Ultrafast Scalable Embedded DCT Image Coding for Tele-immersive Delay-Sensitive Collaboration

Mauritz Panggabean, Maciej Wielgosz, Harald Øverby, and Leif Arne Rønningen
Department of Telematics (ITEM)
Norwegian University of Science and Technology (NTNU)
N-7491, Trondheim, Norway

***Abstract*—A delay-sensitive, real-time, tele-immersive collaboration for the future requires much lower end-to-end delay (EED) for good synchronization than that for existing teleconference systems. Hence, the maximum EED must be guaranteed, and the visual-quality degradation must be graceful. Distributed Multimedia Plays (DMP) architecture addresses the envisioned collaboration and the challenges. We propose a DCT-based, embedded, ultrafast, quality scalable image-compression scheme for the collaboration on the DMP architecture. A parallel FPGA implementation is also designed to show the technical feasibility.**

I. Introduction

Figure 1 shows a simple example of the envisioned collaboration. A and B engage each other in a real-time delay-sensitive communication. They are both a source and a receiver, whereas C only receives data from them. As EED is not critical for C, C can use video-streaming technologies over the Internet. The capacity in the multihop links between A, B and C varies because other users outside the collaboration also use them. Moreover, the target quality of experience (QoE) is so high that it closely approximates reality, i.e. near-natural.

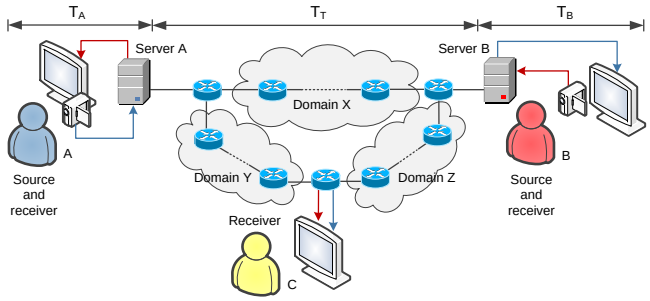


Fig. 1: A simple example of the futuristic collaboration.

The collaboration environment is an immersive collaboration space (CS) more advanced than the CAVE [5]. To achieve the near-natural QoE, each surface in the CS is tiled with autostereoscopic multiview 3D displays with arrays of high-end cameras, microphones, and speakers. At high frame rate, the video traffic from a CS is several orders of magnitude higher than that in typical videoconferencing. It is reduced by *segmenting* only the important objects in the video, such as the faces and bodies of the performers.

A maximum EED for good synchronization between A and B must be guaranteed. Some studies show that the optimal EED for synchronizing rhythmic clapping hands from different places is 11.5ms [4]. Longer delays will produce increasingly severe tempo deceleration while shorter ones yield a modest yet surprising acceleration. Since musical instruments such as percussion are rhythmically very similar to clapping hands, percussion musicians who collaborate from remote places require the same EED for synchronization. It also applies to collaborative dancing because dancers perform based on visual cues from each other [40]. Other cases include collaborative singing and remote conducting [18].

An EED consists of delays due to propagation, transmission, and signal processing. Propagation delay is caused by physical distances, and transmission delay depends on link capacity, queueing delay, and computations at the network nodes. The latter is the electronic bottleneck that limits the achievable capacity of a network [27]. Less capacity in the network causes congestion and increases queuing delay. Instead of multipath transmission, single path is assumed to simplify routing delay. Exploiting temporal redundancy when encoding video data gives better quality but increases encoding delay. *Intraframe* video encoding is, therefore, preferred as shown by a recent experiment that uses JPEG [10]. Since we pursue *very low latency for encoding and decoding* in the order of μs per frame, *parallel* computation must be used as much as possible.

The Distributed Multimedia Plays (DMP) architecture has been proposed to facilitate the envisioned collaboration [17] with the idea that maximum EED is guaranteed if each network node guarantees that the local delay never exceeds its maximum value. This value and the propagation delay can be estimated prior to packet transmission. Because the routers and switches in DMP have advanced functionalities to guarantee Quality of Service (QoS), DMP belongs to the network-centric approach rather than the end-system-based approach [37].

The idea has three important implications. First, a DMP network node must be able to drop parts of the video packets deliberately whenever necessary to guarantee its local delay. The dropping must be fast, and the buffer size must be optimal. Determining the latter is not the goal of this work.

Second, the packet dropping to guarantee graceful video-quality (VQ) degradation must be conducted intelligently. The video contents in the packets must be arranged and transmitted in decreasing order of the importance to VQ. The less important the contents in a packet, the higher the dropping priority. Packets that contain very essential contents, however, must never be dropped. This leads to the property of *quality*

scalability in the wanted image-compression scheme.

Third, fast packet dropping means that it occurs in compressed domain. By providing information necessary for this in the bitstream, the cycle of decoding, dropping, and re-encoding at a node is avoided. This and the second implication mean that the bitstream can be truncated at any point to yield the reconstructed image at a lower bitrate. The quality at the final rate after dropping should be the same with that if it is encoded directly at that rate, i.e. *embedded coding* [22].

The objective of this work is to design an image-compression scheme that has all the properties aforementioned: ultrafast, embedded, quality scalable, fully parallelized, and supporting the processing of segmented objects with arbitrary shapes. Note that we do not pursue better coding performance than that of non-scalable image coding standards because it is unfair and irrelevant. The envisioned collaboration allows for sub-optimal VQ as the price for guaranteeing maximum EED as long as the VQ is gracefully degraded. Tradeoff is normal in image/video coding. For instance, H.264/MPEG-4 AVC [11] and x264 [39] provide profiles and presets to meet various priorities such as low complexity or high performance.

This paper is structured as follows. Section II details the proposed image-compression technique. Experimental results follow in Section III with discussion and analysis. Section IV discusses the complexity of the algorithms for implementation on field-programmable gate array (FPGA). Section V concludes the paper with summary and further ideas.

II. The Proposed Image-Compression Scheme

The DMP approach resembles the concept of layered coding such as in scalable video coding (SVC) [15], [19], [25] and JPEG 2000 [32]–[34]. SVC achieves temporal, spatial, and quality scalability by removing parts of the video bitstream to adapt it to different end-users' preferences and varying terminal capabilities or network conditions. Proposed to supersede JPEG, JPEG 2000 is an image compression standard and coding system based on wavelet transform. Some of the improvements over JPEG are as follows: superior compression performance, multiple resolution representation; progressive transmission by pixel and resolution accuracy; spatial, quality and channel scalability; support of lossless and lossy compression; embedded coding; facilitated processing of regions of interest; error resilience. Consequently, they make JPEG 2000 more complex and computationally demanding.

The properties aforementioned make the proposed image-compression technique (Fig. 2) somewhat different from the existing ones. For example, the quantization, a key step such as in JPEG image compression (Fig. 3), is the principal cause for the loss of information, while the loss in DMP is due to the deliberate packet dropping at network nodes, i.e. the proposed scheme has no such quantization. Moreover, the techniques optimize bandwidth utilization by aiming for the best VQ at a given bitrate with no guarantee over maximum EED.

