Noname manuscript No.
(will be inserted by the editor)

Restricted Affine Motion Compensation and Estimation in Video Coding with Particle Filtering and Importance Sampling: A Multi-Resolution Approach

Mithilesh Kumar Jha, Ravi Chaudhary,
Sumantra Dutta Roy, Mona Mathur, Brejesh Lall

09-01-2017

Abstract In this paper, we propose a multi-resolution affine block-based tracker for motion estimation and compensation, compatible with existing video coding standards such as H.264 and HEVC. We propose three modifications to traditional motion compensation techniques in video coding standards such as H.264 and HEVC. First, we replace traditional search methods with an efficient particle filtering-based method, which incorporates information from both spatial and temporal continuity. Second, we use a higher order linear model in place of the traditional translation motion model in these standards to efficiently represent complex motions such as rotation and zoom. Third, we propose a multi-resolution framework that enables efficient parameter estimation. Results of extensive experimentation show reduced residual energy and better Peak Signal-to-Noise Ratio (PSNR, hereafter) as compared to H.264/HEVC for instance, especially in regions of complex motion such as zooming and rotation.

Keywords- *Restricted Affine Model, Particle Filter, Importance Sampling, Multi-Resolution*

The authors are with the Dept of Elect. Engg., I.I.T. Delhi, Hauz Khas, New Delhi - 110 016, INDIA. Emails: {jham73, ravi.rkgit, mona.mathur}@gmail.com, {sumantra, brejesh}@ee.iitd.ac.in.

1 Introduction

We propose a novel particle filter-based motion estimation strategy, which uses a higher-order linear model (restricted affine). This is better adapted for complex motions such as rotation and zooming. The proposed multi-resolution framework brings computational efficiency in sampling a large search space for the closest matching block. The system uses importance sampling for combining evidence from random sampling-based and deterministic spatial and temporal cues. We integrate the proposed model with the H.264 reference code JM 17.0 and HEVC reference code HM16, and show encouraging compression results, at a reasonable computational complexity.

Conventional video compression schemes such as MPEG [1] and H.264 [2] use block-based translational motion estimation and coding. Advanced video compression schemes such as H.264 [2] and HEVC [3] have further introduced larger block sizes for prediction, larger block transforms and advanced motion vector partitioning [4]. However, they are still based on a translational motion model for inter-frame redundancy exploitation. All these conventional motion estimation techniques are not efficient in the prediction of complex motion regions involving effects such as rotation and zoom, thus limiting the overall compression that can be achieved. HEVC and H.264 approximate such complex motion by dividing bigger blocks and using multiple reference frames, but the corresponding motion vectors, block mode signaling and Rate Distortion (RD, hereafter) optimisation [2] are substantial overheads which affect the overall coding gain specially at high Quantisation Parameters (QPs, hereafter) [5].

Zhang et al. [6] propose a layered affine model. The authors first estimate zoom, rotation and translation at the frame level to account for global motion. A disadvantage of this approach is the relative complication of the layered approach. Further, there is no prediction to help the process of estimation of the affine parame-

ters. Wiegand et al. [7] combine affine motion compensation with long-term memory motion-compensated prediction. A prime disadvantage of this work is the huge computational overhead and larger memory requirement, to store affine warped frames as well as previously decoded frames to be used for motion compensation. Kordasiewicz et al. [8] propose an affine model-based in-loop post processing method benefiting from the advantages of affine motion prediction and using the conventional fixed block-based framework. A similar in-loop pre-processing method in [9], uses motion information provided by the four neighboring coded macro blocks, in addition to the motion vectors of the coding macro block. A limitation of such scheme is the significant computational cost at each macro block, irrespective of the motion of the respective blocks across the frame. Yuan et al. [10] propose an affine motion model with four parameters, to account for only global and local zoom.

Kuo et al. [11] use a Kalman filter-based tracker, but only to refine the motion vector. Kwolek [15] uses a particle filtering-based tracker in a H.264 teleconferencing video and varies the Quality Parameter (QP) for face and non-face regions. Kuo et al. [12] further propose an enhanced motion estimation algorithm based on a Kalman filter. They exploit block motion correlation to achieve high precision in motion estimation and compensation. In this work, a Kalman filter is used as a post-processing tool to increase motion compensation accuracy of conventional rate-constrained motion estimation. Luo et al. [13] propose a block-based motion estimation technique combining Kalman filtering with adaptive block partitioning. In this method, a first order autoregressive (AR, hereafter) model is applied to motion vectors obtained from block-matching algorithms. A particle filtering-based block-wise motion tracking and estimation method [14] is proposed utilizing state information of neighboring blocks. The proposed method can estimate motion and disparity with a smaller number of search points as compared with conventional estimation methods. The above discussed techniques are mostly based on a translational motion model and operate at the base (full) resolution unlike our proposed method.

Kuo et al. [11] use a combination of full search and an autoregressive model (with fixed and empirically determined coefficients). Chung et al. [16] present an affine model by extending their earlier work [17] to the affine domain. The authors determine the translation and rotation using spatio-temporal correlation of motion vectors and estimate scaling using a three-step-search [16]. The above approach suffers from the problem of converging to local minima, as in all existing fast search techniques. Cheung and Siu [9] propose a scheme for affine prediction by sending only the translational motion vectors in H.264 and computing the affine motion parameters at both the encoder and decoder. In addition to the computational complexity, some assumptions are not general enough. All the works above assume that all pixels within the coding macro-block have identical motions. [9] assume that the prediction error is distributed around the periphery of the partition.

Muhit et al. [18] propose a motion model for efficient representation of non-translational motion using a 2-D cosine based basis function. In this scheme, the authors propose to split the coding unit into different coding sizes (similar to standard H.264) to ensure the efficient motion estimation. The limitation with this is that it is still based on the one-to-one block search, increasing the computational cost.

Matthias et al. [19] have explored the affine motion model in combination to the translation model based on the rate-distortion and claim a compression gain upto 23% compared to HEVC. They propose to increase the search block size, however this method will not be efficient for local motion prediction within the block. Huang et al. [20], [21] and Chen et. al. [22], propose a control-point representation for the affine motion model using different block encoding modes (SKIP, DIRECT) addressing the problem of efficient coding of the affine parameters in the motion estimation. In these approaches, the motion parameters are derived from the motion of neighboring coded blocks and incorporated into the standard framework of SKIP and DIRECT modes. This differs from the evolutionary particle filtering based approach of [23]. A multiresolution framework [23] enables a coarse-to-fine refinement and scalability.

Li et. al. [24] recently propose a gradient based affine motion estimation with preliminary results that looks very promising but again lacks the scalability needed for real time video applications. The overhead of the affine parameters is not explained. A higher order motion estimation method [25] is proposed based on spatial and temporal local gradients. The proposed scheme seems more at a conceptual level and seamless integration of such scheme is not stated and therefore may not go along with the standard HEVC hardware and software design for video applications.

In motion estimation, it is desirable to simultaneously have high spatial resolution and accurate motion estimation at an acceptable level of computational complexity. This problem can further be related to the size of the search window as well as the complexity involved in increasing the window size. A multi-resolution approach enables searching in a larger window across multiple resolutions without increasing the computational complexity by a large factor. Motion parameters can be concisely represented at a lower resolution and further refined at higher resolution using a coarse-to-fine strategy. Lee et al. [26] propose a fast multi-resolution block matching algorithm using a translation assumption.

