Higher-Order Motion Models for Temporal Frame Interpolation with Applications to Video Coding

Dominic Rüfenacht, Reji Mathew, and David Taubman
Interactive Visual Media Processing Lab (IVMP)
School of Electrical Engineering and Telecommunications, UNSW, Sydney, Australia
{*d.ruefenacht, reji.mathew, d.taubman*}*@unsw.edu.au*

***Abstract*—We have recently proposed a motion-centric temporal frame interpolation (TFI) method, called BAM-TFI, which is able to produce high quality interpolated frames under a constant velocity assumption. However, for objects that do not follow constant velocity motion, the predictions, although credible, will differ from the "true" target frames, leading to high prediction residuals. In this paper, we show how higher-order motion models can be incorporated into the BAM-TFI scheme to interpolate frames that better predict the target frames. This opens up the door to a seamless integration of TFI with a video coding scheme. Comparisons on a variety of both synthetic and natural video sequences highlight the benefits of a second-order motion model. We further integrate the proposed TFI scheme into HEVC; preliminary comparisons with HEVC show promising results.**

I. Introduction

Motion compensation lies at the heart of *both* temporal frame interpolation (TFI) schemes *and* video compression algorithms. In TFI, it is used to increase the frame-rate by inserting visually pleasing intermediate frames in between existing frames. In a video compression framework, frames are coded in groups of pictures (GOP), and motion-compensated temporal prediction is used to reduce the prediction residual of frames within the GOP; as such, the aim of a video coder is to find the optimal trade-off between spending bits on texture residual, and the cost of coding motion and other *side-information*. The quality (and complexity) of motion vector estimation has steadily increased from one video coding standard to the next. However, even the highly sophisticated motion estimation employed in HEVC [1], the latest standardized codec, is still "opportunistic" in its nature, which means that it does not produce motion that can be used for TFI.

For this reason, most existing TFI schemes (re-)estimate motion at the decoder. This motion estimation process dominates the computational complexity of TFI, which is the reason most methods "resort" to hierarchical block-motion estimation schemes with added smoothness constraints [2]. Such "block" motion is unable to accurately describe motion around moving objects. Furthermore, visually disturbing block-artefacts require time-consuming texture optimizations [3]. Lu *et al.* [4] use four consecutive frames to detect occluded regions, and show improved results compared to [2]. While the use of more than two reference frames allows to observe acceleration, this was not used in [4].

With the exception of just a few TFI methods (e.g., [5]), existing TFI schemes interpolate frames under a constant velocity assumption. One reason for this is that in order to incorporate acceleration, an accurate model of the underlying motion flow is required; such motion models are absent in many of the prior schemes. The problem with the constant velocity assumption is that it leads to abrupt changes in the motion trajectory of accelerated objects at the transition between reference frames.

In [6], we proposed a "motion-centric" TFI scheme (BAM-TFI), which only uses high-quality motion information (and derived motion discontinuities) in the process of mapping the motion to the target frame we seek to interpolate. The BAM-TFI scheme is able to create highly credible interpolated frames under a *constant velocity* assumption. In this paper, we show how by employing more than two reference frames, the scheme can be extended to predict frames following higher-order motion. For example, for a second-order motion model, the scheme needs at least three reference frames to estimate two motion fields, from which velocity and acceleration parameters can be estimated. These parameters can subsequently be used to interpolate a frame at any position between the given reference frames assuming *constant acceleration*.

In a coding framework, the interpolated frames can be used as predictions for the intermediate input frames. By incorporating higher-order motion models into the interpolation scheme, significantly lower prediction residuals can be achieved for regions that undergo non-constant motion.

In order to give more insights into the benefits of the higher-order motion model, we incorporate a second-order motion model extension of the proposed BAM-TFI scheme into HEVC, where preliminary experiments show promising results. The seamless integration of TFI into a video compression system allows for the time-consuming motion estimation to be performed at the encoder; this high-quality motion can then be used at the decoder to perform temporal upsampling and thereby display the video at a frame-rate higher than the one of the input sequence available at the encoder.

