GRADIENT BASED DOMINANT MOTION ESTIMATION WITH INTEGRAL PROJECTIONS FOR REAL TIME VIDEO STABILISATION

A.J. Crawford, H. Denman, F. Kelly, F. Pitié and A.C. Kokaram

Electronic and Electrical Engineering Department
University of Dublin, Trinity College,Ireland
E-mail: crawfofo@tcd.ie

ABSTRACT

This paper presents a new expression of the relationship between Integral Projections and motion in an image pair. The resulting new multiresolution gradient based approach is used to estimate dominant motion in image sequences degraded by random shake. The paper also describes an implementation using the GPU as a co-processor for the CPU that allows, for the first time, real time video stabilisation *in software* on broadcast standard definition television images.

1. INTRODUCTION

The dominant motion in a scene is that motion component that can be ascribed to *most* of the picture material in an image. Terms like *global motion* and *camera motion* are used synonymously to mean the same thing, but they do not quite express the fact that the dominant motion in a scene can be a *combination* of both camera behaviour *and* apparent object behaviour. Dominant motion information has long been recognised as an important feature in many video processing tasks. This motion embodies information about the video event, hence it is a useful feature for content based retrieval [1]. Similarly, because of the large picture area that can be ascribed to dominant motion, it can (in general) be estimated more robustly than local motion, and is useful for compression as in MPEG4 [2].

Image instability manifests as a random, unwanted fluctuation in the dominant motion of a scene. Shake is very common in footage from both hand-held cameras and fixed cameras (despite the image stabilisation technology on most cameras). Instability can be caused by external factors such as wind and unsteadiness in the camera's movement (any instability is magnified at high zoom). Archived video sequences also suffer from unsteadiness introduced during filming or during the digitization of film. As most common compression systems utilise the similarity between consecutive frames, random dominant motion has a large effect on the compressibility of video data since more bandwidth is consumed unnecessarily representing motion locally. Removal of this fluctuation therefore has wide application in a number of different areas.

There are two issues in video stabilisation. Firstly, the dominant motion must be estimated. The unwanted component of this dominant motion must then be extracted and removed, while preserving intentional motion such as pan. To achieve this, it is assumed that the two components of motion have different statistics.

There are many possibilities for estimating dominant motion [2, 3]. Direct matching techniques like Block Matching are simple to implement but computationally expensive. Recently, the method of Integral Projections [4, 5] has been proposed which can reduce substantially the computational load. This employs matching of 1D projections of the image in horizontal and vertical directions. The underlying assumption is that the effect of dominant motion on the interframe pixel difference outweighs the effect of local motion. Provided this holds, there is less of a need for rejecting outliers in the estimation process as in [2, 3]. In this paper, a novel perspective on Integral Projections is presented starting from optical flow. This yields a new gradient based approach which allows the development of a real time software module for video stabilisation at CCIR 601 resolution and 25 or 30 fps. The implementation exploits the GPU as a coprocessor for the CPU and is real time even on a 1GHz Pentium laptop.

2. ESTIMATING DOMINANT MOTION

There are two main categories of method for estimating dominant motion: feature based and image based. Feature based methods, typically employed in computer vision, attempt to locate and match important features, e.g. corners in image pairs, and hence extract the fundamental matrix [6]. Image based methods rely on direct transformation of the image grid and minimise some image difference criterion. The technique discussed here is an image based method.

Dominant motion is estimated based on a single large $N \times N$ block centered on each frame. A value of $N = 512$ pixels is used for 720×576 footage. All methods described use one dimensional, Integral Projections of this block to

estimate global motion. The use of Integral Projections was quantitatively justified in [7] by using the properties of the Radon Transform. In this paper, we present for the first time an alternate perspective using optical flow which brings new insights. Integral projections $\mathbf{p}_{h,n}$ and $\mathbf{p}_{k,n}$ are calculated:

$$\mathbf{p}_{h,n} = \sum_k I_n(h,k) \text{ and } \mathbf{p}_{k,n} = \sum_h I_n(h,k)$$

where $I_n(h,k)$ is the intensity of the pixel (h,k). Using integral projections reduces estimation from a 2D problem to a 1D problem. This gives a very substantial reduction in computational complexity.