In contrast to the above approaches, our work is suited for accurately estimating complex motions at a reasonable computational complexity, and efficiently handling common non-translational effects such as rotations and zooms. Our approach has randomized and deterministic components to effectively search a large space across different resolutions (from coarse to fine in the proposed multi-resolution framework), and avoid convergence to local minima. In the proposed approach, we do not use any object-background segmentation (as in Gehlot et al. [27]) or use any separate object coding. We operate on image blocks irrespective of any semantic separation into objects moving across a background. We have combined our scheme with the standard H.264 and HEVC, and shown better motion estimation and compression results. This work builds on some very preliminary ideas of the authors in [28]. Our previous work [28] was presented more as a proof of concept which we have extended

extensively in this submission by: a) integration of the proposed framework into the H.264 and HEVC standard, b) performing extensive experiments for a wide range of video sequences with various object motions, and c) qualitative and quantitative performance analysis.

The rest of the paper is organized as follows: Sec. 2 provides a detailed description of the proposed motion estimation model. In Sec. 3, we discuss integration of the proposed method with H.264/HEVC, and show experimental results. Sec. 4 concludes the paper.

2 The Proposed Motion Estimation Model

The proposed method considers particle filter-based motion estimation in a multi-resolution framework. There are two prominent problems with existing standards such as H.264 and HEVC. First, a local exhaustive search strategy involves a computationally expensive search in the neighbourhood of the block/coding unit of interest. Such a strategy will not work if there is a considerable amount of motion between frames (the search region is typically fixed in size). Our particle filtering-based approach (Sec. 2.2) is a computationally efficient search procedure to sample a search space with a statistically significant set of search locations (particles). Second, a translational motion model is sometimes a gross approximation to motions which involve rotations and scaling, for instance. Existing standards try to approximate motions more general than translational, by going up to quarter-pel resolution. Sec. 2.1 describes a restricted affine model, which provides an easier alternative to such approximations, within the same H.264/HEVC model. Sec. 2.3 proposes a multi-resolution approach to sampling a larger search space using a smaller number of sampling points, performing a coarse-to-fine refinement in getting the closest matching block/coding unit. This importance sampling combines a deterministic search based on intra-frame proximity and temporal continuity, in conjunction with

a guided randomized search. Sec. 3 integrates the proposed framework with standard H.264/HEVC-based encoders, and shows better compression.

2.1 A Restricted Affine Motion Model

In general, video objects have 3-D motion. One uses 2-D parametric models as a practical approximation to represent the 3-D motion. 2-D models make the motion estimation problem over-determined, and numerically stable [29]. As opposed to the most general linear 8-parameter projective model, a 5-parameter restricted affine model strikes a balance between the number of parameters to estimate, and its generality. A 5-parameter tracker state model (at resolution r in the image at time frame t, Sec. 2.2) $\mathbf{X_t^r} = [x\ y\ h\ w\ \theta]^T$ accounts for commonly observed non-translational effects such as zooming (scale change), and rotation. $\mathbf{X_t^r}$ represents a rotated rectangle, with centroid (x, y), width w and height h, and orientation θ. Fig. 1, shows an example of the residual energy for various 16×16 H.264 macroblocks encoded using an affine model, vis-a-vis that using a translational model. H.264 gives an option to further split the macroblock if the motion estimation is not efficient (with an overhead of increased complexity). Detailed experiments with different coding block sizes are presented in Sec. 3.

2.2 Particle Filtering-based Motion Estimation

We replace the deterministic search of a H.264/HEVC framework with a tracking-based framework. A particle filter/CONDENSATION filter can work with any distribution represented as a collection of N particles [30]. A Kalman filter on the other hand, needs a Gaussian assumption for the prior density, the state/process dynamics model as well as the observation/measurement model. In an H.264/HEVC framework, one searches for the closest matching block of the same size, in a fixed search area. For a particle filter with a particle $\mathbf{X_t^r}$ as in Sec. 2.1, one needs two models:

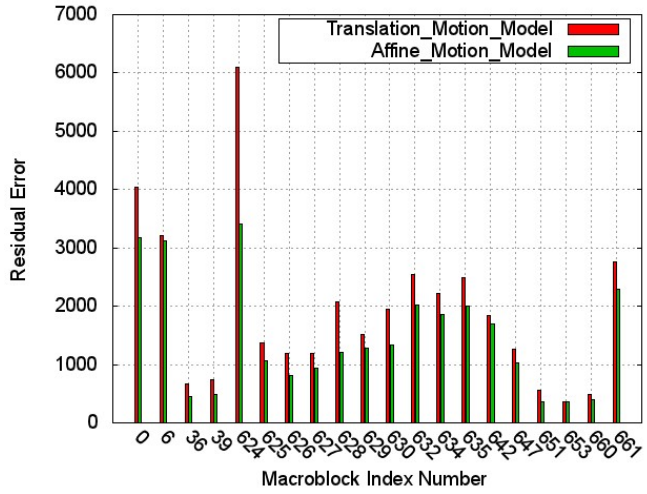


Fig. 1: A restricted affine motion model reduces the residual energy as compared to the standard translational model: an example with a representative pair of frames from a sample video sequence ('cross bridge'). This is for H.264, with the same 16×16 block size, for the sake of a uniform comparison. The $x-$ axis shows the 16×16 macroblock index number.

– *The State/Process Dynamics Model:* In the absence of any prior knowledge about the motion of a particular block in a video, a random walk model is often the best assumption:

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

Here, $\mathbf{n}$ is a time-independent zero-mean-unit-variance Gaussian noise, and $\mathbf{W}^T\mathbf{W}$ is the noise covariance. Sec. 2.3 explains our strategy to handle parameters of such models in a multi-resolution scenario. One can have a similar model across different resolutions [32]:

$$\mathbf{X_t^r} = f(\mathbf{X_t^{r-1}}) + \mathbf{Vm} \tag{2}$$

Sec. 2.3 outlines how one can take information about the prior motion of a particular block. The function $f(\cdot)$ is the resolution level difference, e.g., a factor of 2 for a translational component of $\mathbf{X_t^r}$, and unity, for a rotational component.

– *The Measurement/Observation Model:* Let $\mathbf{Z_t^r}$ denote the measurement/observation at time t at resolution r, for state $\mathbf{X_t^r}$. We estimate the match probability between a block in the target image, and the corresponding warped (5-parameter restricted affine) block, as a function of the sum of squared errors (SSE, hereafter):

$$\Pi_t^r \stackrel{\triangle}{=} P(\mathbf{Z_t^r}|\mathbf{X_t^r}) \propto e^{-SSE} \tag{3}$$

2.3 Multi-Resolution Particle Filtering and Importance Sampling

In this section, we describe the use of particle filtering and importance sampling in a multi-resolution framework.

The proposed method operates in the basic H.264/HEVC philosophy: having intra-coded (I) frames, and predicted (P)/bi-directionally predicted (B) frames. We call any intra-coded frame, a 'reference frame', and in this section, give an example with regard to a 'P' frame. Sec. 3 considers how our framework implementation gels with existing JM and HM implementations of H.264 and HEVC, respectively. Our system builds up a pyramid corresponding to each frame. Fig. 2 outlines the proposed coarse-to-fine resolution strategy.

Corresponding to resolution levels r_1, r_2 $\ldots r_n$ (going up the pyramid in Fig. 2, from the finest to the coarsest level), we have $N(r_i)$ particles at resolution level r_i. For instance, a 16×16 macroblock (in H.264/HEVC notation) at resolution r_1 will correspond to a 8×8 block at resolution r_2, and a 4×4 block at r_3, in a three-stage pyramid. To avoid computational complexity (at the cost of slight inaccuracy), one may omit a low pass filtering operation at every stage, before a sub-sampling [26]. The algorithm starts at the coarsest resolution r_n.

