Video Compression for UAV Applications Using a Global Motion Estimation in the H.264 Standard

Paulo Henrique Fonseca Torres Soares
Instituto Tecnológico de Aeronáutica - ITA
P. Marechal Eduardo Gomes, 50
São José dos Campos, SP Brazil - 12.228-900
paulosoares86@gmail.com

Marcelo da Silva Pinho
Instituto Tecnológico de Aeronáutica - ITA
P. Marechal Eduardo Gomes, 50
São José dos Campos, SP Brazil - 12.228-900
mpinho@ieee.org

***Abstract*— Video coding is used nearly in every video application, but it is a very demanding resource process. In many applications of unmanned aerial vehicles, a video transmission is desired even though the onboard hardware has a low processing capability, in general. Therefore the study of new video compression algorithms with low complexity is an important task in this context. One of the most time consuming step in a video compression scheme is the motion estimation algorithm. To design a new low complexity video coding for remote sensing systems, in 2011, Bhaskaranand and Gibson proposed the use of a global motion estimation. However, the solution presented in 2011 can not be used with any video compression standard. This work deals with the model of Bhaskaranand and Gibson and produces a global motion estimation that can be used in every profile of H.264 in order to significantly reduce encoding time with a little penalty in the bit rate. The algorithm is tested in videos obtained by unmanned aerial vehicles and the results have shown a significant improvement in the time consuming by motion estimation algorithm. In fact this improvement reduces the total encoding time in 10% with a penalty of 1.5% in the bit rate.**

***Index Terms*— video compression, h.264 standard, motion estimation.**

I. Introduction

Video compression (or coding) is widely used in multimedia industry. The most of majority of digital videos used in the world are compressed. It is true because a digital video needs a large bandwidth to be transmitted or a high capacity device to be stored. For example, consider a color video with moderate resolution (640x480 pixels) and a rate of 24 frames per second (fps). If the RGB color space is used, this will result in a requirement of rate of transmission around 200 Mbps. In many Unmanned Aerial Vehicle (UAV) applications, the aircraft has a camera onboard producing a video signal that must be transmitted to an earth station. In general, the communication system of an UAV has the bandwidth constraint and also the needs to be design with a low capability hardware.

The H.264 is a video compression standard [1], [2] that has been used in many applications, such as in Brazilian System for Digital TV (SBTVD -Sistema Brasileiro de TV Digital) [3] and in Blu-Ray discs [4]. This standard is the preferred choice because it gives a substantially higher compression performance than its predecessors [5]. But it also has a higher time complexity [6]. In fact, the time complexity of H.264 baseline decoder is two to three times higher than the H.263 [6].

Video compression standards generally use two algorithms to reduce spatial and temporal redundancies. They are: (a) intra frame prediction; and (b) inter frame prediction, respectively. The first uses information of the own frame to code it while the second uses information of previously coded frames, which improves the compression rate. Motion Estimation (ME) is the algorithm used by inter prediction to determine the offset between an image and its best match from previously coded frames. The difference between these images is called prediction error and the offset position is said to be the motion vector. Even though the ME is very time consuming, it is one of the main stage of a video compression. Although many optimizations methods have been proposed for ME, it is still a bottleneck in the many video encoders. In H.264 standard, depending of the profile and the algorithm for motion estimation, this stage can consume until 90% of the encoding time.

In [7], Bhaskaranand and Gibson introduced the concept of global motion estimation to design a new video coding with a low complexity. This new algorithm was used to compress videos from an aerial platform where the motion observed in the signals was mainly due to the camera movements. In this context, it was shown that the new algorithm was efficient when compared to the standard H.264. Unfortunately, the global motion model proposed in [7] can not be used as the model in different video compression standards and therefore if this algorithm is used, the compression scheme does not keep the compatibility to those standards. In fact, the problem arises from the fact that the motion model deals with transforms like rotations and approximations.