To relate the use of these projections to motion estimation, the analysis begins by expressing the image sequence model as:

$$I_n(\mathbf{x}) = I_{n-1}(\mathbf{x}+\mathbf{d}) + \varepsilon(\mathbf{x})$$

where $\mathbf{x}$ is the vector containing column and row co-ordinates of (h,k), $I_n(\mathbf{x})$ is the intensity of pixel $\mathbf{x}$ of frame n, $\mathbf{d}$ is the image displacement and $\varepsilon(\mathbf{x}) \sim \mathcal{N}(0, \sigma_e^2)$.

Consider calculating an update, $\mathbf{u}$ to an initial estimate $\mathbf{d}_0$:

$$I_n(\mathbf{x}) = I_{n-1}((\mathbf{x}+\mathbf{d}_0)+\mathbf{u}) + \varepsilon(\mathbf{x})$$

Using the Taylor Series Expansion to linearise the left hand side about $\mathbf{x}+\mathbf{d}_0$:

$$I_n(\mathbf{x}) = I_{n-1}(\mathbf{x}+\mathbf{d}_0) + \mathbf{u}^\mathbf{T}\nabla I_{n-1}(\mathbf{x}+\mathbf{d}_0) + \varepsilon(\mathbf{x})$$

Let $Z_n(\mathbf{x}) = I_n(\mathbf{x}) - I_{n-1}(\mathbf{x}+\mathbf{d}_0)$:

$$Z_n(\mathbf{x}) = \mathbf{u}^\mathbf{T}\nabla I_{n-1}(\mathbf{x}+\mathbf{d}_0) + \varepsilon(\mathbf{x})$$

Writing the ∇ operator in full:

$$Z_n(h,k) = u_h G_h(h,k) + u_k G_k(h,k) + \varepsilon(h,k)$$

where

$$G_h = \frac{\partial I_{n-1}}{\partial h} \text{ and } G_k = \frac{\partial I_{n-1}}{\partial k}$$

The crucial next stage is to recognise that assuming the motion is the same for a large image area, summing in a particular direction can allow useful approximations. To simplify matters assume $\sum_h \varepsilon(h,k) = 0$ although it is possible to proceed without this assumption. Summing with respect to h:

$$\underbrace{\sum_h Z_n(h,k)}_{(i)} = \underbrace{u_h \sum_h G_h(h,k)}_{(ii)} + \underbrace{u_k \sum_h G_k(h,k)}_{(iii)}$$

A similar expression exists for summing in the vertical direction. If it were possible to ignore one of the two terms (ii) or (iii) each component of motion could be solved separately. The table below shows the ratio $\sum_h G_k / \sum_h G_h$ for a number of test images.

Image	Ratio
Lena	7.1
Sailboat	24.2
Peppers	76.9

The table shows that term (iii) is much more significant than term (ii) in general. This makes sense since summing with respect to h followed by calculating the gradient also with respect to h is tantamount to applying a low-pass filter followed by a high-pass filter. We would expect this to produce a low-energy output. We can therefore assume part (ii) =0. So:

$$\sum_h Z_n(h,k) = u_k \sum_h G_k(h,k)$$

Let

$$\mathbf{z}_k = \sum_h Z_n(h,k) \text{ and } \mathbf{g}_k = \sum_h G_k(h,k)$$

then

$$\mathbf{z}_k = u_k \mathbf{g}_k$$

We now have an estimate for u_k:

$$u_k = \frac{\mathbf{g}_k{}^\mathbf{T}\mathbf{z}_k}{\mathbf{g}_k{}^\mathbf{T}\mathbf{g}_k}$$

Interestingly, now u_k can be calculated using integral projections as:

$$\mathbf{z}_k = \mathbf{p}_{k,n} - \mathbf{p}_{k,n-1}$$

and

$$\mathbf{g}_k = \mathbf{p}_{k,n-1}(i) - \mathbf{p}_{k,n-1}(i-1)$$

u_h can be calculated similarly, summing with respect to k.Hence the connection between Integral projections and motion estimation.

Multiresolution: The Taylor Series Expansion holds only for small values of dominant motion. This problem can be circumvented by using multiresolution techniques. Coarse to fine refinement of motion estimates on a pyramid of images is one mechanism for dealing with large displacement in the gradient estimation context. Here a 4 level pyramid is employed with a maximum of 10 iterations at each level. The method is called *Multi-Res* in subsequent sections. A further computational savings is had by noting that the pyramid can be generated in the *1D projection space* rather than in the 2D image space. Thus the pyramid is built by downsampling 1D projections rather than projecting downsampled images. The savings is on the order of $N^2/3$ multiply adds.

