimation (BME) together with motion compensation (MC) as a way to reduce or eliminate temporal redundancy in uncompressed video streams [1]. By doing so, BME and MC account for a large proportion in the reduction of the bit rate. However, this technique also represent the most computationally complex module of the coder [2]. In fact, motion estimation can consume up to 60% of the total encoding time of the H.264/AVC codec when just one reference frame is used [3].

To tackle this complexity, several fast algorithms have been proposed. Such algorithms rely on heuristics and metaheuristics to guide how the sampling of the solution space is conducted. Some well known representatives of this class of algorithms include: The Three Step Search algorithm (TSS) [4], one of the first fast ME algorithms proposed; Diamond Search (DS) [5], which uses two geometrically shaped search patterns, and Hexagon-Based Search (HEXBS) [6].

For applications where low bit rate is a key issue, e.g. cell phones, all these algorithms perform well provided that, in this context, fast and complex motion tends to be infrequently [7]. In this paper, we will present two novel algorithms for motion estimation in High Definition (HD) which were specifically designed to achieve high quality in this context while enjoying comparable computational complexity of fast algorithms previously developed. Furthermore, we will also show evidence that fast algorithms do not produce satisfactory results in face of HD digital video coding. Hence the importance to address inherent features of HD in the development of modern ME algorithms.

This paper is organised as follows: Section II gives an overview of the basics of motion estimation and related work. Section III will introduce in detail our two proposed algorithms. In Section IV, experimental results and their impact are described. Finally, we draw the conclusions and point out future research directions.

II. ON MOTION ESTIMATION

An uncompressed digital video has many kinds of information redundancy. One of them is temporal redundancy, which is related to the spatial similarity that temporally correlated frames have. This superfluous information is an effect of the temporal sampling of a video where, commonly, 30 frames are condensed in a second so a human viewer can have a feeling of real-time movement. For a digital video to be more efficiently transmitted or stored, one needs to eliminate this information redundancy with within limits of what is acceptable in terms of image quality loss.

The ME/MC module of a block-based video encoder attempts to reduce the temporal redundancy by compensating the motion in a video through translating or warping the samples of the previously transmitted reference frame, which in its turn is likely to very similar to the actual frame yet to be encoded. The resulting motion-compensated predicted frame is then subtracted from the current frame to produce the residual frame [8], which is potentially a sparse matrix that can be more easily coded by the next stages of the coder. The more efficient the ME process is, the more efficient the compression becomes because less residual energy, or error, needs to be encoded and

less information should be discarded in the quantization step for a fixed bit rate.

In a block-based video coding scheme, a frame is subdivided into non-overlapping blocks of pixels. These blocks may have sizes of $4 \times 4, 8 \times 8, 16 \times 16$ or any combinations of these. The size of the block used and the possibility of adopting multiple block sizes are completely dependent upon the video standard in question in these choices considerably affect the coding efficiency and complexity of the ME process [9].

The motion vector (MV) for a given block is achieved by conducting an algorithmic search which tries to minimize the value of a matching criterion. The most commonly used and one of the simplest criterion is the sum of absolute differences [2], [8], or SAD, which is calculated from the current block (which comes from the actual frame) and an equally sized reference block (which comes from the reference frame). Equation (1) depicts the SAD formula between two blocks of the same size where R represents the reference block and C, the current block.

$$SAD(R, A) = \sum_{i=0}^{N} \sum_{j=0}^{N} \left| R_{(i,j)} - C_{(i,j)} \right| \quad (1)$$

The spatial distance (measured in Cartesian coordinates) related to the best of all candidate reference and the current block is kept and it represents the MV. Regularly, the search for the MV is limited by a range in horizontal e vertical axis, called *search window*. This simplification significantly reduces the number of reference blocks to be analyzed. The search window may also be referenced as search range in the sense that a search window of $N \times N$ is equivalent to a search range of $(\pm N/2, \pm N/2)$. In this work, both concepts will be used interchangeably kept the right meaning.

The complexity of a Block Matching-based encoder largely depends upon motion estimation and the rate-constrained control [1]. However, the main goal of ME is to predict the actual frame as precisely as possible so less residual information needs to be processed and then transmitted. These two objectives are contradictory because usually the more precision is required, the more computationally demanding the search becomes. The trade off between these aspects depends on the application.

