Restricted Affine Motion Compensation in Video Coding using Particle Filtering

Manoj Alwani
The LNM Institute of Information and Technology, Jaipur, India
manoj.07@lnmiit.com

Ravi Chaudhary
Indian Institute of Technology Delhi, India
ravi.iitd09@gmail.com

Mona Mathur
ST MicroElectronics, Noida, India
mona.mathur@st.com

Sumantra Dutta Roy*
Indian Institute of Technology Delhi, India
sumantra@ee.iitd.ac.in

Santanu Chaudhury
Indian Institute of Technology Delhi, India
santanuc@ee.iitd.ac.in

ABSTRACT

We propose a novel particle filter-based motion compensation strategy for video coding. We use a higher order linear model in place of the traditional translational model used in standards such as H.264. The measurement/observation process in the particle filter is a computationally efficient mechanism as opposed to traditional search methods. We use a multi-resolution framework for efficient parameter estimation. Results of our experimentation show reduced residual energy and better PSNR as compared to traditional video coding methods, especially in regions of complex motion such as zooming and rotation.

Keywords

Affine, Video Compression, Motion Compensation, Particle Filtering

1. INTRODUCTION

In this paper, we propose two modifications to traditional motion compensation techniques used in popular standards such as H.264. First, we propose a higher order parametric 2-D linear transformation model in place of a pure translational model. The entire system is in a particle filtering framework. Next, we replace traditional search techniques such as full-search and diamond search with a randomised measurement model of a particle filter. In addition to having a good estimate of the matching block in the previous frame, this offers a considerable computational advantage. Our system uses a multi-resolution framework to get a better motion estimate in a computationally efficient manner as opposed to exhaustive search techniques. We show experimental results in support of the proposed ideas.

Traditional video coding techniques such as H.264 [8] use motion estimation and compensation for rectangular blocks to represent information in a block in terms of a closely matching block in a previous/future frame and the corresponding motion vector, in addition to storing the prediction error in terms of a coded residual image. The assumption is that all pixels in the rectangular block correspond to the same translational motion model. This assumption is quite justified for very small block sizes. Further, it provides advantages of low computational complexity and small motion vector overhead since only two parameters are required to represent translational motion. However, the assumptions of rectangular blocks, and a translational model for the motion of a block are highly restrictive.

The idea of using higher order motion models for estimation and compensation is not new. Zhang et al. [10] propose an affine model for the same. The authors propose a layered approach for the same. They first estimate the zoom, rotation and translation at the frame level. They use a 4 parameter restricted affine model at a 32×32 block level. At a microblock level 16×16, the authors use a standard translational model. A disadvantage of this approach is the relative complication of the layered approach. Further, there is no prediction to help in the process of estimation of the affine parameters. Gahlot et al. [4] use an affine model for motion compensation. The authors however, operate only on MPEG-4 videos, which facilitate object-background segmentation. In our approach we do not use any object-background segmentation, or assume MPEG-4 streams, or use any separate object coding. We operate on image blocks irrespective of any semantic separation into objects moving across a background. In the work by Wiegand et al. [9], affine motion compensation is combined with long-term memory motion-compensated prediction. The idea was to determine several affine motion parameter sets on sub-areas of the image. For each affine motion parameter set, a complete reference frame is warped and inserted into the multi-picture buffer. Given the multi-picture buffer of decoded pictures and affine warped versions thereof, block-based translational motion compensated prediction and Lagrangian coder control are utilized. The prime disadvantage of this work is the