II. Higher-Order Motion Models

Almost all existing TFI schemes interpolate frames under a constant velocity assumption. Unlike most of these schemes, our recently proposed *motion-centric* BAM-TFI scheme [6] employs high-quality motion fields, which makes extensions to higher-order motion models practical. Fig. 1 shows the

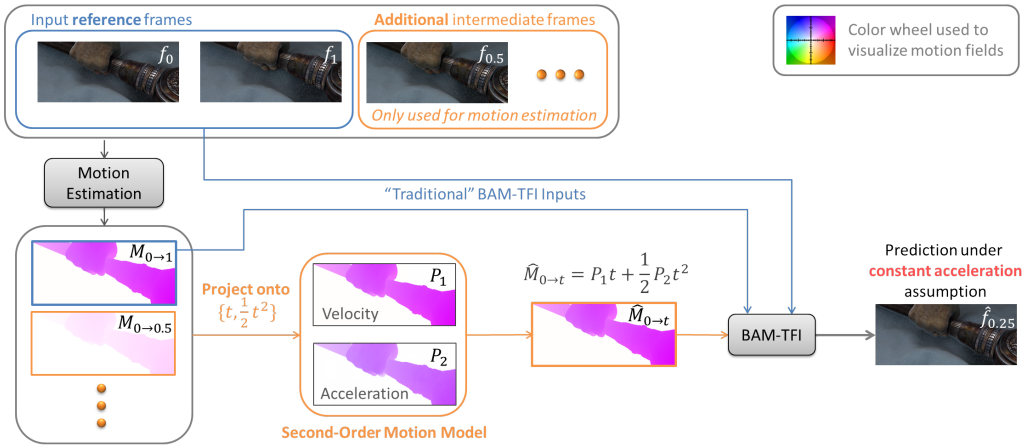


Fig. 1. Extensions to the BAM-TFI framework in order to add a second order motion model that is able to account for accelerating motion (modifications in orange). In addition to the two reference frames, we estimate the motion between f_0 and at least one frame in between f_0 and f_1. These motion fields are then projected onto the subspaces t and $0.5t^2$, which subsequently allow us to compute $\hat{M}_{0\to t}$ under a constant acceleration assumption.

necessary modifications for a second-order motion model. For the purpose of discussion, we use $t \in [0,1]$ to denote the *normalized* time difference between the two reference frames f_0 and f_1. Let $\boldsymbol{u}_{0\to t}(\mathbf{m})$ denote a motion vector which links a location $\mathbf{m}$ in f_0 with its corresponding location in f_t; here, $\boldsymbol{u} = (u, v)$ holds information for the *horizontal* and *vertical* component of the motion field. We further use $\boldsymbol{u}^{(n)}_{0\to t}(\mathbf{m})$ to denote a motion vector following an nth-order motion model.

The general form of a motion vector following an nth-order motion model can be written as

$$u^{(n)}_{0\to t}(\mathbf{m}) = \sum_{k=1}^{n} w_k p_k t^k, \tag{1}$$

where the p_k's are the motion basis vectors, and the w_k's are weights to scale the different basis vectors. For example, in the case of second-order motion model, $w_1 = 1$ and $w_2 = 0.5$, and the two unknowns p_1 and p_2 are generally identified as *velocity* and *acceleration*.