When precision (and thus objective quality) is the main objective, the Full Search (*FS*) algorithm is the best candidate since it evaluates all the possible reference blocks within the search window. However, due to the very high computational cost that this exhaustive search method demands, practical applications are hindered from using it. Generally, algorithms developed for ME try to maximize PSNR while minimizing the computational effort. It is useful to define the most used objective quality metric [10], the peak-signal-to-noise ratio (PSNR), shown in equation (2), in it R represents the reference frame and C, the current frame.

$$PSNR = 20 \cdot \log \left(\frac{255}{\sqrt{\frac{1}{N^2} \sum_{i=0}^{N} \sum_{j=0}^{N} \left(R_{(i,j)} - C_{(i,j)} \right)^2}} \right) \quad (2)$$

A. ME in High Definition

In our findings, we compared SAD maps so we could have a visual comprehension on how the solution space would seem like. These maps were built using the Full Search algorithm by plotting SAD values into a Cartesian coordinate system where each point of these maps represent the SAD value of that block. Figure 1 depicts one set of these maps for the same region of the same video in different resolutions. Note that the search window of each map remains proportionally the same and the center of the space solution is represented exactly at the center of a relative map where darker regions indicate better block matchings than brighter areas. The resolution is seen at the bottom of the respective caption.

In Figure 1a, the global optima can be clearly seen, near the center. This pattern is somehow common for sequences in this resolution. In an optimization context, a greedy approach would yield very satisfactory results since this search process will converge to a global optima solution most of the time. This hypothesis partially explains why fast algorithms reach almost the same quality performance of full search for low definition sequences.

Nevertheless, for a larger resolution in Figure 1b, two regions can be seen in the map. Both, can potentially hold the global optimum and break the previous pattern. The ME in this map can be considered more difficult than in 1a since one needs to widen the search to better evaluate the solution space. This observation is also valid for an enhanced definition sequence in Figure 1c, where the map gets even rougher and one may not notice with certainty the difference between the global optima and local optima. Estimating motion in high definition gets even harder as can be seen in Figures 1d and 1e where a very complex and rough map with lots of valleys and hills that may intricate the problem even further.

Generally, as resolution grows the motion estimation also becomes more difficult in the sense that more search points may need to be evaluated. Considering the recently available consumer electronics devices such as HD digital video broadcasting [7], this is an evidence that ME in HD may become a central issue in the forthcoming years. To the best of our knowledge, there is a paucity of results about this issue, which requires further research.

B. Related Work

Several algorithms have been proposed to efficiently find motion vectors. All the previously mentioned algorithms could be seen as classical algorithms, because they have influenced many modern algorithms and techniques. They are generally considered fast algorithms because they do not search the