*Corresponding Author

ICVGIP '10, December 12-15, 2010, Chennai, India

huge computational overhead, and larger memory requirement - to store affine warped frames as well as previously decoded frames to be used for motion compensation. In contrast, we do not reply on any warped frames to be generated at the decoder side. We use motion compensation in the parameter domain instead of the image domain. Kalman filtering is a possible prediction mechanism which assumes the prior state model, the state/process dynamics model, as well as the observation/measurement model - all three as Gaussians. The authors in [6] use Kalman filtering, but only for refinement of the motion vector. The work is a combination of a full search algorithm, AR model and Kalman Filtering. A limitation of the above approach is that the authors use an AR model for predicting motion vectors for all blocks, but the coefficients of the model are fixed - empirically determined. In contrast, our system avoids the complexity of a full search over the entire image, our motion models are adaptive in getting coefficients using a multi-resolution approach (A pyramidal model [1]). The authors in [6] consider only a translational model. A particle filter generalises the idea of a Kalman filter to work for any distribution: not necessarily Gaussian. Bober and Kittler [2] propose a Hough Transform-based hierarchical algorithm for motion segmentation and estimation. A disadvantage of their technique is the use of non-linear optimisation for motion estimation. In contrast, our technique does not suffer from any such overhead. Particle filtering is a common thread through our approach. The particle filter uses an affine state vector, and the motion vector identification happens through the measurement/observation phase of the particle filter using a randomised approach. An estimate of the motion vectors comes from a computationally efficient multi-resolution approach. To the best of our knowledge, no related work addresses these issues.

The rest of the paper is organised as follows. Section 2 outlines the basic approach in the paper. We show experimental results in support of the proposed ideas in Section 3. Section 4 concludes the paper.

2. RESTRICTED AFFINE MOTION COMPENSATION

2.1 General 2-D Parametric Motion Models

A 2-D affine motion model is particularly attractive since it has 6 parameters (as opposed to the most general linear model: a projective model, which has 8 parameters), estimation of which is easier than for a projective case, for instance. An affine model can handle common cases of motion and deformation, such as translations, rotations, non-uniform scaling/zooming and shear - which a simple translational model cannot handle.However, if the camera is moving, depth parallax cannot be taken care of using translational as well the affine model. But affine definitely increases the no of frames to which the approximation can be assumed to be valid. A commonly used model is a restricted affine one (e.g., [7]). This model has 5 parameters (i.e., an oriented rectangle), which are not difficult to estimate. This strikes a balance between the class of linear deformations that the transformations can handle, and the difficulty in estimating these parameters. In our approach a block in the image is described by an oriented rectangle specified by 5 parameters $[x, y, w, h, \theta]^T$ where, x, y represents centroid of the tracking window, w, h represent its width and height, and θ represents the angle of the tracking window.

2.2 Particle Filtering-based Tracking for Motion Estimation and Compensation

Isard and Blake [5] proposed CONDENSATION/Particle Filtering as a tracking paradigm that works for any distribution which can be represented as a discrete array of samples. In coding standards such as H.264, for every (rectangular) block in an image, one searches for the closest matching block of the same size, in a reference image (e.g., the previous frame), using a simple translational model. For each frame, we use a particle filtering-based approach to find the most suitable block in the previous frame, using a more complex 2-D linear transformation model (restricted affine, Section 2.1). Any predictive tracker has two models

1. *The State/Process Dynamics Model:* In the absence of any prior information about the movement of pixels in a particular block, a random walk model is often the best possible option which can account for any type of motion.

$$\mathbf{X_t} = \mathbf{X_{t-1}} + \mathbf{Wn} \tag{1}$$

We consider $\mathbf{W}$ to be the noise covariance, and the noise term $\mathbf{n}$ to be a time-independent zero-mean unit-variance Gaussian. While there is no general rule for fixing covariance values, one can often make some good estimates if one has some prior knowledge. A common assumption in trackers is to assume different elements of the state vector to be uncorrelated, and hence, have the covariance matrix modelled as 1-D Gaussians in each dimension. For instance, sports videos involve a lot of action, usually. Even in the absence of such prior information, we take high individual variance terms, to start with. We have experimented with an adaptive strategy. We start with high values, and depending on typical values in a region of an image, we gradually increase or decrease these values. Making a tracker robust to such situations is not difficult in a multi-scale scenario [3]. This fits in seamlessly with our framework here. We estimate the parameters of the model i.e., $\mathbf{W}$ using a computationally light multi-resolution framework (described in detail in Section 2.3). Here, t is the subscript indicating the given frame, and $t-1$ refers to the reference frame (e.g., the previous one).