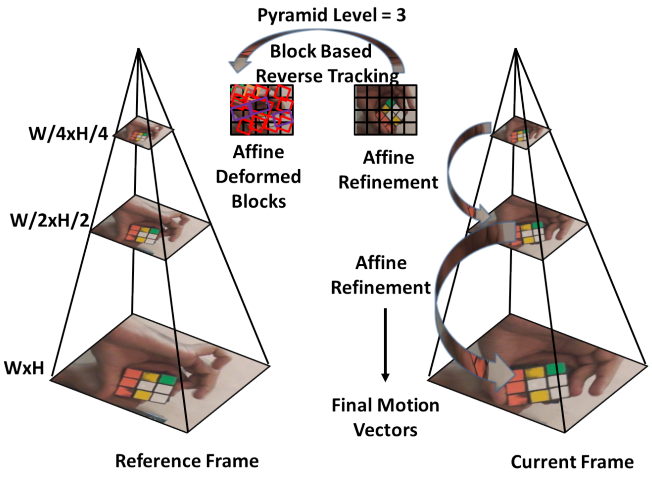


Fig. 2: For a block in the given image, we perform a coarse-to-fine search to efficiently compute motion vectors at the highest resolution level in a robust manner.

2.3.1 Importance Sampling at Resolution Level r_n

The work of Sullivan and Rittscher [31] proposes a basic framework for using deterministic search to guide random particles for a particle filter-based tracker. In our system, three types of $N(r_n)$ particles for a block are generated involving a combination of deterministic and random particles:

1. $N^N(r_n)$ particles, around the neighbourhood of a particular block. These may be generated from a small neighbourhood around the block's state vector e.g., $\pm$ 1 pixel around an end point of a oriented rectangle and $\pm$ the minimum resolution corresponding to an angle.
2. $N^H(r_n)$ particles, corresponding to the history of motion vectors, from the previous non-intracoded (non-I) frame. The $N^H(r_n)$ particles are randomly generated around the temporally previous motion vector corresponding to this block.
3. The remaining $N(r_i)$ - $N^N(r_n)$ - $N^H(r_n)$ particles are generated according to the assumed motion model (Sec. 2.2).

2.3.2 Coarse-to-Fine Refinement and Encompassing a Large Search Space

In this section, we describe how we propagate information across scales for a computationally effective search of the closest matching block.

The main motivation in using a multi-resolution approach is to cover a large part of the search space using a smaller number of search points at a coarse resolution (as in a prior work involving one of the authors[32]). A coarse resolution does not have accuracy in terms of pixel details, but has the advantage of computational efficiency in terms of a smaller number of pixels to process, which cover a larger area in an image. This is especially important for videos with large motion, which traditional H.264/HEVC cannot handle well. For them, the limitations of the search area result in reduced compression. We follow the particle filtering-based strategy of Sec. 2.3.1 both across time (one frame) and across resolutions.

In line with a sequential Bayesian filtering strategy [32], the number of particles at each level $N(r_i)$, $i = n-1, \ldots 1$ can be taken to be the same, or even successively reduced, to reduce the overall computational load. Propagating the best of the particles at a particular resolution level (among others) progressively refines the choices for the best block, allowing a more directed search in a smaller region with possibly the same, or a smaller number of particles. Such a strategy is advantageous in cases involving high motion or non-translational motion for instance, allowing accurate estimation at an acceptable level of computational complexity.

We start at resolution level r_n, and generate $N(r_n)$ particles as described in Sec. 2.3.1. For all particles at level n, we compute the error at each particle as in Eq. 3. We normalise these probabilities as $\sum_{j=1}^{N(r_n)} \Pi_t^{r_n}[j] = 1$. For each successive finer level r_i, $i = n-1 \ldots 1$ the $N(r_i)$ particles are generated as follows:

1. We order the $N(r_{i+1})$ particles above in decreasing order of their probabilities. We take the first $N^R(r_{i+1})$ as those which contribute to 95% of the probability mass.

$$\frac{\sum_{j=1}^{N^R(r_{i+1})} \Pi_t^{r_{i+1}}[j]}{\sum_{j=1}^{N(r_{i+1})} \Pi_t^{r_{i+1}}[j]} \leq 0.95 \tag{4}$$

These are the 'best' particles of the coarser resolution level, which have the state/process dynamics model of Eq. 2 applied to them.

2. The rest of the $N(r_i) - N^R(r_i)$ particles come from a combination of the neighbourhood particles are resolution level r_i and particles according to the motion vector history as in Sec. 2.3.1

At the finest resolution level r_1, we consider the particle j with the highest probability $\Pi_t^{r_n}[j]$, and use it to compute the motion vector for the current block. This is extrapolated to various levels of resolution r_i for the next frame according to Eqs. 1 and 2. Fig. 3 gives a diagrammatic view of the process.

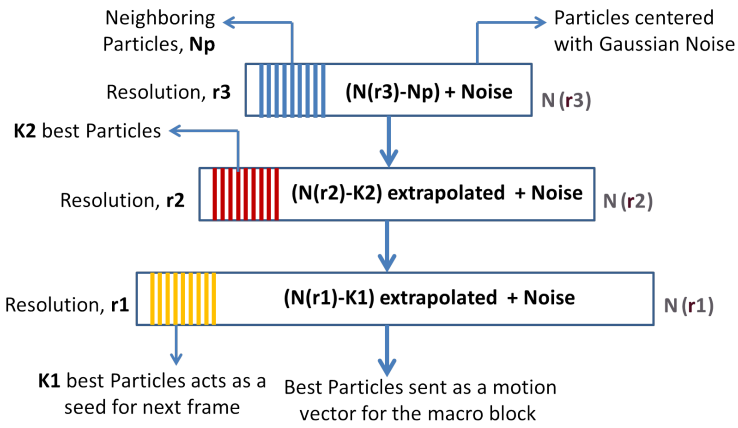


Fig. 3: For a block in the given image, the system has a coarse-to-fine strategy to compute motion vectors at the highest resolution level.

3 Integration Experiments in an H.264/HEVC framework

For all our experiments, we used a Windows 2007 PC with Intel Core- i5 CPU (2.4 GHz), 4GB RAM and a 180 GB HDD. We have experimented with a large number

of standard video sequences commonly used by HEVC (JCT-VC) and H.264/AVC (JVT) that have significant non-translational motion (rotations, zoom-in and zoom-out), and large inter-frame motions (Fig. 4 shows some representative sequences used in our experiments). As already highlighted, we show that the proposed method is more efficient for videos with significant non-translational motion, such as rotation and zoom, which are not efficiently represented by standards such as H.264 and HEVC. Experimental observation suggest that the proposed higher order linear model (restricted affine) in a multi-resolution framework is even more efficient in an HEVC framework, due to the provision of increased coding block size (up to 64×64) in HEVC.