Because the manipulation of integral projections requires so little computation, it is possible to propose another, *hybrid* technique. Direct matching on the projections using

cross correlation is performed, at the integer pixel resolution. The resulting estimate of motion is then used to initialise the gradient based estimator above. This method allows the gradient based method to concentrate on the relatively small motion adjustments required after the gross direct matching is achieved.

3. REAL TIME IMPLEMENTATION AND COMPUTATION

The video frame-rate must be maintained for a real-time implementation. To achieve real-time implementation at this PAL frame rate (25 fps), each frame must be processed in less than 40ms.

The table below compares the computational complexity of block matching with that of each of the methods used in this paper. The first column gives the number of operations required based on a single N x N size block, with a range of $(+/-w)$ (where i is the number of iterations and t is the number of taps used in the low pass filter used by the multi resolution method). This does not include the number of computations required to calculate the projections ($2N^2$). The ratio to block matching is also shown (including the calculation of the projections) given values of $N = 512$, $w = 32$, $i = 20$ and $t = 15$. It is clear from these values the use of integral projections provides a huge reduction in computational complexity.

Method	Operations	Ratio to BM
BM	$(2w+1)^2(N^2)$	1
Gradient based	$2i(7N)$	0.00060
Hybrid	$8wN + 8N + 14iN$	0.00073
Multi-Res	$1\frac{15}{16}N(t+14i)$	0.00074

Implementation: All methods were implemented in C++ using the DirectShow component of the DirectX API. To reduce the computation time taken by each method, the Intel Performance Primitive (IPP) libraries were used. These Libraries use the Single Instruction Multiple Data (SIMD) functionality of the Pentium processor allowing a number of operations can be performed in parallel. The video stream is manipulated directly from the IEEE Firewire input on the PC.

4. SEPARATING UNWANTED COMPONENTS OF MOTION

The task is now to smooth the unwanted variations in motion between consecutive frames. Global motion can be caused by: (1) intentional effects like a pan, and (2) the unsteadiness of the camera which is unintentional. The first effect is generally low frequency and exhibits slow temporal variations, whereas the secondary effect is temporally impulsive. It is possible to extract the first signal by means of a low pass filter [8]. The compensating motion can be found by simple difference of the output of this filter and the measured motion.

As the shake in hand-held cameras is not extreme and only past estimations are available in a real time system, a simple IIR low pass filter is sufficient where

$$H(z) = \frac{0.0201 + 0.0402z^{-1} + 0.2017z^{-2}}{1 + 1.561z^{-1} - 0.6414z^{-2}}$$

5. IMAGE COMPENSATION AND THE GPU

To create the final images for output, each image must be shifted to compensate for the unwanted motion component estimated in previous sections. In order to accurately represent the global motion of a frame, a sub-pixel accurate motion vector is typically required. Interpolation of the image signal is required to motion compensate a frame with a fractional motion vector. Typically bilinear interpolation is sufficient. However this interpolation is computationally very demanding and can be a bottleneck in a real-time shake reduction scheme.

Modern graphics hardware contain very efficient interpolation units which are used in the texture mapping stage of the graphics pipeline. The graphics hardware can compensate each frame with bilinear interpolation accuracy. This can be done much faster than real-time with the motion compensated sequence displayed on screen. Each motion compensated frame can also be retreived from the graphics hardware and saved to file if necessary. Because the graphics hardware can work in parallel with the CPU, using it for motion compensation also frees up valuable CPU cycles for other processes. We do not present here the details of the GPU code needed to achieve this, It is sufficient to note that the interpolation unit on the GPU is used as part of the pipeline for video stabilisation.

6. EXPERIMENTAL RESULTS

Two PAL test sequences were used to test the performance of the Image Stabilisation application. The first is a 10 second cycling sequence with no camera motion and some small local motion. The second is a 10 second sequence with smooth horizontal pan and no local motion. Artificial instability was added to both sequences using Matlab by shifting each frame by a random motion vector (created using a normally distributed Gaussian number generator with variance 49 and mean zero)using bi-linear interpolation. The implementation was tested on a Dell Precision 2.6GHz P4 with 1 GB ram. Video sequences are available for viewing at `www.mee.tcd.ie/~sigmedia/andy`.

Method	$\lvert\mathbf{d} - \hat{\mathbf{d}}\rvert$
Gradient based	0.4089
Hybrid	0.1470
Multi-Res	0.1320

Accuracy: The table above shows the mean error for each of the methods estimated motion vectors, tested using the cycling sequence (d is the motion estimated, $\hat{d}$ is the actual motion). Although the single level gradient based method error is large, on closer examination of the motion estimated (Right: a) it is clear that this method is highly accurate for small displacements while poor for large displacements. Therefore, as shown in the table, when used in conjunction with multi-resolution methods, the error is extremely low. Also, the second sequence was used to test the intentional motion filter. The figure to the Right (b) shows the total motion estimated and the smooth intentional motion calculated by the filter.

