An Image Sensor with Global Motion Estimation for Micro Camera Module

Fabrice Gensolen, Guy Cathébras, Lionel Martin, Michel Robert

An Image Sensor with Global Motion Estimation for Micro Camera Module

F. Gensolen[1,2], G. Cathebras[2], L. Martin[1], M. Robert[2]

[1] STMICROELECTRONICS, ZI de Rousset, BP 2, 13106 Rousset, France
[2] LIRMM, Univ. Montpellier II / CNRS, 161 rue Ada, 34392 Montpellier, France
fabrice.gensolen@lirmm.fr

Abstract. We describe in this paper the building of a vision sensor able to provide video capture and the associated global motion between two consecutive frames. Our objective is to propose embedded solutions for mobile applications. The global motion considered here is the one typically produced by handheld devices movement, which is required for our purpose of video stabilization. We extract this global motion from local motion measures at the periphery of the image acquisition area. Thanks to this peculiar and "task-oriented" configuration, the resulting system architecture can take advantage of CMOS focal plane processing capabilities without sacrificing the sensor fill factor. Our approach is currently implemented in a CMOS 0.13µm technology.

1 Introduction

Our objective is to develop a smart CMOS image sensor for mobile systems (PDA, cell phone). Such handheld devices are very shake prone and often provide trembling video; also we focus in this paper on video stabilization. The best way to stabilize video is to perform an optical correction using gyro sensors and mobile optics/sensors, stabilizing directly the incidence of the light onto the focal plane. Nevertheless, this is a costly and burdening solution for embedded devices.

Another approach is completely electronic. It consists in analyzing the main displacement between two consecutive frames of the video, so called global motion or camera motion, in order to separate the intentional motion from the unwanted one. This last is then compensated, resulting in a video without jolts [1]. This is the stabilization scheme we have adopted. We focus in the present paper on the crucial global motion estimation stage of the processing.

Considering the signal processing architecture, such a motion estimation task can be realized as a post-processing of digital images coming from the imager (Fig. 1). In a time to market point of view, this is a very efficient way to implement an image processing on silicon. However, that means to process the huge amount of video data serially, which is very time and power consuming [2].

In this paper, we investigate another way to perform motion estimation task by reporting part of the processing at pixel level. This approach speeds up the processing time and alleviates the computing power by making use of parallelism of the pixels needed in image sensors for light spatial sampling. The main drawback of this kind of

silicon integration is the increased area per pixel, which decreases the fill-factor and the image resolution. But we present and validate in this paper a new global motion estimation technique based on local motion measures at the periphery of the image acquisition area. Thanks to this peculiar and "task-oriented" configuration, we take advantage of CMOS focal plane processing capabilities without sacrificing the sensor fill factor. Indeed, the silicon area has become the main contribution to the cost of image sensors, accounting for around 70% of the overall cost.

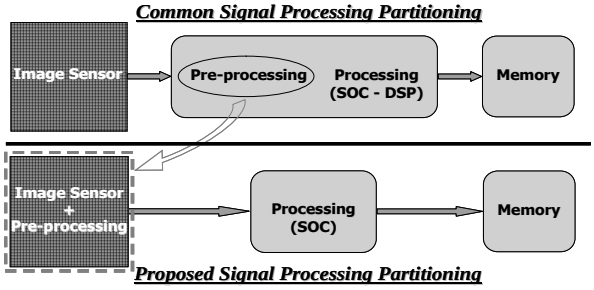


Fig. 1 : Signal processing partitioning.

We describe in section 2 our global motion estimation technique, based on peripheral local motion measures. Section 3 is dedicated to its validation by software implementation, and section 4 to the evaluation and the partitioning of the processing. Then we describe in section 5 the transfer of part of it onto focal plane, performing pixel level processing.

2 Global motion estimation

2.1 Principle and basis

In order to describe the global motion between two consecutive frames, we make use of a four components parametric model, called similarity model. This model allows us to describe the main global movements perceived in the focal plane: that means rotations around the optical axis, zoom, and X-Y translations. Then, such a parametric motion can be ascribed to most of the pixels in an image. It is also a good tradeoff between complexity, noise sensitivity, and description of the inter frames movement.