We focus on the *horizontal* component u of $\boldsymbol{u}^{(n)}_{0\to t}(\mathbf{m})$, noting that the vertical component is handled in the same way. In its general form, there are n *unknowns* p_k, and hence we need at least n motion vectors $\boldsymbol{u}_{0\to t_j}(\mathbf{m})$; this means that we require at least $n+1$ frames as input. In matrix form, we can write the system of equations $A\mathbf{p} = \mathbf{u}$, with

$$A = \begin{bmatrix} w_1 t_1 & w_2 t_1^2 & \dots & w_n t_1^n \\ w_1 t_2 & w_2 t_2^2 & \dots & w_n t_2^n \\ \vdots & \vdots & \ddots & \vdots \\ w_1 t_q & w_2 t_q^2 & \dots & w_n t_q^n \end{bmatrix}, \mathbf{u} = \begin{bmatrix} u_{0\to t_1}(\mathbf{m}) \\ u_{0\to t_2}(\mathbf{m}) \\ \vdots \\ u_{0\to t_q}(\mathbf{m}) \end{bmatrix}, \tag{2}$$

where $q \geq n$. From linear algebra, we know that this system of equations has a *unique* least-squares solution $\mathbf{p}^+$ of smallest norm. One way of finding $\mathbf{p}^+$ is in terms of the *pseudo-inverse* of A, denoted as A^+ ($\mathbf{p}^+ = A^+\mathbf{u}$), which is obtained from the singular value decomposition (SVD) of A.

The motion parameters $\mathbf{p}^+$ allow us to create motion that follows an nth-order motion model for *any* frame f_t in between the two reference frames. For example, in the case of a second order motion model, motion following a *constant acceleration* assumption can be obtained as follows:

$$u^{(2)}_{0\to t}(\mathbf{m}) = p_1 t + 0.5 p_2 t^2. \tag{3}$$

As mentioned earlier, the vertical component $v^{(n)}_{0\to t}(\mathbf{m})$ is obtained following the same development. Combining the horizontal and vertical components yields $\boldsymbol{u}^{(n)}_{0\to t}(\mathbf{m})$.

III. BAM-TFI with Higher-Order Motion

In this section, we show how the BAM-TFI framework we presented in [6] can be extended to account for higher-order motion; we use BAM$^{(n)}$ to refer to the BAM-TFI scheme employing nth-order motion. The proposed scheme employs piecewise-smooth motion fields, with sharp discontinuities at moving object boundaries; suitable motion can be obtained for example using the motion detail preserving optical flow algorithm described in [7]. As mentioned in Sect. II, the input to the motion modelling stage are reference frames f_0 and f_1, plus additional f_{t_j}; the output is an nth-order motion field, denoted as $M^{(n)}_{0\to t}$, which describes the motion between f_0 and *any* f_t, $t \in [0,1]$. As shown in Fig. 1, $M^{(n)}_{0\to t}$ and the two reference frames f_0 and f_1 are then input to BAM-TFI.

In the following discussion, we describe an affine warping procedure that is used to derive from $M^{(n)}_{0\to t}$ the motion fields anchored at the target frame f_t. During this motion mapping procedure, we use the observation that motion discontinuities travel with the foreground object in order to resolve double mappings (occluded regions), and fill in sensible motion in disoccluded regions (i.e., holes).

A. Affine Mesh Warping: Motion Inversion and Inference

We employ the triangular mesh sparsification algorithm presented in [6] to partition the input motion field $M^{(n)}_{0\to t}$ into a triangular mesh; the output is a set of K *vertices* V^k_t, $k \in \{1, ..., K\}$. Each vertex V^k_0 records its coordinates

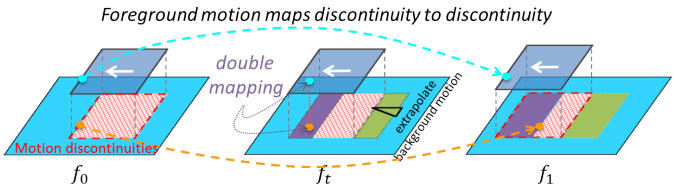


Fig. 2. Illustration of the key ideas used in the proposed TFI method to assign sensible motion in regions around moving objects; see text for details.

in frame f_0, and holds motion vectors relating it with the corresponding location in the succeeding reference frame f_1. Triangles, formed by connecting vertices, are larger in regions of smooth motion within objects, and smaller around moving object boundaries; triangles that straddle motion discontinuities are of size 1×1.

