ORIGINAL PAPER

Non-linear contour-based multidirectional intra coding

THORSTEN LAUDE, JAN TUMBRÄGEL, MARCO MUNDERLOH AND JÖRN OSTERMANN

Intra coding is an essential part of all video coding algorithms and applications. Additionally, intra coding algorithms are pre-destined for an efficient still image coding. To overcome limitations in existing intra coding algorithms (such as linear directional extrapolation, only one direction per block, small reference area), we propose non-linear Contour-based Multidirectional Intra Coding. This coding mode is based on four different non-linear contour models, on the connection of intersecting contours and on a boundary recall-based contour model selection algorithm. The different contour models address robustness against outliers for the detected contours and evasive curvature changes. Additionally, the information for the prediction is derived from already reconstructed pixels in neighboring blocks. The achieved coding efficiency is superior to those of related works from the literature. Compared with the closest related work, BD rate gains of 2.16% are achieved on average.

I. INTRODUCTION

During the past decades, a tremendous improvement of video coding algorithms has been observed. In January 2013, the Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T VCEG and ISO/IEC MPEG finished the technical work for the latest video coding standard, High-Efficiency Video Coding (HEVC) [1, 2], which is also referred to as H.265 by ITU-T and as MPEG-H Part 2 by ISO/IEC. For the same visual quality, depending on the selected configuration and application, HEVC enables a bit rate reduction of 40–60% compared with the predecessor standard Advanced Video Coding (AVC) [3, 4]. After the finalization of HEVC, the research for new coding tools is stimulated by a constant strive for even higher coding efficiency [5].

The prediction tools, which led to the prosperous application of these video coding standards, can be roughly distinguished into inter and intra coding tools. In contrast to intra coding, which solely employs spatial redundancy in the current picture, inter coding additionally uses motion compensated prediction to reduce the temporal redundancy between subsequent pictures to further increase the coding efficiency. For typical video signals, the resulting bit rates to achieve the same visual quality are, therefore, for intra coding considerably higher (10–100 times) compared with inter coding.

Leibniz University Hannover, Institut für Informationsverarbeitung, Appelstr. 9A, 30167 Hannover, Germany

Corresponding author:
Thorsten Laude
Email: laude@tnt.uni-hannover.de

Nevertheless, intra coding is an essential part of all video coding applications and algorithms: it is used to start transmissions, for random access (RA) into ongoing transmissions, for error concealment, in streaming applications for bit rate adaptivity in case of channels with varying capacity, and for the coding of newly appearing content in the currently coded picture. Additionally, intra coding algorithms are predestined for efficient still image coding. This can be considered as a special case of a video sequence with just one picture. For example, the High-Efficiency Image File Format as a container for HEVC encoded pictures is used as the default image file format on mobile devices running iOS which were released since 2017.

The intra prediction as specified by the HEVC standard is based on the spatial prediction of sample values[1] based on neighboring, previously coded sample values (reference values) followed by a transform coding of the prediction error [6]. For the prediction, there are 33 directional modes, one mode for the planar prediction, and one mode for the prediction with the mean value of the reference values (DC mode). The reference values are located in a 1 pel wide column and a 1 pel high row of sample values within previously coded blocks directly to the left and top of the currently coded block (reference area). The directional modes allow for the linear extrapolation of the reference values in the direction of 33 different angles. Considering that there are many horizontal and vertical structures in typical video sequences, a higher angle resolution is selected for these

---

[1]Within this paper, we differentiate between the terms *pixel* and *sample value*. One pixel consists of three sample values for the Y, Cb, and Cr components, respectively.

directions [7]. The directional modes aim at the prediction of blocks with the linear structure or linear contours [6]. Not all blocks to be coded comprise a linear structure. For blocks without a linear structure, the prediction with the DC mode and with the planar mode (weighted combination of four sample values) were developed [6].

An analysis of the intra coding in the HEVC standard reveals multiple limitations:

(i) The reference area is relatively small with a width or height of 1 pel, respectively. Further information could be gained by processing additional neighboring, previously coded sample values.
(ii) The directional modes only enable the prediction of linear structures.
(iii) In the case of directional prediction, only one direction per block can be selected by encoders.

In consequence, blocks with non-linear contours or with multiple contours in different directions are difficult to predict. This conclusion is in line with the finding of Lottermann and Steinbach in their work on signal-dependent bit rate modeling [8]. Their finding (high bit rates for blocks with many edges) matches our premise that blocks with many edges are difficult to predict.

To overcome these limitations, we propose Contour-based Multidirectional Intra Coding (CoMIC). The underlying principle of the CoMIC prediction is the detection and modeling of contours in already reconstructed blocks followed by the extrapolation of the contours into the currently coded block. Namely, the main *contributions* of this paper are:

(i) four non-linear contour models,
(ii) an algorithm for the connection of intersecting contours,
(iii) an along-edge sample value continuation algorithm,
(iv) and a mode decision algorithm for the selection of the best contour model based on the Boundary Recall.

The remainder of this paper is organized as follows: In Section II, we discuss related work from the literature and elaborate on the distinguishing features of our work. In Section III, we start by introducing the CoMIC framework, present our novel non-linear contour modeling and extrapolation algorithms, our sample value prediction algorithm, our mode decision algorithm, and our codec integration. Subsequently, in Section IV, we evaluate our algorithms and demonstrate their efficiency by comparing them with the closest works from the literature.

II. RELATED WORK

The findings of numerous works in the academic literature indicate that valuable information for the prediction of the currently coded block can be extracted from contours in neighboring blocks.

Yuan and Sun demonstrate that a contour map of the currently coded picture is suitable for a fast encoder decision for the selection of a good intra mode [9]. In contrast to applying edge information for the sake of an accelerated encoding process, we utilize extracted edge information to increase the coding efficiency with a new coding mode.

Asheri *et al.* [10] as well as Au and Chan [11] use contour information for error concealment algorithms. For this purpose, contour information from the current picture (Asheri *et al.*,) or from the current and the previous picture (Au and Chan) is utilized to conceal transmission errors. Differently, from our approach, their algorithms do not influence the actual coding process but are used as post-processing in case of errors. Furthermore, we assume a causal encoding system in which the reference sample values can only be in previously coded blocks.

Liu *et al.*, increase the coding efficiency of JPEG (in [12]) and of JPEG 2000 as well as AVC (in [13, 14]) with contour-based inpainting and texture synthesis algorithms. Their algorithms include the signaling of contour shapes similar to JBIG, the linear extrapolation of contours from already coded parts of the picture, the signaling of patches for texture synthesis, the signaling of representative sample values for the inpainting, and the solving of partial differential equations for the inpainting. In contrast to the HEVC intra coding algorithm, this enables the prediction of multiple contours with different directions per block. Liu *et al.*, achieve an increased coding efficiency. However, only linear contours are extrapolated. Moreover, for the sample value prediction it is required to either signal a considerable amount of side information (patches for texture synthesis) or to solve computationally complex systems of partial differential equations for the inpainting (the decoding of a picture with 512×512 pel spatial resolution takes "multiple minutes" [14]). Considering that the coding efficiency of their algorithm is only demonstrated for solely intra-coded pictures (*all-intra*) [13, 14], it cannot be assumed that the visually pleasing results of the texture synthesis are a good prediction for subsequently coded pictures.

To compensate for the described limitations of the algorithm of Liu *et al.*, we proposed CoMIC in our conference paper [15] to increase the coding efficiency of JPEG and HEVC. In general, JPEG coding is extended by a spatial prediction and the DCT is computed on the prediction error instead of the signal to be coded. By combining a prediction with the coding of the prediction error, the problem of Liu *et al.*, that the synthesized picture is unsuitable as a prediction reference is circumvented. Like the approach of Liu *et al.*, this first version of CoMIC (v1) is based on the separate processing of contours and sample value prediction. The contours are detected in the reference area (which is three times the size of the currently coded block) and extrapolated linearly into the currently coded block without the signaling of any side information. The sample value prediction is based on the continuation of neighboring sample values along the extrapolated

contours. In contrast to the sample value prediction as specified by the HEVC standard, the individual contours are treated separately in our approach. This way, the sample values can be continued in multiple directions per block.