(a) LD 144p   (b) UMD 272p

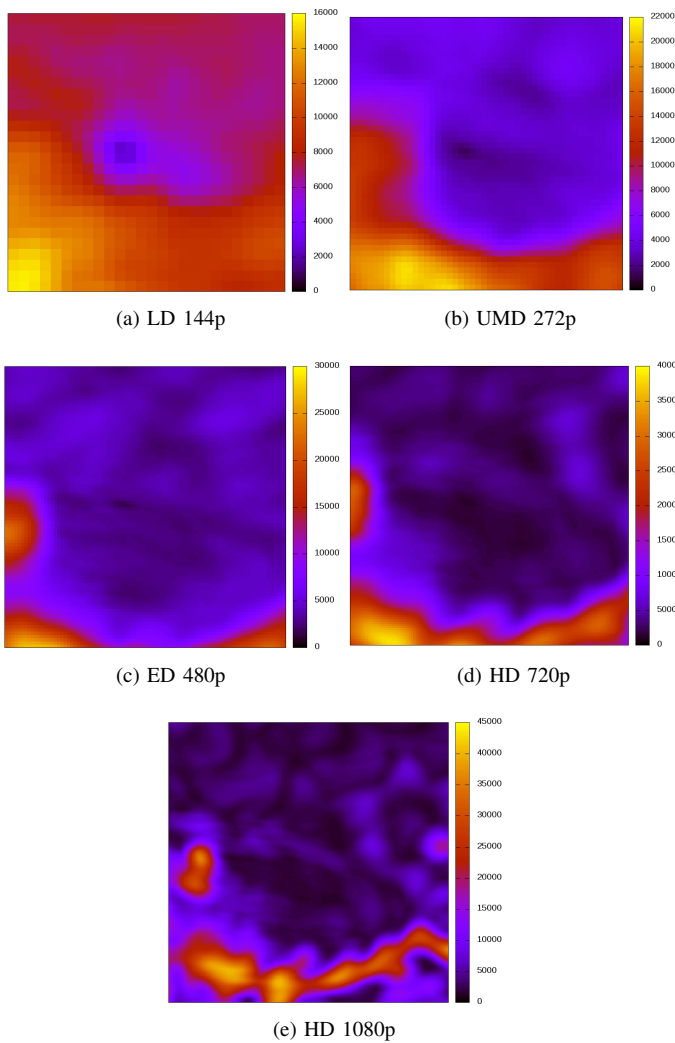(c) ED 480p   (d) HD 720p

(e) HD 1080p

Fig. 1: SAD maps for the block in different resolutions of a same sequence.

entire solution space. As a result, the motion vectors they find are not necessarily the optimum ones.

Three Step Search (TSS) was one of the first algorithms proposed to deal with this problem [4]. The TSS algorithm selects nine search points: one at the center and eight concentrically positioned points at the same distance. This pattern is repeated two times; at each step, the center of the eight points is the best evaluated block from the last step and the distance is divided by a factor of two. This algorithm was very successful and widely adopted in the early stages of video coding [11] due to its simplicity and regularity. The NTSS [12] algorithm is more recent improvement over it.

The Diamond Search Algorithm [5] is a well-known algorithm that led or influenced various other algorithms like the Hexagon Search algorithm (HS) [6] which uses a hexagon shaped geometry to execute the search. In its turn, HS has spawned Unsymmetrical-cross Multi-hexagon-grid Search (UMHexagonS) [3]. The work of [2] proposes the improve-

ment of the UMHexagonS algorithm. In [13], the three-dimensional predict hexagon search (3DPHS) algorithm is proposed. This algorithm uses a rood-shaped search pattern at the fist two searching steps with a higher probability to get motion vectors and it can predict the object movement in horizontal and vertical direction. Most of these algorithms rely on using techniques to improve a common ancestor and in fact should be regarded as algorithmic improvements but not, up to a certain extent, fully original algorithms.

None of these algorithms is focused on high definition or is evaluated as such. In [14], the Dynamically Variable Step Search (DVSS) algorithm is proposed. However, qualitative results in terms of mean absolute differences (MAD) are not carried out in high definition sequences although the hardware architecture proposed is capable of processing 1080p video streams in real-time. In [15], two new random search algorithms are proposed, but their conclusions so far are highly prone to deviate from a HD real case scenario since the evaluation is conducted for only 20 frames of two non-HD sequences.

The need for developing algorithms for high definition ME has only been recently addressed. In [7] the Recursive Dynamically Variable Step Search (RDVSS) ME algorithm for real-time processing of HD video formats is proposed. This algorithm is an improvement over the DVSS algorithm and it dynamically determines the search patterns that will be used for each block based on the MVs of its spatial and temporal neighboring blocks. The algorithm qualitative evaluation is solely performed in high definition videos sequences. Although no quantitative evidence about why fast algorithms perform poorly on HD case studies, the results presented point the same tendency that our qualitative analysis.

III. THE PROPOSED ALGORITHMS

In this section, we will introduce our algorithms and explain each one accordingly.

A. The Random Search Algorithm RS4

Considering that motion estimation is a non-convex optimization problem, randomized techniques are a common tool in this kind of optimization [15]. The sole use of a greedy heuristic to guide the search would not suffice to accurately estimate all kinds of motion. This can be intuitively perceived by analyzing Figure 1e. Following this idea, it is possible to draw that combining these two approaches may have some advantage over a single particular approach. Howerver, we also assume that a completely random search is not efficient since it ignores some common and well-known patterns of motion estimation.