Two kinds of motion are generally present in common video captures with handheld devices like cell phones: the one due to mobile elements in the scene, and the background one (Fig. 2). In our purpose of video stabilization, that is this last background movement which is of main interest as it informs directly about the camera/global motion.

Moreover we point out that the periphery of images are particularly interesting for this task. Indeed, this area of interest contain local motions that better constrain the global motion parameters. Also, these local motions are well distributed in the images and background is often on the periphery of images. Therefore, we only focus on this area to extract the desired global motion (Fig. 2).

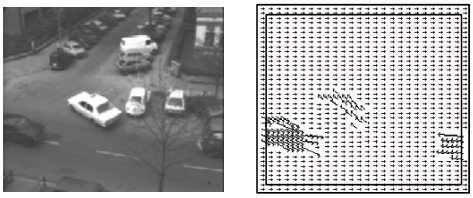
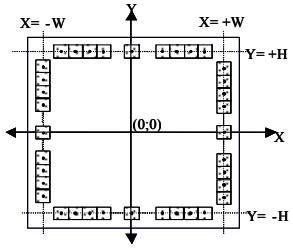


Fig. 2 : Example of a scene, with the associated local motion vectors for a left panning of the camera. Edges on the right picture point out the area of interest.



$$\begin{cases} Xj(t+dt) = \alpha * \cos\theta * Xj(t) + \alpha * \sin\theta * Yj(t) + Tx \\ Yj(t+dt) = -\alpha * \sin\theta * Xj(t) + \alpha * \cos\theta * Yj(t) + Ty \end{cases}$$

Fig. 3 : Geometric system setting.

2.2 Global motion estimation procedure

Let us suppose that a picture "n" is transformed according to a geometric combination of a rotation θ, a zoom factor α, and two translations Tx and Ty, in another picture "n+1": a pixel "j" with cartesian coordinates (Xj(n),Yj(n)) in frame "n" become the pixel with cartesian coordinates (Xj(n+1),Yj(n+1)) in frame "n+1". The system in Fig. 3 describes such a geometric transformation. Applying this transformation to all the points of the area of interest, that leads to the following linear over determined system:

$$P = K \times M \quad (1)$$

Where P is the matrix positions (Xj(n+1),Yj(n+1)), K the matrix linking these positions to the ones of frame "n", composed of (Xj(n),Yj(n)) coordinates with "0" and "1", and M contains the four transformation parameters. These parameters are Tx, Ty, $\alpha.\cos\theta$, and $\alpha.\sin\theta$. Then knowing P and K, we are able to determine the four global motion parameters M thanks to an optimization process.

Matrix P is obtained summing original cartesian positions of pixels in frame "n" with local motions of this pixels between frame "n" and "n+1". The optimization operation is performed here in a least squares sense, and the resulting estimation can be written as [3]:

$$M = (K^T \times K)^{-1} \times K^T \times P \quad (2)$$

3 Software implementation

In order to validate our approach, we first evaluate the performances by software. Hence, we need to compute the local movement estimations at the periphery on the two consecutive images.

3.1 Local motion estimation

Several algorithms exist to perform local motion estimation, but one of the most efficient ways is to carry out pixel, or area, correspondence [4]. We have chosen two algorithms for extracting local motion vectors: the first is the well known full search

block matching (FSBM), and the other comes from [5]. The last technique, called Census transform, consists in a local texture coding which results in a binary code for each pixel, that is then tracked from one frame into the next.

3.2 Performances

Using Matlab software, we have characterized our global motion estimation technique with respect to outdoor and indoor scenes. Firstly we built synthetic video sequences starting from a high resolution picture which we transformed and from which we picked a CIF one with a known displacement. Then we also grabbed real video sequences thanks to the same digital camera, providing a 15 im/s CIF video. Both synthetic and real sequences contain the same image texture. They have been captured in the same illumination conditions with various constant amplitudes of movement. These amplitudes are always lower than : 5% of the image size for translation, 3° for rotation, 2% for zoom. Both represent indoor and outdoor scenes. The indoor ones contain an environment of work with desks and chairs (lowly textured) and the outdoor ones are a nature environment with trees and a river (highly textured).

The inter frame global motion being unknown in real sequences, we apply the precise and robust algorithm developed by [6] in order obtain our reference motions (source code with makefiles are available on the IRISA website: http://www.irisa.fr/Vista/Motion2D/index.html). This algorithm is proven reliable in various applications like underwater vehicle positioning, or super-resolution, and car driving assistance.