CoMIC v1 is suitable for the prediction of linear contours and structures. However, in real images and video sequences, there are plenty of non-linear contours which cannot be predicted by CoMIC v1. As a consequence of this limitation, CoMIC v1 allows a reliable prediction only in those areas of the currently coded block which are in proximity to the reference area. The contour extrapolation with this algorithm into parts of the block which are further away from the reference area results in a too large uncertainty with respect to the contour shape. In turn, this results in a high prediction error and a high bit rate to compensate for this prediction error. Hence, a distance-dependent diminishing of the continued reference sample value to the mean value of all reference samples is applied [15].

In our experimental section, our novel non-linear CoMIC algorithm (CoMIC v2) will be compared with the algorithm of Liu *et al.*, and to the first version of CoMIC from our conference paper.

The main difference between our contribution and related inpainting approaches is that we assume a causal encoding system where only already coded blocks (i.e. above of and to the left of the current block) are available for the contour extrapolation. In contrast to that, inpainting approaches use the signal on all sides of the missing block. For example, Liu *et al.*, signal the sample values on the contours in the currently coded block as side information and construct Dirichlet boundary conditions this way in their respective method. This knowledge is highly beneficial for the prediction. Taking into account that this information is not available for the decoder without signaling, our method solves a harder problem and also requires less signaling overhead. A proof-of-concept for PDE-based video compression is demonstrated by Andris *et al.* [16].

In the work [17], Rares *et al.*, proposed a method for image restoration (i.e. artifact removal). There are multiple commonalities between the related work and our paper: The starting point in both works is similar. Some content is missing, in our case because a block is to be predicted, in the case of Rares *et al.*, because the content is occluded by an artifact. Then, the aim in both works is to "invent" the missing content. In our case by prediction, in the case of Rares *et al.*, by restoration. Additionally, both works are based on contours. Their work is based on modeling the connection of "strongly related" contours on opposing sides of the artifact. The knowledge of where contours end at the other side of the unknown area is highly beneficial for reconstructing the contour in the unknown area. Unfortunately, this knowledge is not available in a causal coding system where blocks are encoded one after another. In consequence, the model proposed by Rares *et al.*, is not transferable to the prediction problem in a causal coding system.

III. CONTOUR-BASED MULTIDIRECTIONAL INTRA CODING

In this section, we discuss the novel contributions of this paper. For the sake of a cohesive presentation, we start by reviewing the CoMIC framework briefly. Subsequently, we elaborate on our novel non-linear contour modeling and extrapolation algorithms, our sample value prediction algorithm, our mode decision algorithm, and describe the integration of the above mentioned algorithms in the CoMIC codec.

With CoMIC, we designed an intra coding mode for the prediction in multiple directions with minor signaling overhead by combining contour detection, contour parameterization, and contour extrapolation algorithms with conventional video coding algorithms like sample value continuation. As a basic working principle, contour information is extracted from reconstructed samples and employed to extrapolate these contours into the currently coded block. In this pipeline, we limit our area of operation to the currently coded block and the adjacent blocks as illustrated in Fig. 1. The blocks in the reference are of the same size as the currently coded block.

A) CoMIC framework

Both versions of the CoMIC codec (v1 published as [15], v2 proposed in this paper) share the same basic working principle which will be discussed with reference to Fig. 2.

The encoding of a block starts by detecting contours in the reference area (see Fig 2a). Our contour detection is based on the well-known Canny edge detection algorithm [18]. A minor additional coding gain (0.3%) can be achieved by using more advanced edge detection algorithms like [19], but this also increases the computational complexity by a factor of 90 for the entire encoder. Thus, we chose to
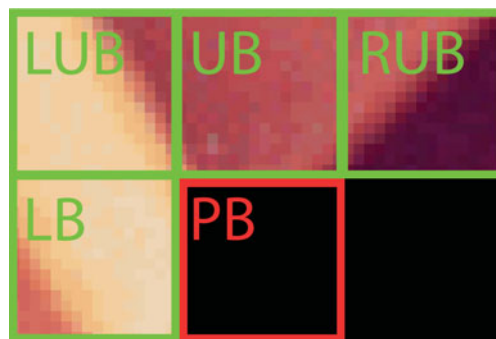

Fig. 1. Area of operation. For the prediction of the current block (predicted block, PB), the following four blocks highlighted by green boundaries are used as reference: left block (LB), left-upper block (LUB), upper block (UB), and right-upper block (RUB). In this example the block size is 16 × 16.
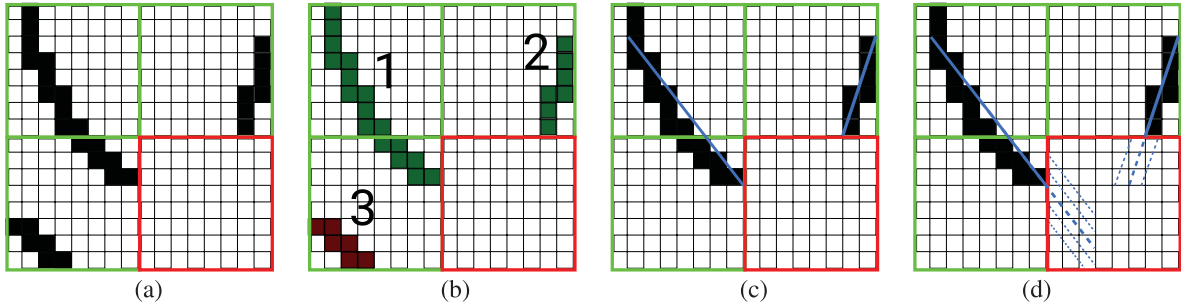
Fig. 2. CoMIC v1 pipeline: the contours are detected using the Canny algorithm, labeled, modeled with linear functions, and extrapolated into the currently coded block. The major limitation of CoMIC v1 is the linear contour model. In consequence, the contours can only be extrapolated partly into the currently coded block because the accuracy of the extrapolated contours drops considerably with increasing distance from the reconstructed area. (a) Detected contours, (b) Labeled contours, (c) Modeled contours, (d) Extrapolated contours.

content ourselves with the original Canny edge detector for this paper.

Once the edges have been detected with the Canny algorithm, a binary image is available that describes all edge pixels in the reconstructed area. Our goal is the extrapolation of distinctive contours. Therefore, we label individual contours in the binary edge picture with the algorithm of Suzuki and Abe [20] (see Fig 2b).

The part of the framework as it is described so far is maintained for this paper. In the following, we briefly present the remainder of the framework as it was used for our conference paper. It is noteworthy that the following parts of the framework will not be used by the contributions of this paper but are described for the sake of cohesiveness.

In [15], each contour is modeled by a straight line (see Fig 2c), i.e. by a polynomial parameterization with two parameters. For contours which hit the left border of the currently coded block, the horizontal coordinate x is used as the independent variable while the vertical coordinate y is used as the independent variable for contours that hit the upper border of the currently coded block. This differentiation facilitates the parameterization of horizontal and vertical structures. Thus, the polynomial $p(a)$, with a being the independent variable (i.e. either x or y) and with coefficients $\beta_i \in \mathbb{R}$ is parameterized as noted in equation (1):

$$p(a) = \beta_0 + \beta_1 a. \tag{1}$$

The coefficients β_i are determined by minimizing the mean squared error between the detected contour and the linear contour model using a least squares regression.

With the parameterization of the contour available, the contour can be extrapolated in the currently coded block. As disclosed in Section II, our linear contour model only allows an accurate extrapolation in parts of the currently coded block (see Fig 2d). The reference sample values are continued parallel to the linearly extrapolated contours to predict the sample values of the current block. In the course of the continuation, the sample values are diminished towards the mean value of the reference sample values due to the discussed limitation of the linear contour model. As a fallback solution for blocks without extrapolatable contours, a DC mode similar to the DC mode as specified by the HEVC standard is available.

B) Non-linear contour modeling

As it was revealed in the previous sections, the major limitation of CoMIC v1 is the lack of capability to model non-linear contours. To overcome this limitation, we propose four different non-linear contour models in this paper. A Boundary Recall-based contour model selection algorithm is applied to select the best model for each block.

