Adaptive In-Loop Prediction Refinement for Video Coding

Shay Har-Noy∗†, Oscar Divorra Escoda∗, Peng Yin∗, Cristina Gomila∗ and Truong Q. Nguyen†

∗Corporate Research, Thomson, Princeton, NJ, USA
†Video Processing Laboratory, University of California, La Jolla, CA, USA
Email: sharnoy@ucsd.edu, {oscar.divorra,peng.yin,cristina.gomila}@thomson.net, nguyent@ece.ucsd.edu

Abstract—Modern video compression codecs achieve high compression efficiency by exploiting the temporal and spatial redundancies present in video sequences. Although state of the art block based intra and inter prediction have been refined to better adapt to the content in the scene, they still exhibit limitations in fully extracting information from the decoded data. We propose the addition of an in-loop prediction refinement stage that is initialized with the conventional intra or inter predicted result and is capable of further reducing the spatial redundancies present in the sequence. By extracting additional information from the decoded neighborhood, the refinement stage is able to bring the prediction closer to the original data, thus improving compression efficiency. In this work, we study one possible method of refinement for the particular case of intra prediction. This uses sparse decomposition of blocks that overlap decoded neighboring blocks and the current predicted block in order to make prediction more coherent with already encoded data.

I. INTRODUCTION

Spatial and temporal prediction are a critical part of modern video codecs as means of improving coding efficiency. Modern hybrid block based codecs, such as H.264/AVC [1], use motion compensation to exploit the temporal redundancies present in video sequences. Spatially coded frames and blocks are used to limit error propagation in places where even sub-pixel motion estimation is unable to efficiently describe the video scene, e.g. in areas suffering from occlusions.

Typical spatial prediction techniques, such as intra prediction, work by splitting the frame into a set of non-overlapping blocks and adaptively predicting each block from its spatially neighboring blocks. The current H.264/AVC standard performs spatial prediction by copying or interpolating the data from the edges of neighboring blocks using a variety of modes and then selecting the mode that offers the best performance in terms of rate-distortion. Temporal prediction, such as inter prediction, also splits the frame into non-overlapping blocks and searches for a match for each block in neighboring frames. Adaptive partitioning of video frames into blocks of various sizes is used to improve the RD efficiency where the finer partitions (e.g. Intra_4x4, Inter_4x4) tend to concentrate around contours, high detail and/or high motion areas.

Despite the ability of intra and inter coding modes to adapt to fine details in areas of the image by varying partition size and coding mode, there are still limitations on the accuracy of prediction. In order to improve spatial and temporal prediction, we propose the addition of an in-loop refinement stage in order to better regularize predicted data according to spatially neighboring already encoded data.

The refinement module can be applied to either intra or inter prediction in any prediction-based video coding framework. The module refines the prediction in the current block using the available decoded spatial neighborhood. Side information can then be used to adaptively indicate, to the decoder, whether or not refinement is in use for a particular image region.

In this paper, we present one possible approach to in-loop data refinement using recent advancements in missing data estimation by sparse signal decompositions [6], [3]. Such an approach can improve consistency among local higher order statistics by locally enforcing sparsity constraints on the predicted signal. Localized dynamic sparsity constraints have also been applied to video compression as a means of reducing quantization noise [4] and improving coding efficiency during fades and blends [7].

The paper is structured as follows: In section II, we introduce the general framework of an in-loop refinement stage and describe a particular adaptive decision structure for intra coding refinement. Section III describes the detail of our refinement filter implementation using sparse signal decompositions. Finally, in sections IV and V results are presented from our refinement filter implementation along with conclusions, respectively.

II. ADAPTIVE IN-LOOP PREDICTION REFINEMENT

A. Refinement on Current Prediction Methods

In the context of video compression, prediction is an approximation step where pictures are locally approximated by adaptively considering different signal-dependent models. Spatial prediction, such as intra prediction in H.264/AVC, assumes that pixels in a block can be interpolated and/or extrapolated from neighboring decoded blocks using simple linear models. Similarly, temporal prediction, such as inter prediction in H.264/AVC, assumes temporal regularity in the direction of purely translational motion.

