Motion Estimation based on Artificial Fish-Swarm in H.264/AVC Coding

[1]CHUN FEI, [2]PING ZHANG, [3]JIANPING LI
[13]School of Computer Science & Engineering
[2]School of Optoelectronic Information
University of Electronic Science and Technology of China
Chengdu 611731
CHINA
[1]chfei1231@gmail.com [2]pingzh@uestc.edu.cn [3]jpli2222@uestc.edu.cn http://www.uestc.edu.cn

Abstract: - Motion estimation plays a key role in H.264/AVC video coding, but it is the most time-consuming task in the encoding process. In order to reduce the computational complexity of motion estimation in H.264/AVC video coding, this paper proposes a new search algorithm based on artificial fish-swarm algorithm (AFSA) which is a new efficient optimizing method. Firstly, some characteristics of AFSA are modified, such as visual, step, moving direction and initial fish positions in order to better adapt to motion estimation. Secondly, an adaptive search strategy based on modified AFSA Algorithm is presented to further reduce computational complexity, including double search mode, dynamic search range and early termination strategy. Experimental results show that the proposed algorithm saves the average motion estimation times up 39.29% and 31.22% for UMHexagonS and EPZS algorithm, which are adopted in H.264/AVC video coding, with almost same rate distortion performance.

Key-Words: - motion estimation, artificial fish-swarm, optimizing method, H.264/AVC coding, search mode, rate distortion

1 Introduction

H.264/AVC [1] video coding standard is developed by the Join Video Team of ITU Video Coding Experts Group (VCEG) and ISO/IEC Moving Picture Experts Group (MPEG), it has been a significant improvement in coding efficiency with higher compressed ratio and better visual quality than previous video coding standards, such as MPEG-1/2/4 and H.261/H.263 [2]. In H.264/AVC video coding, inter-frame prediction is adopted to reduce temporal redundancy and to obtain high compressed performance. For inter-frame prediction, motion estimation (ME) is the most time-consuming task in the encoding process. Several ME methods has been studied such as block matching algorithms, optical flow and per-recursive techniques. Among these methods, block matching seems to be the most popular technique due to its effectiveness and simplicity for both software and hardware implementation.

In block matching motion estimation, the frame to be encoding is partitioned into non-overlapping blocks of pixels. Each block is predicted from a block of equal size in the previous encoded frame(s). Block matching algorithms seek for the best matching block within a search window for each block, whiling minimizing block matching error. Therefore, block matching is essentially an optimization problem whole goal is to find the best matching block.

The computational complexity of block matching methods effects seriously on the coding efficiency of H.264/AVC video coding. According to [3], full search algorithm(FS), a simplest block matching algorithm that checks all candidates and find the best one, consumes 60% (one reference frame case) and 80% (five reference frames case) of the total computation time of H.264/AVC video coding. Hence, many fast search algorithms for motion estimation have been proposed to improve the coding efficiency without reducing video quality significantly, such as diamond search (DS) [4], hexagon-based search (HEXBS) [5] , adaptive root pattern search (ARPS) [6].

Hybrid unsymmetrical-cross multi-hexagon-grid search (UMHexagonS) [7][8] algorithm and enhanced predictive zonal search (EPZS) [9][10] algorithm have been adopted in H.264/AVC coding. UMHexagonS uses the hybrid and hierarchical motion search strategies and uses different kinds of search pattern, such as unsymmetrical-cross, uneven multi-hexagon-grid and extended hexagon. EPZS

employs simple stage pattern (diamond or square) and uses additional predictors and good threshold criteria which mainly is comprised by three features such as the initial predictor selection, the adaptive early termination and the final prediction refinement. Based on these technique UMHexagonS and EPZS algorithm has been proved to have better performance than previous algorithms such as DS, HEXBS, ARPS etc. However, they are still easy to be trapped into local minimum because they are based on unimodal error surface assumption the same as the other algorithms. This assumption is not always valid in real-world video sequences especially for violent motion. And their computational complexity are still some expensive.

To solve this problem, some evolutionary approaches have been applied to the motion estimation of video coding, such as genetic algorithm (GA) [11][12] and particle swarm optimization (PSO) [13][14]. These search algorithms uses the global optimization characteristics of evolutional strategy or swarm intelligence to obtain global optimum. To some extent, they improve the search accuracy for some video sequences, but they have too high computational complexity and could not been used in real video coding standards.