2. *The Measurement/Observation Model:* We use a simple sum-of-squared errors-based observation model to obtain the probability of a block (orientated rectangle in the reference frame) being the closest match to a block being considered in the given image. We estimate the match probability as

$$\Pi_t \stackrel{\triangle}{=} P(\mathbf{Z_t}|\mathbf{X_t}) \propto e^{-SSE} \tag{2}$$

where $\mathbf{Z_t}$ is the measurement/observation at time t corresponding to state $\mathbf{X_t}$, and SSE is the sum-of-squared errors for the oriented rectangular block in the reference image, when warped into the frame of reference of the block in the given image.

Our particle filtering-based motion estimation algorithm has the following steps. We consider an array of N particles (We have considered N to be a function of the resolution for computational efficiency: details in Section 2.3).

1. **Initial Distribution of Particles:** We consider a 5-dimensional Gaussian distribution of particles around the current position of the block, and take N samples from it, to form the set of particles. For our experiments, we have chosen N as a function of the resolution level. In most cases, we had N as 300 for the highest resolution.

2. **Prediction:** We move particles $\mathbf{s_t^{(i)}}$, $i \in \{1, N\}$ according to the deterministic dynamics of motion model (drift), then perturb them individually (diffusion). The above state/process dynamics model (Eqn. 1) comes of use here. Fig. 1 illustrates these two points.

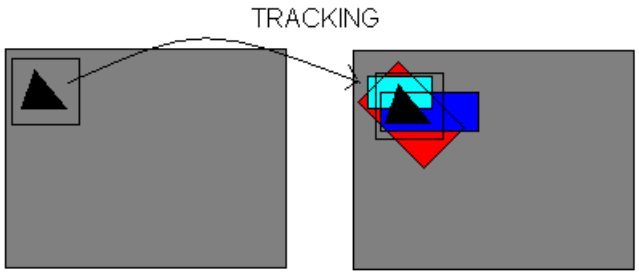


Figure 1: For a block in the given image, the system uses particle filtering to find the best matching block in the reference frame (here, shown to be the previous frame). Section 2.2 outlines the details.

3. **Measurement/Observation:** We use the measurement/observation model (Eqn. 2) for each particle i, $i \in \{1, N\}$.

$$\Pi_t^{(i)} = P(\mathbf{Z_t}|\mathbf{X_t} = \mathbf{s_t^{(i)}}) \quad (3)$$

4. **Output:** The output of the prediction process is the best particle i.e., the particle corresponding to the highest measurement/observation probability.

2.3 Multi-Resolution Analysis to Efficiently Seed the Motion Model for a Block

We use multi-resolution analysis to efficiently estimate the state/prediction model parameters in a computationally efficient manner. We represent the image as a Gaussian pyramid [1]. In Gaussian pyramidisation there is a trade-off between the accuracy and the time complexity of the prediction algorithm. At higher resolutions the accuracy of the prediction is higher but at the same time the computational complexity also high and at the lower resolution time complexity is low but the accuracy of prediction is also reduced. To consider this we take the image up to a level which preserves the basic characteristics of an object and at the same time ensures the reduction in computation complexity. Since video and image resolutions are not completely arbitrary (CIF/QCIF/VGA etc define de-facto standard sizes), we can only have a fixed number of levels of a pyramid before the dimensions of the image start getting too small. For instance, for QCIF resolution, we consider a total of three levels - empirically. After the prediction of motion parameters for the states at the lower resolution we use them as a seed for accurate prediction at the higher resolutions. These form the initial set of values for the estimation of the corresponding parameters at a higher resolution level. At higher resolutions, to get the affine refinement we generate a random particles around the seed point and then Measurement/Observation model is applied to each particle to get a better estimate of the required motion vector. At each stage, the best particle is chosen to be the output of the particle filter.

3. EXPERIMENTAL RESULTS

In our experiments we demonstrate the performance of the proposed affine motion compensation scheme in comparison to the traditional translational model that is commonly used in video coding applications. The standard test video sequences that are used for the experiments are “Mobile ”, “Flower ”, “Tempete ”, “Container ”and “Salesman ”, each of frame-size 352 X 288. Each of these sequences have been sub sampled to 15 frames by frame skipping (1 frame each) to create larger motion displacement. Mobile sequence contains rotation and zooming. Flower sequence contains rotation and translation. Tempete sequence contains zoom-out. Container sequence contains translation. Salesman contains rotation of an object. In the translational scheme videos are encoded with a block-size of 16 X 16 and search window of 32 X 32 pixels.