Discovering a new pattern that can be explored to improve the ME is a central issue in the design of motion estimation algorithms. For instance, it is commonly assumed that the matching error will decrease monotonically when approaching the global best point [3]. It is important to know that this assumption does not hold true for video sequences especially for those with large motion content [5]. These statements may

seem contradictory but by joining these ideas with random search, we can derive that a previously applied random search step could open some solutions of the whole space and then select the best one for positioning the local step search. This local step search can be guided by a greedy heuristic and may use even a common geometrical search shape.

Assuming that high definition sequences particularly benefit from large search ranges [16], the randomization becomes a very attractive approach and it is based on these observations we propose a novel algorithm: the *Random Search 4* (RS4) algorithm.

The search in RS4 is done in Two Steps: the random step and the iterated local step. Before the execution of the random step, the central search point and its four neighbour points at distance 1 are evaluated. Thus, this step guarantees that the search is executed at the middle where for stationary takes the best candidate is likely to be. Then, with uniform probability N points within the search window are chosen to be evaluated (random step). The best candidate among the $5+N$ is set aside so that its position will be used as the center for the iterative local step. This step step iterates using the SDSP pattern [5] until the stop criteria is reached, that is the new evaluated block presents no advantage over the last best candidate.

B. The Memetic Network Algorithm MNA-ME

Memetic Networks is a class of algorithms that define a population which is able to communicate through a network [17]. This approach has the advantage of being flexible enough to adapt to many problems of optimization. Differently from other multi-agent based algorithms, the performance of a Memetic Network is strongly affected by the way the agents are connected, how they exchange information and how this information affects their current context. The Memetic Network model can be seen as a model of cultural evolution in the sense that a *meme* (that is a piece of information) may spread through a population in a fast manner when compared to a *gene* in a Genetic Algorithm (GA). An instance of a memetic network is created from three well-defined rules where each rule applies equally to all agents:

I *Connection Rule*: tells how the agents should connect, that is, to whom one agent should connect to. Distinct rules will generate distinct topologies where some topologies are more adequate to particular problems than other. This rule definition allows one to borrow theorems from network science.

II *Aggregation Rule*: this rule is responsible to manage how one agent's meme (or solution) should influence their respectively connected agents.

III *Appropriation Rule*: tells how the agent should manipulate the meme it has. For instance, it could be any metaheuristic used in a local iterated search algorithm.

As one may realize, this model is extremely flexible in the sense that rules are defined from a very abstract context. Moreover, this model is extremely powerful because it could unify different metaheuristics and techniques into a single and hybrid instance.

The memetic network model has some advantages over a purely GA approach. One of them is that an instance of this model may converge faster than a GA to a good solution and this is particularly interesting in the case of the ME problem especially for real-time video coders. This advantage and the structural simplicity of a memetic network form together the main motivation to the use of an instance of this model focused on the HD ME problem. We thus propose a new memetic network-based algorithm called MNA-ME.

In the MNA-ME, one agent always start in the center and the others should start in random positions inside the search window. Given that, the three rules are defined as follows:

I *Connection Rule*: each agent should necessarily connect to the central agent and with the agent who holds the best current solution. This way, each agent should know where is the best ranked agent, that is the best block matching achieved so far. This rule is represented by a matrix which is updated at each iteration.

II *Aggregation Rule*: the agent which has incoming connections spatially "attracts" the connected agent by a factor named *aggressiveness* denoted by α. The higher this factor is, the stronger will be its influence.

III *Appropriation Rule*: the agent, after exchanging information, performs a full search in the range $(\pm 1, \pm 1)$ and changes its current location to better solution if one is found.

It is useful to further explain the *aggressiveness* parameter which denotes the factor by which an agent should change its positions according to connections made so far. This parameter may vary from 0.0 to 1.0 but is fixed throughout the computation. The value 0.0 denotes no influence at all, while the value 1.0 replicates the best agent rendering itself irrelevant. A value of 0.5 puts the connected agents at the middle of the original distance between them.