Li et al. has recently proposed a new swarm-intelligence algorithm to solve complex optimization problems which is known as Artificial Fish-Swarm Algorithm (AFSA) [15]. The basic idea of AFSA is to imitate the fish behaviors such as preying, swarming, following and moving within local search of fish individual for reaching the global optimum. Each food position represents a feasible solution for the problem under consideration and corresponding food density represents the quality of such solution (fitness value). The AFSA algorithm starts by producing a randomly distributed population. After initialization, each fish carries out three behaviors to move around via exchanging data with their adjacent members. After a maximum number of cycles of moving or a criterion meeting, the algorithm terminates. Based on these series of instinctive behaviors, the fish try to maintain their colonies and therefore demonstrate intelligent behaviors. The results show that AFSA can get a better performance than other methods in several optimization problems. AFSA have been applied in various fields like optimization, control, image processing, data mining, improving neural networks, networks, scheduling and signal processing and so on[16][17][18].

In this paper, a new motion estimation algorithm based on AFSA is proposed. Firstly, some characteristics of AFSA are modified, such as visual, step, moving direction and initial fish positions, in order to better adapt to motion estimation. Secondly, an adaptive search strategy based on modified AFSA Algorithm is presented to further reduce computational complexity, including double search mode, dynamic search range and early termination strategy. Experimental results show that the proposed algorithm saves the average motion estimation times up 39.29% and 31.22% for UMHexagonS and EPZS algorithm.

The rest of this paper is organized as follows. In section 2, AFSA algorithm is briefly introduced. Section 3 introduces motion estimation in H.264/AVC video coding. The details of motion estimation based on modified AFSA in H.264/AVC coding are described in Section 4. And Section 5 discusses the simulation results and the performance of three algorithms. Finally, conclusion is given in section 6.

2 Artificial Fish-Swarm Algorithm

Artificial Fish-Swarm Algorithm (AFSA) is a biologic optimization algorithm by imitating fish behavior, which was first proposed by Li et al. in 2002[15]. In nature, the fish can discover the more nutritious area by individual search or following after other fish, the area with much more fish is generally most nutritious. We simulate the behaviors of fish based on this characteristic to find the global optimum, which is the basic idea of the AFSA. In AFSA, artificial fish is a fictitious entity of true fish, which we can regard as an entity encapsulated with one's own data and a series of behaviors.

The artificial fish realizes external perception by its vision showed in Fig.1. X is the current state of artificial fish, *Visual* is the visual distance, *Step* is the moving step length and X_v is the visual position at some moment. X_{n1} and X_{n2} are companions. If the state at the visual position is better than the current state, it goes forward a step in this direction, and arrives the X_{next} state; otherwise, continues an inspecting tour in the vision. The greater number of inspecting tour the artificial fish does, the more knowledge about overall states of the vision the artificial fish obtains. Certainly, it does not need to travel throughout complex or infinite state, which is helpful to find the global optimum by allowing certain local optimum with some uncertainty.

the best matched block in the previous frame is called motion vector (MV). The process of block matching motion estimation is showed in Fig. 2. Once MV is got and residual error is calculated, encoder writes these data into a binary stream. Under such perspective, block matching motion estimation can be approached as an optimization problem aiming to find the best MV within a search window.

The rate-distortion cost function is used as a matching criterion for motion estimation in H.264/AVC video coding [19].

$$J(mv, \lambda_{MOTION}) = \lambda_{MOTION} \cdot R(mv - pmv) + SAD(s, c(mv)) \quad (7)$$

Where $mv = (mv_x, mv_y)^T$ is actual MV of current block, $pmv = (pmv_x, pmv_y)^T$ is predict MV of current block, $R(mv - pmv)$ is bit rate for coding the difference between actual MV and predict MV. λ_{MOTION} is Lagrange coefficients. s is data of current frame and c is data of reference frame.SAD is sum of absolute difference as follow,

$$SAD(x_i, y_i) = \sum_{m=1}^{M}\sum_{n=1}^{N} \left| f_k(m,n) - f_{k-1}(m + x_i, n + y_i) \right| \quad (8)$$

Where (x_i, y_i) is motion vector, f_k is pixel on (m,n) of current frame and f_{k-1} is pixel on $(m + x_i, n + y_i)$ of reference frame. M is width of block and N is height of block.