Let \( x \) be the video signal, \( p_i(.) \) the predictor function selected by the encoder, and \( \hat{x}_d^{pi} \) the already encoded/decoded part of the signal selected by \( p_i(.) \) for prediction, then \( x \) can be, in general, expressed as

\[
\begin{align*}
    x = p_i(\hat{x}_d^{pi}) + \epsilon_{pi},
\end{align*}
\]

where
where $\epsilon_{p_i}$ is the prediction error. In most cases, $p_i(\cdot)$ is rarely complete enough to fully represent $x$ from $\hat{x}_d$. Consequently, $\epsilon_{p_i}$ or a transformed version of it is often required to represent $x$.

In current prediction strategies, neither intra nor inter prediction take full advantage of all spatial dependencies present in a typical video frame. Inter prediction neglects most of the spatial redundancies within the same frame, while the copying nature of intra prediction results in a non-linear directional low pass characteristic that neglects higher order components.

In order to improve prediction beyond conventional intra and inter prediction schemes, one can impose spatial constraints to improve regularity in signal features. Indeed, let $f_R (x_1, x_2)$ be a filtering operation where $x_1$ is regularized according to the statistics and/or characteristics of a given $x_2$, a new representation of $x$ is:

$$x = f_R \left( p_i \left( \hat{x}_d^{p_i} \right), \hat{x}_{ds} \right) + \epsilon'_{p_i},$$

where, when $p_i(\cdot)$ and $\hat{x}_{ds}$ have enough common components, it is possible that $\|\epsilon'_{p_i}\| \leq \|\epsilon_{p_i}\|^2$. In Eq. 2, $\hat{x}_{ds}$ stands for all available pixels from neighboring, already encoded/decoded, blocks. Eventually, this can lead to a reduction in the amount of residual information to code.

The proposed approach seeks to improve consistency of image features across boundaries of blocks by adding a refinement filter, $f_R$, after the standard prediction stage. In practice, the refinement module can be applied on the standard intra or inter prediction result for the current block, where the filter uses the available spatially neighboring decoded blocks available at both the encoder and decoder to refine the prediction (Fig. 1).

In the particular case of intra prediction, refinement is conducted using the causally decoded neighborhood, meaning the upper, upper-left, upper-right and left macroblocks of the predicted data when raster-scan coding order is used. In addition, for sub-block intra prediction, such as that allowed in H.264 (INTRA_4x4), those sub-blocks already decoded can also be used (Fig. 2). Inter prediction refinement can be made much more flexible by conducting refinement after motion compensation is performed on the entire frame. Indeed, an out-of-order coding of prediction residual can allow the refinement procedure access to all decoded blocks surrounding those regions which are more poorly predicted. In the following, though, we will concentrate on the intra prediction refinement case.

### B. Intra Prediction Refinement

In the context of H.264/AVC intra prediction, the encoder selects the best R-D directional prediction mode (e.g. vertical, horizontal, etc.) for every coded block. In our proposed approach, a refinement procedure is applied on the best selected prediction by H.264/AVC, generating a new coded residual and R-D cost.

Depending on the signal nature, and on its local characteristics, refinement may not always be able to improve prediction, and in some cases, can even degrade it. Hence, after the refinement procedure terminates, the coding algorithm determines whether or not the refinement procedure improves the prediction of the current block in a R-D sense. If prediction is improved, the result produced by the refinement procedure is stored for that block, otherwise the original prediction is used. Appropriate side information can then be used to signal the adaptive use of prediction refinement.