A. Block Ranking and Transform

The encoder of the proposed scheme consists of three major steps: block ranking, transform, and entropy coding (variable length encoding, VLE, and run-length encoding, RLE). After

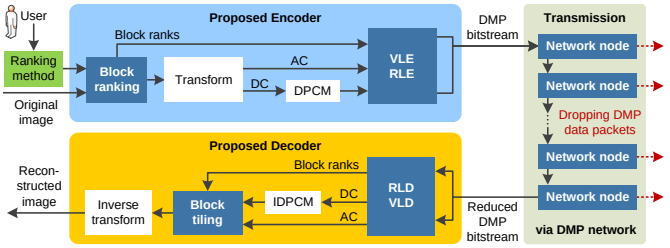


Fig. 2: The proposed image-compression technique.

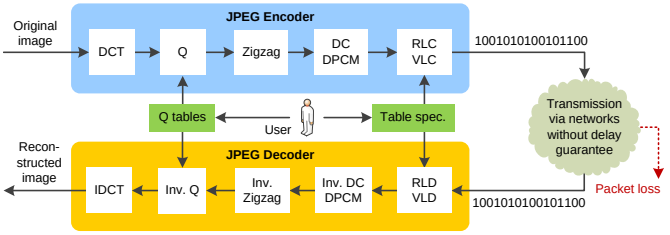


Fig. 3: Block diagram of JPEG image compression.

an input picture is divided into $N \times N$ blocks and the color space is converted to YCbCr, the block ranking automatically classifies each block into one of several ranks according to how important the block contents are to VQ. The importance of a block is indicated by the level of distortion to human perception when the binary representation of the block content becomes less precise, e.g. via quantization or packet dropping. The blocks are independent from each other, and thus can be processed concurrently. In this work $N = 8$ pixels, and users can define their own ranking method.

For the transform, two dimensional DCT (2D-DCT) [1] is selected for two main reasons. First, it is widely used because of the excellent energy compaction. Second, many fast hardware (HW) implementations of 2D-DCT have been reported, e.g. in [28]. The most recent work closest to ours is that by van der Vleuten et al. [35], which incorporates quality scalability to JPEG by encoding the DCT coefficients bit-plane by bit-plane, starting at the most significant one. Although the performance is similar to that of JPEG without quantization or entropy coding, the algorithm, particularly the scan order, must be adapted to each image. Our scheme is agnostic to the input image.

The block ranking can be applied before or after the block transform. The first only has 64 pixel values of the block luma available for analysis and ranking, whereas 64 DCT coefficients are additionally present in the latter. We choose the first option because various statistical properties of pixel values have been used for content classifications in images [6], [41]. Furthermore, luma values are integers, but DCT coefficients use floating points. Therefore, computing pixel values requires less resources and time than if DCT coefficients are added to the computation. Moreover, DCT coefficients in natural images are more complex to use for classification purposes [24].

The statistical measures for ranking the blocks must be highly accurate and fast to compute. We use the entropy E, which measures the amount of information and uncertainty

contained in data [21]. For a grayscale image with N unique pixel values, it characterizes the texture therein as

$$E = -\sum_{i=1}^{N} p_i \log_2 p_i$$

where p_i is the probability of the ith pixel value from the histogram counts. The block entropy BE rises when the frequency content of the block increases.

Shown in Fig. 4 for LENA image using 8×8 blocks, the BE values are between 2 and 6 in all images tested (Fig. 5). Since the colors correspond well with human perception, BE is a good indicator of the frequency content in a block. The constant range of BE can be used to define the thresholds for arbitrary number of block ranks for dropping. We use the following four block ranks with the ranges: low ($\lceil BE \rceil \leq T_L$), low-medium ($T_L < \lceil BE \rceil \leq T_{LM}$), medium-high ($T_{LM} < \lceil BE \rceil \leq T_{MH}$), and high ($\lceil BE \rceil > T_{MH}$), where T_L, T_{LM} and T_{MH} are thresholds in positive integers.

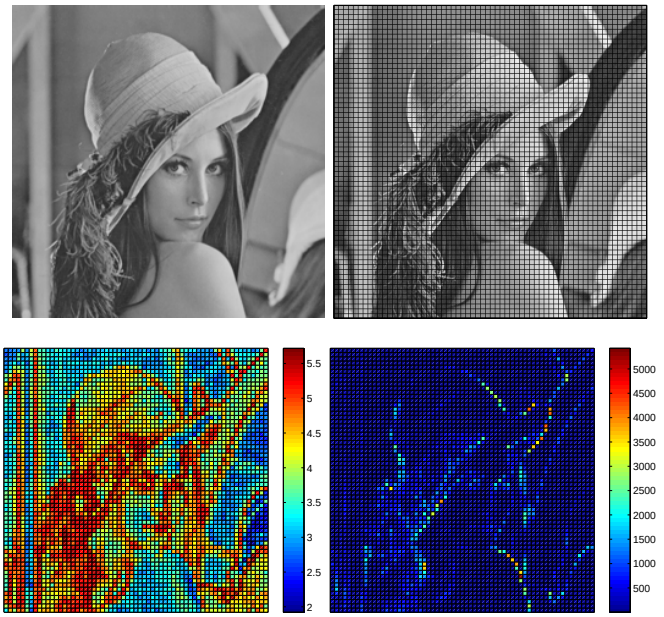


Fig. 4: Clockwise from top left: the original LENA image, the tiled 8×8 blocks, the BV, and the BE.

The criterion for the high-frequency rank is not very accurate because some of the blocks are grouped into the medium-high rank. It leads to the use of block variance BV to improve the block-ranking accuracy. The variance of a grayscale image with M pixel values is given by

$$V = \frac{1}{M-1} \sum_{i=1}^{M} (x_i - \hat{x})^2$$

where x_i denotes the intensity value of the ith pixel, and $\hat{x}$ is the average of all the pixel values. In the proposed block-ranking algorithm (Algorithm 1), $\lceil x \rceil$ rounds the scalar x to the nearest integer towards plus infinity, $T_V = 1, T_L = 3$, and $T_{LM} = 4$. The resulting BV values for LENA image are shown in Fig. 4. The BE and BV are not only fast to compute in HW (section IV), but also correspond well with human perception (section III).

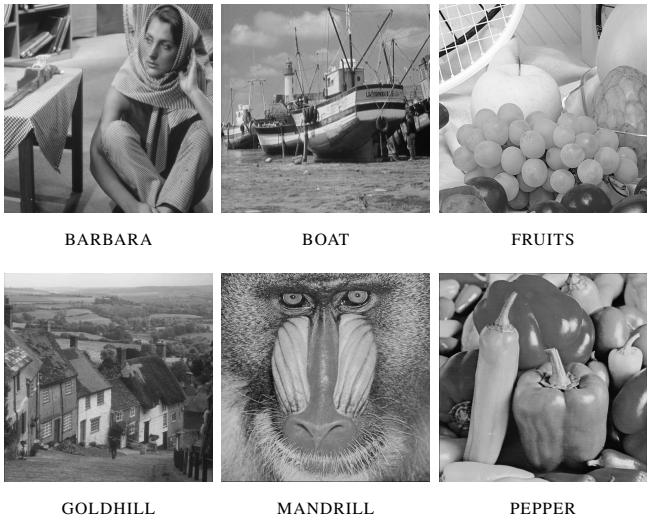


Fig. 5: The test images besides LENA.