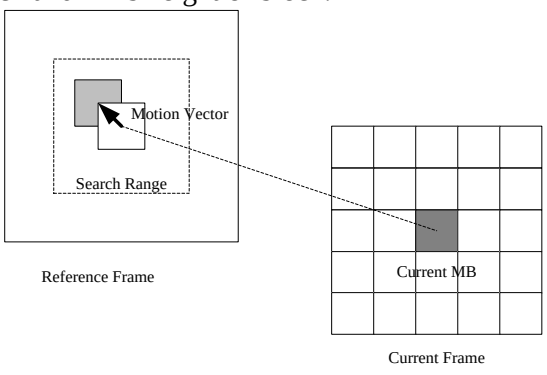


Fig. 2 The process of block matching motion estimation

4 Motion Estimation based on modified AFSA in H.264/AVC Coding

AFSA algorithm can find the global minimum but it takes a high computation cost for practical use. In this paper, the new algorithm based on modified AFSA is proposed for motion estimation in H.264/AVC. Firstly, some characteristics of AFSA are modified, such as visual, step, moving direction and initial fish positions, in order to better adapt to motion estimation. Secondly, an adaptive search strategy based on modified AFSA Algorithm is presented to further reduce computational complexity, including double search mode, dynamic search range and early termination strategy.

4.1 Modified AFSA Algorithm for motion estimation

AFSA has many advantages, such as good robustness, global search ability, tolerance of parameter settings and so on. But it still has disadvantages including higher time complexity, lower convergence speed and lack of balance between global search and local search. For motion estimation in H.264/AVC, the modified strategy in detail is as follow.

4.1.1 Adaptive VISUAL and STEP

At the beginning iteration of AFSA, larger *Visual* and *Step* will lead to a better global search result. As growing of iteration count, larger *Visual* and *Step* will reduce the stability of convergence. Thus, we use a linear *Visual* and *Step* which become smaller as the iteration count increased. Adaptive *Visual* and *Step* is calculated by Eq. 9 and Eq.10 respectively, where $Visual(i)$ and $Step(i)$ are *Visual* and *Step* value in the i th iteration. K_{visual} and K_{step} are the changing slope rates.

$$Visual(i) = Visual - K_{visual} \bullet i \quad (9)$$

$$Step(i) = Step - K_{step} \bullet i \quad (10)$$

4.1.2 Moving direction of Preying Behavior

Preying behavior is the basic behavior of AFSA. To further enhance the performance of AFSA, the preying behavior is modified as follows:

Let X_i be the artificial fish current state and select a state X_j randomly in its visual distance and Let Y be the food density (objective function value). If $Y_i > Y_j$, the fish goes forward a step in the direction of the vector sum of the X_j and the X_{best}. X_{best} is the best artificial fish state till now. Otherwise, select a state X_j randomly again. If it cannot satisfy after Try_number times, it moves a step randomly.

$$X_i^{(t+1)} = X_i^{(t)} + \left(\frac{X_j - X_j^{(t)}}{\| X_j - X_j^{(t)} \|} + \frac{X_{best} - X_i^{(t)}}{\| X_{best} - X_i^{(t)} \|}\right) \bullet Step \bullet rand() \quad (11)$$

4.1.2Selection of initial fish positions

According to [20], motion vectors usually locate around neighboring MVs in a high probability. Therefore, it's not effective that AFSA starts by producing a totally randomly distributed population without taking their neighboring MVs into account. We propose that part of food positions are initialized to spread around the neighboring MVs instead of generating randomly.

A total of 9 search points are chosen as the initial fish positions. They are:

1. One search point is the center point of current search area.
2. One search point is the position of the predictive motion vector, which description in detail is as follows.
3. Four search points locate on the four corners with two pixels distance away from the MVP in both X and Y directions, as shown in Fig. 3, where the predictive motion vector is represented by a solid black point and four corner positions are represented by hollow points.
4. One search point is the co-located MV position in the previous encoded frame.
5. Two search points are generated randomly to maintain the randomness of AFSA, as shown in Eq.12.

$$SP_{i,j} = \lfloor R_{\min} + (R_{\max} - R_{\min} + 1) * rand(0,1) \rfloor \quad (12)$$

Where $(R_{\min}, R_{\max})$ is the search area for motion estimation. $rand(0,1)$ is a random number uniformly distributing in (0, 1).

