HEVC-EPIC: EDGE-PRESERVING INTERPOLATION OF CODED HEVC MOTION WITH APPLICATIONS TO FRAMERATE UPSAMPLING

Dominic Ruefenacht and David Taubman

Interactive Visual Media Processing Lab (IVMP), School of EE&T, UNSW Sydney, Australia

ABSTRACT

We propose a method to obtain a high quality motion field from decoded HEVC motion. We use the block motion vectors to establish a sparse set of correspondences, and then employ an affine, edge-preserving interpolation of correspondences (EPIC) to obtain a dense optical flow. Experimental results on a variety of sequences coded at a range of QP values show that the proposed HEVC-EPIC is over *five times as fast* as the original EPIC flow, which uses a sophisticated correspondence estimator, while only slightly decreasing the flow accuracy. The proposed work opens the door to leveraging HEVC motion into video enhancement and analysis methods. To provide some evidence of what can be achieved, we show that when used as input to a framerate upsampling scheme, the average Y-PSNR of the interpolated frames obtained using HEVC-EPIC motion is slightly lower (0.2dB) than when original EPIC flow is used, with hardly any visible differences.

***Index Terms*—** Edge-Preserving Motion Interpolation, Optical flow, Temporal Frame Interpolation (TFI), HEVC.

1. INTRODUCTION

Motion estimation lies at the heart of modern video compression algorithms, and is one of the fundamental problems of computer vision. However, motion information is used for different purposes. In compression, it is used to exploit the temporal redundancy between frames, which plays an essential role in the reduction of the coding cost of video content. In computer vision, so-called optical flow estimation algorithms aim at estimating the "physical" flow between frames, which can then be used for a variety of video analysis and enhancement techniques. The aim of this paper is to build more meaningful motion from decoded "block" motion, and hence bridge two fields that traditionally are treated separately.

Most of the video content that is consumed is compressed content, for which (block) motion information has been estimated. However, this motion estimation is opportunistic and does not in general attempt to follow actual scene motion, in particular around moving objects. Focussing on the frame upsampling task, Chen *et al.* [1] add spatial regularization to the motion estimation scheme on the encoder side, and then use the decoded motion to obtain temporally upsampled frames of higher quality. In order to be compliant with existing video codecs, Wu *et al.* [2] use decoded block motion directly to interpolate frames, and propose an iterative refinement technique to conceal inevitable interpolation errors in the target frame. Rather than fixing up texture information, Yang and Yang [3] instead propose to improve the decoded motion field by considering causal neighbour blocks as additional motion candidates, and select the one that has the smallest prediction residual; the resulting (block) motion field is then used as input for a traditional TFI method.

Our approach to obtain a high quality motion field from decoded (block) motion is inspired by the recent trend followed by several top-performing optical flow algorithms (e.g., [4, 5]), which first estimate a sparse set of reliable correspondences, and then employ an edge-preserving interpolation strategy [4] to compute a dense optical flow. In order to avoid the time-consuming part of finding correspondences, we propose to directly use the motion of each decoded block motion vector as a correspondence. We then use edge information, estimated on decoded texture using a structured edge detector [6], to guide the edge-preserving interpolation of correspondences (EPIC) algorithm [4] to obtain the final motion.

Not surprisingly, the use of decoded block motion as "seeds" for the sparse-to-dense interpolation significantly reduces the computational complexity of optical flow estimation. Much more surprising, however, is the quality of the motion flows that can be obtained in this way. Although the source HEVC motion field is block-based, containing holes and many opportunistic vectors, a convincing flow field can be recovered with remarkably high accuracy. To test the suitability of HEVC-EPIC motion for video analysis and enhancement methods, we use the estimated flow as input to a state-of-the-art framerate upsampling algorithm [7], and show that it is able to create high quality interpolated frames.