Algorithm 1 Proposed algorithm for block ranking

```
if ⌈BV/100⌉ ≤ T_V then
    if ⌈BE⌉ ≤ T_L then
        Rank 4: low frequency (blue)
    else if T_L < ⌈BE⌉ ≤ T_LM then
        Rank 3: low-medium frequency (green)
    else if ⌈BE⌉ > T_LM then
        Rank 2: medium-high frequency (yellow)
    end if
else
    Rank 1: high frequency (red)
end if
```

Encoded and included in the bitstream as side information, the produced block ranks must never be lost because it will jeopardize the image reconstruction from the transmitted packets at the receiver. They are also used in structuring the encoded DCT coefficients into the data packets to enable packet dropping in the compressed domain.

After the block ranking, 2D-DCT is applied to each of the luma block independently, and it produces 64 DCT coefficients per block, which comprise one DC coefficient and 63 AC coefficients. As the DC coefficient contains the average value of the block, it is essential for reconstruction and must not be dropped. The location of an AC coefficient in a block indicates the importance. The only information loss in the proposed encoder, rounding the values to the nearest integers reduces the precision for faster computation with less memory use.

B. Universal Codes for Entropy Coding

The distribution of the rounded DCT coefficients is key in encoding them losslessy and efficiently. Fig. 6 depicts the empirical probability density functions (PDFs) of the rounded AC coefficients between -10 and 10 from DCT and Walsh-Hadamard transform (WHT) for comparison. They include more than 99% of all the coefficients for WHT in all the test images, but it is between 80% and 99% for the DCT. Note the symmetry around zeros in the PDFs. The quality of the

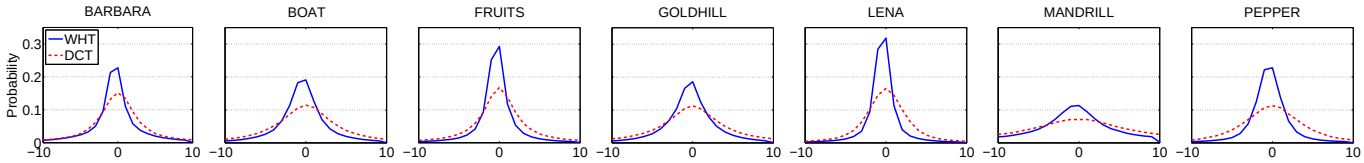


Fig. 6: The empirical PDFs of the AC coefficients from DCT and WHT for the test images.

reconstructed images using DCT in peak-signal-to-noise ratio (PSNR) is always a few dB higher than that for WHT due to the rounding of the DCT coefficients.

The probability of zeros is always less than 0.20 for DCT and slightly higher for WHT. Producing shorter average length of codewords with higher probability of zeros [23], the Huffman code is not efficient for this case. Moreover, the distribution of the DCT coefficients could not be known beforehand. The Huffman code is favored when more zeros are present in the high-frequency coefficients caused by stronger quantization.

The dropping of the DC coefficients starting from the least important implies that RLE is not suitable for this work because it introduces more dependencies in the resulting bitsream. The run lengths of the coefficients can also be very short because of no quantization. This is a challenge because RLE can increase performance in lossless coding.

Furthermore, coding techniques that can only be decoded when the bitstream is complete, such as the Burrows-Wheeler transform (BWT) [3], are also not suitable. It is, in fact, impossible because the received stream at the end are truncated due to dropping. Nevertheless, they are useful for encoding the block ranks and the rounded differences after applying differential pulse-code modulation (DPCM) to the DC coefficients.

Excellent texts such as [23] comprehensively discuss and compare various coding techniques available. They lead us to the use of universal codes (UCs) for entropy coding for the applicability regardless of the data distribution. We propose using the Fraenkel and Klein C^1 Fibonacci code [8] (FK_1) based on the comparison of well-known UCs in [7]. The recurrence relation $F(i) = F(i-1)+F(i-2)$ with seed values $F(0) = 0$ and $F(1) = 1$ defines the sequence $F(i)$ of the famous Fibonacci numbers. The Zeckendorf's theorem states that any integer can be formed as the sum of Fibonacci numbers [29]. Thus, for a positive integer number n, if $d_0, d_1, ..., d_k$ represent n, then we have $n = \sum_{i=0}^{k-1} d_i F_{i+2}$ and $d_k = d_{k-1} = 1$ where F_i is the ith Fibonacci number. A Zeckendorf representation $Z(n)$ is coded by writing a binary vector with a 1 wherever that Fibonacci number is included, but F_1 is omitted due to redundancy. For example, since $19 = 13(F_7) + 5(F_5) + 1(F_2)$, it means $19 = (1\times13)+(0\times8)+(1\times5)+(0\times3)+(0\times2)+(1\times1)$ which gives $Z(19) = 101001$.

A very important property of $Z(n)$ is that two adjacent 1's never occur. Therefore, the FK_1 code produces $FK(n)$ by writing $Z(n)$ in the reverse order and appending another 1 as a terminating comma; hence, $FK(19) = 1001011$. Decoding n from $FK(n)$ is straightforward and only involves additions, making it fast for HW implementation. Using the code for signed integers is possible after *bijection*, i.e. mapping the real values in signed integers into symbols in positive values. Table I shows the FK_1 codewords of the symbols n from applying bijection to the real values x.

TABLE I: Some examples of $FK_1(n)$ for symbols from real values after bijection

x	n	$FK(n)$	x	n	$FK(n)$
0	1	11	3	6	10011
1	2	011	-3	7	01011
-1	3	0011	4	8	000011
2	4	1011	-4	9	100011
-2	5	00011	...	...	...

The FK_1 code offers several advantages [7]. First, unlike using adaptive parametrized codes, storing tables of codewords in the network nodes for packet dropping is unnecessary; hence, more efficient use of resources. Second, using two 1's as the delimiter between consecutive coded symbols gives more robustness against transmission errors than table-based codes such as the Huffman code. Third, because of the universal codewords, simply reading from lookup tables (LUTs) allows fast encoding and decoding. Fourth, the memory allocated for the LUTs is also very small (section IV). Fifth, no prefix code used also means higher efficiency.

The encoding strategy for the DCT coefficients and the block ranks is proposed as follows. First, the block ranks are encoded in the raster fashion using BWT because only four integer symbols are used, which saves around 18% than using 2-bit binary encoding. Using run lengths gives very little gain, merely around 0.01 kilobytes. The BWT is currently the best lossless compression technique, especially for text, with fast implementations available [23] such as the *bzip2* [20].

Second, the AC coefficients are processed and encoded separately from the DC. Prior to encoding, DPCM and bijection are applied to the signed coefficients. The resulting symbols for the DC coefficients are then encoded into binary string using the FK_1 code. The bitstream from the block ranks and the DC coefficients must not be dropped.

Third, the 63 AC coefficients from each block are grouped into 63 series according to their position, following the zigzag direction as used in JPEG. The symbols after bijection are encoded in parallel using the FK_1 code. The symmetry of their distribution (Fig. 6) motivates the use of bijection. The coefficients starting from the most top-left block are transmitted first in the raster fashion, and the series are sent according to their series number.

If the probability of zeros in the resulting binary string is very high, i.e. higher than 0.9, the run lengths of ones and zeros can be encoded further, for example, using the Golomb codes [9]. In all the test images, however, the probability is

only around 0.7. The bitstream from the Golomb code must be decoded before dropping. Since the reduced bitstream after dropping must be encoded again using the Golomb code, processing time at the network nodes increases.