To evaluate the performance of the proposed scheme we use PSNR (dB) and residual energy (in terms of SSE) as measurement.Table 1 summarizes the comparison of the PSNR and residual energy.

The real motivation for using affine compensation can be seen in the results for PSNR in Table 1 which show an improvement of up to 1.2 dB and an average improvement of 0.7 dB over the translational model. In these standard test sequences the object has not been segmented out, if the motion estimation is performed over the segmented regions rather than pure block based search then the algorithm would perform much better. For this purpose we have captured our own video sequences in which different motions are carried.These videos can be made available on demand. Table 2 summarizes the results for different types of motions of an object.Results show that the proposed model gives lower residual energy and better PSNR for affine motions like rotation, zooming and translation as compared to the translation model.

Figs. 3 and 5 display a comparison of PSNR and residue energy of “Salesman ”and “Mobile ”using respectively for both the schemes. It is observed that the PSNR in the reconstructed “Salesman ”frame no. 12 is 35.8 dB by our scheme as compared to 34.1 dB by the translational model. It is due to the sudden rotation in the left hand of the salesman and movement in his face as highlighted in Fig. 2. The most important point to note is that the proposed algorithm compensates this sudden motion and thereby reduces residual energy and increases PSNR as shown in Fig. 3.

For the “Mobile ”sequence due to the constant zooming and rotation of the ball and train a constant improvement of about 1.2 dB in the PSNR is achieved as shown in Fig. 5. This is due to the zoom out in the sequence along with the rotation of the ball in the sequence. The improvement in residual energy can be seen from Fig. 5.

In addition, the visual quality of the reconstructed image is also improved considerably. This can be seen in Fig. 2

Table 1: Residual energy and PSNR comparison for standard test sequences

Sequence type	Full Search(Avg SSE)	Proposed Scheme (Avg SSE)	Residual Energy Gain(%)	Full Search (Avg PSNR dB)	Proposed Scheme (Avg PSNR dB)	PSNR Gain (dB)
Mobile	21263000	16116242	24.21	23.99	25.19	1.2
Container	2268000	1844300	18.68	33.71	34.61	0.89
Salesman	568580	490720	13.69	34038	35.13	0.75
Flower	14265000	12175000	14.65	26.08	26.73	0.65
Tempete	12063000	11025011	8.60	26.46	26.85	0.39

Table 2: Residual energy and PSNR comparison for object motion in our own captured videos

Test Sequence	Full Search(Avg SSE)	Proposed Scheme (Avg SSE)	Residual Energy Gain(%)	Full Search (Avg PSNR dB)	Proposed Scheme (Avg PSNR dB)	PSNR Gain (dB)
Translation	373890	263670	29.48	39.03	40.15	1.12
Rotation	156230	131520	15.82	34.27	35.57	0.3
Zooming	679120	582140	14.5	36.92	37.21	0.29

which shows the reconstructed images of frame no. 12 for the Salesman sequence. For the Mobile sequence the improvement can be seen in Fig. 4. The frames have been reconstructed without adding the residue for both models.

In order to validate our results we have performed Intra-coding on residue at different values of QP obtained from both the schemes. As we have seen that the residue energy in proposed scheme is less as compared to the other scheme so at different values of QP compression is also better than the other scheme. In Fig. 6 we have shown the percentage compression gain of proposed scheme over translation model. We have shown these results on "mobile ","Tempete ", and "Flower "videos which contains different types of motions. Fig. 6 also shows the variation in compression gain with change in QP. We have used standard H.264 JM encoder (ver 17.0) for Intra Coding. The configuration file is modified to work in Intra mode only.

4. CONCLUSIONS

An efficient affine motion estimation algorithm using particle filtering is presented. In sequences where complex motion is involved, frame prediction on the basis of affine motion estimation is more suitable than that involving estimation of translational motion only. Experimental results indicate proposed algorithm outperforms standard translational model in terms of PSNR, residual energy and compression. The proposed motion estimation algorithm takes care of object motion due to translation, zooming and rotation. This in turn results in high quality video coding. However, the computational complexity involved in this proposed algorithm is high as compared to traditional translational model. We may overcome this limitation if a dedicated architecture is developed for this purpose.