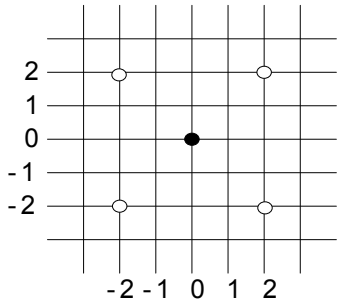


Fig. 3 Initial fish position distribution

The motion vector of a block is highly correlated to its neighboring blocks in one frame. As shown in Fig. 4, the motion vector of block E is spatially correlated to the motion vectors of neighboring blocks (A, B and C). The predictive motion vector is used as the median MV of neighboring MVs, which is called MV predictor (MVP).

$$MVP_E = median(MVP_A, MVP_B, MVP_C) \quad (13)$$

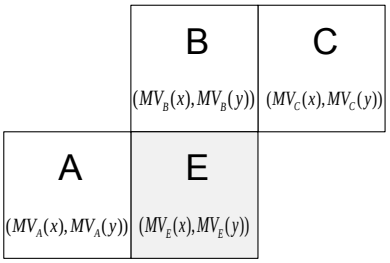


Fig. 4 Neighboring blocks of motion vectors

There may be duplicated search points in these initial search points. If such points are found, they are replaced with randomly generated points.

4.2 Adaptive search strategy based on modified AFSA Algorithm for H.264/AVC

Modified AFSA algorithm for motion estimation obtains accurate motion vectors of blocks using the global characteristic of AFSA. However, it still has very high computational complexity. Thus, we propose some adaptive search strategies for further reducing computational complexity.

4.2.1Double search mode

If the movement of the current block in a frame is very slow, AFSA algorithm will cause excessive waste of computation. To solve this problem, we propose double search mode.

Firstly, the degree of the movement of the current block needs to be accurately determined. Then, if the movement of the block is slow, DS algorithm is used. If the movement of the block is moderate or violent, modified AFSA algorithm is used. For slow movement situation the unimodal error surface assumption is valid, DS algorithm can achieve satisfactory accuracy and speed for its good robustness and effective local search ability.

The key of double search mode is the ability to accurately determine the degree of motion. In this paper we use the predictive motion vector information to determine. Let $SVar$ be the standard variance of neighboring MVs, which includes three spatial neighboring MVs (top, left, top-right MVs) as shown in Fig. 4 and the co-located temporal neighboring MV.

$$SVar = \frac{1}{N}\sum((mvp_{ix} - (\sum mvp_{ix})/N)^2 + (mvp_{iy} - (\sum mvp_{iy})/N)^2) \quad (14)$$

Where mvp_{ix} is the X component of predictive

motion vectors and mvp_{iy} is the Y component of predictive motion vectors.

If $SVar$ is less than the threshold Th_{sv}, the slow movement is assumed and DS is used. Otherwise, modified AFSA algorithm is used.

4.2.2 Dynamic search range

In H. 264/AVC video coding, a large fixed search range is set. This search range can guarantee the accuracy of motion vector for the violent movement sequences, but it will waste unnecessary search time for slow movement sequences. In order to ensure the search accuracy while reducing the computational complexity, we propose the adaptive search range.

According to the statistical distribution of the motion vector in video sequences, the majority motion vectors are in 5×5 search area of the search original point [21]. Therefore, firstly we set a basic search range $SR_{base}=(-5,+5)$. If the movement is slow, the search range is SR_{base}. Otherwise, the adaptive search range is defined as follow:

$$SR_{dy}=SR_{base}+\frac{SVar_{cur}}{SVar_{\max}}(SR_{input}-SR_{base}) \quad (15)$$

Where SR_{input} is the original fixed search range, Var_{cur} is the current variance and $Var_{\max}$ is the maximum variance.

4.2.3 Early termination strategy

As we know, the number of iterations of AFSA impacts computational complexity significantly. To reduce the number of iterations, two termination strategies are proposed here.

Firstly, the maximum iterative number $K_{\max}$ is set. Secondly, the iteration can be terminated without reaching the maximum iterative number if the rate-distortion cost value of current block is less than threshold Th_{stop}. The threshold Th_{stop} is as follows:

$$Th_{stop}=Min\{J_U,J_L,J_{RU}\} \quad (16)$$

Where J_U,J_L,J_{RU} is the rate-distortion cost value of top, left, top-right block, respectively.