C. Data Structure and Packet Format

The data structure also affects the coding. The proposed data structure depicted in Fig. 7 can be used to arrange the blocks, ranks and coefficients for encoding. Since all the AC coefficients of the same rank and index are grouped together, dropping them as a group when necessary is straightforward. Packet dropping is fast because checking each block rank for dropping is not needed. Moreover, since the bitstream of the group is long, more compression gain can be achieved using Golomb codes on the run lengths of the zeros. The proposed HW design and implementation of the DMP network node also obtain higher throughput with longer input bitstream. This area opens many interesting questions for future work. For this work we use fixed-length packets.

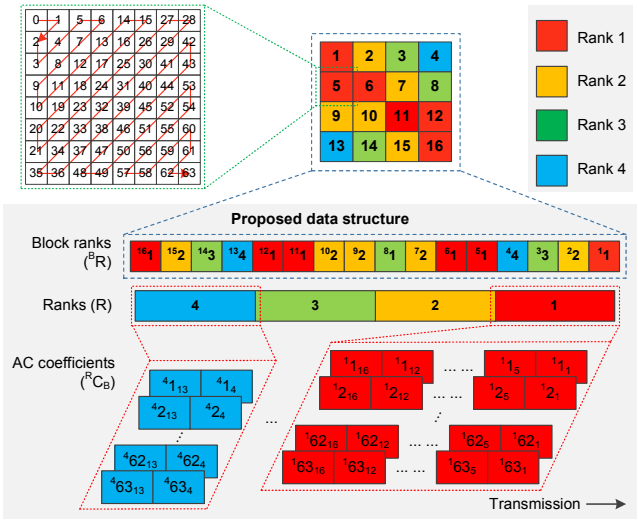


Fig. 7: Proposed data structure for packetization and transmission of the encoded blocks, ranks, and DCT coefficients.

III. Results and Discussion

Seven standard grayscale test images are used in the experiments. The results are produced using MATLAB, and the *bzip2* codec [20] is used for the BWT-based compression. All images exhibited in this section should be seen with magnification on screen for the best perceptual quality.

The first two images in Fig. 8 depict the block maps of LENA image using only entropy and that using the proposed block-ranking method. Both maps have the same blue and green blocks, but not those in the other two colors. There are more red blocks in the second image, for example on the edges and in the area of the fur. This illustrates the importance of the block variance in ranking the blocks. It produces more red blocks because they are more sensitive to visual distortion.

Fig. 8 also displays four sets of areas according to the four ranks produced by the proposed ranking method. They show the high classification accuracy of the proposed ranking

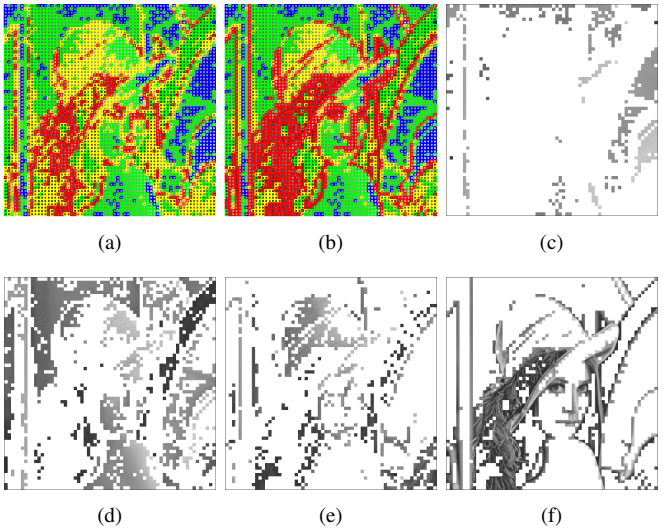


Fig. 8: The four-rank block map of LENA image using only entropy (a) and that using the proposed ranking method (b). The image is decomposed into five ranks as follows (with decreasing dropping priority): low frequency in blue (c), low-medium in green (d), medium-high in yellow (e), and high in red (f). Borders are added for better view.

method. Different techniques to classify the contents can be employed, for example using edge and texture detection to detect the edges and the textured areas. This idea, however, is not necessary because the blocks containing them can be successfully categorized as those in red by the proposed method.

The distribution of the four block ranks in the test images as shown in Fig. 9 indicates that all the rank maps of the images correspond well with human perception. It can be checked by visually comparing the distribution with the original images in Fig. 5. Rank-1 blocks are dominant in BARBARA, BOAT, GOLDHILL, and especially MANDRILL due to many textured areas present therein. In the other images, the portions of the blocks of Rank 1 and Rank 3 are almost the same because of the flat areas (Rank 3) and the edges (Rank 1).

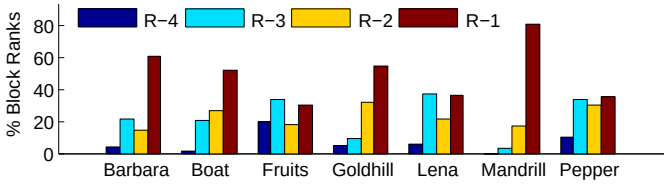
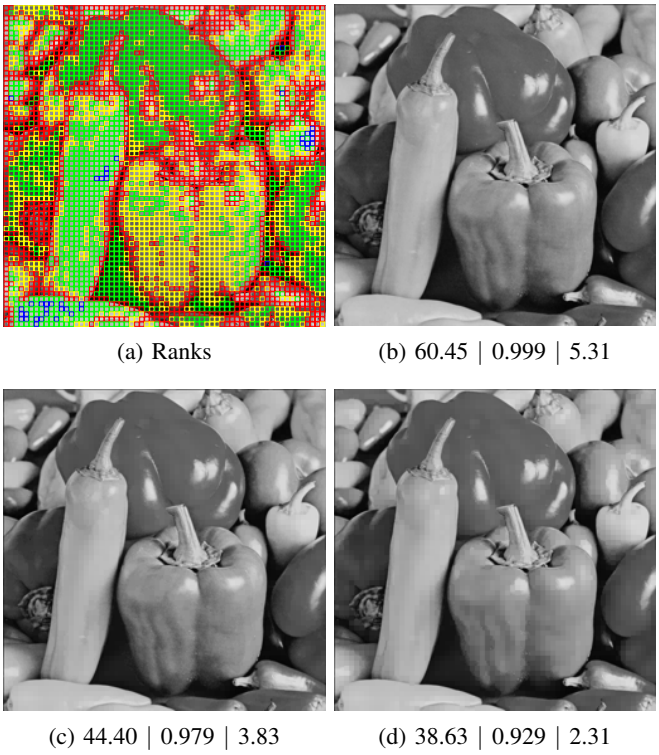


Fig. 9: The distribution of the four ranks in the test images.

Some examples of the rank map and the reconstructed images are shown in Fig. 10 from PEPPER image. Fig. 10 (b) is reconstructed from the received bitstream without the DC coefficients of Rank 4 because they have been dropped completely. When the network capacity is reduced, the nodes start dropping the AC coefficients of Rank 3 beginning from those of the 63th index. When all the AC coefficients of Rank 3 have been dropped, the resulting image quality is shown in



(a) Ranks (b) 60.45 | 0.999 | 5.31

(c) 44.40 | 0.979 | 3.83 (d) 38.63 | 0.929 | 2.31

Fig. 10: The examples from PEPPER image: the rank map with entropy and variance (a); the images reconstructed without the DCT coefficients from Rank 4 (b), from Ranks 4 and 3 (c), and from Ranks 4, 3, and 2 (d). The PSNR (dB), MSSIM and bitrate (bpp) are provided underneath.