Naturally, the straightforward extension of our linear contour model towards a non-linear model is a polynomial model with three or more parameters. Thus, as first non-linear contour model, we propose a quadratic function. Extending equation (1), the model is formulated as in equation (2):

$$p(a) = \beta_0 + \beta_1 a + \beta_2 a^2, \tag{2}$$

This model will be referred to as *Contour Model 1*. Similarly to the linear model, the independent variable is chosen depending on whether the contour enters the predicted block horizontally (x is the independent variable) or vertically (y is the independent variable). In the following, we describe the polynomial calculation only for x as an independent variable. For n given edge pixels (x_i, y_i) with $i = 1, \ldots, n$ we write equation (3):

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n \times 1}, \quad \boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^{n \times 1}. \tag{3}$$

The approximation of the contour is written as in equation (4) with ϵ as approximation error:

$$\boldsymbol{y} = \begin{bmatrix} 1 \\ \vdots & \boldsymbol{x} & \boldsymbol{x}^2 \\ 1 \end{bmatrix} \boldsymbol{\beta} + \boldsymbol{\epsilon} \quad \text{with } \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix}, \boldsymbol{\epsilon} \in \mathbb{R}^{n \times 1}$$

$$= \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}. \tag{4}$$

This linear regression problem (with β_i as regression parameters) can be solved with a least squares approach, i.e.

by minimizing $f(\boldsymbol{\beta})$ in equation (5):

$$f(\boldsymbol{\beta}) = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}). \tag{5}$$

The estimated regression parameters, denoted as $\hat{\boldsymbol{\beta}}$, are calculated as in equation (6):

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}. \tag{6}$$

This model enables a favorable approximation for some contours. However, we observed that this model is not robust enough for all contours by reason of the small number of contour pixels (on which the regression is performed) and that to some extend imprecise contour location and volatile contour slope. Considering these reasons, we project the contour information into a different space prior to the modeling for the remaining three contour models. To be precise, we model the curvature of the contour by modeling the slope of the contour location linearly. This model is more robust than the quadratic model since it has fewer parameters. Additionally, the slope is smoothed to reduce the volatility of the curvature.

Let m_i denote the slope of the contour at location i. Furthermore, let

$$\boldsymbol{m} = \begin{bmatrix} m_1 \\ \vdots \\ m_n \end{bmatrix} \in \mathbb{R}^{n \times 1},$$

with

$$m_i = \frac{y_{i+1} - y_i}{x_{i+1} - x_i}.$$

Note that the denominator could be zero if no precautions were taken. This is due to the fact that the contour location is only determined with full-pel precision. The difference of the independent variable for two neighboring contour pixels is calculated in the denominator. Therefore, the problem arises in cases when two neighboring contour pixels share the same value for the independent variable (e.g. for narrow bends). To circumvent the problem, a single value of the dependent variable is interpolated from the two values of the neighboring pixels. Thereby, only one slope value (instead of two) needs to be calculated for this value of the independent variable. No difference of the independent variable needs to be computed if two pixels share the same value for this variable as these two pixels are combined to one data point. Then, the second non-linear contour model which we refer to as *Contour Model 2* is formulated as in equation (7):

$$\boldsymbol{m} = \begin{bmatrix} 1 \\ \vdots \quad \boldsymbol{x} \\ 1 \end{bmatrix} \boldsymbol{\gamma} + \boldsymbol{\epsilon} \quad \text{with } \boldsymbol{\gamma} = \begin{pmatrix} \gamma_0 \\ \gamma_1 \end{pmatrix}, \; \boldsymbol{\epsilon} \in \mathbb{R}^{n \times 1}$$

$$= \boldsymbol{M}\boldsymbol{\gamma} + \boldsymbol{\epsilon}, \tag{7}$$

where $\boldsymbol{\gamma}$ describes the regression coefficients which can be approximated by applying equation (6) analogously.

It is known that it can be beneficial to address outliers when formulating approximation models. Therefore, adopting the approach of Holland and Welsch [21] to iteratively reweighten the least squares residues, we propose the third contour model (*Contour Model 3*). The weights ω_i for the residues r_i are calculated with an exponential function which is parameterized by the largest residue (r_{max}) as noted in equation (8):

$$\omega_i = e^{-(2r_i/r_{\max})^2}. \tag{8}$$

Given the weights from one iteration, the regression parameters for the next iteration can be calculated by reformulating equation (6) to:

$$\hat{\boldsymbol{\gamma}} = (\boldsymbol{M}^T\boldsymbol{W}\boldsymbol{M})^{-1}\boldsymbol{M}^T\boldsymbol{W}\boldsymbol{m}, \tag{9}$$

with

$$\boldsymbol{W} = \begin{pmatrix} \omega_1 & 0 & \cdots & 0 \\ 0 & \omega_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \omega_n \end{pmatrix}, \tag{10}$$

where $\hat{\boldsymbol{\gamma}}$ is the estimation of the regression parameters $\boldsymbol{\gamma}$. The actual number of reweighting iterations is not of crucial importance for the regression result as long as the number is sufficiently high. Hence, it is chosen as 15 since the computational complexity is negligible as evaluated in Section IV.

None of the three proposed non-linear contour models considers that the curvature of the contour may change unexpectedly – i.e. evasive by equation (7) – in the reference area. As means against such evasive contour changes in the reference area, we propose a fourth contour model (*Contour Model 4*). The rational for this model is that in cases of such contours, it is undesirable to consider parts of the contour which are far away from the currently coded block. Thus, the residues for the regression are weighted depending on their distance to the currently coded block. To be precise, the weights decrease linearly with increasing distance. In contrast to Contour Model 3, the regression is only performed once for Contour Model 4 since the distance does not depend on the weights.

It is worth mentioning that it became apparent in our research that splines – although being often used for signal interpolation – are not capable of achieving a good contour extrapolation. Moreover, our research revealed that higher order polynomial models (i.e. cubic and higher) are not robust enough.

C) Contour extrapolation

With the contour modeling being accomplished by the four proposed models from Section III.B, the extrapolation of the contours which are modeled will now be described.

For Contour Model 1, the contour is extrapolated by directly calculating new function values with equation (2) for the independent variable in the predicted block. For the other three models which model the slope of the contour,
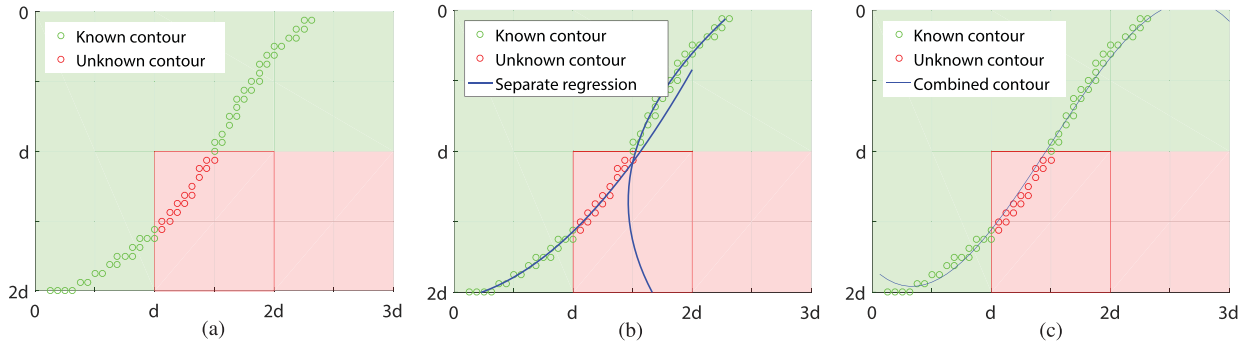
Fig. 3. Combination of intersecting contour parts. (a) Original contour shape, (b) Separate extrapolation of the two contour parts results in inaccurate prediction, (c) Combination of the two contour parts and transfer of the extrapolation problem into an interpolation problem results in accurate prediction.
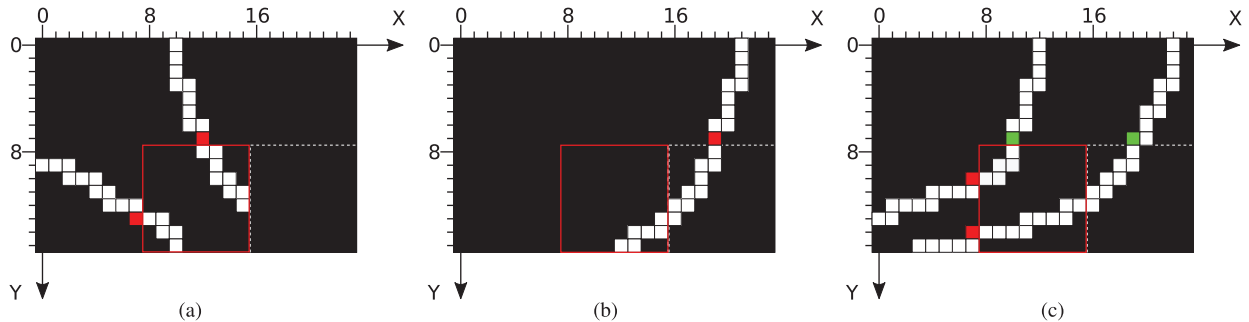