2. OVERVIEW

We start with a high-level overview of the proposed method, which is guided by Fig. 1. Input to the proposed method is decoded block motion from a standard HEVC codec in low-delay mode (IPPP structure), as well as the reconstructed frame texture data. More precisely, for each P-frame f_k, a block motion field $B_{k\to k-1}$ is decoded that contains motion

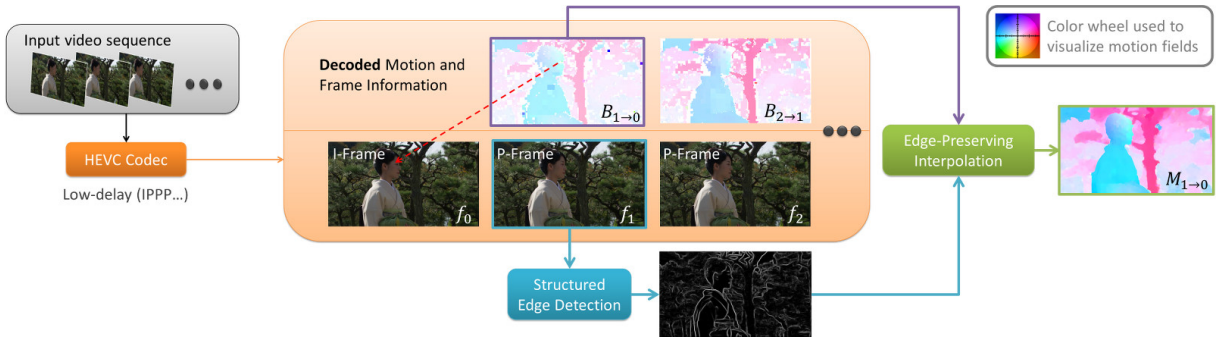


Fig. 1. Overview of the proposed HEVC-EPIC method. An input video sequence is coded using HEVC low-delay mode (IPPP structure). For each P-frame f_k, a block motion field $B_{k\to k-1}$ is decoded, indicating the displacement between f_k and f_{k-1}. We estimate edge information on the decoded frame f_k using [6], which is used to guide the sparse-to-dense affine motion interpolation procedure to obtain a dense motion field $M_{k\to k-1}$ that is suitable for computer vision tasks.

vectors pointing to frame f_{k-1}. $B_{k\to k-1}$ is "opportunistic" in the sense that the encoder does not aim to find the "true" motion between the frames f_k and f_{k-1}, but rather a motion vector that minimizes the number of bits to code the prediction error (see Sect. 3) of the texture information at frame f_k. The decoded $B_{k\to k-1}$ exhibits a number of problems that make it ill-suited for video enhancement and analysis tasks (see Fig. 1). Firstly, it contains artificial discontinuities at block boundaries. In addition, blocks are unable to accurately describe object boundaries, which results in visibly disturbing artefacts if such motion is used for computer vision tasks. Lastly, the encoder can decide to disable temporal prediction for any block, and only use spatial prediction (i.e., $Intra$ prediction), which leaves "holes" in the motion field; the $Intra$ mode is selected often in regions around moving objects.

In this work, we propose a motion post-processing method to alleviate the above-mentioned issues. We get rid of artificial block boundaries and fill in blocks without motion information (i.e., $Intra$ blocks) by smoothly interpolating between the motion vectors – anchored at the block center – of each block. In order to allow for discontinuous motion at object boundaries, we incorporate edge information in the form of an edge probability map [6] into the motion interpolation strategy; this successful strategy is used in a number of top-performing optical flow estimation algorithms [4, 5]. In this paper, we focus on motion estimated for P-frames, and leave the interesting extension to incorporate B-frames for future work.

3. MOTION-COMPENSATED PREDICTION IN VIDEO CODECS

There is no explicit attempt in standardised video codecs to estimate the "physical" motion of the scene. Instead, the motion is chosen in an R-D optimal way. That is, the objective is to get the minimum distortion subject to an upper bound on the overall bit-rate; or, equivalently, to get the minimum bit-rate subject to an upper bound on overall distortion. Both objectives are equivalent to minimizing the unconstrained Lagrangian cost function $J = D + \lambda R$, where D is the overall distortion (typically measured as mean squared error), R is the overall bit-rate, and $\lambda > 0$ determines the constraint for which the solution is optimal.

Finding the *global* minimum of J is infeasible, primarily because the rate term depends on choices made in neighbouring spatial and temporal blocks. Instead, existing video coders find the solution on a per-block basis. That is, for each block K_l, the *block prediction mode* I_l (see end of this section) is found by minimizing the Lagrangian cost function [8]

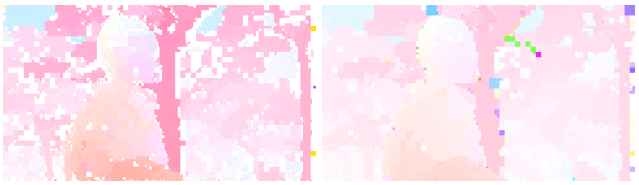
$$J(K_l, I_l) = D_{rec}(K_l, I_l, \mathbf{u}) + \lambda R(K_l, I_l, \mathbf{u}). \qquad (1)$$