4.3 The MAFSA algorithm structure for motion estimation

The MAFSA algorithm for motion estimation is summarized as follows:

Step1: Select the MAFSA initial parameters such as search points, iterative number, *Visual* and *Step* etc.

Step 2: Calculate the standard variance value $SVar$ for the current block. If it's less than the threshold Th_{sv}, go to step 3. Otherwise, go to step 4.

Step 3: Perform DS search algorithm then go to step 5.

Step 4: Perform modified AFSA search algorithm. Firstly, calculate the rate-distortion cost function of initialize search points. Secondly, calculate four behaviors of AFSA: preying, swarming, following and moving. Meanwhile, adjust search range, *Visual* , *Step* and moving direction. Then, repeat four behaviors of AFSA until the early termination condition is satisfied. If the iteration is terminated, go to step5.

Step 5: Set the resulting search point obtained by DS or modified AFSA algorithm as the final motion vector.

5 Experimental results in H.264/AVC video coding

We integrated MAFSA algorithm into JM 17.2 encoder of H.264/AVC video coding. The maximum iteration time is set as 5, initial *Visual* is 32, initial *Step* is 16, the crowd factor $\delta=10$, the threshold $Th_{sv}=20$ and $Th_{stop}=512$.

The test condition in JM17.2 is: GOP structure IPPP…, size of macro block 16×16, the values quantization parameter (QP) 16/20/24/28, number of reference frames three, number of encoded frame 100, RDO on, rate control off and CABAC encoding. Other parameters are default.

To compare MAFSA, UMHexagonS and EPZS, six test video sequences which have variety formats, as shown in Table 1.These test sequences can download from http://media.xiph.org/video/derf/.

Table 1. Test video sequences

Sequences	Format	Search range
Mobile	QCIF(176×144)	(-8, +8)
Coastguard	CIF (352×288)	(-16, +16)
Crew	4CIF(704×576)	(-32, +32)
Walk	SD(720×576)	(-32, +32)
Parkrun	HD(1280×576)	(-64, +64)
Rushhour	HD(1920×1280)	(-64, +64)

Evaluation of motion estimation algorithm is in two ways. One is search accuracy and the other is computational complexity. In H.264/AVC video coding, rate-distortion optimization technique is

used to encode actual code stream. Therefore, the evaluation of search accuracy needs to consider peak signal to noise ratio (PSNR) value and encoding bit rate. And the evaluation of computational complexity usually adopts search time of motion estimation algorithm. In this paper we compares the time saving. $\Delta T_{MA/U}$ is defined as search time ratio of MAFSA relative to UMHexagonS and $\Delta T_{MA/E}$ is defined as search time ratio of MAFSA relative to EPZS. The detail is as follows:

$$\Delta T_{MA/U} = (T_{UMHexagonS} - T_{MAFSA}) / T_{UMHexagonS} \times 100\% \quad (17)$$

$$\Delta T_{MA/E} = (T_{EPZS} - T_{MAFSA}) / T_{MAFSA} \times 100\% \quad (18)$$

Table 2 shows the performance comparison of MAFSA,UMHexagonS and EPZS for six sequences. As shown in Table 2, the encoding efficiency of MAFSA is almost the same as that of UMHexagonS and EPZS. They have very close PSNR and bit rate values. MAFSA has an average increase of 0.02dB in PSNR value and an average increase of 0.46% of bit rate value compared with UMHexagonS. And MAFSA has an average increase of 0.03 dB in PSNR value and an average increase of 1.73% of bit rate value compared with EPZS. However, MAFSA has obvious advantages of computational complexity. From Table 2, we can see that MAFSA has an average saving of 39.29% and 31.22% in motion estimation time compared with UMHexagonS and EPZS, respectively.

Fig. 5 and Fig. 6 how the rate-distortion performance curve and average motion estimation time of MAFSA, UMHexagonS and EPZS for six sequences. According to Fig. 5, MAFSA can achieve very similar rate-distortion performances to UMHexagonS and EPZS. From Fig. 6, MAFSA not only saves significantly the average motion estimation time-consuming for UMHexagonS and EPZS, but also MAFSA has a little fluctuation for all sequences.

In order to give the visual results, subjective images of motion compensation of various algorithms are shown in Fig. 7. We can see that three images have almost no difference.Therefore, the overall performance of MAFSA has absolute advantage and much more easily to be applied in real-time video encoding.