We report in the following Table 1 the first results of our characterizations in terms of error percentage to the reference, or known, motion. As we can see, raw Census transform motion estimation gives lower results than block matching.

	*Census 5*5*	*Bloc matching*
Synthetic outdoor	5.8 %	0.02 %
Real outdoor	71 %	12%
Synthetic. Indoor	6.2 %	0 %
Real indoor	82 %	15 %

Table 1 : First results on performances achieved with a software implementation of the global motion estimation (in terms of error percentage to the reference motion).

Search area	*FSBM 5×5* $\sim 2 \times M^2 \times S^2 \times N$	*Census 5*5*
+/-16 pixels	~15 232 068 op 99.92 %	~4 394 495 op 99.73 %
+/- 12 pixels	~8 736 042 op 99.87 %	~3 369 087 99.65 %
+/- 8 pixels	~4 032 002 op 99.31 %	~ 2 330 367 99.49 %

Table 2 : Processing load of local motion estimation in elementary operations, and the associated ratio to the total computational load of the global motion estimation (with N=280 local motions on the periphery).

4 Towards a vision system on chip

Our final objective is to integrate the proposed technique to an image sensor. The main constraint, as discussed in the introduction of the paper is the silicon area. But the sensor has to perform the global motion estimation in real time and is dedicated to embedded devices, hence additional constraints have to be taken into account.

4.1 Complexity analysis

In the hierarchy of vision sensor design flow, complexity analysis is the first step to optimize the digital hardware required.

Firstly we can point out that the lines number for all matrix involved in our technique can be half the original one, while keeping exactly the same estimation robustness. Indeed, let us suppose that we consider N positions of local motion estimations, the resulting over determined system of equation (1) contains 2N lines, as each position is described in the image plane by two coordinates.

Then if a local motion is erroneous, it constitutes a proportion of 2/2N=1/N of the system, which is the same proportion as if we consider only one of the two new coordinates (1/N). Therefore we choose this last solution and we will involve in the global estimation only the coordinate parallel to the considered side of the image periphery. This is the same as computing one-dimensional motion estimation, and the overall processing load in terms of elementary operations number is half of the original one. We have quantified the total number of elementary operations to perform the global motion estimation, leading to 42N+207 operations, where N is the number of local motion considered in the periphery of the image.

Let us now consider the local motion estimation processing load which constitutes the main part of the total load. Indeed, as described earlier, we compute local movement estimations thanks to matching algorithms, just as the well-used MPEG scheme. Unfortunately, these algorithms lead to highly regular low-level tasks, and a huge amount of data access through frame buffer is also required. In a typical video encoder for example, it accounts for as much as 60% of total CPU cycles [2]. In our case, FSBM and Census algorithms are both quadratic algorithms, with respective complexity of $2\times M^2\times S^2\times N$ and $S^2\times N$. Where $M\times M$ are blocks of pixels, S is the search area in pixels, and N the number of local movements considered. Let us consider that we perform one local motion measurement each 10 pixels of a SVGA frame, that means that N=280. In that case, it accounts for as much as the overall number of operations presented in the following Table 2.

4.2 Hardware requirements and partitioning

As pointed out in Table 2, local motion estimations accounts for around as much as 99% of the total processing load required to extract the global motion between two consecutive frames.

On the other hand, we can perform in CMOS technology numerous kinds of analog and/or mixed signal processing, as soon as the phototransduction [7], avoiding to perform the computation in the processing stage (Fig. 1). This computation saving being then allocated to higher level tasks as image segmentation for tracking for example. This is the processing partitioning that we propose, performing the peripheral local motion measurements at pixel level thanks to dedicated motion detectors, and adding the least square global motion estimation procedure to the processing hardware (Fig. 1). Therefore, the resulting vision sensor architecture is the one shown on the right in Fig. 4.