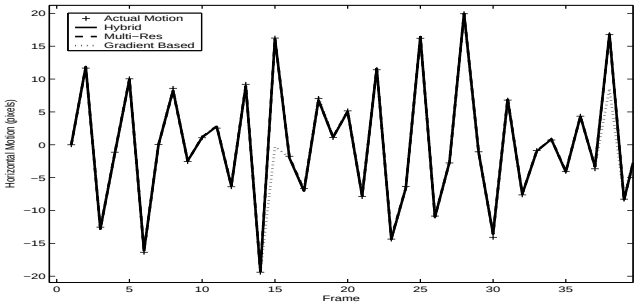
Method	Time per frame (ms)
Gradient based	3.9775
Hybrid	4.8932
Multi-Res	4.3907

Speed: The Table above shows the average time taken to estimate motion for a single frame for each method. The figures do not correspond directly to the computations required for each method due to the implementation using the IPP Libraries. The figures are all well below the 40ms maximum for real-time processing. This allows further time for image compensation and other processing.

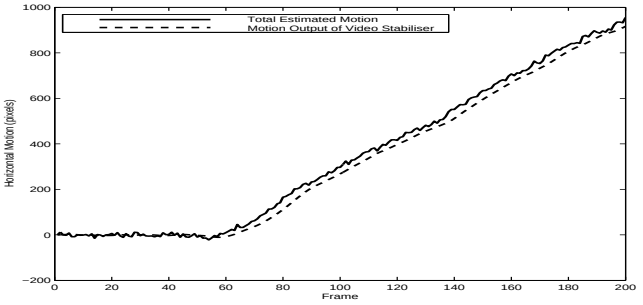
7. CONCLUSIONS

Dominant motion estimation, once a slow process, can now be implemented in real time on a standard PC for PAL resolution. The approximations that allow this reduction in complexity do not significantly reduce accuracy as long as large motion is catered for. The method described calculates projections only along the image axes. This may not always be the correct direction depending on the picture information. From the optic flow approach taken, it is possible to notice that the error in using projections in this way depends on the ratio between the sum of horizontal and vertical gradients in each direction. Therefore a test for the validity of the projection can be made by measuring the ratio $\sum_h G_k / \sum_h G_h$. If this ratio is not large, then the projection along image axes is not valid. Finding new projection angles is made feasible by analysing principal axes of gradient across the entire image. This idea is not pursued further in this paper, but will be presented in related work.

Finally, it is to be emphasised that it is the use of integral projections *in combination with* the GPU for motion compensation that makes real time achievable across a wide range of compute platforms available now.



(a) Motion estimated from cycling sequence



(b) Intentional motion filter

8. REFERENCES

[1] P. Bouthémy, M. Gelgon, and F. Ganansia, "A unified approach to shot change detection and camera motion characterization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, pp. 1030–1044, 1999.

[2] F. Dufaux and J. Konrad, "Efficient, robust and fast global motion estimation for video coding," *IEEE Transactions on Image Processing*, vol. 9, pp. 497–501, 2000.

[3] J-M. Odobez and P. Bouthémy, "Robust multiresolution estimation of parametric motion models," *Journal of visual communication and image representation*, vol. 6, pp. 348–365, 1995.

[4] J.-S. Kim and R.-H. Park, "A fast feature-based block matching algorithm using integral projections," *IEEE J. Selected Areas in Communications*, vol. 10, no. 5, pp. 986–971, June 1992.

[5] J. H. Lee and J. B. Ra, "Block motion estimation based on selective integral projections," in *IEEE ICIP*, 2002, vol. I, pp. 689–693.

[6] P.H.S. Torr, "Geometric motion segmentation and model selection," *Philosophical Transactions of the Royal Society A*, pp. 1321–1340, 1998.

[7] P. Milanfar, "A model of the effect of image motion in the radon transform domain," *IEEE Trans. on Image Processing*, vol. 8, no. 9, pp. 1276–1281, 1999.

[8] A. Kokaram, R. Dahyot, F. Pitié, and H. Denman, "Simultaneous luminance and position stabilization for film and video," in *Visual Communications and Image Processing,San Jose, California USA*, January 2003.