Fig. 10 (c). The dropping is continued until the worst quality is achieved by reconstructing only from the DC coefficients of all ranks (Fig. 10 (d)). The results are accompanied with the bitrate in bits per pixel (bpp) as well as the PSNR and the mean structure-similitary index (MSSIM) [30] as the objective VQ metrics. Furthermore, the rate-distortion plots using the two quality measures for the test images are shown in Fig. 11. The plots from JPEG are also provided for comparison.

The performance of the proposed scheme can be improved by using deblocking filter as post-processing step at the receiver. An important role for estimating the bitrate, this step is conducted at the source in many image/video coding techniques. Other possible improvements include intra-prediction, which is not considered here because it creates dependencies between the adjacent blocks and increase the complexity. Note again that we do not aim at better coding performance than that of JPEG or even JPEG 2000.

Fig.10 shows that the reconstructed images suffer from blocking artifacts that occur only at the blocks which AC coefficients are dropped. The artifact, however, is different from the typical blocking artifact in JPEG because the latter occupies much larger areas consisting of many blocks. The distortion is called *pixelation artifact* due to the resemblance to it. The encoded rank maps in the side information of the bitstream plays another important function; they inform the receiver of the exact locations of the pixelated blocks. Thus those blocks can be directly restored without searching their locations as in typical deblocking algorithms.

For the worst distortion because all the AC coefficients are discarded, a fast depixelization algorithm is proposed in Algorithm 2 which refers to Fig. 12. Fig. 13 shows some examples of the depixelization for FRUIT and PEPPER images with PSNR and MSSIM values. The algorithm successfully restores the flattened blocks to be more appealing to human perception.

The blocks of Rank 2, 3 and 4 can be reconstructed only from the DC coefficients without pixelated blocks because they can be repaired fast using Algorithm 2 (Fig. 13). In fact, Rank-1 blocks with strong textures and no edges can be made free from pixelation. Nevertheless, the artifacts are still visible in Rank-1 blocks with edges even after depixelization. Repairing the pixelated edges can benefit from more advanced techniques such as in [13].

Fig. 14 (left) shows a collaborating person as a segmented object from a video frame of an HD test video sequence from [26]. It is expected to be produced by the cameras on a surface of a CS. Applying the proposed block ranking algorithm produces the blocks in Fig. 14 (right). The blocks of the background can be assigned an additional rank, for

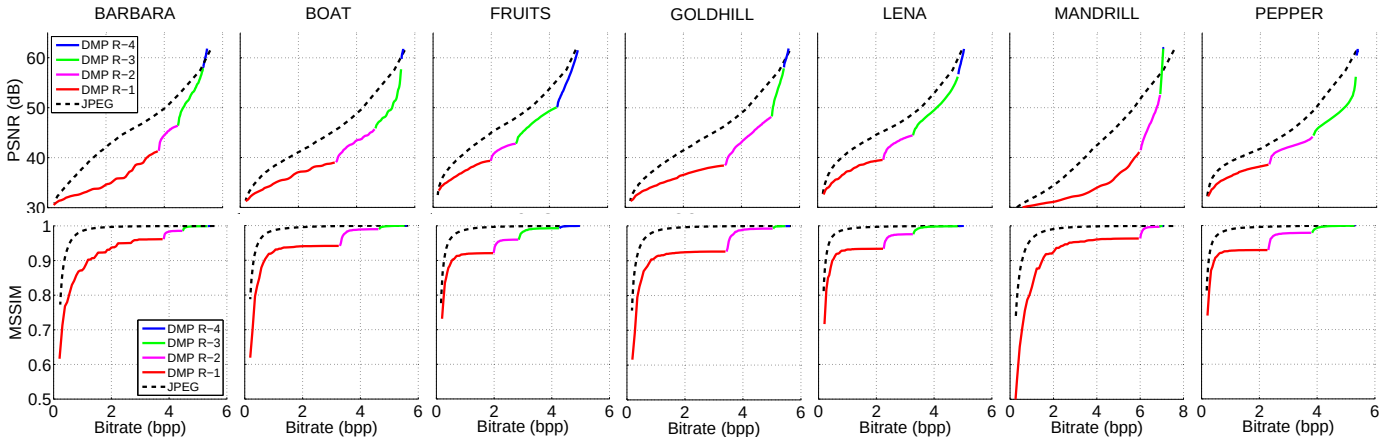


Fig. 11: The rate-distortion plots of PSNR (top) and MSSIM (bottom) against bitrate in bits per pixel (bpp).

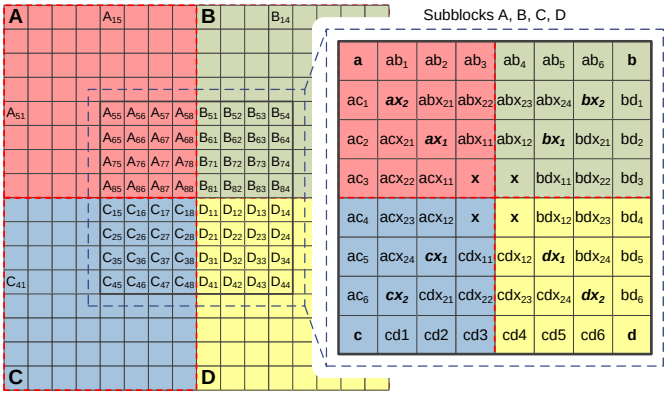


Fig. 12: The proposed depixelization in Algorithm 2.

Algorithm 2 A depixelization algorithm for the worst distortion

Reference elements of the subblocks A, B, C and D
$X \leftarrow (A_{88} + B_{81} + C_{18} + D_{11})/4$
$a \leftarrow A_{55}$, $b \leftarrow B_{54}$, $c \leftarrow C_{45}$, $d \leftarrow D_{44}$
for Block $M = \{A, B, C, D\}$ **do**
  $\{m, mx_1, mx_2, X\} \leftarrow \text{LI}(m, X, 2)$
end for

Non-reference elements of the subblocks A, B, C and D
if Rank of block $A \geq T_R$ and $C_A \leq T_C$ **then**
  if Rank of block $B \geq T_R$ and $C_B \leq T_C$ **then**
    $\{a, ab_1, ..., ab_6, b\} \leftarrow \text{LI}(a, b, 6)$
    $\{ax_2, abx_{21}, ..., abx_{24}, bx_2\} \leftarrow \text{LI}(ax_2, bx_2, 4)$
    $\{ax_1, abx_{11}, abx_{12}, bx_1\} \leftarrow \text{LI}(ax_1, bx_1, 2)$
  else if Rank of block $B \geq T_R$ and $C_B > T_C$ **then**
    $\{a, ab_1, ab_2, ab_3, B_{51}\} \leftarrow \text{LI}(a, B_{51}, 3)$
    $\{ax_2, abx_{21}, abx_{22}, B_{61}\} \leftarrow \text{LI}(ax_2, B_{61}, 2)$
    $\{ax_1, abx_{11}, B_{71}\} \leftarrow \text{LI}(ax_1, B_{71}, 1)$
  end if
end if
Run Steps 10-18 to compute the elements with suffix $ac-$
Compute the elements of subblocks B, C and D with the logic as in Steps 9-20

Terminal elements in boundary blocks such as block A
if Rank of block $A \geq T_R$ and $C_A \leq T_C$ **then**
  Non-corner elements A_{ij} (i=1:4, j=5:8; i=5:8, j=1:4)
  for $i = 1$ to 4 **do**
    $\{A_{i5}, ..., A_{i8}\} \leftarrow \{a, ab_1, ab_2, ab_3\}$
    $\{A_{5i}, ..., A_{8i}\} \leftarrow \{a, ac_1, ac_2, ac_3\}$
  end for
  Corner elements A_{ij} (i=1:4, j=1:4)
  for $i = 1$ to 4 **do**
    $\{A_{11}, A_{22}, ..., A_{55}\} \leftarrow \text{LI}(A_{11}, A_{55}, 3)$
    Compute the other non-corner elements with the logic as in Steps 15-17
  end for
end if

example, Rank 5. They contribute an insignificant increase in bits (much less than 1 Kbits) and their DCT coefficients are all discarded. This illustrates that the proposed scheme can encode regions-of-interest with arbitrary shapes.