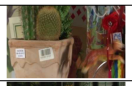
Test Sequence	Sample Frame	Sequence description
Cactus (1080P)		-- High local motion with stationary camera -- Objects rotating in different directions -- Shadows moving with rotating objects
Tennis (1080P)		-- High angular camera motion -- Fast movement of players and racket -- Motion of tree leaves -- Shadows moving with objects and camera
Party Scene (480P)		-- Constant zoom-out of camera -- Running girl, toy waving its hands -- High texture details across the wall
Car Crossing (VGA)		-- Zoom/in and zoom/ out of objects -- Hand-held camera -- Tree leaves moving in wind -- Texture along roadside
Football (CIF)		-- Very high local motion -- Fast motion of players -- Rapid movement of camera -- Fast movement of ball
Foreman (QCIF/CIF)		-- Angular texture on the wall -- Angular camera motion -- Person talking to the camera with facial expressions and moving his head and hands
Mobile (CIF)		-- High Texture details -- Constant zoom-out of camera -- Movement of objects in different direction
Park Scene (1080P)		-- Motion of different object in the scene -- Constant panning of camera -- Tree leaves moving with the wind and camera

Fig. 4: Test sequences used for experiments with their characteristics

We have used block sizes of 16×16 for QCIF (176×144) and CIF (352×488) videos, and 32×32 and 64×64 for high resolution videos (480P (832×480)/1080P

(1920×1080)). Larger block sizes are chosen for high resolution videos to ensure a better chance of capturing a representative large motion in the sequence (and hence, enabling better compression), as opposed to trying to capture it in parts with a large number of smaller blocks. In our experiments, we chose 2 levels of the Gaussian pyramid [36] for 16×16 and 3, for 32×32 and 64×64 blocks. The distribution of particles have been empirically chosen to be 200 at each resolution of the Gaussian pyramid. In order to test the performance of our algorithm at sub-pel accuracy we have used a simple bicubic filter which would generate similar results as a six-tap followed by a bi-linear filter used in JM-17.0 [33].

We have compared the results of the proposed scheme with H.264 and HEVC (JM 17.0 baseline profile and HM 3.1_dev, respectively), and integrated our scheme into the above standard frameworks. For instance, if the motion is closer to pure translational than restricted affine (5 parameters), the encoder sends them as translational blocks, to save on the extra number of bits needed to encode a translational motion as affine. For both H.264 and HEVC in the RD loop before the mode decision, we replace the original prediction with the new local affine prediction as a pre-processing step. We use the motion compensation framework of the standard, where we replace the predicted macroblock defined by the current slice pointer with the proposed affine based prediction model. The pre-processing decision is made on the basis of the Sum-of-Squared-Errors (SSE) of the predicted blocks by the standard reference code (SSE_{Trans}) and our scheme (SSE_{Affine}). The coding of a block as restricted affine has a cost of three extra motion vectors h, w and θ (as two translational parameter represent the traditional motion vectors). We restrict the variation in these three parameters to ± 7 and therefore introduce a worst case extra bit-overhead of 12 bits per coding unit. In our experiments, we choose a block to be affine if SSE_{Affine} is less than 90% of SSE_{Trans}. Fig. 6 shows a representative frame along with its categorisation of macroblocks. In H.264/HEVC, there is an option to further split macroblocks if the motion estimation is not efficient. We have done a compara-

tive study with varying the coding unit size in the standard HEVC and the proposed framework in HEVC as shown in Fig. 7.

Bit streaming is an important part in video coding standards. In order to achieve flexibility in a variety of applications and transport methods, H.264 separates the Video Coding Layer (VCL) and Network Abstraction Layer (NAL) as shown in Fig. 5. We propose to use unspecified NAL unit types in H.264 (24 to 31 in Table 7.1 of the H.264 standard) encapsulating all information related to the affine coded blocks and slices for the decoder to reconstruct. Similar to the handling of a skip-block, an affine coded block is indicated for entropy coding using CAVLC or CABAC in H.264. The video coding layer (VCL) (which encodes and writes each macroblock data in the NAL unit of slice to the *AnnexB* byte stream [34]) encapsulates information about the motion.

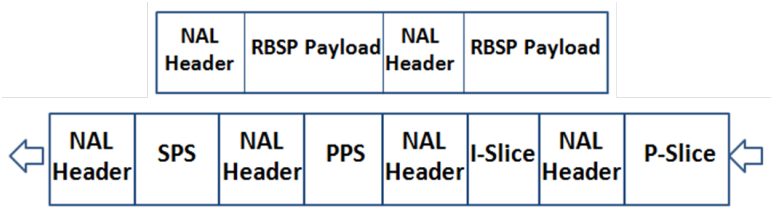


Fig. 5: NAL Unit Sequence [35]

For simplicity, we use an 'IPPP...' structure for the frames for most of our experiments, however we also use the default HM low delay configuration with default prediction sequences to ensure the seamless integration of the proposed framework with HEVC. Each P frame uses its immediate previous frame as reference. For coding efficiency, the average bit-rate and PSNR values have been compared for the first 25 frames at 4 different QP values, 22, 27, 32 and 37. After RD optimization, the percentage of affine coded macroblocks varies from 33% to 65%, which corresponds to a reduction in bit-rate. The baseline configuration is used from the standard for encoding in H.264 (JM references) and the encoder low delay configuration is used

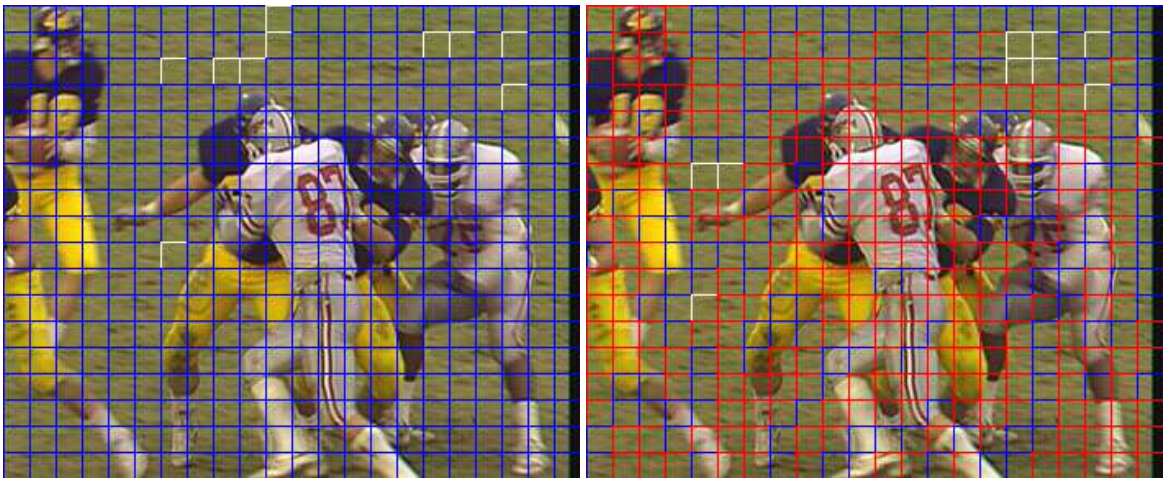


Fig. 6: Categorisation of macroblocks in a representative frame of the 'Football' Sequence (CIF) coded with standard JM and the proposed JM_{Affine}. Blocks in red are those coded as (restricted) affine blocks, blocks in blue are those coded in translation mode and those in white are those coded as skip blocks.

for encoding in HEVC (HM references). The detailed encoding parameters used are described in Table 1 and Table 2.

We compare the performance of the proposed model with HM-16 using the Bjøntegaard Delta (BD, hereafter) rate parameter [2]. Fig. 7 and Fig. 8 shows a representative example: a comparison of the RD plot for the 'Car crossing (VGA)', 'Park Scene (1080P)' and 'Cactus (1080P)' for the proposed method (HM_{Affine}), and that of a pure HEVC encoding (HM). The proposed method shows a trend of higher PSNR values at smaller bit rates, an indication of better coding efficiency with higher fidelity to the original unencoded video.