IV. EXPERIMENTAL RESULTS

As our objective with both algorithms is to achieve high quality in high definition, the test sequence set should be composed exclusively of HD sequences. These sequences are freely available at [18] and their resolution is 1920x1080 pixels progressive (1080p). Motion estimation and motion compensation were executed solely on luminance samples and the PSNR was obtained comparing the original frame and the motion compensated frame that is, the output of the MC (the residual frame was discarded). This decision is justified in the sense that this work is not tied to a single video coding standard, but rather to a conceptual point of view. Adopting this principle is useful in the sense that it contributes to the generalization of our contribution.[1]

Our main complexity metric is the number of evaluated search points (ESP). Since the processing time grows linearly with the number of evaluated blocks, the time length of each simulation was also discarded. The block size used was fixed

---

[1]The algorithms RS4 and MNA-ME were implemented in C so they could, in principle, be evaluated in terms of complexity and objective quality.

in 4×4 pixels because more vectors would be generated and consequently a more precise motion estimation process would be carried out. Using a small block size is not a problem at all since a bottom-up variable block size algorithm [19], [20] can be used to achieve larger block sizes that in turn may reduce the number of vectors to be encoded. Thus improving the coding efficiency of the coder as a whole. No SAD sampling technique was used in our experiments and only the first two hundreds frames of each sequence were considered (approximately 8 seconds).

Since the RS4 and the MNA-ME are essentially stochastic algorithms, their results may change from distinct executions for the same input. This feature renders the need of running the execution many times, so the expected behaviour can be better know. For our experiments, both algorithms were executed ten times for each sequence for a given set of parameters and the mean absolute deviation rendered itself not significant, being less then 0.02 in average for PSNR. Figure 2 presents the results of the RS4 algorithm for each sequence video test set where each line represents a different size of the search window used. A search window of 16×16 means that the range of search is from -8 to $+8$ and so on. The number of search points chosen for the random step was 16 since this value is approximately the upper bound for the DS algorithm with 4 iterations.

The first result to be noticed is that sequences like *pedestrian area* and *riverbed* can be better coded using a bigger search range. These sequences are motion intensive and the 64×64 search window presents considerable gains over the 32×32 search window of 1.46db and 1db respectively. These values are considerable high considering the logarithm scale of the PSNR metric. On the other hand, the RS4 algorithm shows better efficiency with a small search window on sequences with low motion information, namely *station2* and *sunflower*. For the sequences *tractor* and *rush hour*, the 32×32 search window yielded the best results.

The MNA-ME algorithm was tested with the fixed connection rule described in Section III and three agents only to keep complexity under an acceptable limit. However, the aggressiveness parameter was evaluated in the range $[0, 1]$ with 0.2 incremental steps. This evaluation would allow us to better understand how much a single node may interfere on its neighbor nodes in different space solutions. For example, a rough space solution search may be privileged by a higher aggressiveness since less time would be spent in potentially low quality solutions. In Figure 3, the results of this study are presented.

All results in Figure 3 were evaluated for a window size of 256x256. The decision to use such a relatively large search area is to make the motion estimation difficult especially for the random step so a real hard test would be endured by the MNA-ME algorithm. According to the results, there is a correlation between aggressiveness and the objective quality which holds true for all sequences except *sunflower* and *station2*. The optimum value for this parameter considering the test set was 0.2, which is a relatively small value.

For a more wider comparison with our algorithms, one should carry out evaluations of classical algorithms even if their focus is not HD. Thus, for completeness we considered both fast classical algorithms and one exhaustive search. The former with relatively large fixed search window range and the later with different search window ranges. This data is available in Table I where the average results of each algorithm for the test sequences are introduced. The table contains the quality metric (PSNR in db) and the complexity metric (ESP) for a given search range. The algorithms implemented and evaluated were: the Diamond Search (DS) algorithm, the Three Step Search algorithm (TSS), the 4 Step Search algorithm (4SS) and the Full Search algorithm.

TABLE I: Comparative of the proposed with the classic algorithms.

Algorithm	PSNR	ESP ($\times 10^9$)	Search Range
RS4	**34.43**	**0.723**	**($\pm 8, \pm 8$)**
RS4	**34,87**	**0.716**	**($\pm 16, \pm 16$)**
RS4	**34,82**	**0.719**	**($\pm 32, \pm 32$)**
RS4*	**35,51**	**-**	**-**
MNA-ME	**33,27**	**0.983**	**($\pm 128, \pm 128$)**
TSS	32.77	0.697	($\pm 64, \pm 64$)
4SS	34.7	0.928	($\pm 64, \pm 64$)
DS	33.46	0.637	($\pm 64, \pm 64$)
FS	35.08	7.429	($\pm 8, \pm 8$)
FS	38.33	26.542	($\pm 16, \pm 16$)
FS	40.21	106.168	($\pm 32, \pm 32$)
FS	41.31	424.673	($\pm 64, \pm 64$)