6 Conclusion

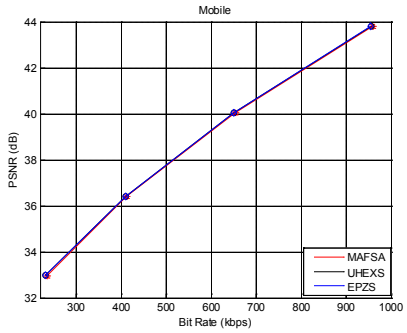
In this paper, a new search algorithm based on artificial fish-swarm algorithm for motion estimation is proposed. The proposed algorithm takes advantage of global optimization of AFSA and modifies some characteristics of AFSA to better adapt to motion estimation. Meanwhile, the proposed algorithm uses some adaptive search strategies to further reduce computational complexity. Compared with UMHexagonS and EPZS algorithm, the proposed algorithm can provide almost rate distortion performance and save the average motion estimation times up 39.29% and 31.22%, respectively. Consequently, the proposed algorithm is easy to be applied to real-time video coding systems.

7 Acknowledgement

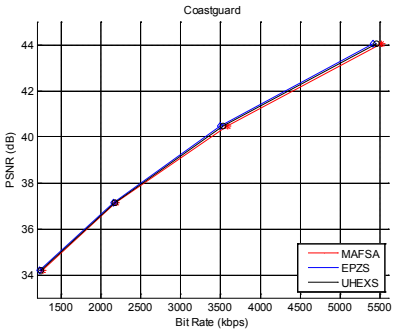
This research was supported by the National Natural Science Foundation of China (Grant No. 61308102), SRFDP (Grant No. A09010501301851200 38), China Postdoctoral Science Foundation (Grant No. 2013M531946), Fundamental Research Funds for the Central Universities, China (No. ZYGX-2009J024). The authors are also grateful to a referee for his/her very helpful comments and suggestions.

Table 2. Performance comparison of MAFSA, UMHS and EPZS (PSNR(dB), BitRate(BR,kb/s),Time(sec), QP=28)

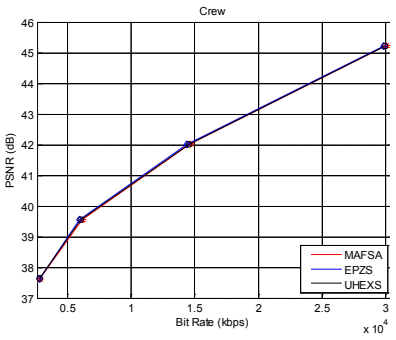
Sequences	UMHexagonS			EPZS			MAFSA			Time Saving	
	PSNR	BR	Time	PSNR	BR	Time	PSNR	BR	Time	$\Delta T_{MA/U}$	$\Delta T_{MA/E}$
Mobile	32.97	232.26	7.52	32.98	231.54	7.38	32.96	234.71	6.26	16.76%	15.18%
Coastguard	34.20	1250.50	41.50	34.20	1228.16	35.79	34.19	1263.30	24.99	39.78%	30.18%
Crew	37.64	2813.47	158.09	37.63	2777.00	138.88	37.63	2818.45	99.63	36.98%	28.26%
Walk	36.81	4767.17	359.35	36.73	4711.56	186.34	36.91	4808.74	103.02	71.33%	44.71%
Parkrun	33.20	14381.37	467.28	33.20	14220.14	428.47	33.21	14497.68	232.18	50.31%	45.81%
Rushhour	40.77	3605.72	658.02	40.79	3537.95	680.54	40.77	3552.53	522.63	20.58%	23.20%
Average	**35.93**	**4508.42**	**281.96**	**35.92**	**4451.06**	**246.23**	**35.95**	**4529.24**	**164.79**	**39.29%**	**31.22%**