In this work, we present a new video coding which uses a global motion estimation and which the coded video can be recovered by any H.264 decoder. The new procedure is based on a simplification of the model global estimation introduced in [7] and its adaptation to work in the standard H.264. The new scheme is tested in videos produced by an UAV and its performance is compared to the standard. This paper is organized as follows. In Section II, the concept of motion estimation and compensation is presented. The idea of global motion estimation is treated in Section III. The results obtained in this work is shown in Section IV. Closing the paper, Section V presents the conclusion.

II. Motion Estimation

For each block, a ME algorithm looks in the pictures of the Decoded Picture Buffer (DPB) for the block that minimizes a cost function. This block is called the best match one. Once this block is found, the motion vector is coded into bit stream. Since the difference between pixels of the given block and the best match is expected to be smooth, then it is transformed to the frequency domain using Discrete Cosine Transform and the higher frequency are discarded by quantization.

The Inter Prediction attempts to exploit the temporal correlation between frames and ME is the algorithm that finds the best match candidate for a given block. Thus, given a frame $F[i]$ and the target block $B_i[j]$ that are being encoded in a given moment, the ME uses blocks $B_l[k]$ of already encoded frames $F[l]$, $(l < i)$ to encode $B_i[j]$. Note that $B_i[j]$ and $B_l[k]$ must have the same dimensions. The standard can work with seven different block sizes: 16x16, 16x8, 8x16, 8x8, 4x8, 8x4, 4x4.

Suppose that the upper left pixel of the target block $B_i[j]$ is at line l and column c and that the candidate block is at line l' and column c', then the motion vector candidate is by definition $\mathbf{mv} = (l' - l, c' - c)$. This motion vector, which represents the offset between the target block $B_i[j]$ and the candidate $B_l[k]$, is also encoded into compressed video, using an entropy code. Following the rules of this entropy code, a motion vector with a large magnitude will produce a large codeword and therefore the size of the output video (i.e., the result of compression) is dependent of this magnitude too.

Ideally speaking, to know which would be the best candidate, it would be necessary to transform and quantize residues for each candidate, and then applying entropy coding to these residues and to the corresponding motion vectors. Since this procedure is impractical, it is used a cost function that takes two factors into consideration: the magnitude of the motion vector and the magnitude of the residuals.

Therefore, the cost must first evaluate the amount of bits needed to encode the motion vector. This cost partial motion vector $\mathbf{mv}(B_l[k]) = (mv_x, mv_y)$ is given by

$$Cost_{mv} = bits(mv_x) + bits(mv_y)$$

where the function $bits$ can be defined as

$$bits(x) = \begin{cases} 0 & \text{if } x = 0 \\ 2.\lfloor log_2(\ |x|\ )\rfloor + 3 & \text{otherwise} \end{cases}$$

Moreover, it must be also incorporated into the overall cost the candidate error the difference $B_l[k] - B_i[j]$ using some error metric. This metric is usually the Sum of Absolute Differences because of its simplicity (avoids multiplications) and performance (comparable to mean square error) [8]. Thus, a solution adopted in the H.264 reference model to evaluate a candidate according to these two costs is the Lagrangian cost[1].

[1]Information taken from file $./lencod/src/me\_fullsearch.c$

$$Cost = \sum_{m=1}^{M}\sum_{n=1}^{N} |B_l[k](m,n) - B_i[j](m,n)| + \\ + \lambda[bits(mv_x) + bits(mv_y)] \quad (1)$$

where $\lambda = 187$ is the default value of this constant and $l < i$, i.e., to encode a block of i-th frame the ME must search in the previously encoded frames. Thus, Inter Prediction can not be used in the first table.

Since in many videos, the rate of frames per second is greater, then it is expected that the correlation between consecutive frames to be very large. In fact, the time correlation is bigger than the spatial correlation exploited by Intra Prediction. Thus, Inter Prediction outperforms Intra Prediction and for this reason it is used as the primary prediction method. In some cases the Intra Prediction procedure can be useful. The most obvious case is the first frame, where there are no frames of reference. Furthermore, the use of the Intra Prediction periodically can be employed to reduce the encoder complexity and to avoid an error propagation. However, if the aim is to achieve the best compression rate, the Inter Prediction is the right choice. Therefore, the ME algorithm is called many times in the procedure of video compression and its complexity can not be neglected.