Tables 1 and 2 show results comparing the performance of our method with H.264 and HEVC respectively (JM_{Affine} and HM_{Affine}, versus the original H.264 and HEVC versions JM and HM) for some representative test sequences at sub-pel resolution. Table 3, shows comparative results of HM_{Affine} vs $HM16.0$ for the default encoding configuration from the standard.

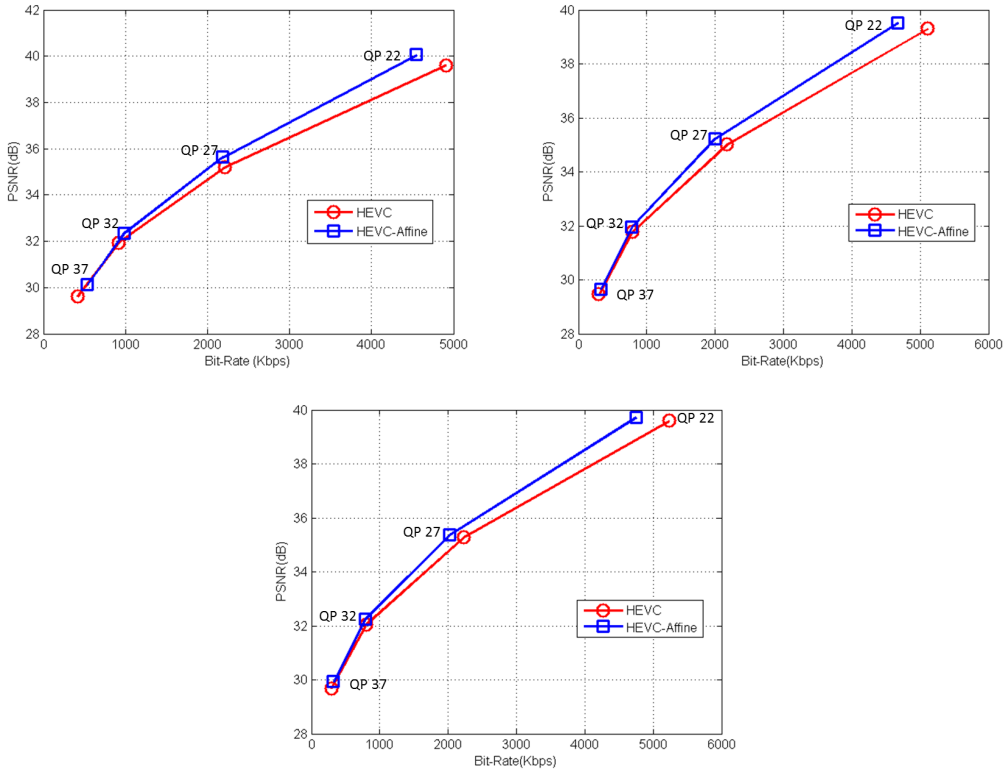


Fig. 7: RD Plot comparing HM and HM_{Affine} (pure HEVC, and the proposed method, respectively) for the sequence, 'Car Crossing' (VGA) using the coding block size as 16×16, 32×32 and 64×64 respectively, in raster scan order. The proposed method gives a higher PSNR value corresponding to a lower bit rate, indicating better coding efficiency with greater fidelity to the original unencoded video.

3.1 Discussion on the Computational Complexity

In this subsection, we present the time and storage complexity of the proposed algorithm. We have profiled JM and HM for encoding flow and generated the data for JM, JM_affine and HM, HM_affine. The comparative results are presented in Table 6.

The computational complexity of both algorithms is computed in terms of the number of addition and multiplication operations required. For an exhaustive full search with a search radius p, the number of addition operations required to calculate

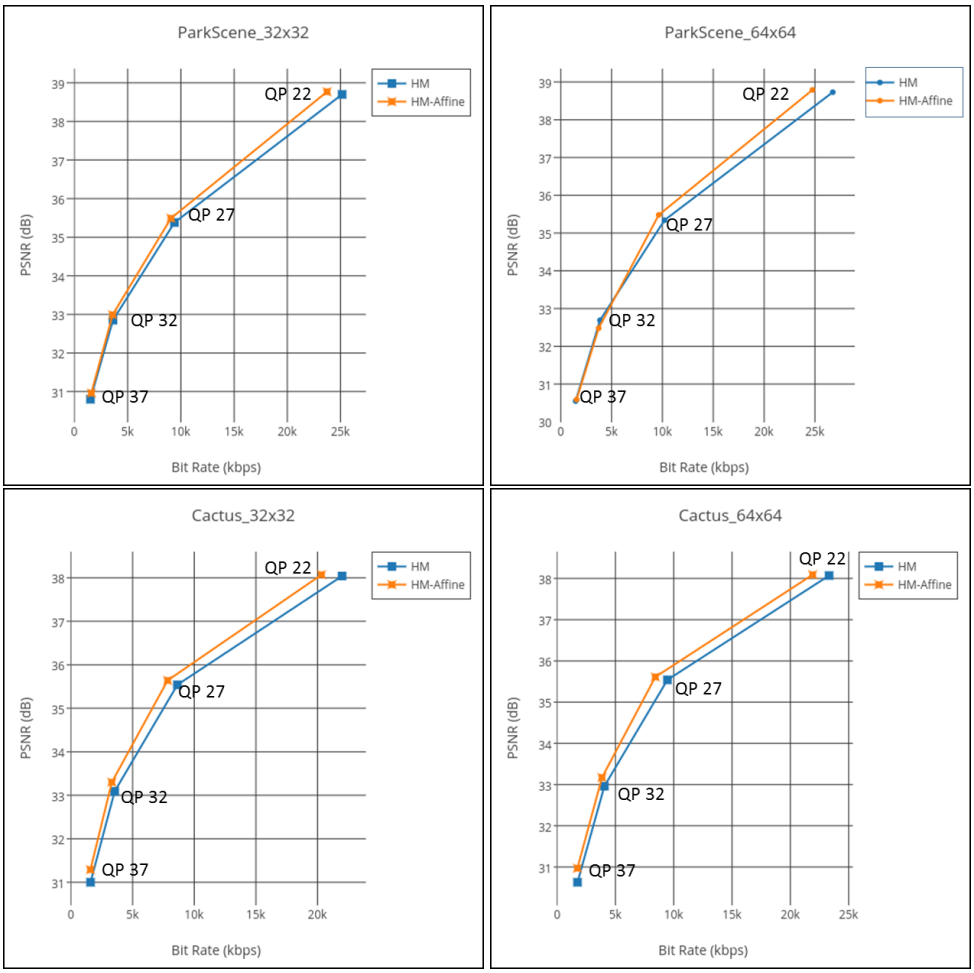


Fig. 8: RD Plot comparing HM and HM_{Affine} (pure HEVC, and the proposed method, respectively) for the sequence 'Park Scene' (the top row) and 'Cactus' (the bottom row) using the coding block sizes of 32×32 (the graph on the left side on each row) and 64×64 (the one on the right).

the SSE are $(2p+1)^2 * 2 * B_{size}^2$ and the number of multiplication operations required are $(2p+1)^2 * B_{size}^2$. Our motion model involves an extra warping overhead which depends upon the number of particles $N(r_i)$ and $B_{size}(r_i)$ at a resolution r_i. Warping each block requires $N(r_i) * 4 * B_{size}(r_i)^2$ additions and subtractions. Also for SSE calculation at each resolution level r_i, we require $N(r_i) * 2 * B_{size}^2(r_i)$ additions and $N(r_i) * B_{size}^2(r_i)$ multiplication operations. The 'Foreman' sequence as explained