For the description of the mapping process, we focus on how an individual vertex V_0^k is mapped to the target frame; the mapped vertices can then be connected together to form triangles, which completely cover the target frame; motion $\hat{M}_{t\to 0}^{(n)}$ and $\hat{M}_{t\to 1}^{(n)}$, which relates f_t with its preceding and succeeding reference frame f_0 and f_1, respectively, can then be obtained through affine interpolation of the motion vectors $\boldsymbol{u}_{t\to 0}^{(n)}(V_t^k)$ and $\boldsymbol{u}_{t\to 1}^{(n)}(V_t^k)$.

We use $\boldsymbol{u}_{0\to t}^{(n)}(V_0^k)$, derived from an nth-order motion model as described in Sect. II, to map V_0^k to the target frame; that is, $V_t^k = V_0^k + \boldsymbol{u}_{0\to t}^{(n)}(V_0^k)$. Negating its motion yields the motion linking the target frame with f_0,

$$\boldsymbol{u}_{t\to 0}^{(n)}(V_t^k) = -\boldsymbol{u}_{0\to t}^{(n)}(V_0^k). \tag{4}$$

From $\boldsymbol{u}_{0\to 1}^{(n)}(V_0^k)$ and $\boldsymbol{u}_{0\to t}^{(n)}(V_0^k)$, we *infer* the forward pointing motion vector $\boldsymbol{u}_{t\to 1}^{(n)}(V_t^k)$:

$$\boldsymbol{u}_{t\to 1}^{(n)}(V_t^k) = \boldsymbol{u}_{0\to 1}^{(n)}(V_0^k) - \boldsymbol{u}_{0\to t}^{(n)}(V_0^k), \tag{5}$$

which links the vertices between the target frame and the succeeding reference frame f_1. Importantly, because $\boldsymbol{u}_{t\to 1}^{(n)}(V_t^k)$ is composed of motion vectors $\boldsymbol{u}_{0\to 1}(V_0^k)$ and $\boldsymbol{u}_{t\to 0}^{(n)}(V_t^k)$, $\boldsymbol{u}_{t\to 0}^{(n)}(V_t^k)$ and $\boldsymbol{u}_{t\to 1}^{(n)}(V_t^k)$ point to the same geometrical location in f_0 and f_1.

B. Handling of Regions around Moving Objects

Using Fig. 2, where a foreground rectangle moves on top of static[1] background, we now give a high-level overview of how problematic regions around moving objects are handled; the approach adopted here is the same as that in [6], where it is expounded more comprehensively.

1) Handling of Double Mappings: As we are mapping triangles from f_0 to f_t, multiple triangles may overlap in the target frame; this happens on the leading side of moving objects, where a foreground object moves on top of a background object (see purple region in Fig. 2). This ambiguity is resolved by observing that the foreground motion maps the nearby motion discontinuity in f_0 closer to a motion discontinuity in the other reference frame f_1 (cyan arrow) than the motion belonging to the background object (orange arrow).

2) Background Motion Extrapolation in Disoccluded Regions: Triangles that straddle a region of large divergence in $M_{0\to 1}^{(n)}$ are likely disoccluding when mapped to the target frame (see green region in Fig. 2). In such regions, the affine interpolated motion linearly interpolates between foreground and background motion vectors, which creates "non-physical" motion. Again, using motion discontinuity information, we identify which vertex/vertices of the disoccluding triangle belong to the background; their motion is then extrapolated to the foreground vertex/vertices of the disoccluding triangle.

C. Weighted, Occlusion-Aware Frame Interpolation

In the BAM-TFI framework, we readily observe regions that become disoccluded between f_0 and f_t (e.g., *forward* disocclusions); these are the regions where the piecewise-smooth motion fields exhibit large local divergence, where background motion is extrapolated as explained above. With a slightly more involved method, we compute *reverse* disocclusions, i.e. regions in f_t that are not visible in f_1 (see [6] for details); these correspond to a subset of the regions in which double mappings are observed in the second reference frame. We store this valuable information in *visibility masks*

$$I_t^j[\mathbf{m}] = \begin{cases} 1 & \mathbf{m} \text{ visible in } f_j \\ 0 & \text{otherwise} \end{cases}, \tag{6}$$

where $j = 0$ and $j = 1$ refer to f_0 and f_1, respectively.