III. The Global Motion Model

Since the motion estimation algorithm must be applied to every block of a frame, it is a very time consuming process. However, in UAV applications, the motion in the video signal is mainly due to the motion of the camera, which is in the aircraft. In fact, it was this property that motivates the use of a global motion model, that is, a motion estimation of the entire frame, in an algorithm to compress videos. In [7], this model is used to design an efficient algorithm for video compression when used to encode videos captured by aerial platforms. This solution reduces significantly the complexity of the encoder with a small penalty in its performance. Following a global motion model consisted by rotation, scaling and translation, the work in [7], [9] defines a pixel transform as

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \times \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}$$

where $[x\ y]^T$ is the position of the pixel in the original frame, $[x'\ y']^T$ is the position of the pixel in the transformed frame, $[e\ f]^T$ is the translation vector and a, b, c, d are the parameters describing the rotation and scaling. It is important to notice that the motion estimation used in many video compression algorithms can not deal with rotation and scaling. Therefore, the algorithm in [7], [9] is completely different from many standards. Actually, the rotation and scaling were introduced in the model since the tested videos were obtained by a camera with zoom mounted in an helicopter.

In many UAV applications, the used platform is a fixed wing aircraft with a constant altitude and the camera does not change its zoom. In these cases, the model can be used with no scale effect. Furthermore, if the video has a high frame rate, the rotation effect can be neglected. In fact, using a set of UAV videos, this work measured the performance loss when the rotation model is withdraw from the model proposed in [7]. The results are shown in Section IV and it can be observed that the Sum of Absolute Difference (SAD) is reduced by a factor less than 0.5% if the rotation is considered. Based on these results, this work proposes a new solution using only the effect of translation. It is important to notice that if the rotation and scale models can not be discarded, an encoder based in the H.264 standard can not be designed.

Ignoring the rotation and zoom between frames, the model can be written as

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}$$

But, by the definition of motion vectors, the motion vector of any target block using any candidate block of a past frame will be

$$\mathbf{mv} = (x - x', y - y') = (-e, -f)$$

This equations shows that if the video is according to the model, than every motion vector of a frame will be constant. Thus, the ME can be modified to evaluate just one motion vector by frame, which will reduce the encoding time. However, to guarantee that the compressed video can be recovered by a H.264 decoder, all the blocks of the frame are encoded using the same motion vector.

IV. Results

Using a set of UAV videos and the version 18.4 of the H.264/AVC JM Reference Software, available in http://iphome.hhi.de/suehring/tml/, this work measures the performance of the new scheme. In a first test, the performance of a global motion model with the effect of rotation was compared to the performance obtained when this effect is omitted. Table I shows the the comparison between SAD considering the rotation and not considering it, when the tested block was of the size of one quarter of the frame. It can be observed from the results presented in Table I that the penalty is less than 2.5% when the rotation effect is not used. So, it can be concluded that ignoring this effect will not severely impact the compression performance in our applications. Besides that, considering rotation makes impossible to use H.264 codec and the time to find the motion vectors would be higher. Thus, it seems to be a good choice to ignore it.

To find the motion vector of each frame, it was designed a pre processor. It takes as input the video sequence and return the motion vectors of each frame as output. This pre processor works with a block of a variable length. In this way, the encoder has to perform just one motion estimation per frame and the length of the block used in this estimation can be adjusted. The fractional motion estimation of the standard was not modified in order to give some flexibility to the codec, specially to deal with cases where video does not suit perfectly to the model.

TABLE I
COMPARISON BETWEEN SAD CONSIDERING THE ROTATION AND NOT CONSIDERING IT.