Except for the range $(\pm 8, \pm 8)$, the RS4 algorithm presented PSNR gains over the DS, TSS and 4SS algorithms. In the case of RS4 algorithm in the search range of $(\pm 16, \pm 16)$ gains of 2.1db, 1.41db and 0.17db in PSNR were achieved over TSS, DS and 4SS respectively. Considering the ESP, the RS4's complexity is very close to the TSS and is just 11% higher. It should be noted that a fixed N value was used and that this ESP is closely related with it. As it can be seen, the RS4 algorithm better evaluates a search window when compared to other fast classical algorithms. This in turn reflects in less burden to memory buses since small data chunks are needed. Compared to the smallest FS, the RS4 achieves in average 0.65db less in PSNR but does approximately 10 times less block operations. Since the FS explores all possible candidate blocks, its ESP grows exponentially as the search range grows. However, the PSNR seems saturated at search ranges larger than $(\pm 64, \pm 64)$. The RS4* row presents the average result of the best search range among $(\pm 8, \pm 8)$, $(\pm 16, \pm 16)$ and $(\pm 32, \pm 32)$, that is the range maximize PSNR for a given sequence and its relevance is to show that a adaptive search range technique may improve considerably the quality results for a fixed N value. Furthermore, it is important to note that the results in this case were better than the Full Search algorithm.
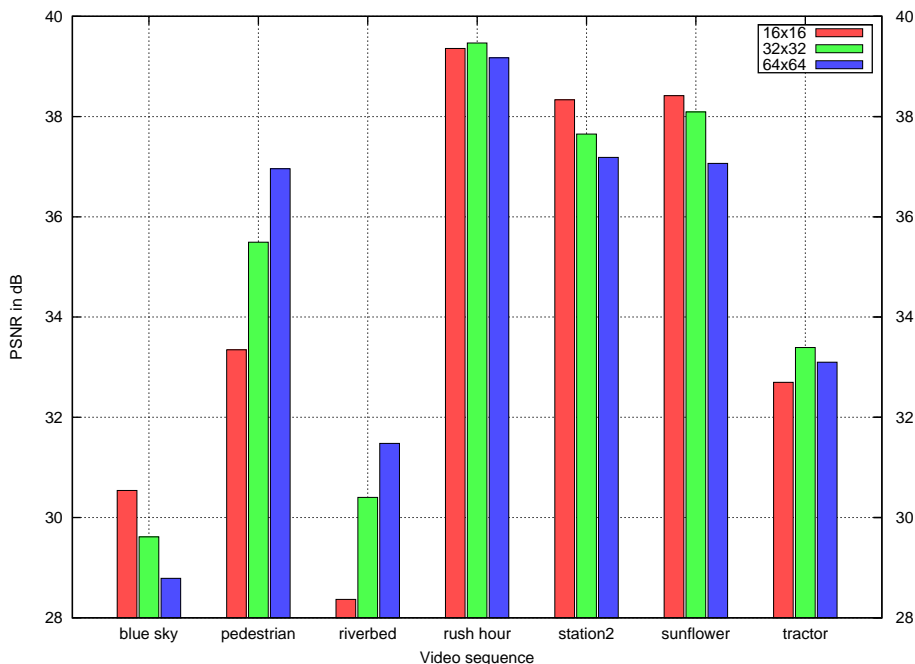
Fig. 2: The effect of search window size in RS4 PSNR results.

The MNA-ME algorithm did poorly in quality and complexity since it did not outperform the DS and the 4SS, achieving quality gains only over the TSS algorithm. These results present evidence that our current approach may not be the optimal one. In particular, the failure to achieve a good quality on HD ME merits further investigation about the problem, instance of the model and the model itself. This may also suggest that a fundamentally different approach is needed, combining a more straightforward approach in the design of the algorithm. One hypothesis is that distinct agents may leave too soon their current local optimal in favour of other agent information. This in turn could be a trap since the former agent could possible reach a better solution give time for it. Moreover, the agent that provided the information could be stuck in a local minimal.