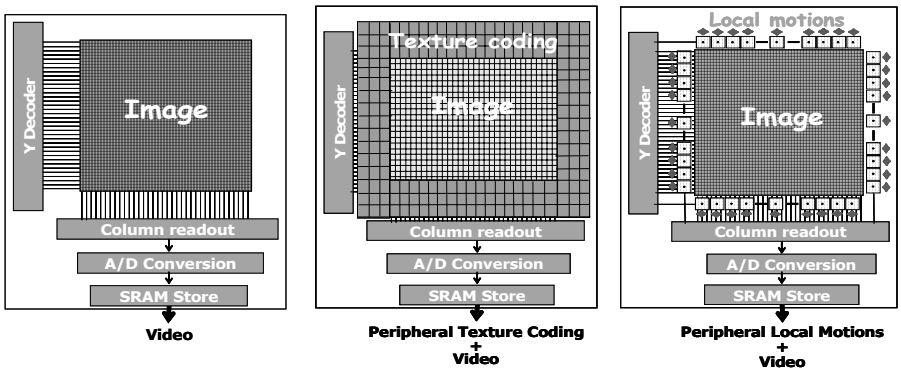


Fig. 4 : Common image sensor architecture (left), and our two proposed achitectures (center and right). The prefered architecture is the right one.

5 Focal plane local motion measures

Focal plane signal processing performed in vision sensors are often elementary operations in order to keep relative simple pixels and preserve the sensor fill-factor while benefiting from the intrinsic massive parallelism of image sensors to obtain powerful computing architecture. It is also possible to design specific architectures dedicated to particular tasks.

Motion estimation is one of them, and several local motion detectors have been designed for this purpose [8]. Some of them are inspired from biology and constitute silicon models of elementary biological functions. Each of these smart pixels embeds additional electronics (around 20 transistors), leading to larger silicon area per pixel, hence lower resolution and higher cost compared to pixels dedicated only to the image acquisition (3 transistors). That is the main reason why these kind of smart pixels are not widely used in industry.

However, fixing our task of global motion estimation considering only the periphery of the image acquisition area avoids this antagonism between pixel level processing and silicon area required. Also, we propose in the following section two kinds of focal plane processing estimating the desired local motions. In the first solution we integrate a modified version of the census transform (center in Fig. 4), and in the second solution we propose the integration of the entire local motion estimations (right in Fig. 4).

5.1 Ternary census transform

We firstly consider the silicon integration of the census transform, which has been previously detailed in [9]. Then, based on our circuit characterizations, we have carried out further validations on the census transform. That brought us to introduce a new version of it, especially dedicated to a focal plane processing type, where no noise reduction is performed (due to silicon area saving). This new census transform has been shown to be more robust to fixed pattern noise due to CMOS process

dispersions [10]. The resulting pixel architecture involves an hysteresis comparator to perform luminance comparisons between pixels, associated to a 3 transistors active pixel sensor (Fig. 5).

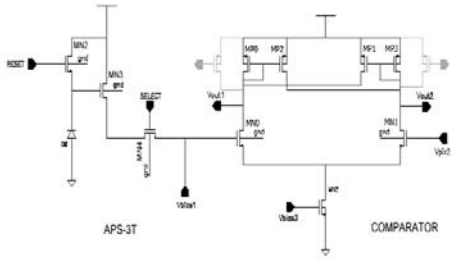


Fig. 5 : Pixel architecture integrating the ternary census transform.

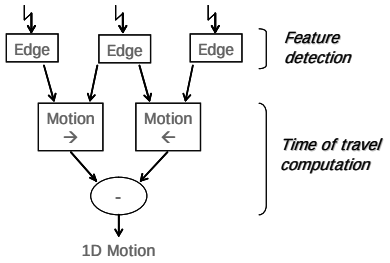


Fig. 6 : Local motion measurement.

Each pixel results in a 10×10µm² area, instead of 4×4µm² for pixels specialized in image acquisition only. Moreover, as explained in section 3, local motions are obtained performing pixel correspondence with neighbours. This implies to integrate not only a single line of ternary census pixels around the image, but several lines in order to be able to determine the magnitude of inter frame movements (b. in Fig. 4).

For a SVGA video module for example (sensor size which is currently on sale), the magnitude is about 3% of the image size, which equals about 21 pixels (in terms of image pixels size, i.e. 4×4µm² area). However in terms of ternary census pixels, it is equal to a displacement between 8 and 9 pixels in both directions. Therefore we need to integrate 2*9+1=19 lines of ternary pixels, increasing the image area by 22% of the original SVGA image acquisition area. This integration induces 50% saving of the overall computation load to perform global motion estimation (see Table 2).