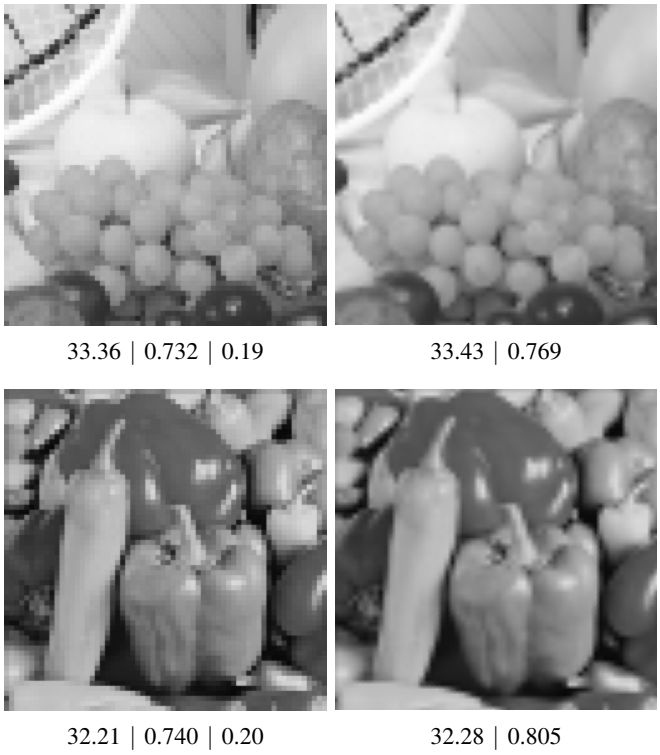


Fig. 13: Some examples of the worst distortion (left) and the improved quality after de-pixelization (right) for FRUIT (top) and PEPPER (bottom) images. The numbers denote PSNR and MSSIM, respectively.

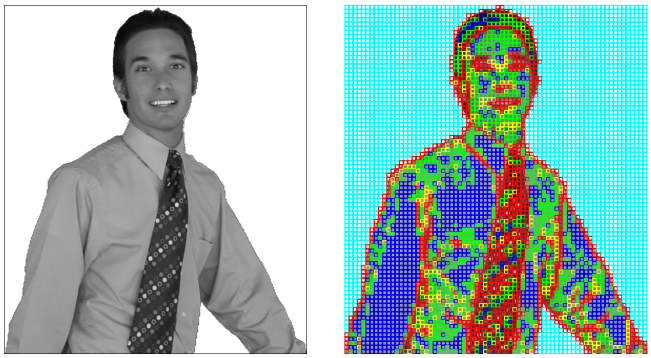


Fig. 14: An image with a segmented object as part of a video frame from a CS's surface (left). The blocks after applying the proposed block ranking algorithm (right). Image border is added for better view by readers.

IV. Algorithm Complexity and FPGA Design

We have proposed an FPGA-based platform for the design and implementation of a DMP network node [31] (Fig. 15). It provides a detailed introduction to the platform architecture and the simulation-implementation environment for the design. Our compact implementation on a Xilinx Virtex-6 ML605 board consumes very small amount of the available resources. Moreover, the elementary operations in our implementation take (much) less than 5 μs as desired to meet the low-latency

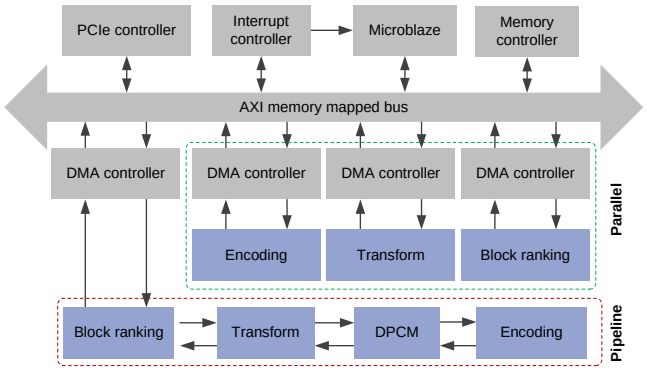


Fig. 15: The FPGA-based architecture of a DMP transmitter with the pipeline and parallel approaches.

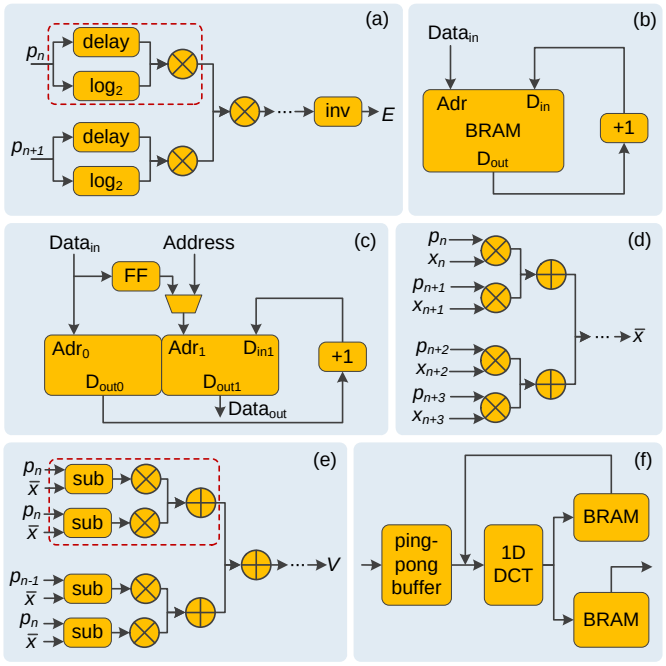


Fig. 16: The modules for calculating entropy (a), histogram (b,c), mean(d), variance (e), and 2D-DCT (f).

requirement. The AXI bus and the EDK environment are used to implement both the transmitter and receiver in DMP. Although the architecture of the access node has different number of compression-scheme components than that of the network node, their core components and adopted processing approach are the same. In addition to controlling the data flow within the FPGA system, Microblaze is also used to establish and maintain the communication with the external DMP servers located on a host (PC machine).

The design's modularity and scalability ease the integration of the external modules into the platform, which can follow parallel, pipeline or hybrid approaches. The first two approaches are depicted in Fig. 15. By assuming equal access in the memory-mapped AXI bus, the parallel approach offers flexibility because it permits software elimination in certain steps of the processing chain if necessary, e.g. the encoding. This is possible because the Microblaze governs all the execution steps of the chain, and they are independently connected to a single AXI bus. On the other hand, the pipeline implementation is more efficient provided that all the modules are used in the processing chain and the pipeline latency is not critical. Adopting both approaches in a hybrid fashion is also an alternative depending on the application.