Fig. 4. Different cases for the sample value prediction along the contours. (a) Contours hit the predicted block directly, (b) Contour hits the predicted block indirectly, (c) Combination of contours.

the extrapolation is performed analogously by respectively derived equations.

This extrapolation algorithm works reliably in most cases. However, it may fail for contours which start in the reference area, proceed through the predicted block, and reenter the reference area. An example of such a contour is illustrated in Fig. 3a. For the contour modeling, it appears as if there are two independent contours in the reference area. The independent extrapolation of these two contours is shown in Fig. 3b. Although the extrapolation of one contour is accurate with respect to the original contour, the extrapolation of the second contour would cause a considerable prediction error. As a solution to this issue, it is detected that these two contours form one contour and they are processed together. For this purpose, the contour points of the two to-be-combined contours are summarized to one contour at first. Then, the contour pixels of this combined contour are modeled using the same independent variable. For this combined contour, considering that a section in the middle of the contour is to be predicted, the extrapolation problem turns into an interpolation problem which is approached with a cubic polynomial regression as illustrated in Fig. 3c.

D) Sample value prediction

With the contours extrapolated following the description in Section III.C, the sample values can be predicted by utilizing the extrapolated contours. The sample value prediction is based on the continuation of boundary sample values.

Three cases can be distinguished. They are discussed following the illustration in Fig. 4.

In the first case as illustrated in Fig. 4a, the contours directly hit the predicted block either from the top or from the left. In this case, the sample value of the contour pixel directly adjacent to the predicted block (highlighted in red) is used as a reference and extrapolated along the contour. Slightly different is the second case (Fig 4b) in which the contour hits the predicted block indirectly via the neighboring block on the right side. Here, the sample value of the adjacent pixel on the right side of the predicted block is unavailable as a reference since this pixel is not yet predicted. Hence, the sample value of the nearest available reference pixel (which is highlighted in red) from the top-right block is used as reference. As for the first case, this reference sample is continued along the contour. The commonality of these two cases is that the reference sample value is continued unaltered. This is different in the third case which is used for combined contours as shown in Fig. 4c: For such contours, there are two reference sample values, one at each end of the extrapolated contour. Since these two reference sample values may be different, we chose to linearly blend the predicted sample values between them.

So far, only the sample values on the contour are predicted. To predict the remaining sample values, the above-described process is extended horizontally or vertically, depending on the orientation of the contour. The reference sample values whose pixel location has an offset to the hit point between contour and block boundary in horizontal or vertical direction, respectively, are continued parallel to
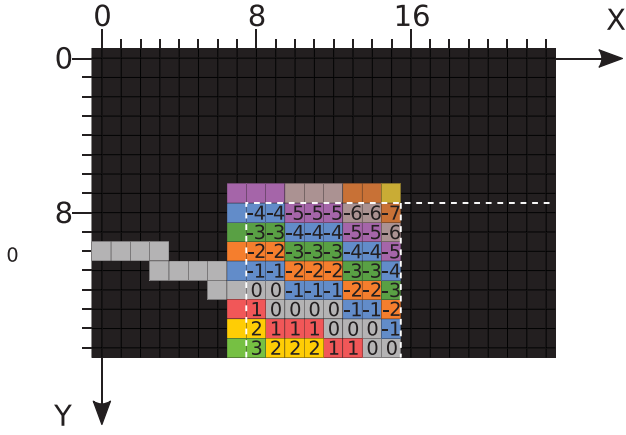
Fig. 5. Illustration of the sample value prediction. The numbers indicate the offset of the location of the reference pixel with respect to the reference pixel on the contour.

the extrapolated contour. An example is illustrated in Fig. 5. Here, the numbers indicate the offset of the location of the reference pixels.

In the presence of multiple contours, the predicted sample value of a non-contour pixel can be obtained from multiple references. One of our premises is that contours are located at the borders of objects. Therefore, sample values are continued up to the next contour. Individual contours are processed in a clockwise direction.

Our decision to base the sample value prediction on the continuation of neighboring reference sample values is backed up by multiple reasons: (1) The continuation of the reference sample is well understood from the intra coding algorithms in recent video coding standards [6, 22]. (2) In contrast to inpainting algorithms, the complexity is moderate (cf. Section IV). (3) In contrast to texture synthesis algorithms, the reconstructed signal (i.e. the sum of the predicted signal and the transmitted quantized prediction error) is suitable as a reference for future predictions.

E) Contour model selection

In total, there are four non-linear contour models. One of them needs to be selected per block by the encoder. The premise of our work is that it is beneficial to process structural information (i.e. contours) and textural information (i.e. sample values) separately. Hence, the decision for the best contour model cannot be based on the quality of the predicted sample values (e.g. by measuring the MSE). Instead, the decision is based on the accuracy of the extrapolated contours. As the original contours are known at the encoder, the extrapolated contours can be compared with the original contours.

The Boundary Recall [23] is chosen as a metric to measure the accuracy of the extrapolated contours. This metric is commonly used to measure the accuracy of superpixel boundaries [24, 25]. Since these two tasks (measuring the accuracy of extrapolated contours and superpixel boundaries, respectively) are very similar, we believe that the BR is a suitable metric for our case. Letting n_{orig}, n_{ext}, and n_{mutual} denote the number of pixels of the original contour, the number of pixels of the extrapolated contour, and the number of pixels which are mutual for the original and the extrapolated contour, respectively. Then, the BR is defined as in equation (11):

$$\text{BR} = \begin{cases} \dfrac{n_{\text{mutual}}^2}{n_{\text{orig}} \cdot n_{\text{ext}}} & \text{for } n_{\text{orig}} > 0, n_{\text{ext}} > 0 \\ 0 & \text{otherwise.} \end{cases} \quad (11)$$