A widely accepted strategy to find a solution for (1) is to first find the motion vectors for each block, and to consider the optimization of transform and coding modes as a second step. The motion is found using

$$\mathbf{u}_l = \arg\min_{\mathbf{u}\in U} E(K_l, \mathbf{u}) + \lambda_{mv} R_{mv}(K_l, \mathbf{u}), \qquad (2)$$

where $E(K_l, \mathbf{u})$ is the *block distortion measure*, computed as the sum of squared differences between the target block and the motion-compensated prediction of that block. Furthermore, U is the set of all possible *partitions* of the block K_l that are allowed by the standard. For motion-compensated prediction, HEVC considers a large variety of block partitions, starting from 64×64, going all the way down to 8×4 pixels [9].

We now briefly present the two most commonly used prediction modes, which allows us to get some more insight into the opportunistic nature of the motion estimation performed in video codecs. These modes are intrapicture and interpicture prediction, commonly referred to as $Intra$ and $Inter$, respectively. $Intra$ prediction is performed exploiting only spatial redundancies. That is, K_l is predicted from already decoded blocks of the same frame, and the distortion $D_{rec}(K_l, Intra)$ is the squared prediction residual;



(a) HEVC (b) HEVC **no** $Intra$

Fig. 2. Decoded block motion for (a) HEVC, and (b) HEVC without $Intra$ prediction. One can observe how around moving objects, HEVC resorts to $Intra$ prediction (zero motion, white), whereas the forced $Inter$ prediction results in non-physical motion vectors in such regions.

$R(K_l, Intra)$ is the rate obtained after entropy coding of the texture residual. In $Inter$ prediction for P-frames (i.e., uni-directional prediction), an already decoded frame f_a is used as prediction reference for motion-compensated prediction of the target frame f_b. For each block K_l, the distortion is

$$D_{rec}(K_l, Inter, \mathbf{u}_l) = \sum_{\mathbf{m} \in K_l} (f_a[\mathbf{m} + \mathbf{u}_l] - f_b[\mathbf{m}])^2, \quad (3)$$

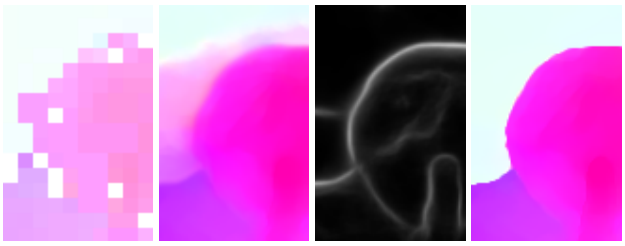
and $R(K_l, Inter, \mathbf{u}_l)$ is the sum of the rates for the motion vectors, transform coefficients, and mode information. As a consequence, for any block K_l where $J(K_l, Intra) < J(K_l, Inter)$ in (1), no motion information will be communicated, leaving "holes" in the block motion field. The $Intra$ mode is particularly useful in regions that are not visible in the reference frame(s); as illustrated in Fig. 2a, the $Intra$ mode is often chosen around moving objects. To further illustrate the opportunistic nature of the motion, Fig. 2b, we show a motion field estimated where the $Intra$ mode has been disabled, and hence only $Inter$ prediction is used. One can see how there is "random" motion selected around moving objects, which is both expensive to code, as well as highly "non-physical".

4. EDGE-PRESERVING AFFINE INTERPOLATION OF BLOCK MOTION VECTORS

The decoded block motion from HEVC provides a set of N motion vectors. In HEVC, the same motion vector is used for the whole block. In this work, we propose to use these N motion vectors as "seeds" to drive an edge-preserving affine interpolation; we refer to the resulting method as HEVC-EPIC. For each motion vector $\mathbf{u}^n_{i\to j}$, $n \in \{1, ..., N\}$, we construct correspondence pairs $(\mathbf{x}^n_i, \mathbf{x}^n_j)$ as follows; here, $\mathbf{x}^n_i$ is the location of nth motion vector $\mathbf{u}^n_{i\to j}$ in frame f_i. Then, its corresponding location in frame f_j is simply

$$\mathbf{x}^n_j = \mathbf{x}^n_i + \mathbf{u}^n_{i\to j}. \quad (4)$$

We now describe how this sparse set of motion vectors can be interpolated to obtain a dense motion field. That is, for any integer pixel location $\mathbf{m}$ of a given frame f_i, we want to find a motion vector that maps the point to a corresponding location