A. Calculation of Entropy

The complexity for HW design of the major parts of the proposed system is presented as follows. Fig. 16 (a) shows the entropy module, and the dashed line covers the parallel structure. The logarithm operation can be implemented as a registered LUT for 8-bit input data at one clock (CLK) [2]. Thus, the overall latency is 7 CLK, i.e. 3 CLK for each multiplier, and given n parallel structures, it becomes $1 + 3\log(n)$ clocks. Consisting of a block RAM (BRAM) memory and an incremental logic, a histogram module with 64 input integer values provides the probability values p_i.

Fig. 16 (b) and (c) show the simplified and real diagrams, respectively. The input data for the evaluation of the histogram address the BRAM and the BRAM's D_{out} stores the count of $Data_{\text{in}}$'s prior to the occurrences at the BRAM address bus. The counter is incremented by one and written back to the BRAM at the same address. The BRAM limits the calculation speed as the output data is one CLK delayed with respect to the address bus (a synchronous memory data read); hence, evaluating a single input pixel involves two CLK. Strong parallelization of the computations is possible [12], and the histogram computation needs 34 LUTs and 23 FFs.

B. Calculation of Mean and Variance

Computing mean values of n inputs of $p_i x_i$ (Fig. 16 (d)) takes 3+log(n) CLK, and the variance-calculation module consists of the mean-calculation unit and a set of subtractors, multipliers and adders. They are strongly parallel modules which process the data every clock cycle. The parallelization determines the computation time, and generally it is 4+log(n) CLK plus the latency from the mean-calculation module.

C. Calculation of 2D-DCT, IDCT and DPCM

By employing a two-pass 1D-DCT transform [28], computing a complete 8×8 2D-DCT takes 80 clock cycles and can work at 107 MHz. Adopting a ping-pong fashion, it stores the results of the 1D-DCT by means of an intermediate buffer (Fig. 16 (f)). It is a trade-off between resource consumption and speed which complies well with the idea of an AXI-based Microblaze-controlled architecture. Nevertheless, other implementation approaches for 2D-DCT can be considered, such as replacing the time-consuming multiplications with LUT accesses [14]. As for the DPCM, its sequential execution flow favors software implementation in Microblaze, and the processing power will not be absorbed because DC coefficients are fewer than AC coefficients.

D. Encoding and Decoding

Encoding Fibonacci code is simple, but straightforward implementation in iterative procedures needs substantial clock

cycles. Therefore, it is better implemented as LUT and executed in one clock, which is feasible because Fibonacci coder for 8-bit numbers consumes merely 8×12 bits, and 3072 bits can fit into a single BRAM memory of 18 Kbits. Thus, it occupies only 2 BRAMs for both encoding and decoding. Moreover, the *bzip2* algorithm can be implemented in software in Microblaze.

E. Packet Dropping

The dropping module in Fig. 17 is the core component of the QS scheme in a DMP node. The module is integrated to the platform in Fig. 15 also via a direct memory access (DMA) controller. Our strategy is to extensively use AXIS (AXI Streaming) bus which provides system flexibility. All the modules connected to the network node are AXIS-compatible.

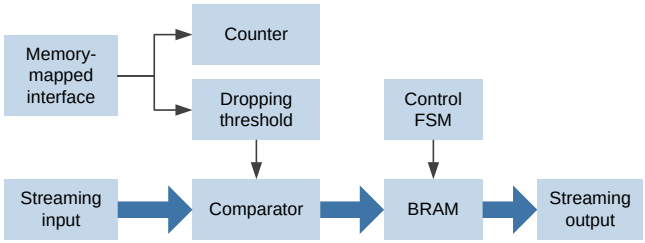


Fig. 17: The proposed structure for DMP dropping module.

The dropping module works as follows. The network packets carrying image data are sent over the PCIe to the external memory (DDR3 in ML605) and stored on a long queue. The Microblaze monitors the status of the queue, programs DMA controller to read the data from the external memory, and writes them to the dropping module. Based on the data received from the other nodes in the network, a current threshold value for dropping is computed and written to the internal register of the dropping module. The data fed into the dropping module from the external memory by the DMA are either dropped or passed through to the internal memory (BRAM) after compared with the threshold. Once the DMA write-operation is finished, the Microblaze is interrupted and informed that the dropping statistics can be read from the internal register of the dropping module. The Microblaze then programs the DMA controller based on the statistics, and the data stored in the internal memory are transferred to the external memory. The dropping process is finished when the data is read from the internal memory of the dropping module and written to the external RAM. All HW modules in both access and network nodes are interconnected with AXIS bus and controlled by the Microblaze.

F. Depixelization as Post-Processing

Depixelization is essentially a finite-impulse response (FIR) filter operation conducted on block borders. Xilinx delivers a FIR compiler tool which can be used to compile the FIR architectures to generate the depixelization filter [38]. It includes 12-bit coefficients and 60-tap input data which can work at 150 MHz and consume 3382 LUT-FF pairs.

G. Overall Performance

The system performance strictly depends on the chosen architecture (parallel, pipeline or hybrid) and the synchronization between the modules. The following is the gross estimate of the computational time for processing a video frame as a 1920×1080 color image. By assuming 256-bit AXI bus width, 64-pixel block, and 100 MHz FGPA clock cycle as the main constraints, the internal processing speed per single thread becomes 50 Mblocks/s. As the luma and 25% subsampled chroma images equal 48,600 blocks, the essential processing time becomes 1 ms per frame. Since the object-based processing reduces the number of blocks processed per frame, the processing time per frame is less than 1 ms.

The consumed resources are detailed in Table II, and the additional pre- and post-processing steps are excluded. LUTs consume the most resources which are roughly 10% of all the XCVLX240T resources, the FPGA used in ML605 [36]. It is possible to balance the usage of LUTs with DSP implementation to equalize resource consumption. Consequently, 15 to 20 parallel processing streams can be implemented as in Fig. 15 which reduce the processing time to approximately 50 μs per frame. Moreover, several FPGA boards can be used as a one-stop system [16] to achieve chip-level parallelism.

TABLE II: Total consumption of resources

Module	#LUT	#FF	#BRAM
Entropy	$m{\cdot}\log(m){\cdot}115 + n \cdot 19$	$110 + n \cdot 64$	0
Histogram	34	23	1
Variance	$n \cdot 115 + [n + n{\cdot}\log(n)] \cdot 8$	$n \cdot 110$	0
DCT	123	110	2
Fibonacci	0	0	2

m denotes the number of parallel inputs for entropy module

V. Conclusion and Future Work

We have presented an ultrafast, DCT-based, embedded image-compression scheme which is quality scalable and can process objects with arbitrary shapes. It is designed for network architecture such as DMP that guarantees maximum EED. The encoder mainly consists of block ranking, 2D-DCT and entropy coding. As the quantization as in existing image coding schemes is not present, the main loss of information in this approach is due to the intelligent dropping of data packets by network nodes during transmission to guarantee local delay. Since simplicity and parallelization are favored for minimizing processing time, block entropy and variance are used for block ranking, which work satisfactorily by yielding four ranks of 8×8-blocks with increasing importance. Universal codes are employed to encode the resulting block ranks and DCT coefficients. The VQ in PSNR and MSSIM of several common test images due to dropping is given against the bitrates, which are also compared to the results from JPEG. JPEG performs better as expected because it can exploit global redundancies in an image and the bitstream, but lacks the scalability. Fundamental differences between the two schemes make such a performance comparison essentially irrelevant. Excessive dropping results in pixelation artifacts that are faithfully contained in the blocks which the receiver can immediately locate from the side information available in the

packet headers of the remaining bitstream. A depixelization algorithm, a post-processing step at the receiver, is proposed for the worst distortion. We show how the scheme can be applied to objects of arbitrary non-rectangular areas in images after segmentation. Every video frame, channel, segmented object, and block are processed independently, allowing fully parallel HW implementation. Finally, as indicated by the estimated complexity and resource consumption of the proposed scheme for FPGA implementation, a video frame as a 1920×1080 color image can be processed, encoded and decoded in less than 1ms, sufficient to meet the maximum EED at 11.5ms. Ideas for further performance improvement include incorporating fast intra-prediction and advanced depixelization.