The same contour model is selected for all contours in the predicted block. It is worth noting that different models might be better for different contours within the predicted block. Thus, the selection is not optimal. However, the selection of the contour model needs to be signaled to the decoder because the BR cannot be calculated at the decoder due to the absence of the knowledge of the original contour location. Hence, the better trade-off between the selection of a good contour model and signaling overhead is to use only one contour model per block.
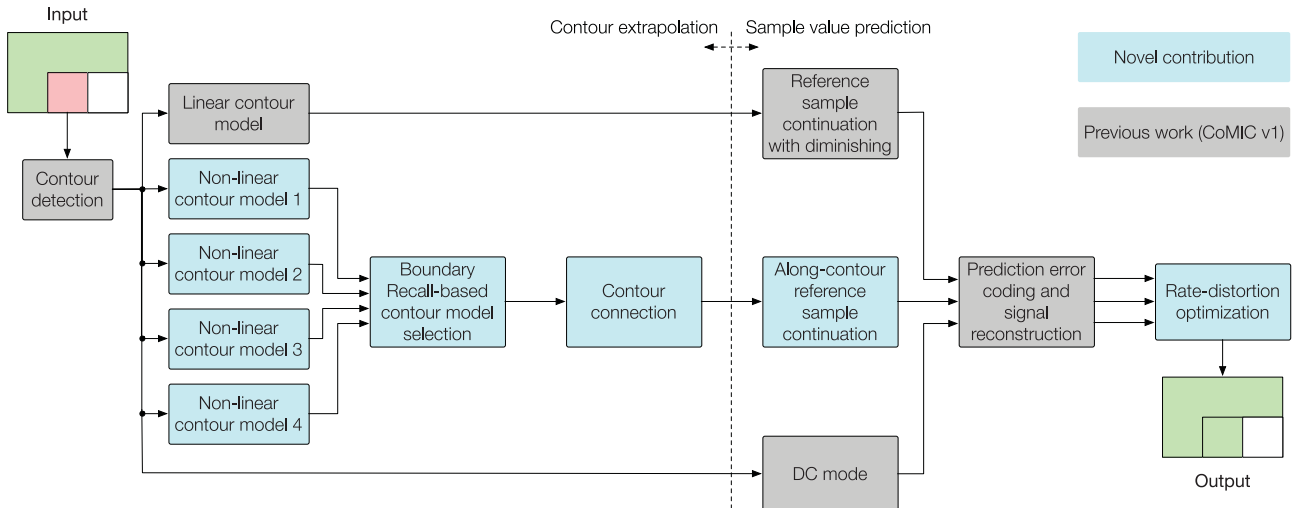

Fig. 6. CoMIC v2 pipeline. Novel contributions of this manuscript are in blue blocks, contributions from our previous work [15] are in gray blocks.

Table 1. Bit rate savings over JPEG (quality 75): we compare our algorithm (CoMIC v2) with the results from the related work of Liu *et al.*, [12] and with the results from our conference paper [15]. Both works are outperformed.

Image	Liu [12] (%)	CoMIC v1 [15][2] (%)	CoMIC v2 (this paper) (%)
Airplane		35.0	36.3
Baboon		25.7	27.1
Barbara		28.4	30.4
Boats		32.8	34.2
Goldhill		30.8	32.1
Jet*	20.6	35.0	36.3
Lena*	20.1	32.4	35.1
Milk*	44.0	34.4	36.9
Pepper*	20.8	34.9	37.4
Kodim 1		22.7	23.6
Kodim 2*	33.0	24.2	25.2
Kodim 3*	32.1	28.4	33.5
Kodim 4		28.6	32.2
Kodim 5*	15.2	26.4	26.7
Kodim 6		24.5	27.6
Kodim 7*	25.7	30.7	32.4
Kodim 8		27.9	28.6
Kodim 9		33.0	34.4
Kodim 10		31.0	32.1
Kodim 11*	23.5	25.0	27.4
Kodim 12		27.9	30.1
Kodim 13		23.9	23.9
Kodim 14		26.5	27.6
Kodim 15		28.0	30.3
Kodim 16		23.2	23.9
Kodim 17		28.2	29.5
Kodim 18		25.4	28.1
Kodim 19*	28.4	29.6	31.1
Kodim 20*	28.9	29.9	32.1
Kodim 21		27.2	30.8
Kodim 22		29.3	30.0
Kodim 23*	30.9	34.3	37.1
Kodim 24		24.9	34.7
Average*	26.9	30.4	32.6
Average		28.8	30.9

Table 2. BD rate gains over JPEG. Positive values indicate increased coding efficiency. CoMIC v1 is outperformed by CoMIC v2

Image	CoMIC v1 versus JPEG (%)	CoMIC v2 versus JPEG (%)	CoMIC v2 versus CoMIC v1 (%)
Airplane	33.47	35.02	2.35
Baboon	24.55	25.06	0.60
Barbara	32.13	33.70	2.38
Boats	30.29	31.92	2.34
Goldhill	27.64	29.04	1.93
Jet	33.47	35.02	2.35
Lena	31.18	33.90	3.87
Milk	33.35	37.03	5.31
Pepper	33.68	36.09	3.58
Kodim 1	22.29	23.44	1.49
Kodim 2	23.79	24.79	1.29
Kodim 3	27.19	29.93	3.89
Kodim 4	27.55	29.54	2.73
Kodim 5	25.36	25.78	0.61
Kodim 6	23.84	25.00	1.51
Kodim 7	29.80	31.84	3.01
Kodim 8	25.69	26.56	1.19
Kodim 9	30.29	32.08	2.65
Kodim 10	28.15	30.07	2.70
Kodim 11	23.81	24.82	1.34
Kodim 12	25.46	27.53	2.72
Kodim 13	22.50	22.87	0.51
Kodim 14	25.18	26.15	1.30
Kodim 15	24.61	27.53	3.68
Kodim 16	22.98	24.12	1.52
Kodim 17	28.31	29.84	2.20
Kodim 18	25.93	26.27	0.48
Kodim 19	26.90	28.05	1.60
Kodim 20	28.52	30.15	2.18
Kodim 21	26.87	27.90	1.47
Kodim 22	27.10	28.01	1.27
Kodim 23	33.25	35.95	3.96
Kodim 24	24.11	25.17	1.40
Average	27.55	29.10	2.16

F) Codec integration

The proposed algorithms were integrated into the CoMIC stand-alone codec software which was originally proposed in [15]. This new version of the codec is referred to as CoMIC v2.

In the following, we will describe the integration of the proposed algorithms in the CoMIC v2 pipeline with reference to Fig. 6. At the encoder, the picture is partitioned into blocks. For the prediction, we use blocks with a size of 32×32. Subsequent to the contour detection, the four different contour models are organized in parallel tracks to the linear contour model. Among the four non-linearly extrapolated contours, the most accurate extrapolation is chosen with the Boundary Recall-based contour model selection. In the case of intersecting contours, the contour connection algorithm is applied. Thereby, the contour extrapolation is finished. Consecutively, the sample values are predicted along the non-linearly extrapolated contour. After this stage, there are three predicted signals: one for the linear prediction, one for the non-linear prediction, and as fallback one for the DC mode. For these three predicted signals, the prediction error is transform coded (DCT and quantization) and the signal is reconstructed with the quantized prediction error. It is worth noting that we use a different block size for the encoding of the prediction error than for the prediction, namely the same 8×8 block size which is used for JPEG. The reason is that this allows to measure the gain of our prediction over the direct encoding of the picture as it is done by JPEG. For the same reason, we utilize the same quantization matrix which is suggested by the Independent JPEG Group and implemented in the JPEG reference software [26].

Finally, the best mode is selected by a rate-distortion optimization [27]. The selection of a Lagrange multiplier for rate-distortion optimization is non-trivial. This states true especially since the rate-distortion optimization is an application-specific task. Hence, there is no single correct λ value. Basically, choosing a Lagrange multiplier is about expressing how the distortion would change in case of a slightly changing bit rate. Therefore, we modeled this

Table 3. BD rate gains over HM. Positive values indicate increased coding efficieny. It is observed that HM is outperformed.