Frame	Angle (degrees)	SAD (with rotation)	Error between SAD and SAD with rotation (%)
1	0,00	1844675	0,00
2	0,00	1546309	0,00
3	0,00	1983688	0,00
4	0,00	1910732	0,00
5	0,00	1607829	0,00
6	0,00	2141527	0,00
7	0,00	1987224	0,00
8	0,00	1709617	0,00
9	0,00	2041279	0,00
10	0,00	2203261	0,00
11	0,00	1880256	0,00
12	0,00	1935201	0,00
13	0,00	1989397	0,00
14	0,00	1859170	0,00
15	0,00	1974313	0,00
16	0,00	2107120	0,00
17	0,00	1682194	0,00
18	0,00	2083678	0,00
19	0,00	2206558	0,00
20	0,00	1911341	0,00
21	0,00	1794271	0,00
22	0,00	1753599	0,00
23	0,00	1845973	0,00
24	0,00	1789962	0,00
25	0,00	1499732	0,00
26	0,00	1886640	0,00
27	0,00	1952025	0,00
28	0,00	1540284	0,00
29	0,23	2034234	-2,44
30	0,23	1981299	-2,10
31	0,00	1549717	0,00
32	0,23	1880876	-2,78
33	-0,18	1800430	-0,75
34	0,00	1627944	0,00
35	0,22	1776902	-0,93
36	0,00	1665960	0,00
37	0,00	1631929	0,00
38	0,00	1780951	0,00
39	0,00	1652871	0,00
40	0,00	1465239	0,00
41	0,00	1718688	0,00
42	0,20	1845338	-1,18
43	0,23	1929215	-1,74
44	0,00	1923940	0,00
45	0,19	1886839	-0,99
46	0,00	1907903	0,00
47	0,00	1973602	0,00
48	0,00	1697784	0,00
49	0,00	1831392	0,00

Table II shows the bit rate of H.264 using the motion vectors got from the pre processor. These vectors are calculated evaluating using the cost (1) and assume that the simplified Global Motion Model is valid. This table shows how the block size changes the combinations between the total time (encoding time plus pre processor execution time) and the bit rate.

From Table II it is possible to conclude that the total time increases when the block size is bigger, which is true because more computations are necessary in the pre processor. But, the bit rate does not decrease in a relevant way when this

TABLE II

COMPARISON BETWEEN BIT RATE AND TOTAL TIME USING DIFFERENT BLOCK SIZES.

Rate (kbps)	Block size	Encoding time (s)	Pre processor execution time (s)	Total time (s)
6312,89	16	54,566	0,24	54,77
6314,68	32	54,192	0,28	54,77
6315,67	48	54,355	0,37	54,76
6315,67	64	54,172	0,36	54,74
6309,53	80	54,548	0,44	54,73
6309,53	96	54,220	0,54	55,02
6309,53	112	54,247	0,63	55,00
6309,53	128	54,186	0,77	55,20
6312,08	144	54,099	0,93	55,22
6317,73	160	54,308	1,24	55,46
6313,84	176	54,496	1,30	55,54
6317,49	192	54,208	1,41	55,60
6319,83	208	54,500	1,66	56,13
6320,34	224	54,148	1,84	55,99
6318,39	240	54,339	2,06	56,37
6320,27	256	54,262	2,35	56,57
6318,82	272	54,321	2,79	57,09
6316,23	288	54,316	2,84	57,11
6321,10	304	54,316	3,17	57,45
6321,10	320	54,506	3,46	57,75
6321,10	336	54,761	3,81	58,17
6323,43	352	54,287	4,10	58,38
6316,76	368	54,481	4,41	58,63
6316,64	384	54,416	5,15	59,35
6319,55	400	54,305	5,30	59,59
6317,69	416	54,224	5,48	59,80
6316,09	432	54,234	5,86	60,16
6316,09	448	54,343	6,22	60,73
6316,43	464	54,282	6,61	61,01

happens. So, the best combination between total time and bit rate is using the block of size 16x16, since the objective is to reduce the encoding time with little penalty in the bit rate.