A possible solution would be allowing to have agents that do not share information neither allow others to see their solution since this approach would allow some agents to better explore its neighbour blocks. Another solution would be adopting an information exchange strategy similar to a simulated annealing where one may locally explore its close solution at the beginning and as time advances they become more willing to influence of other agents. This approach may guarantee better local space solution evaluating we believe.

Comparing the results of new algorithms would be *per se* a relatively important contribution but this is not the focus of this work. For simplicity, we assume that the improvements they present over the original classical algorithms are essentially dependent of the performance of the root algorithm for a specific video sequence.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we introduced two novel algorithms for high quality motion estimation for high definition coding. The first algorithm, the RS4, relies in an initial random sampling that is further refined by a greedy search step. Although very simple, the RS4 algorithm presented good results with the advantage that it could be better tuned in real time by adjusting the N parameter dynamically according to the video input. This feature has not been implemented yet, but it has the potential to reduce the number of search points calculated for low motion sequences and then spend this saved computations when needed and thus providing a better solution at all. Additionally, dynamically adjusting the search window and using a motion prediction vector algorithm may help the RS4 algorithm to achieve a better trade off between complexity and quality.

The other algorithm was based on a population-based stochastic optimization class of algorithms in which individuals exchange information through the underlying network. This algorithm was based on a memetic network algorithm. The rules were appropriately defined for the ME application and the resulting instance was called MNA-ME. This instance was tested in for high definitions sequences and our results implied that the choice of the rules (especially the connection rule) may not be the optimum one since the results were somewhat limited in terms of quality and complexity. Perhaps, this empirical study also revealed more fundamental issues related to the MNA class. Although addressing and solving these issues with the model is not the focus of this work, it is worth to mention that one still lacks an efficient methodology to instantiate an optimization algorithm of this class for a
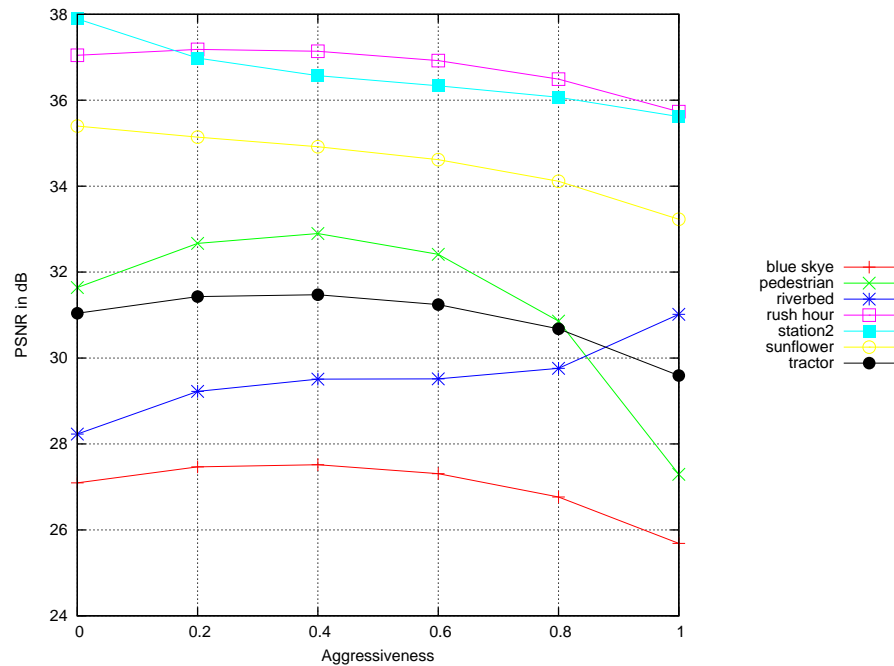
Fig. 3: Variation of the aggressiveness for the MNA-ME algorithm.

specific problem. This issue is completely understandable in the sense that MNA is a relatively new general model for solving problems.

Another contribution of this work is that we provided further evidence that fast algorithms do not perform well for high definition video sequences and that all the work related to this field should consider addressing this point by executing the simulations for HD contents also. This point is consistent with the ever increasing demand for high definition content to the final user. Moreover, we believe that it is important to bring this point into consideration.