Sequence	All intra				Random Access				Low Delay			
	Y (%)	Cb (%)	Cr (%)	YCbCr (%)	Y (%)	Cb (%)	Cr (%)	YCbCr (%)	Y (%)	Cb (%)	Cr (%)	YCbCr (%)
Steam Locomotive	0.30	0.04	0.05	0.23	0.19	0.38	0.28	0.22	0.15	−0.11	0.02	0.10
Kimono	0.50	0.35	0.30	0.46	−0.16	−0.06	0.10	−0.11	0.12	0.18	0.02	0.11
Park Scene	0.19	0.14	0.24	0.19	−0.01	0.22	0.06	0.03	0.10	0.16	0.16	0.11
Basketball Drive	0.39	0.12	0.16	0.33	0.36	−0.09	−0.15	0.24	0.24	0.17	−0.13	0.18
BQ Terrace	0.37	0.03	0.16	0.30	0.37	0.32	0.29	0.36	0.15	−0.05	0.14	0.12
Johnny	0.16	0.50	0.50	0.24	0.38	−0.13	0.67	0.36	−0.01	2.33	2.65	0.61
Ball Under Water	0.98	1.58	2.32	1.23	1.45	2.18	2.21	1.64	1.12	2.24	1.72	1.34
Bubbles Clear	0.41	0.78	0.77	0.50	2.48	0.27	0.35	1.94	2.17	0.86	0.54	1.80
Calming Water	0.23	0.05	0.11	0.19	0.18	0.16	0.13	0.17	0.19	0.11	0.14	0.17
Carpet Circle Fast	0.32	0.11	0.35	0.29	0.65	0.71	0.59	0.65	0.08	−0.49	−0.06	−0.01
Drops on Water	0.27	0.07	0.13	0.22	0.19	0.10	0.15	0.18	0.20	0.11	0.20	0.19
Flowers 2	0.29	0.01	0.03	0.22	0.12	0.05	0.31	0.14	0.20	−0.07	−0.04	0.14
Paper Static	0.17	0.21	0.84	0.26	0.15	0.26	0.01	0.14	0.07	0.28	0.83	0.19
Smoke Clear	0.58	0.00	0.00	0.44	0.51	0.00	0.00	0.38	0.51	0.00	0.00	0.38
Sparkler	0.45	0.60	0.73	0.51	0.99	0.24	0.73	0.86	0.66	0.30	0.77	0.63
Squirrel	0.17	0.01	0.01	0.13	0.16	0.22	0.27	0.18	−0.03	0.02	0.19	0.01
Bike 1	0.16	0.19	0.39	0.19	0.18	0.14	0.69	0.24	0.13	0.27	0.26	0.17
Bike 2	0.16	0.26	0.35	0.20	0.35	−0.32	0.23	0.25	0.20	0.62	1.11	0.37
Bike 3	0.24	0.39	0.68	0.31	0.49	0.14	1.36	0.56	0.33	−0.10	1.26	0.39
Bike 4	0.21	0.32	0.51	0.26	0.50	0.15	0.39	0.44	0.39	0.54	1.00	0.49
Bike 5	0.30	0.61	1.19	0.45	0.43	−0.21	0.56	0.37	0.31	0.47	0.96	0.41
Bike 6	0.31	0.43	0.59	0.36	0.36	0.06	0.34	0.32	0.36	0.16	0.52	0.35
Bike 7	0.28	0.21	0.47	0.30	0.59	0.01	0.48	0.51	0.37	0.13	0.79	0.40
Bike 8	0.24	0.39	0.44	0.28	0.74	−0.01	−0.23	0.52	0.44	0.06	0.21	0.36
Bike 9	0.12	0.20	0.44	0.17	0.36	0.17	0.70	0.38	0.29	0.43	0.20	0.29
Bike 10	0.14	0.17	0.17	0.15	0.57	0.05	0.70	0.52	0.34	0.15	0.68	0.35
Bike 11	0.35	0.70	0.63	0.43	0.52	0.06	0.44	0.45	0.36	0.46	0.09	0.34
Bike 12	0.10	0.34	0.34	0.16	0.39	0.24	0.51	0.38	0.20	0.19	0.13	0.19
Bike 13	0.19	0.05	0.24	0.18	0.31	0.27	0.55	0.34	0.15	0.38	0.25	0.19
Bike 14	0.42	0.97	1.32	0.60	0.71	0.56	1.62	0.81	0.59	0.89	0.96	0.67
Average	0.30	0.33	0.48	0.33	0.48	0.21	0.48	0.45	0.35	0.36	0.52	0.37

change as a function of the quality parameter for the MSE(rate) curve for the picture *Milk*.

IV. EVALUATION

In this section, we will describe the experiments we performed to evaluate the proposed methods. In [15], we integrated our algorithm into the HEVC reference implementation (HM) and proposed a stand-alone codec implementation to enable a comparison with related works from the literature. Similarly, for this paper, we implemented the proposed algorithms into the stand-alone CoMIC software and into HM as well.

Liu *et al.*, chose to measure the performance of their method in terms of bit rate savings over JPEG at a JPEG quality of 75. In [15], we adopted this procedure. We do the same for this paper to compare the performance of our new contour models with these two related works. The resulting bit rate savings are listed in Table 1[2]. It is worth noting that only a limited number of test pictures were used in [12]. The subset is highlighted by a star (*). For this paper, we extend the set of used pictures to all pictures listed in the table. However, to facilitate an overview at a glance, we calculate a separate average value for the *-subset. CoMIC v2 achieves bit rate savings between 23.6% and 37.4% over JPEG. In average, 30.9% (and 32.6% for the *-subset) are achieved. Thereby, both related works are outperformed. With the new contribution proposed in this paper and incorporated in CoMIC v2, the bit rate savings are increased by on average 2.1 percentage points compared with CoMIC v1. Another interesting finding is that the gap over the results of Liu *et al.*, depends on the content in the encoded images. For images with lots of structural parts (e.g. Lena) the gap is larger than for images with many homogeneously colored areas (e.g. Milk or Kodim 02). For the latter, Liu *et al.*, can max out the benefits of their inpainting algorithm.

The evaluation of bit rate savings for a quality of 75 is a necessary experiment to compare the performance between CoMIC and the related work of Liu *et al.* In addition to this experiment, we also calculate BD rate [28] gains over a wider quality range (30–90) to assess our algorithm for different

---

[2]Various minor improvements of the CoMIC framework were added after the release of CoMIC v1. The binarization of the prediction error was changed from one descriptor with signed values to two descriptors, one with unsigned values and one for the sign. Entropy coding was introduced for the mode usage. As these improvements are not considered as contributions for this paper, we re-ran the simulations for CoMIC v1 with the additional improvements. This way, only the increased coding efficiency as result of the proposed methods is measured.
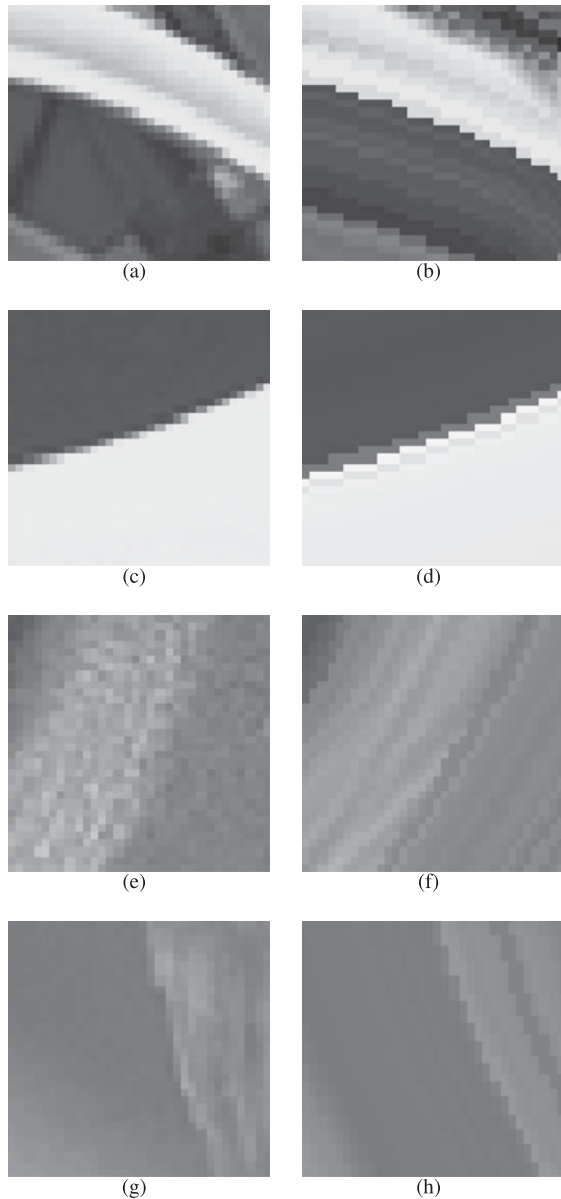
Fig. 7. Exemplary prediction results for the proposed algorithms. The left column (Sub-figures a, c, e, g) shows the original blocks and the right column (Sub figures b, d, f, h) shows the corresponding predicted blocks.

qualities. BD rate gains indicate bit rate savings for equal PSNR. The results are summarized in Table 2. It is observed that CoMIC v1 and CoMIC v2 achieve BD rate gains of 27.55 and 29.10% over JPEG, respectively. These values are almost the same than those for the bit rates savings at quality 75 (28.8 and 30.9%). It can be concluded that the coding gain is achieved consistently over different qualities. When comparing CoMIC v2 with CoMIC v1, the gain is 2.16%.