(a) Block Motion (b) Affine Interp. (c) Edges (d) HEVC-EPIC

Fig. 3. Euclidean versus geodesic distance measure for interpolation. (a) shows decoded block motion; (b) shows the affine interpolated motion; (c) are the edges estimated using SED [6] on the decoded texture; (d) shows how motion boundaries are preserved using the proposed HEVC-EPIC, which uses an edge-preserving interpolation strategy.

in frame f_j. We use an affine model to interpolate motion vectors, as it offers a good trade-off between complexity and ability to describe typical motion (e.g., translation, rotation, zoom, shear). That is, each location $\mathbf{m}$ of the motion field $\hat{M}_{i\to j}$ is interpolated using a weighted affine estimator,

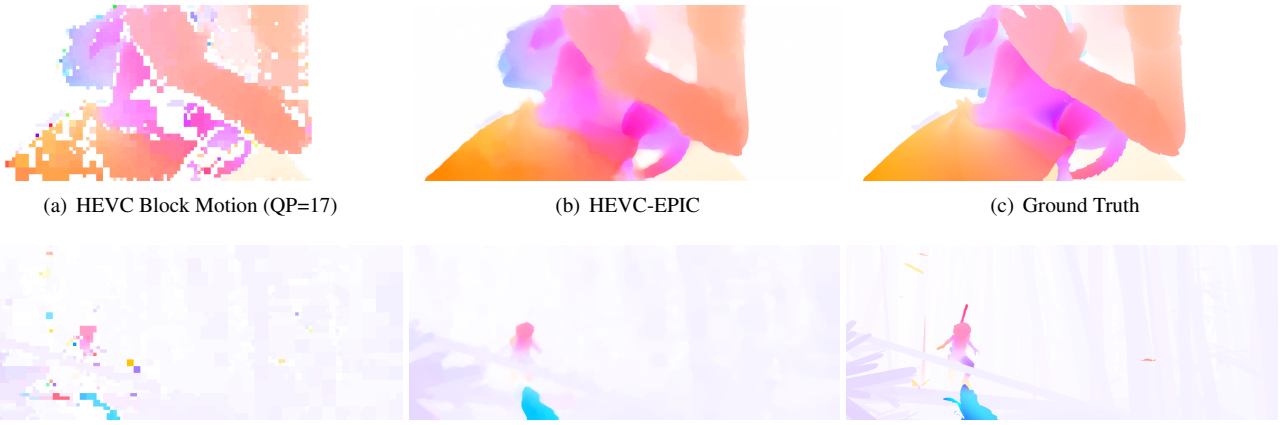
$$\hat{M}_{i\to j}[\mathbf{m}] = A_{\mathbf{m}}\mathbf{m} + t_{\mathbf{m}}, \quad (5)$$

where $A_{\mathbf{m}}$ (2×2 matrix) and $t_{\mathbf{m}}$ (2×1 vector) are parameters of the affine transform at pixel location $\mathbf{m}$. In order to find these parameters, we need at least the *three* points $\mathbf{x}^s_i$ in frame f_i that are *closest* to $\mathbf{m}$, and their corresponding locations $\mathbf{x}^s_j$ in frame f_j. To add robustness to outliers, we use $S > 3$ correspondences (we empirically set $S = 25$), and compute the least-squares solution of the overdetermined system

$$(A_{\mathbf{m}}, t_{\mathbf{m}}) = \arg\min_{(A,t)} \sum_{s=1}^{S} e^{-D(\mathbf{x}^s_i, \mathbf{m})} (A\mathbf{x}^s_i + t - \mathbf{x}^s_j), \quad (6)$$

where $D(\mathbf{a}, \mathbf{b})$ measures the *distance* between the points $\mathbf{a}$ and $\mathbf{b}$. The implication of (6) is that the "closer" – according to the distance measure $D(\cdot, \cdot)$ – the point $\mathbf{x}^s_i$ is to the location $\mathbf{m}$ we seek to interpolate, the more weight is put on fitting the affine model to match the correspondence pair $(\mathbf{x}^s_i, \mathbf{x}^s_j)$.