One of the most efficient algorithm to estimate the motion vector is the algorithm EPZS (Enhanced Predictive Zonal Search). In fact, in [8] there are comparisons between the algorithms EPZS and some alternatives, such as the PMVFAST (Predictive Motion Vector Field Adaptive Search Technique), the Diamond Search and the MVFAST (Predictive Motion Vector Field Adaptive Search Technique). In general, the results show that EPZS is a good choice which outperforms the others algorithms for motion estimation in speed and which leads to a good bit rate performance. Therefore, this work uses the standard H.264 with the EPZS algorithm to evaluate the improvement in coding time that can be achieved when the global motion estimation is adopted. Table III shows the comparison between the proposed model operating with block size of 16×16 and the standard H.264 with EPZS algorithm. These results were obtained using the main profile. From Table III it can be observed that it was achieved a reduction of 9.35% in terms of encoding time but it was noticed that the penalty was of 1.50% in the bit rate.

It is important to notice that the reduction in execution time obtained by the use of the global estimation is not only 9.35%. In fact, the results from Table II show that the encoding time is around 54.57 seconds and the pre processor execution time is 0.24 second. Table III shows that the encoding time changes

TABLE III

COMPARISON BETWEEN BIT RATE OF EPZS AND THE PROPOSED ALGORITHM.

Motion Estimation algorithm	Encoding time (s)	Bit rate (kbps)
Proposed	54,77	6312,89
EPZS	60,42	6219,62

from $54.57$to 60.42 seconds when the EPZS algorithm is used. Therefore, the difference between the time consuming by the EPZS algorithm is around 5.85 seconds (against 0.24 second (spent by the global model). It is easy to see that in a different framework (with a faster coding procedure), the improvement can be much higher than 9.35%.

V. CONCLUSION

In this paper, it was presented a new scheme to compress video using a global motion model in the standard H.264 which keep the compatibility with this standard (i.e, a H.264 decoder can recover the compressed video). This new scheme was used to compress videos obtained by unmanned aerial vehicles and the measured performance shows a significant improvement in terms of execution time with a small penalty in the achieved bit rate. The tests were done using the version 18.4 of the JM reference software in the main profile, but the algorithm can be used in every profile of H.264 with the proper modification. When compared to the H.264 using the algorithm EPZS for motion estimation, the total coding time is reduced in 9.35% and the bit rate is increased in 1.5% only.

ACKNOWLEDGEMENTS

Marcelo da Silva Pinho would like to thank FAPESP (Proc. 2010/11695-1) for financial support.

REFERENCES

[1] I. E. G. Richardson, *The H.264 Advanced Video Compression Standard*, Wiley and Sons, 2010.

[2] T. Wiegand, G. J. Sullivan, G. Bjntegaard, A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits, Systems and Video Technology,* vol. 13, pp. 560—576, 2003.

[3] *Digital Terrestrial Television — Video Coding and Multiplexing. Part 1: Video Coding,* ABNT, NBR 15602-1: 2008.

[4] *Application Definition Blu-ray Disc Format - BD-J Baseline Application and Logical Model Definition for BD-ROM,* Blu-ray Disk Association (BDA), 2005.

[5] N. Kamaci, Y. Altunbasak, "Performance comparison of the emerging h.264 video coding standard with the existing standard," *Proceedings of IEEE International Conference on Multimedia & Expo*, pp. 345—348, 2003.

[6] M. Horowitz, A. Joch, F. Kossentini, A. Hallapuro, "H.264/AVC baseline profile decoder complexity analysis," *IEEE Transactions on Circuits, Systems and Video Technology,* vol. 13, no. 7, pp. 704—716, 2003.

[7] M. Bhaskaranand, J. D. Gibson, "Low complexity video encoding for uav reconnaissance and surveillance," *Proceedings of 2011 Military Communications Conference - Track 4 - Middleware Services and Applications*, 2011.

[8] Alexis M. Tourapis, "Enhanced predictive zonal search for single and multiple frame motion estimation," *Proceedings of SPIE — Visual Communications and Image Processing*, vol. 4671, pp. 1069—1079, 2002.

[9] M. Bhaskaranand, J. D. Gibson, "Global motion compensation and spectral entropy bit allocation for low complexity video coding," *Proceedings of IEEE International Conference on Communications*, 2012.