We use a weighted bidirectional prediction of the motion compensated reference frames, denoted as $f_{0\to t}$ and $f_{1\to t}$, whenever the location $\mathbf{m}$ is visible in either *both* or *neither* of the reference frames, and switch to uni-directional prediction whenever a location is only visible in one reference frame.

$$\hat{f}_t[\mathbf{m}] = \begin{cases} (1-t)f_{0\to t}[\mathbf{m}] + t f_{1\to t}[\mathbf{m}] & I_t^0[\mathbf{m}] = I_t^1[\mathbf{m}] \\ f_{j\to t}[\mathbf{m}] & \text{otherwise} \end{cases}, \tag{7}$$

where j refers to the reference frame f_j where location $\mathbf{m}$ is visible.

IV. Experimental Results and Discussion

In this section, we evaluate various aspects of the proposed work. For the experiments, we limit the model order to $n = 2$, and leave a thorough investigation of the benefits of motion models of order $n > 2$ for future work. First, we compare BAM$^{(1)}$ with other TFI schemes. Next, we compare BAM$^{(1)}$ with BAM$^{(2)}$ to assess the improvement in prediction performance achieved by employing a second-order motion model. Lastly, we integrate the TFI scheme into a video coder, and compare the performance with HEVC.

A. Evaluation of Temporal Frame Interpolation Performance

In a typical TFI setting, constant velocity between reference frames is assumed. In this section, we show results on a variety of common natural test sequences, and compare the

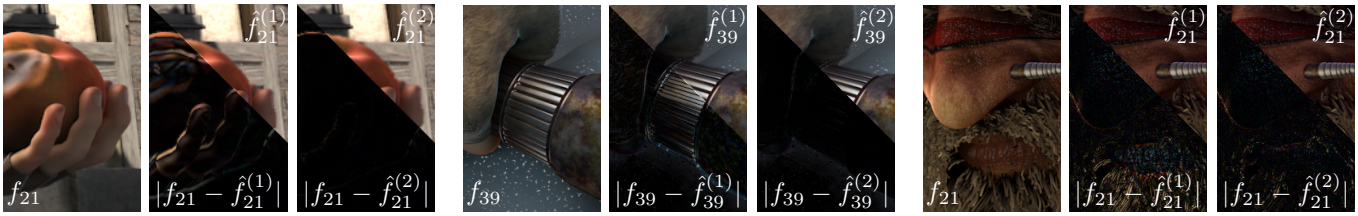
[1]The background is static for ease of explanation; the same reasoning generalizes to scenes with moving background.

TABLE I
QUANTITATIVE COMPARISON OF BAM$^{(1)}$ WITH [3], [8], AND [4]. WE FURTHER SHOW THE PREDICTION PERFORMANCE OF BAM$^{(2)}$. **BOLD** INDICATES BEST PER-ROW PERFORMANCE (BAM$^{(2)}$ EXCLUDED).