As second implementation, the proposed algorithms were implemented as additional coding mode in the HEVC reference software HM. The usage of the additional mode is determined by the rate-distortion optimization [27] on coding unit level. CoMIC v2 is chosen if it provides the lowest rate-distortion costs. One binary flag for the mode usage is signaled in the bitstream as part of the coding unit syntax. For blocks for which CoMIC v2 is used, an additional

syntax element is introduced to indicate the selected contour model. To cover a variety of different contents, 30 test sequences as listed in Table 3 were selected from various databases: MPEG Common Test Conditions [29] (Steam Locomotive, Kimono, Park Scene, Basketball Drive, BQ Terrace, Johnny), BVI Texture database [30] (Ball Under Water, Bubbles Clear, Calming Water, Carpet Circle Fast, Drops on Water, Flowers 2, Paper Static, Smoke Clear, Sparkler, Squirrel) and action camera content [31] (Bike). The encoder was configured in all the intra, RA, and low delay configurations [29]. BD rates are calculated following [28]. Additionally, as suggested in [32], weighted average BD rates BD_{YCbCr} were calculated with weighting factors of 6/1/1 for the three color components Y/Cb/Cr. The experimental results are summarized in Table 3. It is observable that HM is noticeably outperformed with BD rate gains of up to 2.48%.

Some subjective results for the predicted signal are illustrated in Fig. 7. The left column shows the original blocks and the right column shows the corresponding predicted blocks. It can be observed that the non-linear contour are predicted accurately.

We measure the computational complexity of CoMIC by using the run time on an Intel Core i7-4770 K CPU @ 3.50 GHz as metric. For a 512×512 pel picture (Lena), the encoding requires 1.65 s (excluding the time for file reading and writing). Similar run times are observed for other pictures. Thereby, the CoMIC is very fast compared with the related work from Liu *et al.*, for which the authors indicate a run time of "multiple minutes" for a picture of the same size. A breakdown of this time is illustrated in Fig. 8: The largest part in the run time is caused by the non-linear prediction, followed by the linear prediction, the transform coding, and the mode selection. The decoding time is lower than the encoding time. This is because the encoder tests all modes per block while the decoder only performs the mode which is signaled in the bit stream for a given block.

Conceptually, for pictures with linear content, the predictions with CoMIC and with HEVC should be similar. This is demonstrated with the synthetic example in Fig. 9. The most noticeable difference in this example are two blocks that are very close to a contour but next to it. For these blocks the CoMIC DC mode produces a prediction which is slightly too dark. This is caused by the different reference areas for the calculation of the mean sample value in the DC modes of HEVC and CoMIC: While HEVC uses only one row/column of sample values, CoMIC uses the entire reference area as demonstrated in Fig. 1. This is
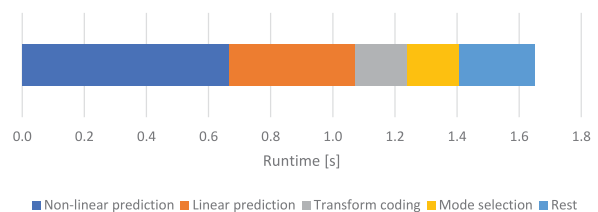

Fig. 8. Run time analysis.

Fig. 9. Predicted signals for a synthetic example: it is observable that CoMIC (left) and HEVC (right) can generate a similar prediction for linear structures.
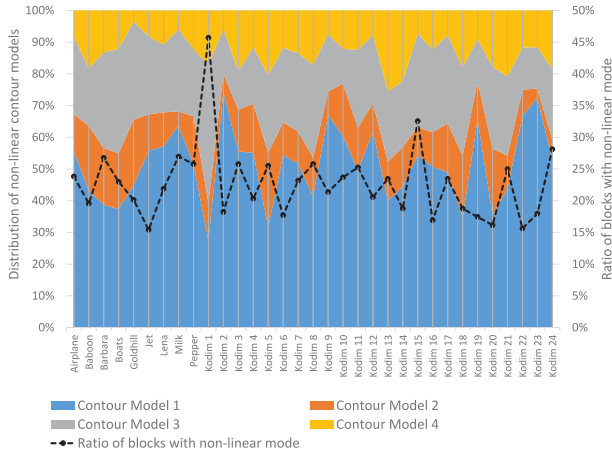

Fig. 11. Pictures *Kodim 1* and *Kodim 13*.

The color of the artifacts is similar to the color of the joints. We hypothesize that these artifacts result in outliers after the contour detection, which are subsequently suppressed by the re-weighting of Contour Model 3. *Kodim 13* includes wavy water with contours whose slope varies. For such contours the distance-dependent weights of Contour Model 4 are beneficial. This hypothesis is supported by the observation that this contour model is used with an above-average share for this picture. Additionally, the ratio of blocks which are coded using non-linear contour models is illustrated on the right vertical axis of the same figure. Naturally, the number of blocks for which the (non-linear) contour models are applicable is limited by the number of blocks which contain such contours. Our analysis is that the non-linear contour models are primarily used for such blocks.

Fig. 10. Distribution of the selected contour models (left vertical axis) and ratio of blocks coded with non-linear contour models (right vertical axis). Contour Model 1 is the regular polynomial model, Contour Model 2 models the slope, Contour Model 3 additionally ensures robustness in case of outliers by iterative reweighting, and Contour Model 4 includes distance-dependent weighting.

more stable in general, but not for this particular synthetic example in which a contour passes through the otherwise homogeneous reference area. Also, some lines appear to be straighter in case of HEVC. The reason is that for CoMIC v2 they are approximated by a quadratic polynomial whose coefficient for the squared term is not exactly determined as zero by the least-squares estimation. Besides that, this example reveals that CoMIC (which extracts all information from the reconstructed signal) is as good as brute-force selection on block-level of one of 35 HEVC intra prediction modes.

To get a deeper understanding for the four different proposed contour models, we analyze their distribution over the test set. The results are visualized in Fig. 10. On average, in 50.2% Contour Model 1 is selected, in 12.7% Contour Model 2, in 23.8% Contour Model 3, and in 13.3% Contour Model 4. The data also suggest that the modes with small average values have a reason for existence: For some pictures (Fig 11), their share in the distribution is considerably larger than their average values. One example is *Kodim 1* for which Contour Model 3 is used more often. This picture shows a brick wall with lots of artifacts in the bricks.

V. CONCLUSION

In this paper, we proposed non-linear CoMIC. It is based on the following main contributions: four non-linear contour models, an algorithm for the connection of intersecting contours, an along-edge sample value continuation algorithm, and a mode decision algorithm for the selection of the best contour model based on the Boundary Recall. The evaluation reveals that related works are outperformed. Compared with the closest related work, BD rate gains of 2.16% are achieved on average. Furthermore, it is worth noting that the CoMIC method can be integrated into any modern video codec where it could be combined with other beneficial tools like advanced prediction error coding or variable block sizes. Namely, BD rate gains of up to 2.48% were achieved over the HEVC reference implementation HM.

REFERENCES

[1] ITU-T: Recommendation H.265/ ISO/IEC 23008-2:2013 MPEG-H Part 2: High Efficiency Video Coding (HEVC).2013.

[2] Wien, M.: High Efficiency Video Coding – Coding Tools and Specification, *Springer*, Heidelberg, 2015.

[3] Hanhart, P.; Rerabek, M.; De Simone, F.; Ebrahimi, T.: Subjective quality evaluation of the upcoming HEVC video compression standard. in SPIE Optical Engineering + Applications, p. 84990V, October 2012.

[4] De Cock, J.; Mavlankar, A.; Moorthy, A.; Aaron, A.: A large-scale video codec comparison of x264, x265 and libvpx for practical VOD applications, in Tescher, A.G., (eds), Applications of Digital Image

Processing XXXIX, *International Society for Optics and Photonics*, San Diego, 2016, 997116.