In this work, two possible refinement configurations are introduced for INTRA_4x4 mode. In the first one (REFINE_4x4), the refinement filter is applied to all sub-macroblocks using the available 4x4 and 16x16 causally decoded neighbors (Fig. 2). A single flag is, then, sent to the decoder for the entire macroblock indicating whether or not to perform refinement. In the second configuration (REFINE_SUB4x4), a decision on whether or not to maintain the refined result is made for each sub-macroblock and 16 flags are sent to the decoder indicating which of the sub-macroblocks have been refined.

In order to limit the number of overhead bits sent to the decoder, flag coding into the bit stream is done on a hierarchical basis. For every macroblock the following coding structure is used:

- Code flag $mbRefineFlag$

  - if ($mbRefineFlag == 1$ and INTRA_4x4)
    - Code flag $subMbRefineFlag$
      - if ($subMbRefineFlag == 1$)
        - Code 16 flag bits $refineFlags4x4[16]$

where $mbRefineFlag$ indicates whether refinement is performed on the current macroblock, $subMbRefineFlag$ indicates whether refinement decisions are made on a 4x4 block basis, and the $refineFlags4x4$ flags indicate which of the 4x4 blocks have been refixed. The R-D computation for each macroblock takes into account the number of flag bits required for each refinement mode.

INTRA_16x16 mode is always refined on 16x16 block basis, hence only one flag per macroblock is necessary.

Clearly, directional modes could be jointly optimized with the refinement procedure. However, such a joint optimization is not considered at this point due to its high computational complexity.
one possible approach for prediction refinement is the use of a sparsity based non-linear filter. Such a filtering process refines the prediction on the current block by enforcing an assumption of sparseness of a transform over the entire region containing the known decoded neighborhood and the current predicted block.

A. Sparsity Based Filtering

Sparsity based filtering is typically performed by following three steps: i) forward transform, ii) coefficient thresholding, iii) inverse transform. In such a procedure, one assumes that only those signal components free of error are kept by the thresholding operation. When there is no clear separation between error components and good signal, iterative filtering strategies can be considered, using conservative thresholds. This is the case presented in [5], [6] for the recovery of missing image regions. In [5], [6], missing data is iteratively estimated by successive de-noising steps on available data and the missing region with a dynamic thresholding on an over-complete transform domain.

In the present work, we adapt [5], [6] for the purpose of studying the use of prediction refinement within video coding schemes. We consider the over-complete transform formed by all integer shifted DCT transforms in the region of interest (the current block and its decoded neighbors, Fig. 2). The refinement filter works by splitting the current block into pixel layers and updating the prediction on each layer within the inner loop of the following procedure [5], [6]:

- Determine a starting threshold $T_0$ based on statistics of the surrounding decoded neighborhood.
- Determine a final threshold $T_f$ based on coding quality.

1) $T = T_0$.
2) Decompose current block into L layers.
3) Set prediction values on layer pixels, $Iter = 0$.
4) While ($T > T_f$ & $Iter < MAX_{Iter}$)
   a) For $i = 1, \ldots, L$
      i) Find DCT blocks with significant overlap with layers $i, \ldots, L$.
      ii) Hard threshold coefficients with $T$.
   iii) Inverse transform and update the pixels in layer $i$ by averaging the over-complete inverse transforms that overlap the relevant part of the layer.
   b) $T = T - \Delta T$, $Iter = Iter + 1$.

B. Sparsity Based Intra Prediction Refinement

In order to limit the number of overhead bits sent to the decoder, the initial threshold is computed from the statistics of the decoded neighboring data. As $T_0$ increases, the amount of prediction error is assumed to be higher, discarding a higher amount of components from the initial prediction. As we allow more of its transform coefficients to be set to zero, the amount of refinement performed on the current block is greater, adopting more information from all of its available neighbors. This is in contrast with intra prediction, where only a selection of all available pixels are used for prediction. Thus, the initial threshold can be thought of as a measure of trust we have in the initial prediction.