Sequence	Frames	Jeong [3]	Veselov [8]	Lu [4]	BAM$^{(1)}$	BAM$^{(2)}$
Cactus	007-025	33.15 (-1.07)	31.27 (-2.94)	34.12 (-0.09)	**34.22**	34.31 (+0.09)
Kimono	001-019	33.93 (+0.41)	33.40 (-0.13)	**34.51** (+0.99)	33.53	34.07 (+0.54)
Kimono	175-193	39.97 (-1.42)	40.21 (-1.18)	39.51 (-1.87)	**41.39**	41.94 (+0.56)
Rushhour	049-067	35.18 (+0.41)	34.93 (+0.16)	**35.30** (+0.53)	34.77	35.50 (+0.73)
Shields	101-119	35.90 (-0.30)	35.10 (-1.09)	35.89 (-0.30)	**36.20**	36.70 (+0.50)
Shields	385-403	33.87 (-3.88)	35.58 (-2.17)	33.52 (-4.23)	**37.75**	37.88 (+0.13)
Park	139-157	38.29 (-1.65)	38.84 (-1.10)	38.74 (-1.20)	**39.94**	39.75 (-0.19)
Parkrun	145-163	30.63 (-1.42)	30.97 (-1.07)	30.50 (-1.54)	**32.04**	32.20 (+0.16)
Station2	021-039	41.10 (-2.06)	41.41 (-1.75)	40.54 (-2.63)	**43.16**	43.26 (+0.10)
Mobcal	361-379	29.13 (-9.38)	34.75 (-3.75)	29.53 (-8.97)	**38.51**	38.38 (-0.13)
Average	-	35.11 (-2.03)	35.65 (-1.50)	35.22 (-1.93)	**37.15**	37.40 (+0.25)

TABLE II
AVERAGE Y-PSNR FOR BAM$^{(1)}$ AND BAM$^{(2)}$ ON FULL SINTEL SEQUENCES (23 INTERPOLATED FRAMES PER SEQUENCE).

Sequence	BAM$^{(1)}$	BAM$^{(2)}$
Alley 1	31.64 (-1.62)	**33.26**
Alley 2	32.86 (-2.02)	**34.88**
Ambush 7	30.58 (-5.07)	**35.66**
Bandage 1	31.97 (-1.26)	**33.22**
Bandage 2	33.82 (-2.07)	**35.89**
Bamboo 1	29.23 (-0.06)	**29.29**
Bamboo 2	28.54 (-0.81)	**29.35**
Market 2	28.62 (-0.96)	**29.58**
Shaman 2	37.58 (-1.04)	**38.62**
Shaman 3	37.05 (-0.01)	**37.06**
Average	32.19 (-1.49)	**33.68**



(a) Alley 1 (b) BAM$^{(1)}$ (c) BAM$^{(2)}$ (d) Ambush 7 (e) BAM$^{(1)}$ (f) BAM$^{(2)}$ (g) Shaman 2 (h) BAM$^{(1)}$ (i) BAM$^{(2)}$

Fig. 3. Crops of three sequences from the Sintel dataset [9]; we show the ground truth frame, followed by the predicted frame using the first order motion and the second order motion model, denoted as BAM$^{(1)}$ and BAM$^{(2)}$. The upper right part shows the predicted frame, and the lower left part the (absolute) difference between the prediction and the ground truth frame.

results with state-of-the-art TFI schemes [3], [8], [4]. We selected 21 consecutive frames from various sequences available on https://media.xiph.org/video/derf/, and dropped every odd frame. The task is then to interpolate the dropped odd frames from the existing even frames. For the BAM-TFI scheme, we used the optical flow estimator from Xu *et al.* [7] to estimate motion between the two (even-indexed) reference frames.

Table I shows quantitative results in terms of Y-PSNR. In most of the sequences, BAM$^{(1)}$ outperforms the tested state-of-the-art TFI schemes.

B. Prediction Performance of Second-Order Motion Model

Next, we employ a second motion field between each even frame and succeeding odd frame, and evaluate the improved *prediction* performance of BAM$^{(2)}$ compared to BAM$^{(1)}$.[2]

The last column of Table I shows quantitative results of the BAM$^{(2)}$ scheme, where in addition to the motion between any two *even* reference frames, we estimated motion between each even frame and the succeeding odd frame. One can see that in 8 out of the 10 tested sequences, BAM$^{(2)}$ performs better than BAM$^{(1)}$ by varying amounts. Notable improvements are achieved on sequences that contain fairly large regions that are accelerated, such as the "Kimono" sequence, as well as "Rush Hour" and "Shields". Clearly, the second-order model should be at least as good as the first order one. However, as mentioned earlier, the motion fields have to be temporally consistent in order for a higher order motion model to be meaningful, which cannot be guaranteed if the two fields are independently estimated.