5. REFERENCES

[1] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden. Pyramid Methods in Image Processing. *RCA Engineer*, 29:33 – 41, 1984.

[2] M. Bober and J. Kittler. A Hough Transform based Hierarchical Algorithm for Motion Segmentation and Estimation. In *International Workshop on Time Varying Image Processing and Moving Object Recognition*, pages 335 – 342, 1993.

[3] S. Dutta Roy, S. D. Tran, L. S. Davis, and B. S. Vikram. Multi-Resolution Tracking in Space and Time. In *Proc. IAPR- and IEEE-sponsored Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP)*, pages 352 – 358, 2008.

[4] A. Gahlot, S. Arya, and D. Ghosh. Object-based Affine Motion Estimation. In *Proc. IEEE Region 10 Conference*, pages 1343 – 1347, 2003.

[5] M. Isard and A. Blake. CONDENSATION - Conditional Density Propagation For Visual Tracking. *International Journal of Computer Vision*, 28(1):5 – 28, 1998.

[6] C. M. Kuo, C. Hsiesh, Y. D. Jou, H. C. Lin, and P. C. Lu. Motion Estimation for Video Compression using Kalman Filtering. *IEEE Transactions on Broadcasting*, 42(2), 1996.

[7] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-Based Probabilistic Tracking. In *Proc. European Conference on Computer Vision (ECCV)*, pages 661 – 675, 2002.

[8] I. E. G. Richardson. *H.264 and MPEG-4: Video Compression*. Wiley, 2003.

[9] T. Wiegand, E. Steinbach, and B. Girod. Affine Multipicture Motion-Compensated Prediction. *IEEE Transactions on Circuits and Sytems for Video Technology*, 15(2):197 – 209, 2005.

[10] K. Zhang, M. Bober, and J. Kittler. Video Coding using Affine Motion Compensated Prediction. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1978 – 1981, 1996.

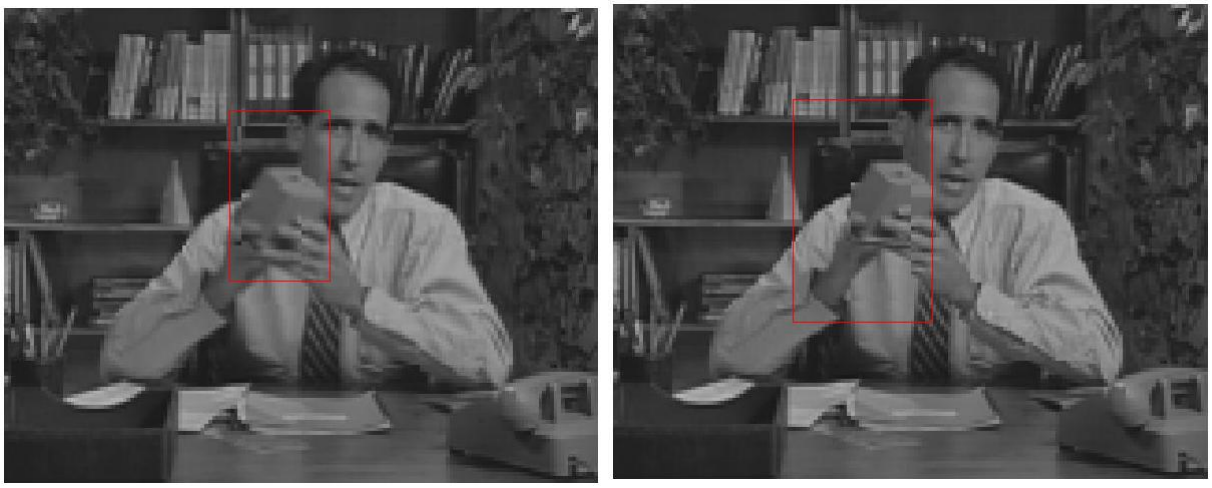


Figure 2: Reconstructed image (for a frame of the Salesman sequence) using the proposed method, and that using a translational model