We now provide more details into the choice of distance measure. Fig. 3a shows a crop of a decoded HEVC motion field, and Fig. 3b shows the corresponding affine interpolation, where each location $\mathbf{m}$ was interpolated according to (6), with Euclidean distance as distance measure $D(\cdot, \cdot)$. One can see how the motion is interpolated across object boundaries, which leads to wrong motion assignments around moving objects. In this work, we use the edge-aware geodesic distance measure proposed by Revaud *et al.* [4]. The idea is to find the cheapest path between two points, where the "cost" is measured by an edge probability map [6], computed on the texture of the frame. Fig. 3c shows an example edge probability map. In Fig. 3d, we show the dense motion field obtained by applying (6) with the edge-aware geodesic distance measure, where the motion boundaries are preserved.



(a) HEVC Block Motion (QP=17) (b) HEVC-EPIC (c) Ground Truth

(d) HEVC Block Motion (QP=29) (e) HEVC-EPIC (f) Ground Truth

Fig. 4. Visualization of HEVC-EPIC motion fields. (a/d) show decoded block motion, (b/e) show the edge-aware, affine interpolated motion obtained by HEVC-EPIC, and (c/f) show the ground truth motion.

Table 1. Average EPE values (lower is better) of the motion fields for different QP values. We compare the proposed HEVC-EPIC (HE) with the original EPIC flow [4], as well as HEVC-EPIC where we disabled the $Intra$ prediction (HE-NoIntra).

Sequence	QP=17			QP=29			QP=40		
	HE-NoIntra	EPIC	HE	HE-NoIntra	EPIC	HE	HE-NoIntra	EPIC	HE
alley_1	**0.21** (+0.00)	0.23 (+0.02)	**0.21**	0.40 (+0.05)	**0.33** (-0.02)	0.35	0.72 (+0.00)	**0.66** (-0.06)	0.72
alley_2	0.23 (+0.02)	**0.21** (+0.00)	**0.21**	0.40 (+0.06)	**0.33** (-0.01)	0.34	0.84 (+0.04)	**0.78** (-0.02)	0.80
bamboo_1	**0.26** (+0.00)	**0.26** (+0.00)	**0.26**	0.32 (+0.01)	**0.31** (+0.00)	**0.31**	0.51 (+0.01)	**0.49** (-0.01)	0.50
bamboo_2	0.32 (+0.03)	0.30 (+0.01)	**0.29**	0.38 (+0.05)	**0.33** (+0.00)	**0.33**	0.47 (+0.02)	**0.43** (-0.02)	0.45
bandage_1	0.76 (+0.06)	**0.60** (-0.10)	0.70	1.05 (+0.02)	**0.95** (-0.08)	1.03	**1.67** (-0.08)	1.71 (-0.04)	1.75
bandage_2	0.42 (+0.14)	**0.25** (-0.03)	0.28	0.65 (+0.21)	**0.37** (-0.07)	0.44	1.15 (+0.27)	**0.80** (-0.08)	0.88
cave_4	4.76 (+0.45)	**3.69** (-0.62)	4.31	5.62 (+0.20)	**4.31** (-1.11)	5.42	8.46 (-0.08)	**7.17** (-1.37)	8.54
market_2	1.31 (+0.21)	**1.00** (-0.10)	1.10	1.58 (+0.16)	**1.13** (-0.29)	1.42	2.14 (+0.06)	**1.43** (-0.65)	2.08
shaman_2	**0.15** (+0.00)	**0.15** (+0.00)	**0.15**	**0.29** (-0.01)	0.30 (+0.00)	0.30	**0.59** (-0.01)	0.61 (+0.01)	0.60
shaman_3	0.25 (+0.03)	0.25 (+0.03)	**0.22**	0.39 (+0.03)	0.44 (+0.08)	**0.36**	**1.25** (-0.05)	1.31 (+0.01)	1.30
Average EPE	0.87 (+0.09)	**0.69** (-0.08)	0.77	1.11 (+0.08)	**0.88** (-0.15)	1.03	1.78 (+0.02)	**1.54** (-0.22)	1.76

5. EXPERIMENTAL RESULTS

The experimental evaluation consists of two parts. First, we assess the accuracy of the estimated motion fields the proposed HEVC-EPIC produces. Motivated by the results, we then investigate the suitability of the motion fields obtained from decoded HEVC block motion for the application of TFI. In both experiments, we used the HEVC reference software HM 16.12 with the low-delay (P-frames) coding settings (IPPP...), and a QP offset of 2 for P-frames.