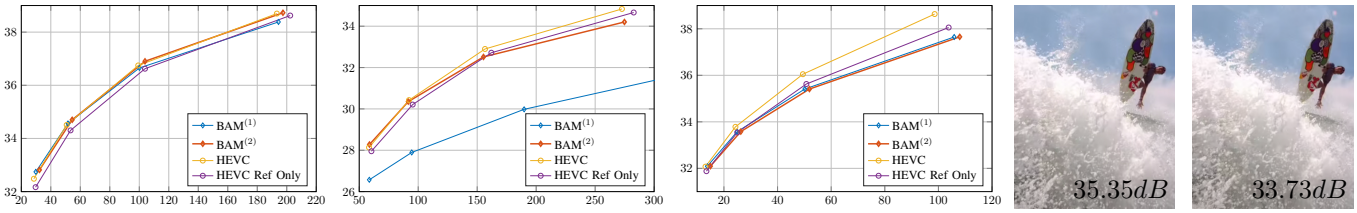
To reduce the impact of suboptimal motion estimation, we further conducted a comprehensive experiment on challenging sequences from the Sintel dataset [9], where forward motion between adjacent frames (i.e., 1-hop motion) is available as ground truth. We construct a 2-hop motion field by concatenating two 1-hop flows; for locations that move out of the frame in the first motion field, we assume constant velocity to create the 2-hop motion vectors. As in the previous experiment, we drop every odd frame, and use the proposed BAM-TFI scheme to predict the odd frames. Table II shows the average (of 23 interpolated frames) Y-PSNR on a number of sequences.

In this experiment, the BAM$^{(2)}$ *consistently* outperforms BAM$^{(1)}$. Significant improvements are observed on sequences that contain large acceleration, such as the "Ambush 7" sequence, as well as "Bandage 1" and "Alley 2". Fig. 3 shows crops of frames from three different sequences, which visualize how BAM$^{(2)}$ predicts the actual location of objects much better than BAM$^{(1)}$.

C. Integrating BAM-TFI into a Video Coder

The experiments in the previous section have shown that BAM$^{(2)}$ is able to better predict the "true" position of moving objects that are under non-constant motion. In this section, we integrate the BAM-TFI scheme into HEVC, and provide

[2]Note that while the odd frame is not available in a traditional TFI framework, it is available in a coding framework, which this work builds towards.



(a) Station 2 (1080p) (b) Shields (720p) (c) Surfer Jump (720p) (d) HEVC QP33 (e) $\hat{f}_{84}^{(1)}$ **no resi**

Fig. 4. (a)-(c) show RD-plots for three test sequences; the x-axis show the average number of kbits per frame, and the y-axis the Y-PSNR. (d) shows a crop of a *middle* frame f_{84} (GOP is $f_{80} - f_{88}$) from the "Surfer Jump" sequence, decoded using HEVC at QP value of 33; (e) shows the same frame produced by BAM$^{(1)}$ *without* residual coding. While there are differences in the splashes, the difference between (d) and (e) is hardly visible.

preliminary coding results on three sequences.

We estimate motion between the first frame (i.e., the base frame) and eight successive frames of the group of pictures (GOP). From these eight motion fields, we estimate a second-order motion model, as described in Sect. II. We then predict the seven target frames of the GOP, and code the texture residuals using HEVC; the two reference frames are coded in Intra-mode.[3] The motion fields are coded using a modified JPEG-2000 codec, which uses a breakpoint-adaptive wavelet transform to conserve motion boundaries, as used in [10]. In the current implementation, all motion fields are *separately* coded; further improvements can be expected from a *joint* motion field coding. Fig. 4 shows the average per-frame rate-distortion performance for three sequences, with QP-values of $\{28, 33, 38, 42\}$; the QP-offset for the target frame residuals is set to 4 for all methods. We show results obtained for BAM$^{(1)}$ (blue) and BAM$^{(2)}$ (red), as well as for HEVC using hierarchical B-frames (yellow), and a modified version of HEVC (purple), where only the two reference frames are used as prediction references, as is the case in our framework.