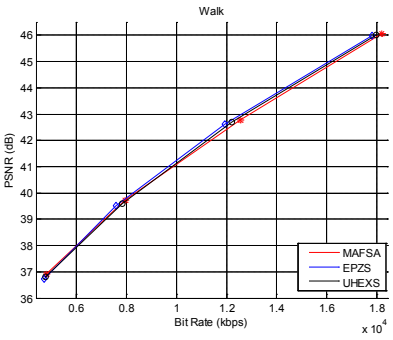
(a)Mobile



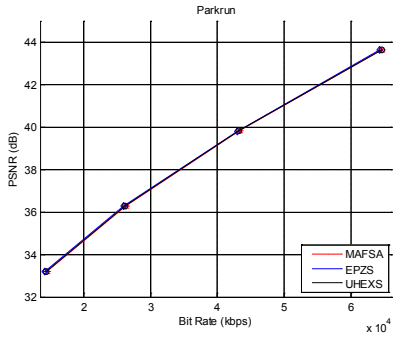
(b)Coastguard



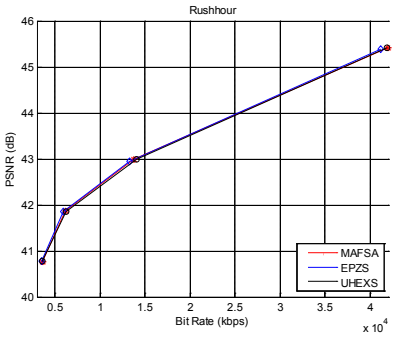
(c)Crew



(d)Walk

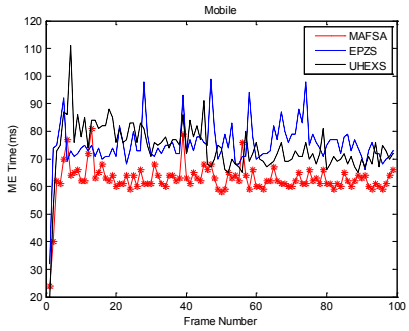


(e)Parkrun

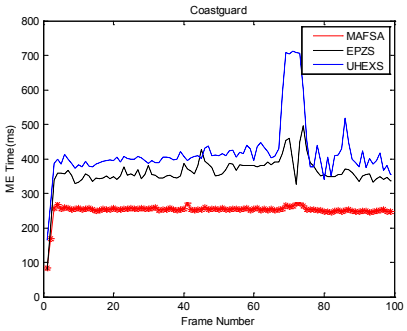


(f)Rushhour

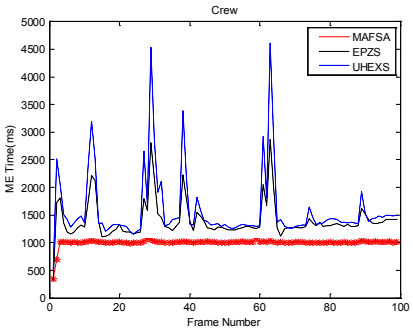
Fig. 5 Rate-distortion performance of MAFSA, UMHexagonS and EPZS(QP=16/20/24/28)



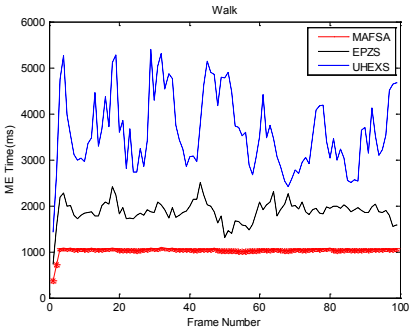
(a)Mobile



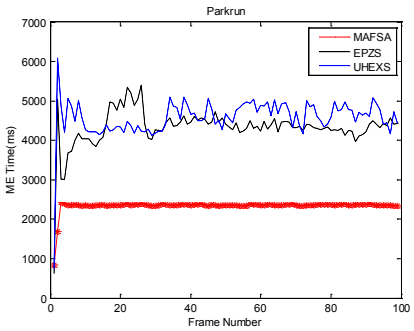
(b)Coastguard



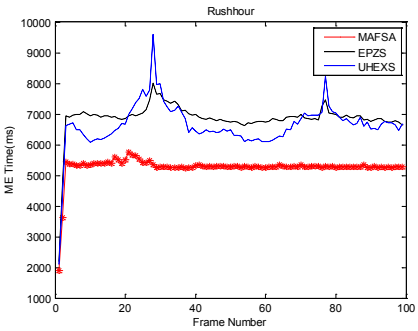
(c)Crew



(d)Walk



(e)Parkrun



(f)Rushhour

Fig. 6 Average motion estimation time of MAFSA, UMHexagonS and EPZS (QP=28)

(a) MAFSA(38.00dB)

(b) EPZS(37.99dB)

(c) UMHexagonS (38.02dB)

Fig. 7 Crew No.10 frame: subjective image of motion compensation of UMHexagonS, EPZS and MAFSA

References:

[1] T.Wiegand, G.J.Sullivan, G.Bjntegaard, et al., Overview of the H.264/AVC video coding standard, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 12, No. 7, 2003,pp.560–576.