It is important to point out that this silicon area proportion is more and more decreasing as the image sensor resolution grows (which is the actually evolution).

5.2 Local motion detector

As introduced at the beginning of this section, several motion detectors have been designed to measure local motions. We focus currently on the ones described in [11]. The main advantage of such a processing is the continuous time mode of measuring local motion. Indeed it avoids the problem of temporal aliasing [8]. The principle illustrated in Fig. 6 is to measure the time Δt of travel of a spatial or temporal feature of the scene (an edge for example) between two photo elements distant of L, resulting in a crossing speed of :

$$V = \frac{L}{\Delta t}$$

This measure is asynchronous, that is why we are currently developing a technique to process a temporal integration over the inter frame period in order to synchronize our data with the video.

This elementary motion sensor is integrated in a 30×30µm² pixel. Moreover, thanks to the continuous time processing, only one line of such detectors are necessary, resulting in an increase in silicon area of 4% of the original SVGA image

acquisition area. These local motion measurements induce around 99% saving of the overall computational load (see Table 2).

6 Conclusion

We have presented in this paper a new approach in performing video stabilization. The global motion required to fix this task is extracted from local motion estimations at the periphery of the image acquisition area. We performed a least-squares global motion estimation and local motion estimations with two pixel correspondence techniques: the Census transform and the block matching. The block matching technique gave us the best results, allowing to get a global motion estimation error of 12% of the true motion in real video sequences. Such an error is suitable for standard video captures but appears not precise enough in cases of large movements.

We have also described the building of a vision sensor able to provide video capture and the associated global motion. The main advantage of the proposed technique, in a vision system architecture point of view, is to perform each task of image acquisition and motion estimation independently, with the optimized focal plane processing. Indeed it avoids the sensible tradeoff between image pixel area and pixel level processing.

References

1. C. Morimotto and R. Chellappa, "Fast Electronic Digital Image Stabilization", *in Proceedings of the 13th Int. Conf. on Pattern Recognition*, Aug. 1996, vol. 3, pp. 284-288.
2. P. M. Kuhn and W. Stechele, "Complexity analysis of the emerging MPEG-4 standard as a basis for VLSI implementation", *in Proceedings of the Int. Conf on Visual Communications and Image Processing*, San Jose, Jan. 1998, vol. SPIE 3309, pp. 498-509.
3. N.R. Draper and H. Smith, "*Applied Regression Analysis*", 3rd Ed., John Wiley & Sons, New York, 1998.
4. F. Dufaux and F. Moscheni, "Motion estimation techniques for digital TV: a review and a new contribution", *in Proceedings of the IEEE* , vol. 83, Issue 6, June 1995, pp. 858-876.
5. R. Zabih and J. Woodfill, "Non-Parametric Local Transforms For Computing Visual Correspondance", *in 3rd European Conference on Computer Vision*, 1994, pp 151-158.
6. J.M. Odobez, P. Bouthemy, "Robust multiresolution estimation of parametric motion models", *Journal of Visual Communication and Image Representation*, December 1995, 6(4), pp.348-365.
7. C. Mead, "Analog VLSI and neural systems," *Addison-Wesley*, 1989.
8. R. Sarpeshkar, J, Kramer, G. Indiveri and C. Koch "Analog VLSI Architectures for Motion Processing: from Fundamental Limits to System Applications", *Proceedings of the IEEE*, Vol. 84, Issue: 7, , July 1996, pp. 969-987.
9. D. Navarro, G. Cathébras and F. Gensolen, "A block matching approach for movement estimation in a CMOS retina: principle and results," *proceedings of the European Solid-State Circuits Conference ESSCIRC'03*, 2003, pp. 615-618.
10. F. Gensolen, G. Cathebras, L. Martin, M. Robert, «Pixel level silicon integration of motion estimation », *IEEE workshop on Design and Diagnostic of Electronic Circuits and System*s, Sopron, Hungary, April 2005.
11. J. Kramer, "Compact integrated motion sensor with three-pixel interaction", in IEEE trans. On Pattern Analysis ans Machine Intelligence, April 1996, vol. 18, Issue 4, pp. 455-460.