Table 1: Comparison of the proposed method ('JM_{Affine}') with H.264 (JM-17.0, 'JM') for a search range of 16 for low resolution sequences and 32 for high resolution sequences. In general, JM_{Affine} results in greater compression at an improved PSNR. Encoding parameters used are: SearchRange=16/32, SearchMode=Full Search, Search Precision = Integer pixel, sub-pixel, Macroblock Partition Modes = P16×16, PSKIP, De-blocking Filter=ON, RD Optimization = ON, Entropy Coding = CAVLC

Sequence	QP	JM		JM_{Affine}		BD Rate Gain
		Bit-rate	PSNR	Bit-rate	PSNR	
Foreman (QCIF)	22	812	40.44	732	40.69	-14.96
	27	391	36.33	337	36.54	
	32	155	32.49	131	32.57	
	37	51	29.11	60	29.48	
Mobile (QCIF)	22	2139	40.7	2042	40.89	-10.16
	27	1426	35.47	1323	37.26	
	32	816	30.3	717	34.04	
	37	322	25.53	276	31.53	
Foreman (CIF)	22	2299	40.56	2076	40.89	-12.58
	27	962	37.03	869	37.26	
	32	383	33.68	375	34.04	
	37	170	30.74	209	31.53	
Football (CIF)	22	5040	40.93	4819	41.51	-9.61
	27	3021	36.99	2947	37.17	
	32	1636	33.03	1602	33.57	
	37	852	29.44	777	30.31	
Party Scene (480P)	22	23528	40.14	22913	40.2	-8.25
	27	13567	35.41	12986	35.5	
	32	7223	30.96	6570	31.1	
	37	3181	27.05	2916	27.4	
Car Crossing (VGA)	22	8870	41.86	8017	42.21	-9.02
	27	4962	37.73	4698	38.11	
	32	2298	33.7	2389	34.1	
	37	854	30.24	937	30.79	
Tennis (1080P)	22	39311	40.67	35983	40.79	-9.49
	27	18258	38.72	17563	38.91	
	32	9000	36.34	8900	36.69	
	37	4732	33.7	5017	34.16	
Cactus (1080P)	22	74114	39.45	66495	39.59	-9.12
	27	26198	36.75	25079	36.96	
	32	12709	34.35	13263	34.70	
	37	6245	31.70	6413	32.12	
Park Scene (1080P)	22	67362	40.25	61345	40.45	-12.06
	27	28629	36.91	27327	37.29	
	32	12369	33.66	11905	34.02	
	37	5448	30.86	5944	31.28	

earlier has a lot of angular motion, whereas the 'Football' sequence is characterized by very high and complex motion. In both sequences, the translational motion model is not able to perform well even at sub-pel level.

Table 2: Comparison of the proposed method ('HM_{Affine}') with HEVC (HM 16, 'HM') for a block size = 16×16 for low resolution videos and a block size = 32×32 for high resolution videos. In general, HM_{Affine} results in greater compression at an improved PSNR. Encoding parameters used are: SearchRange=16/32, SearchMode=Full Search, Search Precision = Integer pixel, Coding Unit (CU) size = 16/32/64, De-blocking Filter=ON, RD Optimization = ON, Internal Bit-Depth = 8, Entropy Coding = CAVLC

Sequence	QP	JM		JM_{Affine}		BD Rate Gain
		Bit-rate	PSNR	Bit-rate	PSNR	
Foreman (CIF)	22	1181	38.65	1071	38.74	-10.09
	27	459	35.25	436	35.5	
	32	210	32.76	222	33.22	
	37	120	30.57	144	31.44	
Football (CIF)	22	2664	39.12	2524	39.45	-11.87
	27	1409	34.55	1336	35.0	
	32	698	30.77	685	31.31	
	37	347	27.88	365	28.68	
Mobile (CIF)	22	5601	37.53	5250	37.5	-8.64
	27	3185	31.69	2858	31.69	
	32	698	30.77	685	31.31	
	37	347	27.88	365	28.68	
Party Scene (480P)	22	15206	36.91	14232	36.89	-7.06
	27	7645	31.86	7196	31.95	
	32	3165	28.14	3104	28.43	
	37	1226	25.48	1325	25.73	
Car Crossing (VGA)	22	4907	39.60	4548	40.06	-7.48
	27	2216	35.22	2184	35.64	
	32	912	31.93	983	32.35	
	37	414	29.63	537	30.12	
Tennis (1080P)	22	16179	40.29	15576	40.40	-7.61
	27	7548	37.84	7293	37.98	
	32	3682	35.40	3595	35.60	
	37	1948	33.26	1939	33.54	
Cactus (1080P)	22	21986	38.04	20316	38.07	-12.13
	27	8633	35.54	7839	35.64	
	32	3543	33.09	3313	33.3	
	37	1586	31	1578	31.29	
Park Scene (1080P)	22	25143	38.7	23731	38.77	-6.39
	27	9396	35.38	9068	35.49	
	32	3620	32.85	3589	32.99	
	37	1502	30.8	1583	30.96	

An important parameter in the process is the number of particles. A larger number of particles can better explore the search space at a given resolution level, at the cost of a larger number of computations. It is important to strike a balance between the two. For a representative set of videos, we considered the average performance of the strategy with 16×16 and two levels of resolution with the restricted affine

Table 3: Comparison of the proposed method ('HM_{Affine}') with HEVC ($HM16$) with default HM configuration of low delay main profile with GOP size=4,SearchRange=64, FastSearch=1, BipredSearchRange=4, InternalBitDepth=8

Sequence	QP	HM		HM_{Affine}	
		Bit-rate	PSNR	Bit-rate	PSNR
Foreman (CIF)	22	756	33.04	685	33.12
	27	294	30.14	279	30.35
	32	134	28.01	142	28.4
	37	77	26.14	92	26.88
Football (CIF)	22	1705	33.45	1615	33.73
	27	901	29.54	855	29.93
	32	447	26.31	438	26.77
	37	223	23.84	233	24.52
Mobile (CIF)	22	3585	32.09	3360	32.06
	27	2038	27.09	1829	27.09
	32	447	26.31	438	26.77
	37	223	23.84	233	24.52
Party Scene (480P)	22	9732	31.56	9108	31.54
	27	4893	27.24	4605	27.32
	32	2026	24.06	1987	28.21
	37	785	21.79	848	22.0
Car Crossing (VGA)	22	3140	33.86	2911	34.25
	27	1418	30.11	1398	30.47
	32	584	27.3	629	27.66
	37	265	25.33	344	25.75
Tennis (1080P)	22	10355	34.45	9969	34.54
	27	4831	32.35	4668	32.47
	32	2356	30.27	2301	30.44
	37	1247	28.44	1241	28.68
Cactus (1080P)	22	14071	32.52	13002	32.55
	27	5525	30.39	5017	30.47
	32	2268	28.29	2120	28.47
	37	1015	26.51	1009	26.75
Park Scene (1080P)	22	16092	33.09	15188	33.15
	27	6013	30.25	5804	30.34
	32	2317	28.09	2297	28.21
	37	961	26.33	1013	26.48