[5] Laude, T.; Adhisantoso, Y.G.; Voges, J.; Munderloh, M.; Ostermann, J.: A Comparison of JEM and AV1 with HEVC: Coding Tools Coding Efficiency and Complexity, in *Proc. IEEE Picture Coding Symposium (PCS)*, 2018.

[6] Lainema, J.; Bossen, F.; Han, W.-J.; Min, J.; Ugur, K.: Intra coding of the HEVC standard. IEEE Trans. Circuits. Syst. Video Technol., **22** (2012), 1792–1801.

[7] Min, J.; Lee, S.; Kim, I.; Han, W.-J.; Lainema, J.; Ugur, K.: Unification of the Directional Intra Prediction Methods in TMuC, JCTVC-B100, Geneva, Switzerland, July 2010.

[8] Lottermann, C.; Steinbach, E.: Modeling the bit rate of H.264/AVC video encoding as a function of quantization parameter, frame rate and GoP characteristics, in *2014 IEEE Int. Conf. on Multimedia and Expo Workshops (ICMEW)*, July 2014, 1–6.

[9] Yuan, Y.; Sun, X.: Edge information based effective intra mode decision algorithm, in *2012 IEEE Int. Conf. on Signal Processing, Communication and Computing (ICSPCC)*, August 2012, 628–633.

[10] Asheri, H.; Rabiee, H.; Pourdamghani, N.; Ghanbari, M.: Multidirectional spatial error concealment using adaptive edge thresholding. IEEE Trans. Consum. Electron., **58** (2012), 880–885.

[11] Au, O.; Chan, S.-H.: Edge-directed error concealment. IEEE Trans. Circuits. Syst. Video Technol., **20** (2010), 382–395.

[12] Liu, D.; Sun, X.; Wu, F.; Li, S.; Zhang, Y.-Q.: Image compression with edge-based inpainting. IEEE Trans. Circuits. Syst. Video. Technol., **17** (2007), 1273–1287.

[13] Liu, D.; Sun, X.; Wu, F.; Zhang, Y.-Q.: Edge-oriented uniform intra prediction. IEEE Trans. Image. Process., **17** (2008), 1827–36.

[14] Liu, D.; Sun, X.; Wu, F.: Edge-based inpainting and texture synthesis for image compression, in *IEEE Int. Conf. on Multimedia and Expo (ICME)*, July 2007, 1443–1446.

[15] Laude, T.; Ostermann, J.: Contour-based multidirectional intra coding for HEVC, in *IEEE Proc. of 32nd Picture Coding Symposium (PCS)*, Nuremberg, Germany, 2016.

[16] Andris, S.; Peter, P.; Weickert, J.: A proof-of-concept framework for PDE-based video compression, in *Proc. of the Picture Coding Symposium (PCS)*, Nuremberg, Germany, 2016.

[17] Rares, A.; Reinders, M.; Biemond, J.: Edge-based image restoration. IEEE Trans. Image. Process., **14** (2005), 1454–1468.

[18] Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. (PAMI), **8** (1986), 679–698.

[19] Dollar, P.; Zitnick, C.L.: Fast edge detection using structured forests. IEEE Trans. Pattern. Anal. Mach. Intell., **37** (2015), 1558–1570.

[20] Suzuki, S.; Abe, K.: Topological structural analysis of digitized binary images by border following. Comput. Vis. Graph. Image. Process., **30** (1985), 32–46.

[21] Holland, P.W.; Welsch, R.E.: Robust regression using iteratively reweighted least-squares. Commun. Stat. – Theory Methods, **6** (1977), 813–827.

[22] Ostermann, J. *et al.*: Video coding with H.264/AVC: tools, performance, and complexity. IEEE Circuits. Syst. Mag., **4**(1) (2004), 7–28.

[23] Ren, X.; Malik, J.: Learning a classification model for segmentation, in *IEEE Proc. Ninth IEEE Int. Conf. on Computer Vision*, 2003, 10–17.

[24] Achanta, R.; Shaji, A.; Smith, K.; Lucchi, A.; Fua, P.; Süsstrunk, S.: SLIC superpixels compared to state-of-the-art superpixel methods. IEEE Trans. Pattern. Anal. Mach. Intell., **34** (2012), 2274–2282.

[25] Reso, M.; Jachalsky, J.; Rosenhahn, B.; Ostermann, J.: Temporally consistent superpixels, in *The IEEE Int. Conf. on Computer Vision (ICCV)*, 2013, 385–392.

[26] I. J. Group: "http://www.ijg.org/ (accessed 2018-03-19)".

[27] Sullivan, G.; Wiegand, T.: Rate-distortion optimization for video compression. IEEE Signal. Process. Mag., **15** (6) (1998), 74–90.

[28] Bjøntegaard, G.: 35th Meeting VCEG-AI11: Improvements of the BD-PSNR model. ITU-T Study Group 16 Question 6, Berlin, Germany, 2008.

[29] Bossen, F.: JCT-VC L1100: Common HM test conditions and software reference configurations. 12th Meeting of the Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Geneva, CH, 2013.

[30] Papadopoulos, M.A.; Zhang, F.; Agrafiotis, D.; Bull, D.: A video texture database for perceptual compression and quality assessment, in *Int. Conf. Image Processing (ICIP)*, Quebec, Canada, 2015.

[31] Laude, T.; Meuel, H.; Liu, Y.; Ostermann, J.: Motion blur compensation in scalable HEVC hybrid video coding, in *IEEE 2013 Picture Coding Symposium (PCS)*, San Jose, CA, USA, 313–316, December 2013.

[32] Sullivan, G.J.; Ohm, J.-R.: Meeting Report of the Fourth Meeting of the Joint Collaborative Team on Video Coding. *ITU-T/ISO/IEC JCT-VC Document JCTVC-D500*, 2011.

Thorsten Laude Thorsten Laude studied electrical engineering at *Leibniz University Hannover* with specialization on communication engineering and received his Dipl.-Ing. degree in 2013. In his diploma thesis, he developed a motion blur compensation algorithm for the scalable extension of HEVC. He joined *InterDigital* for an internship in 2013. At *InterDigital*, he intensified his research for HEVC. After graduating, he joined *Institut für Informationsverarbeitung* at *Leibniz University Hannover* where he currently pursues a Ph.D. He contributed to several standardization meetings for HEVC and its extensions. His current research interests are intra coding for HEVC, still image coding, and machine learning for inter prediction, intra prediction and encoder control.

Jan Tumbrägel Jan Tumbrägel studied electrical engineering at *Leibniz University Hannover* and received his Dipl.-Ing. degree in 2016.

Marco Munderloh Marco Munderloh achieved his Dipl.-Ing. degree in computer engineering with an emphasis on multimedia information and communication systems from the *Technical University of Ilmenau*, Germany, in 2004. His diploma thesis at the *Fraunhofer Institute for Digital Media Technology* dealt with holographic sound reproduction, so-called wave field synthesis (WFS) where he holds a patent. During his work at the Fraunhofer Institute, he was involved in the development of the first WFS-enabled movie theater. At *Institut für Informationsverarbeitung* of *Leibniz University Hannover*, Marco Munderloh wrote his thesis with a focus on motion detection in scenes with non-static cameras for aerial surveillance applications and received his Dr.-Ing. degree in 2015.

Jörn Ostermann Jörn Ostermann studied Electrical Engineering and Communications Engineering at *University of Hannover* and *Imperial College London*. He received Dipl.-Ing. and Dr.-Ing. from the *University of Hannover* in 1988 and 1994, respectively. In 1994, he joined *AT&T Bell Labs*. From 1996 to 2003 he was with *AT&T Labs – Research*. Since 2003 he is Full Professor and Head of *Institut für Informationsverarbeitung* at *Leibniz University Hannover*, Germany. Since 2008, Jörn Ostermann is the Chair of the Requirements Group of MPEG (ISO/IEC JTC1 SC29 WG11). Jörn Ostermann received several international awards and is a Fellow of the IEEE. He published more than 100 research papers and book chapters. He is coauthor of a graduate level text book on video communications. He holds more than 30 patents. His current research interests are video coding and streaming, computer vision, 3D modeling, face animation, and computer–human interfaces.