[2] D.Marpe, T.Wiegand, G.J.Sullivan, The H.264/MPEG4 advanced video coding standard and its applications, *IEEE Communication Magazine*, Vol. 44, No. 8, 2006, pp. 134–144.

[3] Y.W. Huang, B.Y. Hsieh, S. Y. Chien, et al., Analysis and complexity reduction of multiple reference frames motion estimation in H.264/AVC, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.16, No.4, 2006, pp. 507–522.

[4] S. Zhu and K.K. Ma, A new diamond search algorithm for fast block matching motion estimation, *IEEE Transactions Image Processing*, Vol. 9, No.2, 2000, pp.287–290.

[5] C. Zhu, X. Lin, 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, 2002, pp. 349−355.

[6] Y. Nie, K.K. Ma, Adaptive rood pattern search for fast block-matching motion estimation, *IEEE Transactions Image Processing*, Vol.11, No.12, 2002, pp.1442-1448.

[7] Z.B. Chen, P. Zhou, Y. He, et al., Fast Motion Estimation for JVT, *ISO/IEC JTC1 and ITU-T JVT-G016*,Pattaya Thailand, Mar 2003.

[8] Z.B. Chen, J.F. Xu, Y. He Yun, et al., Fast integer-pel and fractional-pel motion estimation

for H.264/AVC, *Journal of Visual Communication and Image Representation*, Vol.17, No.2, 2006, pp. 264-290.

[9] A.M. Tourapis, C. A. Oscar, M.L. Liou, Highly efficient predictive zonal algorithms for fast block-matching motion estimation, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.12, No.10, 2002, pp. 934-947.

[10] A.M. Tourapis, C. A. Oscar, M.L. Liou, Fast ME in the JM reference software, *ISO/IEC JTC1 and ITU-T JVT-P026*, Poznan Poland, Jul 2005.

[11] S. Li, W.P. Xu, N.N. Zhang, et al., A novel fast motion estimation method based on genetic algorithm, *Acta Electronica Sinica* , Vol.28, No.6, 2000, pp. 114 – 117.

[12] T. Gong, R.T. Ding, A modified genetic algorithm based block matching motion estimation method, *Signal Processing*, Vol.19, No.3, 2003, pp. 207-210.

[13] X.D. Yuan, X.J. Shen, Block matching algorithm based on particle swarm optimization for motion estimation, *The 2008 International Conference on Embedded Software and Systems (ICESS2008)*,2008, pp.191–194.

[14] K. M. Bakwad, S.S. Pattnaik , et al., Small population based modified parallel particle swarm optimization for motion estimation, 1*6th International Conference on Adavanced Computing & Communication (ADCOM)*, 2008, pp. 367-373.

[15] X.L. Li, Z.J. Shao, J.X. Qian, An optimizing method based on autonomous animats: fish-swarm algorithm, *Systems Engineering Theory & Practice*, Vol.22, 2002, pp.32-38.

[16] W.J. Tian, Y. Geng Y, J. Liu J, et al., Optimal parameter algorithm for image segmentation, *IEEE second international conference on future information technology and management engineering*, 2009,pp 179–182.

[17] D. Yazdani, S. Golyari S, M.R. Meybodi, A new hybrid algorithm for optimization based on artificial fish swarm algorithm and cellular learning automata, *2010 5th International Symposium on Telecommunications*, 2010, pp. 932–937.

[18] L. Wang, L.J. Ma, A hybrid artificial fish swarm algorithm for bin-packing problem, *2011 International Cconference on Electronic & Mechanical Engineering and Information Technology*, 2011, pp. 27–29.

[19] Y.J. Xiang, N. Lei, W.Y. Yu, et al., Research of block matching criterion for motion estimation, *Computer Science*, Vol. 36, No. 9, 2009, pp. 278-280.

[20] C. H. Cheung and L.M. Po, A novel cross-diamond search algorithm for fast block motion estimation, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 12, No. 12, 2002, pp. 1168-1177.

[21] C.C. Lou, S. W. Lee, C.C. Kuo, Adaptive motion search range prediction for video encoding, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 20, No. 12, 2010, pp. 1903-1908.