Table 4: Comparative study of the proposed scheme and a fast search method (TZ). Encoding configurations are: Fastsearch:1, SearchRange:32, MaxCUWidth: 64, MaxCUHeight: 64

Sequence	SSE (TZ)	SSE (Proposed)	% decrease in SSE
Tennis(1080P)	66778	45348	32.09
Park Scene(1080P)	254462	169367	33.44
Cactus(1080P)	388013	260395	32.89
Party Scene(480P)	138886	101306	27.06
Football(704X576)	151713	113754	25.02

model against a similar configuration with a pure translational model employing a full search, as in standard video coding standards. The performance was averaged over 10 sample runs. Fig. 9 shows the percentage reduction in the SSE with the number of particles varying from 100 to 250. As expected, increasing the number of particles results in a better approximation of the actual motion (which is far from being a simple

Table 5: Experimental results of the multiresolution motion estimation algorithm. Encoding configurations are: Fastsearch:0, SearchRange:32, MaxCUWidth: 64, MaxCUHeight: 64

Sequence	% decrease in SSE in Multiresolution Framework		
	2-levels	3-levels	4-levels
Tennis(1080P)	19.72	31.21	32.12
Park Scene(1080P)	20.28	32.98	34.01
Cactus(1080P)	21.26	31.22	32.69
Party Scene(480P)	17.73	26.18	31.83
Football(704X576)	14.57	22.33	23.59

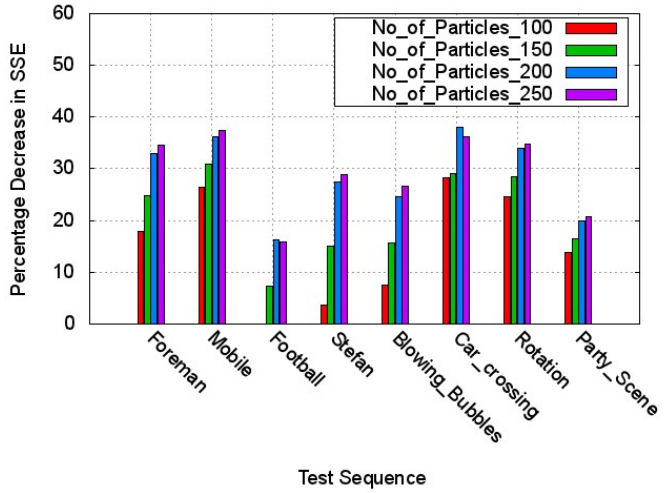


Fig. 9: Illustration of the % decrease in the SSE with a variation in the number of particles. The results compare our restricted affine configuration approach with the translational model.

translational one in the representative videos), showing a better match to the affine model, and better search towards the parameters of this model. The standard deviation varies from 0.1 to 2.7 for the different videos with 100 particles, with the average SSE steadily decreasing as the number of particles increases. For 200 particles, the range of standard deviations is 0.06 to 0.54, which does not change appreciably when the number of particles is increased to 250 (while reducing the SSE, but the amount of reduction is not as drastic as with a smaller number of particles, as in Fig. 9). The

associated computational cost in going from 200 to 250 particles however registers an increase of about 25%. We observe similar results for larger block sizes as well.

Further, if we augment the translational model with full search with a pyramid, we observe an approximate 30% saving in the total number of computations with the proposed method. For these experiments, we considered 16×16 block sizes, and a search range of 32. This illustrates the advantage of using a particle filter-based strategy (as in our proposed method, with 200 particles in this experiment), as against one which involves a full search.

Another parameter in the process is the number of levels of resolution. Fig. 10 shows the percentage decrease in the SSE with increasing the number of levels of the Gaussian pyramid. Smaller frame size videos for H.264 with block sizes of 16×16 are not appropriate for this experiment, since with 3 levels of resolution, a block size of 4×4 is too small to contain any appropriate information. In this experiment, we have considered high resolution videos and hence have made a comparison with standard HEVC. As can be seen from these results, the percentage decrease in the SSE tends to taper off as the number of levels is increased. For instance, the percentage SSE decrease from 3 to 4 levels is not considerable.

Table 6: Time complexity analysis for JM, JM_{affine}, HM and HM_{affine}. Columns 2 & 3 give the encoding time (In Seconds) using standard JM (QP=28, Search Range = 32, encoded frames = 25), and the Affine-based proposed scheme in JM framework ('JM_affine'). Similarly, columns 4 &5 give the motion estimation time for JM and JM_affine. Columns 6 & 7 give the encode time using HM (QP=28, Search Range=32, encoded frames = 25) and the Affine-based proposed scheme in HM frame work('HM_affine').

Sequence	Encode time JM	Encode time JM_affine	ME time JM	ME time JM_affine	Encode time HM	Encode time HM_affine
foreman(176X144)	5.17	8.52	3.03	3.15	10.82	8.10
flower(352X288)	21.68	33.114	13.49	15.76	33.39	46.24
football(352X288)	26.415	38.07	18.06	27.28	39.99	44.2
car_crossing(640X480)	68.31	105	44.47	54.5	80.45	96.38
party_scene(832X480)	88.69	136.5	55.66	68.05	123.18	143.63

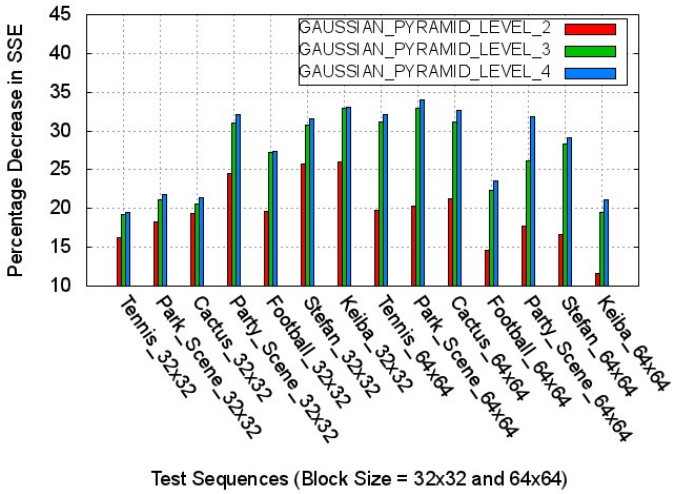


Fig. 10: The percentage SSE decrease over a single level of resolution, with 2, 3 and 4 levels, for representative high resolution videos, for block sizes of 32×32 and 64×64. Sec. 3

4 Conclusion and Outlook

In this paper we propose a novel multi-resolution motion compensation scheme based on a restricted 5-parameter affine representation. We use a particle filter-based tracker in a multi-resolution framework (for computational efficiency in sampling a large search space for the closest matching block) and use importance sampling for using evidence from deterministic spatial and temporal cues, to guide a particle filter-based block tracker for fast and efficient motion estimation. We have combined our scheme with a standard H.264 and HEVC framework and shown improved motion estimation and compression results.

One of the known improvement areas of the framework is the handling of multiple motions if they are close enough and the energy distribution within a region corresponds heavily to one particular motion. With the increase in block-size, the probability that more than one motion within the block exists increases. Although two motions can be detected and tracked using the coarse-to-fine strategy, we select

only a single restricted affine estimator for the block. Further partitioning of the block on the basis of motion peaks could be future work. A quick solution to these problems could be segmentation prior to tracking. First, identify the dominant motion peaks, remove all dominant motion peaks corresponding to the primary motion. Now since all dominant peaks corresponding to primary motion are removed we can again identify the dominant motion peaks for secondary motion. But further research would be required in this area to get the best possible segmentation technique.