5.1. Motion Field Quality

In this section, we evaluate the quality of the motion fields we obtain using HEVC-EPIC, and compare it to the recently proposed EPIC-flow scheme [4]; in addition, we also report results for HEVC-EPIC with disabled $Intra$ prediction. We selected 11 consecutive frames from a number of sequences from the Sintel dataset [10]. This dataset contains highly challenging sequences where the 1-hop ground truth motion field is known, which allows us to evaluate the quality of the estimated motion fields. Fig. 4 shows decoded HEVC motion (the input to our method), the motion fields estimated by the proposed HEVC-EPIC, as well as the ground truth motion field. In Fig. 5, we provide further qualitative results, and compare HEVC-EPIC to the high quality results of EPIC flow. Table 1 reports the average *endpoint-error (EPE)* for different QP values. Using $\hat{\mathbf{u}} = (\hat{\boldsymbol{u}}, \hat{\boldsymbol{v}})$ and $\mathbf{u} = (\boldsymbol{u}, \boldsymbol{v})$ to denote the estimated and the ground truth motion vector, respectively, the EPE for a motion vector $\hat{\mathbf{u}}[\mathbf{m}]$ at location $\mathbf{m}$ is computed as

$$EPE[\mathbf{m}] = \sqrt{(\hat{\boldsymbol{u}}[\mathbf{m}] - \boldsymbol{u}[\mathbf{m}])^2 + (\hat{\boldsymbol{v}}[\mathbf{m}] - \boldsymbol{v}[\mathbf{m}])^2}. \quad (7)$$

The first thing to note is that quantization affects all three methods in a similar way. In general, the average EPE is increasing as the quantization step size increases, which can be explained by the fact that the decoded texture information contains more artefacts. This can mislead the edge detector which is used to guide the edge-preserving sparse-to-dense motion vector interpolation in all three methods. Fur-

Table 2. Average timings for EPIC flow [4], as well as the proposed HEVC-EPIC. The difference in timings is the time EPIC spends on finding the sparse set of correspondences.

Resolution	EPIC	HEVC-EPIC	Speedup
1024x416	8.2s	**1.6**s	5.1×
1280x720	16.6s	**3.2**s	5.2×
1920x1080	39.6s	**7.5**s	5.3×

thermore, the block size used in HEVC in general increases, meaning that there are fewer decoded motion vectors.

In Table 2, we report average processing times for different resolutions. One can see how HEVC-EPIC is significantly faster than EPIC flow, as it "skips" the expensive feature matching stage by recycling the decoded motion vectors. On average, HEVC-EPIC is around *five times* as fast as the original EPIC.

5.2. Application: Temporal Frame Interpolation (TFI)

In this section, we turn our attention to a common application that benefits from high quality motion, namely temporal frame interpolation (TFI). We use the so-called "BAM-TFI" scheme reported in [7], because it is specifically designed to use piecewise-smooth motion fields with discontinuities at moving object boundaries. [7] reports excellent performance for this scheme, in comparison to other state-of-the-art TFI methods.

Experimental Setup We selected 21 frames from a number of challenging sequences from the Sintel (1024×416) [10] dataset, as well as commonly used 720p and 1080p natural test sequences containing a variety of motion activities (e.g., translation, rotation, zoom, ...). For each sequence, we dropped every second (odd) frame, and then used motion fields estimated using EPIC flow and the proposed HEVC-EPIC as input to BAM-TFI [7] to interpolate the odd frames, resulting in a total of 10 interpolated frames per sequence.

Quantitative Results We quantitatively evaluate how well BAM-TFI works with motion fields estimated using HEVC-EPIC, and compare it to the original EPIC flow. Table 3 gives per-sequence results for six natural test sequences, as well as the average performance on the 10 Sintel sequences listed in Table 1. On average, HEVC-EPIC performs slightly worse than EPIC flow (0.2 dB), with closest performance to EPIC at the medium (QP=29) bitrate.

Qualitative Results We further provide qualitative results of the TFI performance, which perhaps is more significant than quantitative results. In Fig. 5, we show results for three sequences, decoded at different QP values. The first row of the figure shows the decoded block motion field, overlaid with the edge information that was estimated on the decoded texture using SED [6]. The second row shows crops of the decoded block motion field, the motion field estimated using EPIC flow, as well as the proposed HEVC-EPIC. One can see that the obtained motion fields from HEVC-EPIC are very similar to the ones estimated using a state-of-the-art optical flow estimator. The last row of the figure shows crops of the target frame; for each sequence, we show from left to right the uncompressed reference frame, followed by the interpolated frames, where we used motion estimated from EPIC flow and using motion from the proposed HEVC-EPIC, respectively, as input motion to BAM-TFI [7]. As can be seen, BAM-TFI with HEVC-EPIC motion produces very similar results to BAM-TFI with EPIC flow as input, which serves as evidence of the suitability of the motion produced by HEVC-EPIC for video enhancement tasks.