On the "Station 2" sequence, which is dominated by zoom-out motion, BAM$^{(2)}$ performs on par with HEVC. The "Shields" sequence contains an inconsistent zoom, for which the (credible) constant-velocity prediction of BAM$^{(1)}$ creates large residuals. BAM$^{(2)}$ is able to much better predict the intermediate frames, which is evidenced by the large gap between the blue and the red curve in Fig. 4b. Lastly, we tested the performance on the "Surfer Jump" sequence, which contains complex motion such as splashing waves, which is not captured in the estimated motion fields. The offset between BAM$^{(2)}$ and BAM$^{(1)}$ is caused by the additional motion field that has to be coded, which in this sequence does not reduce the texture residual. While HEVC outperforms the proposed scheme on this sequence, it is worth noting that the interpolated frames produced by our scheme are *highly credible*, as can be seen in Fig. 4e. The whole sequence without residual coding is available on our website.[4]

To end the discussion, we highlight that with the proposed framework, the frame-rate can easily be increased at the decoder without having to re-estimate motion, which is not possible in current video codecs.

[3]Reference frames could also be coded as P- or B-frames.

[4]http://ivmp.unsw.edu.au/~dominicr/pcs_2016.html

V. Conclusions and Future Work

In this paper, we show how our recently proposed motion-centric BAM-TFI scheme can be extended to account for higher-order motion. This is achieved by estimating at least one additional motion field between f_0 and a frame in between the two reference frames f_0 and f_1. In the case of second-order motion, these motion fields are projected onto velocity and acceleration subspaces. Adding acceleration to the framework opens up interesting links to compression, where the aim is not only to create a credible interpolation, but one that is close to the target frame to be predicted. To this end, we integrate the BAM-TFI scheme into HEVC; preliminary coding results are encouraging and motivate further research for seamlessly integrating TFI with video compression systems.

In future work, we intend to add temporal regularization to the motion field estimation to improve the temporal consistency; furthermore, we will investigate the benefits of motion models with higher order than 2.

References

[1] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," *IEEE Trans. Circ. Syst. for Video Tech.*, vol. 22, no. 12, pp. 1649–1668, 2012.

[2] S. Dikbas and Y. Altunbasak, "Novel true-motion estimation algorithm and its application to motion-compensated temporal frame interpolation," *IEEE Trans. Image Proc.*, vol. 22, no. 8, pp. 2931–2945, 2013.

[3] S.-G. Jeong, C. Lee, and C.-S. Kim, "Motion-compensated frame interpolation based on multihypothesis motion estimation and texture optimization," *IEEE Trans. Image Proc.*, pp. 4497–4509, 2013.

[4] Q. Lu, N. Xu, and X. Fang, "Motion-Compensated Frame Interpolation With Multiframe Based Occlusion Handling," *Journal of Display Technology*, vol. 11, no. 4, 2015.

[5] P. Csillag and L. Boroczky, "Estimation of accelerated motion for motion-compensated frame interpolation," *Visual Comm. and Image Proc.*, pp. 604–614, 1996.

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

[7] L. Xu, J. Jia, and Y. Matsushita, "Motion detail preserving optical flow estimation," *IEEE Trans. Patt. Anal. and Mach. Intell.*, pp. 1744–1757, 2012.

[8] A. Veselov and M. Gilmutdinov, "Iterative Hierarchical True Motion Estimation for Temporal Frame Interpolation," *IEEE Int. Workshop on Multimedia Sig. Proc.*, 2014.

[9] 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.*, pp. 611–625, Oct. 2012.

[10] D. Rüfenacht, R. Mathew, and D. Taubman, "A Novel Motion Field Anchoring Paradigm for Highly Scalable Wavelet-based Video Coding," *IEEE Trans. Image Proc.*, 2015.