References

1. ITU-T Rec. H.262 and ISO/IEC 13818-2 (MPEG-2), Generic coding of moving pictures and associated audio information - Part 2: Video, (Nov. 1994)
2. ITU-T Rec. H.264 and ISO/IEC 14496-10 (MPEG4-AVC), Advanced video coding for generic audio-visual services, (v1, May 2003; v2, Jan. 2004; v3, Sept. 2004; v4, July 2005)
3. ITU-T Rec. H.265 and ISO/IEC 23008-2 (HEVC), High Efficiency Video Coding, (v1, Apr. 2013; v2, Oct. 2014; v3, Apr. 2015)
4. MINEZAWA, A., SEKIGUCH, S., AND SUGIMOTO, K. Te12.2 report (m18142) on motion vector (MV) prediction AMVP/IMVP. In *Mitsubishi Electric, Technical Report JCTVC-C119*, (2010)
5. KAMP, S. Video coding using decoder-side motion vector derivation. In *RWTH Aachen University, Germany, Technical Report (Online)*, (2008)
6. ZHANG, K., BOBER, M., AND KITTLER, J. Video coding using affine motion compensated prediction. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp.1978 – 1981, (1996)
7. WIEGAND, T., STEINBACH, E., AND GIROD, B. Affine multi-picture motion-compensated prediction. *IEEE Transactions on Circuits and Systems for Video Technology* , vol.15, no.2, pp.197 – 209, (2005)
8. KORDASIEWICZ, R. C., GALLANT, M. D., AND SHIRANI, S. Affine motion prediction based on translational motion vectors. *IEEE Transactions on Circuits and Systems for Video Technology* , vol.17, no.11, pp.1388 – 1394, (2007)
9. CHEUNG, H. K., AND SIU, W. C. Local affine motion prediction for H.264 without extra overhead. *IEEE International Symposium on Circuits and Systems,*, pp.1555 – 1558, (2010)
10. YUAN, H., LIU, J., SUN, J., LIU, H., AND LI, Y. Affine model based motion compensation prediction for zoom. *IEEE Transactions on Multimedia,* vol.14,no.4, pp.1370 – 1375, (2012)

11. KUO, C. M., HSIESG, C., JOU, Y. D., LIN, H. C., AND LIU, P. C. Motion estimation for video compression using Kalman filtering. *IEEE Transactions on Broadcasting,* vol.42,no.2, pp.111 – 116, (1996)

12. KUO, C. M., CHUNG, S. C., AND SHIH, P. Y. Kalman filtering based rate-constrained motion estimation for very low bit rate video coding. *IEEE Transactions on Circuits and Systems for Video Technology,* vol.16,no.1, pp.3 – 18, (2006)

13. LUO, Y., AND CELENK, M. Kalman filtering based motion estimation for video coding with adaptive block partitioning. *IEEE Workshop on Signal Processing Systems,* pp.129 – 134, (2008)

14. YANG, S. Particle filtering based estimation of consistent motion and disparity with reduced search points. *IEEE Transactions on Circuits and Systems for Video Technology*, vol.22,no.1, pp.91 – 104,(2012)

15. KWOLEK, B. Face tracking for H.264 encoded video sequences. In *Proc. IEEE International Conference on Image Processing (ICIP),* pp.449 – 452, (2005)

16. CHUNG, K. L., AND YAO, T. J. New prediction and affine transformation - based three-step search scheme for motion estimation with applications. *Journal of Information Science and Engineering*, vol.24, pp.1095 – 1109, (2008)

17. JING, X., AND CHAU, L. P. An efficient three-step search algorithm for block motion estimation. *IEEE Transactions on Multimedia*, vol.6, pp.435–438, (2004)

18. MUHIT, A., PICKERING, M. R., FRATER, M. R., AND ARNOLD, J. F. Video coding using elastic motion model and larger blocks. *IEEE Transactions on Circuits and Systems for Video Technology* , vol.20,no.5, pp.661 – 672, (2010)

19. MATTHIAS, N., AND SWOBODA, R. Extending HEVC by an affine motion model. *Picture Coding Symposium (PCS)*, pp.321 – 324, (2013)

20. HUANG, H., WOODS, J. W., ZHAO, Y., AND BAI, H. Affine SKIP and DIRECT modes for efficient video coding. *Visual Communications and Image Processing (VCIP)* , pp.1 – 6, (2012)

21. HUANG, H., WOODS, J. W., ZHAO, Y., AND BAI, H. Control-point representation and differential coding affine-motion compensation. *IEEE Transactions on Circuits and Systems for Video Technology* , vol.23, no.10, pp.1651 – 1660, (2013)

22. CHEN, H., LIANG, F., AND LIN, S. Affine SKIP and MERGE modes for video coding. *IEEE 17th International Workshop on Multimedia Signal Processing (MMSP)*, pp.1 – 5, (2015)

23. YOO, J. H, L., SEOK, H., AND ZHANG, B. Evolutionary particle filtering for sequential dependency learning from video data *IEEE Congress on Evolutionary Computation*, pp.1 – 8, (2012)

24. LI, L., LI, H., LV, H., AND YANG, H. An affine motion compensation framework for high efficiency video coding. *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp.525 – 528, (2015)

25. HEITHAUSEN, C., AND VORWERK, J. H. Motion compensation with higher order motion models for HEVC. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp.1438 – 1442, (2015)

26. LEE, J. H., LIM, K. W., SONG, B. C., AND RA, J. B. A fast multi-resolution block matching algorithm and its LSI architecture for low bit-rate video coding. *IEEE Transactions on Circuits and Systems for Video Technology* , vol.11, no.12, pp.1289 – 1301, (2001)

27. GAHLOT, A., ARYA, S., AND GHOSH, D. Object-based affine motion estimation. In *Proc. IEEE Region 10 Conference*, pp.1343 – 1347, (2003)

28. ALWANI, M., CHAUDHARY, R., MATHUR, M., DUTTA ROY, S., AND CHAUDHURY, S. Restricted affine motion compensation in video coding using particle filtering. In *Proc. IAPR-sponsored Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP)*, pp.479 – 484, (2010)

29. IRANI, M., ROUSSO, B., AND PELEG, S. Computing occluding and transparent motions. *International Journal of Computer Vision*, vol.12, no.1, pp.5 – 16, (1994)

30. ISARD, M., AND BLAKE, A. CONDENSATION - Conditional density propagation for visual tracking. *International Journal of Computer Vision*, vol.28,no.1, pp.5 – 28, (1998)

31. SULLIVAN, J., AND RITTSCHER, J. Guiding random particles by deterministic search. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pp.1 – 18, (2001)

32. DUTTA ROY, S., TRAN, S. D., DAVIS, L. S., AND VIKRAM, B. S. Multi-resolution tracking in space and time. In *Proc. IAPR-sponsored Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP)*, pp.352 – 358, (2008)

33. LAI, Y. K., AND LAI, Y. F. Quality enhancement for scalable view window in touchable display systems. In *Proc. IEEE International Conference on Consumer Electronics*, pp.539 – 540, (2011)

34. RICHARDSON, I. E. G. H.264 and MPEG-4: Video compression. In *Wiley*, (2003)

35. WIEGAND, T., SULLIVAN, G. J., BJONTEGAARD, G., AND LUTHRA, A. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, vol.13,no.7, pp.688 – 703, (2003)

36. BURT, P. J., AND ADELSON, E. H. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, vol.31, no.4, pp.532 – 540, (1983)