As future work, we pretend to improve both algorithms and implement them inside a real coder so that bit rate data could be obtained from simulations. For instance, new topologies should be evaluated for a new MNA-ME instance like small-worlds, hierarchical and sparsely-connected topologies, etc. We also pretend to experiment SAD subsampling techniques with our algorithms since they do not affect the quality in a significant way [21]. Besides, a hardware implementation in an FPGA device for RS4 algorithm is currently being planned.

REFERENCES

[1] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, "Video coding with h.264/avc: Tools, performance and complexity"," *IEEE Circuits and Systems Mag.*, 2004.
[2] T. Toivonen and J. Heikkila, "Improved unsymmetric-cross multi-hexagon-grid search algorithm for fast block motion estimation," in *Proceedings of the IEEE International Conference on Image Processing*, oct. 2006, pp. 2369–2372.
[3] Z. Chen, J. Xu, Y. He, and J. Zheng, "Fast integer-pel and fractional-pel motion estimation for h.264/avc," *Journal of Visual Communication and Image Representation*, vol. 17, no. 2, pp. 264–290, 2006.
[4] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," *Nature*, 1981.
[5] S. Zhu and K.-K. Ma, "A new diamond search algorithm for fast block-matching motion estimation," *IEEE Transactions on Image Processing*, vol. 9, no. 2, pp. 287–290, feb 2000.
[6] C. Zhu, X. Lin, and L.-P. Chau, "Hexagon-based search pattern for fast block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 5, pp. 349–355, may 2002.
[7] O. Tasdizen and I. Hamzaoglu, "Recursive dynamically variable step search motion estimation algorithm for high definition video," in *Proceedings of the 20th International Conference on Pattern Recognition (ICPR)*, aug. 2010, pp. 2354–2357.
[8] I. Richardson, *Video codec design: developing image and video compression systems*. Wiley, 2002.
[9] C. Cai, H. Zeng, and S. K. Mitra, "Fast motion estimation for h.264," *Image Communication*, vol. 24, no. 8, pp. 630–636, 2009.
[10] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, april 2004.
[11] L.-M. Po and W.-C. Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 313–317, jun 1996.
[12] R. Li, B. Zeng, and M. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 4, no. 4, pp. 438–442, aug 1994.
[13] T.-H. Tsai and Y.-N. Pan, "A novel 3-d predict hexagon search algorithm for fast block motion estimation on h.264 video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 12, pp. 1542 –1549, dec. 2006.
[14] O. Tasdizen, A. Akin, H. Kukner, and I. Hamzaoglu, "Dynamically variable step search motion estimation algorithm and a dynamically reconfigurable hardware for its implementation," *IEEE Transactions on Consumer Electronics*, vol. 55, no. 3, pp. 1645–1653, august 2009.
[15] S. Boltz and F. Nielsen, "Randomized motion estimation," in *Proceedings of the IEEE International Conference on Image Processing*, sept. 2010, pp. 781–784.
[16] J. Vanne, E. Aho, K. Kuusilinna, and T. Hamalainen, "A configurable motion estimation architecture for block-matching algorithms," *IEEE*

Transactions on Circuits and Systems for Video Technology, vol. 19, no. 4, pp. 466 –477, april 2009.

[17] R. M. Araujo and L. C. Lamb, "Memetic networks: Analyzing the effects of network properties in multi-agent performance," in *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI-08)*, 2008, pp. 3–8.

[18] Xiph.org Test Media Repository, 2011. [Online]. Available: http://media.xiph.org/video/derf/

[19] I. Rhee, G. Martin, S. Muthukrishnan, and R. Packwood, "Quadtree-structured variable-size block-matching motion estimation with minimal error," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 10, no. 1, pp. 42–50, feb 2000.

[20] R. S. Dornelles, F. M. Sampaio, and L. V. Agostini, "Variable block size motion estimation architecture with a fast bottom-up decision mode and an integrated motion compensation targeting the h.264/avc video coding standard," in *Proceedings of the 23rd symposium on Integrated circuits and system design*. New York, NY, USA: ACM, 2010, pp. 186–191.

[21] F. Moschetti, M. Kunt, and E. Debes, "A statistical adaptive block-matching motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 5, pp. 417–431, may 2003.