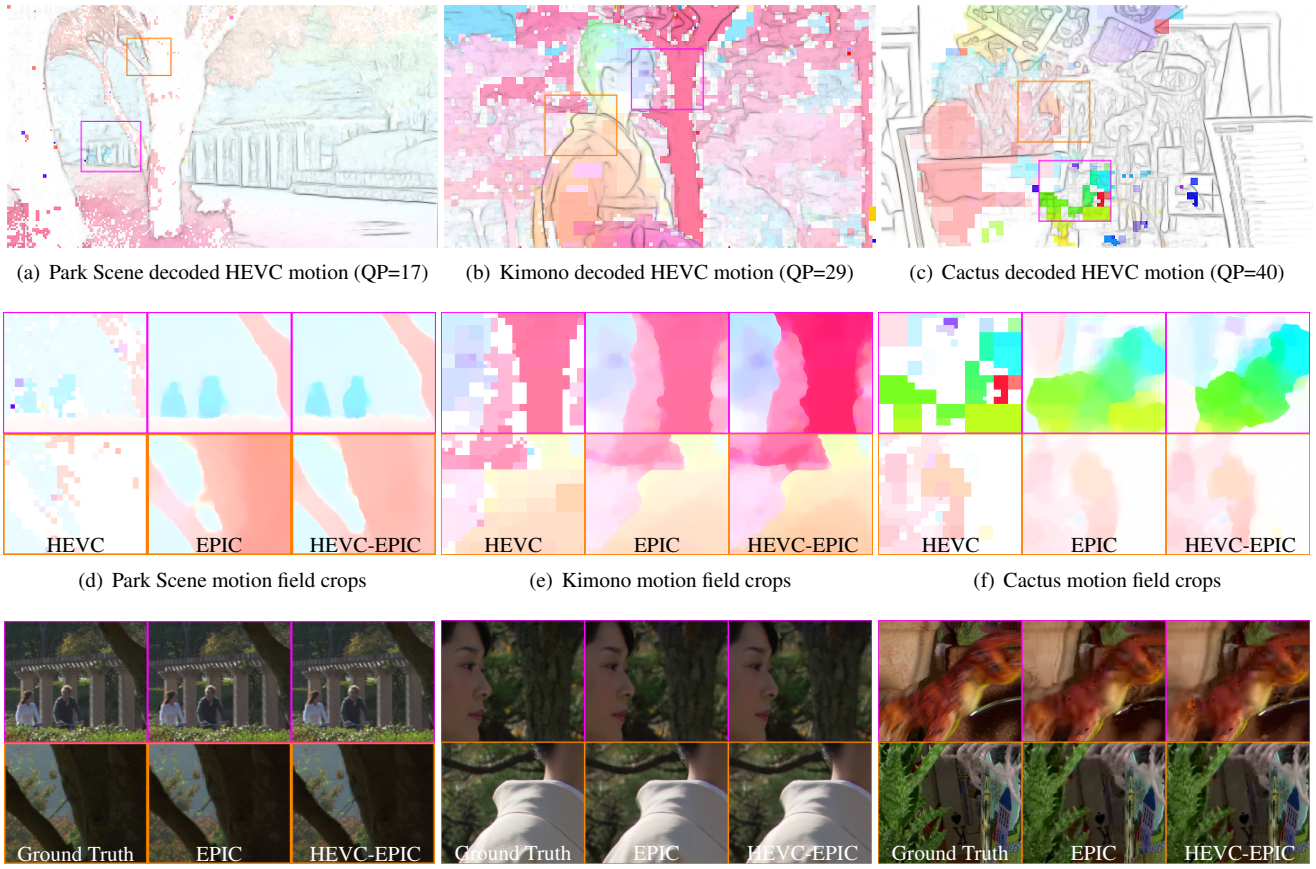
Table 3. Y-PSNR comparison of TFI performance for different QP values. We compare the proposed HEVC-EPIC (HE) with the original EPIC optical flow estimator [4].