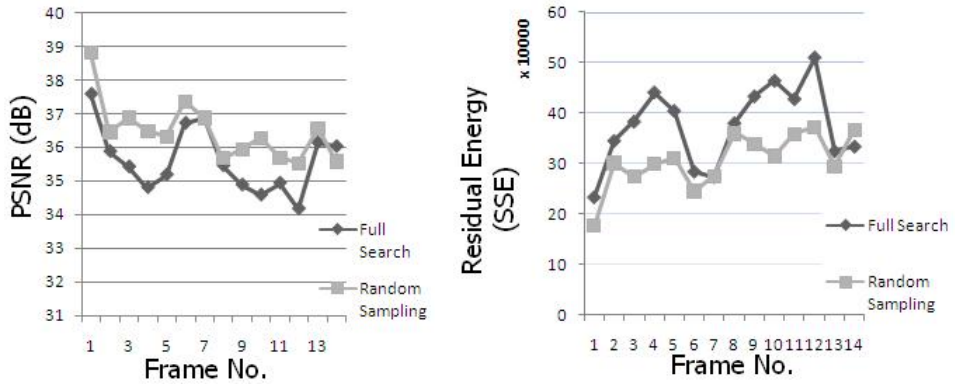


Figure 3: The PSNR, and residual energy comparison of both schemes (proposed, translational model) for the Salesman Sequence

Figure 4: Reconstructed image (for a frame in the Mobile sequence) using proposed scheme, and that using a translational model

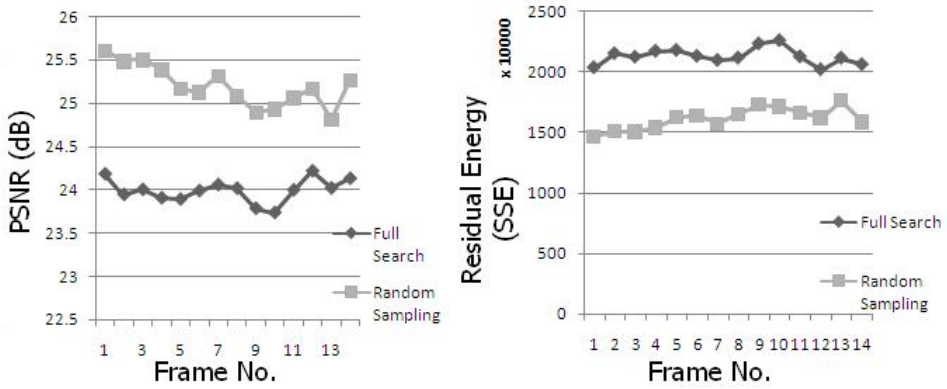
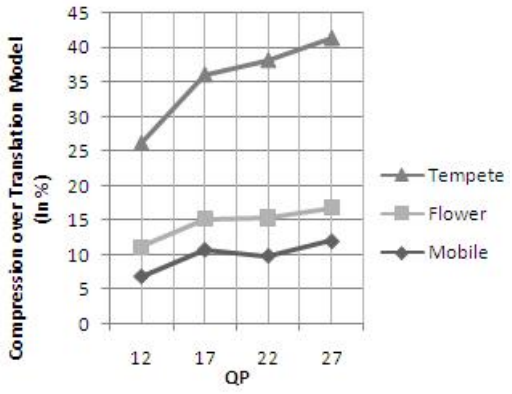


Figure 5: The PSNR, and residual energy comparison of both schemes (proposed, translational model) for the Mobile Sequence



Sequence	QP	Size in KB (Full Search)	Size in KB (Proposed Scheme)	Compression over Full Search(In %)
Mobile	12	586	546	6.83
	17	418	373	10.76
	22	264	238	9.85
	27	150	132	12
Flower	12	713	683	4.21
	17	522	499	4.41
	22	368	348	5.43
	27	231	220	4.76
Tempete	12	246	209	15.04
	17	177	140	20.9
	22	118	91	22.88
	27	69	52	24.64

Figure 6: Residue Compression as a function of QP