The refinement filter adaptively sets the initial threshold, $T_0$, based on the information in the surrounding neighborhood. In [6] the missing region is initialized with the mean of the known data in the image and the initial threshold, $T_0$, is set to be the standard deviation of that data. However, since we initialize the block with the intra coded result, we expect our initialization to be closer to the true values than the mean of the surrounding regions. Moreover, the proximity of the initialized block to the true values depends on the quality of the intra prediction and thus on the residual energy for that block. However, since the decoder does not have access to the residual of the current block before refinement, the decoded residual of the neighboring blocks is used as an approximation.

Combining these factors, we arrive at two possible configurations for selecting a reasonable initial threshold:

- $T_0 = \sigma_s / \gamma$
- $T_0 = \sigma_r / \gamma$

where $\gamma$ is a scaling factor and $\sigma_s$ and $\sigma_r$ correspond to the standard deviations of the surrounding neighborhood and of the residual of the surrounding neighborhood, respectively. The decision on whether to use $T_0$ or $T_0$ and the $\gamma$ factor can be made on a slice, picture or sequence level and communicated to the decoder in any of the headers.

The final threshold $T_f$ (s.t. $T_f < T_0$) corresponds to the noise floor of the image and thus the point beyond which we do not expect further improvement. In our context, the quantization step size may be used to determine the noise floor as the quality of the decoded neighboring data is limited by the quantization error introduced in the encoding process.

IV. EXPERIMENTAL RESULTS

The refinement algorithm described in section [III] is implemented on the JSVM 6 [8] H.264/AVC reference software. All of the presented results were computed using QCIF resolution sequences with the prediction refinement procedure only operating on the luma band. Prediction refinement parameters ($\gamma$, $\Delta T$, $T_f$, selection between 16x16 or 8x8 DCT size, and selection between $\{T_0^s, T_0^r\}$) were globally optimized for
best performance for every sequence and QP. Throughout this section, average coding gains are given according to the convention proposed in [2].

Fig. 3 shows a frame from the Foreman sequence overlaid with the refinement decisions using QP=36. The large congregations of REFINE_16x16 decisions in the upper right corner of the image and on the face epitomize the strength of the refinement procedure. Indeed, these macroblocks contain either texture or structure highly correlated with that found in their causal neighbors. Thus, the refinement filter is able to propagate reliable information into these blocks using sparse decompositions of overlapping blocks.

Fig. 4 compares the RD performance of the H.264 codec with the refinement filter enabled to that of the standard H.264 codec for the Foreman sequence. The plot shows a consistent improvement in RD across all QPs (28, 32, 36, 40) for sequence Foreman, indicating that the proposed refinement framework, even in this preliminary implementation, is indeed capable of improving the coding efficiency of state-of-the-art video codecs. Table I shows the average compression gains of the refinement filter for additional sequences.

Additional gains can be achieved by further development of the refinement filter and by possibly predicting from the neighboring decoded data which are the blocks that might benefit from refinement. Doing so, it would reduce the number of overhead bits required to signal to the decoder which blocks to perform refinement on.

V. CONCLUSION

In this work, adaptive in-loop prediction refinement for video coding is discussed. In particular, we study the application of a sparsity based spatial constraint for intra prediction refinement. This is able to enhance state-of-the-art intra prediction by extracting information from the decoded neighborhood.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Average dB Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foreman</td>
<td>0.259 dB</td>
</tr>
<tr>
<td>Carphone</td>
<td>0.189 dB</td>
</tr>
<tr>
<td>Silent</td>
<td>0.116 dB</td>
</tr>
<tr>
<td>Bus</td>
<td>0.108 dB</td>
</tr>
<tr>
<td>Container</td>
<td>0.166 dB</td>
</tr>
</tbody>
</table>

Table I

Our results show that the addition of an adaptive in-loop prediction refinement stage improves compression efficiency. In order to further improve coding performance, future work involves studying more efficient and less complex refinement strategies, as well as detailed experimentation with refinement filters for the inter prediction framework.

REFERENCES