Sequence	QP=17		QP=29		QP=40	
	EPIC	HE	EPIC	HE	EPIC	HE
Mobcal	**33.03**	33.01	**32.51**	32.40	**29.93**	29.04
Stockholm	32.97	**33.44**	**32.85**	32.72	**29.67**	29.58
Cactus	**31.20**	30.61	**30.69**	30.41	**28.43**	28.19
ParkScene	**35.94**	35.35	**33.80**	33.72	**29.27**	29.22
Kimono1	**31.57**	30.98	31.15	**31.30**	29.34	**29.51**
BQTerrace	31.85	**32.07**	32.37	**32.74**	**30.53**	30.44
Average Natural	**32.76**	32.58	**32.23**	32.22	**29.53**	29.33
Average Sintel†	**30.13**	29.91	**29.41**	29.21	**27.18**	26.90

† Average of all sequences presented in Table 1 (100 frames per QP value).

6. CONCLUSIONS AND FUTURE WORK

In this paper, we propose HEVC-EPIC, a method to obtain high quality motion from decoded HEVC motion. We use the decoded "block" motion vectors as "seeds" for a locally affine, edge-preserving sparse-to-dense interpolation procedure. Extensive tests at different QP values on a number of challenging sequences show that the resulting method is able to produce motion fields that are close – in terms of average endpoint error – to what a state-of-the-art optical flow estimator produces. By "recycling" the motion vectors estimated at the encoder side, we can significantly reduce the motion estimation time at the decoder. The obtained motion information can be beneficial for a variety of video enhancement and analysis tasks. In this work, we focus on the application of framerate upsampling. Experiments on a large number of challenging synthetic and natural sequences show that the interpolation performance is very close to the one using motion estimated using a state-of-the-art optical flow estimator. In this work, we focused on motion estimated using the low-delay profile of HEVC, which uses P-frames for all but the key frames. The next step will be to apply the same framework to a hierarchical B-frame structure, which is commonly used in video compression.



(a) Park Scene decoded HEVC motion (QP=17) (b) Kimono decoded HEVC motion (QP=29) (c) Cactus decoded HEVC motion (QP=40)

(d) Park Scene motion field crops (e) Kimono motion field crops (f) Cactus motion field crops

(g) Park Scene TFI results (h) Kimono TFI results (i) Cactus TFI results

Fig. 5. Qualitative results of TFI performance. (a)-(c) show the decoded block motion for three different scenes, where we overlaid the estimated edge map; (d)-(f) each show crops of the decoded block motion (left), EPIC flow (middle), and proposed HEVC-EPIC (right); (g)-(i) each show the ground truth target frame, as well as the interpolated frame obtained using EPIC flow (middle), and HEVC-EPIC (right) as input motion for BAM-TFI.

7. REFERENCES

[1] Y.-K. Chen, A. Vetro, H. Sun, and S.-Y. Kung, "Frame-rate up-conversion using transmitted true motion vectors," *IEEE Int. Workshop on Mult. Sig. Proc.*, 1998.

[2] Y. Wu, M. N. S. Swamy, and M. Ahmad, "Error concealment for motion-compensated interpolation," *IET Image Proc.*, vol. 4, no. 3, pp. 195–210, 2010.

[3] S.-H. Yang and C.-C. Yang, "Fast frame rate up-conversion based on multiple frames," *IEEE Int. Conf. on Mult. and Expo*, 2011.

[4] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow," *Proc. IEEE Conf. Comp. Vis. and Patt. Rec.*, 2015.

[5] C. Bailer, B. Taetz, and D. Stricker, "Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation," *Int. Conf. on Comp. Vis.*, 2015.

[6] P. Dollár and C. L. Zitnick, "Fast edge detection using structured forests," *IEEE Trans. Patt. Anal. and Mach. Intell.*, vol. 37, no. 8, pp. 1558–1570, 2015.

[7] D. Rüfenacht and D. Taubman, "Temporally Consistent High Frame-Rate Upsampling with Motion Sparsification," *IEEE Int. Workshop on Mult. Sig. Proc.*, 2016.

[8] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G. J. Sullivan, "Rate-constrained Coder Control and Comparison of Video Coding Standards," *IEEE Trans. Circ. Syst. for Video Tech.*, vol. 13, no. 7, pp. 688–703, 2003.

[9] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding," *IEEE Trans. Circ. Syst. for Video Tech.*, vol. 22, no. 12, pp. 1649–1668, 2012.

[10] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," *European Conf. on Comp. Vis.*, 2012.