REFERENCES

[1] N. Ahmed, T. Natarajan, R. K. Rao, "Discrete cosine transform," *IEEE Trans. Computers*, vol. 23, no. 1, pp. 90–93, 1974.

[2] N. Alachiotis, A. Stamatakis, "Efficient floating-point logarithm unit for FPGAs," in *Proc. IEEE Int'l Sym. Parallel & Distributed Processing, Workshops and PhD Forum (IPDPSW)*, 2010.

[3] M. Burrows, D. Wheeler, *A Block Sorting Lossless Data Compression Algorithm*, Technical Report 124, Digital Equipment Corporation, 1994.

[4] C. Chafe, M. Gurevich, G. Leslie, S. Tyan, "Effect of time delay on ensemble accuracy," in *Proc. Int'l Symp. Music Acoustics*, 2004.

[5] T.A. DeFanti, D. Acevedo, R.A. Ainsworth, M.D. Brown, S. Cutchin, G. Dawe, K.U. Doerr, A. Johnson, C. Knox, R. Kooima, F. Kuester, J. Leigh, L. Long, P. Otto, V. Petrovic, K. Ponto, A. Prudhomme, R. Rao, L. Renambot, D.J. Sandin, J.P. Schulze, L. Smarr, M. Srinivasan, P. Weber, G. Wickham, "The future of the CAVE," *Central European J. Eng.*, vol. 1, no. 1, pp. 16–37, 2011.

[6] Y. Du, J. Wang, S.-M. Guo, P.D. Thouin, "Survey and comparative analysis of entropy and relative entropy thresholding techniques," *IEE Proc. - Vision, Image and Signal Processing*, vol. 153, no. 6, pp. 837–850, 2006.

[7] P. Fenwick, Universal Codes, in K. Sayood (ed.), *Lossless Compression Handbook*, Academic Press, 2003.

[8] A.S. Fraenkel, S.T. Klein, "Robust universal complete codes for transmission and compression," *Discrete Applied Mathematics*, vol. 64, no. 1, pp. 31–55, 1996.

[9] S. W. Golomb, "Run-length encodings," *IEEE Trans. Information Theory*, vol. 12, no. 3, pp. 399–400, 1966.

[10] P. Holub, J. Matela, M. Pulec, M. Šrom, "Ultragrid: low-latency high-quality video transmissions on commodity hardware," in *Proc. ACM Multimedia*, 2012, pp. 1457–1460.

[11] *Advanced Video Coding for Generic Audio-Visual Services*, ITU-T Rec. H.264 and ISO/IEC 14496-10 (AVC), ITU-T and ISO/IEC JTC 1, May 2003 (and subsequent editions).

[12] E. Jamro, M. Wielgosz, K. Wiatr, "FPGA implementaton of strongly parallel histogram equalization," in *Proc. IEEE Design and Diagnostics of Electronic Circuits and Systems (DDECS)*, 2007, pp. 1–6.

[13] J. Kopf, D. Lischinski, "Depixelizing pixel art," *ACM Trans. Graphics*, vol.30, no. 4, pp. 99:1–99:8, 2011.

[14] R. Kutka, "Fast computation of DCT by statistic adapted look-up tables," in *Proc. IEEE Int'l Conf. Multimedia and Expo (ICME)*, 2002, pp. 781–784.

[15] J-R. Ohm, "Advances in scalable video coding," *Proc. IEEE*, vol. 93, vol. 1, pp. 42–56, 2005.

[16] One Stop Systems, http://www.onestopsystems.com/

[17] L. A. Rønningen, *The Distributed Multimedia Plays Architecture (version 3.20)*, Technical Report, ITEM, NTNU, 2011.

[18] L.A. Rønningen, O.J. Wittner, "Experiments on remote conducting between Trondheim and Lisbon," ITEM, NTNU, 2011.

[19] H. Schwarz, D. Marpe, T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1103–1120, 2007.

[20] J. Seward, bzip2 codec, http://www.bzip.org/

[21] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[22] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Processing*, vol. 41, no. 12, pp. 3445–3462, 1993.

[23] D. Solomon, G. Motta, *Handbook of Data Compression*, Springer, 2010.

[24] G. Sorwar, A. Abraham, L.S. Dooley, "Texture classification based on DCT and soft computing," in *Proc. 10th IEEE Int'l Conf. Fuzzy Systems*, 2001.

[25] H. Sun, A. Vetro, J. Xin, "An overview of scalable video streaming," *Wireless Communications and Mobile Computing*, vol. 7, pp. 159–172, 2007.

[26] TGFX. http://www.timelinegfx.com/.

[27] R. Tucker, "The role of optics and electronics in high-capacity routers," *J. Lightwave Tech.*, vol. 24, no. 12, pp. 4655–4673, 2006.

[28] A. Tumeo, M. Monchiero, G. Palermo, F. Ferrandi, D. Sciuto, "A pipelined fast 2D-DCT accelerator for FPGA-based SoCs," in *Proc. IEEE Computer Society Annual Symp. VLSI*, 2007, pp. 331–336.

[29] S. Vajda, *Fibonacci and Lucas Numbers, and the Golden Section Theory and Applications*, Ellis Horwood, Chichester, 1989.

[30] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Processing*, vol. 13, pp. 600–612, 2004.

[31] M. Wielgosz, M. Panggabean, J. Wang, L. A. Rønningen, "An FPGA-based platform for a network architecture with delay guarantee," *J. Circuits, Systems and Computers*, vol. 22, no. 06, 2013.

[32] P. Schelkens, A. Skodras, T. Ebrahimi, *The JPEG 2000 Suite*, Wiley, Series: Wiley-IS&T Series in Imaging Science and Technology 2009.

[33] D. Taubman, "High performance scalable image compression with EBCOT," *IEEE Trans. Image Processing*, vol. 9, no. 7, pp. 1158–1170, 2000.

[34] D. Taubman, M. Marcellin, *JPEG 2000: Image Compression Fundamentals, Standards and Practice*, Kluwer Academic Publishers, 2001.

[35] R.J. Van der Vleuten, R.P. Kleihorst, C. Hentschel, "Low-compexity scalable DCT image compression," in *Proc. IEEE Int'l Conf. Image Processing*, 2000, pp. 837–840.

[36] Virtex series, http://www.xilinx.com/publications/matrix/Virtex_Series.pdf

[37] D. Wu, Y. Hou, Y-Q. Zhang, "Transporting real-time video over the Internet: challenges and approaches," *Proc. IEEE*, vol. 88, no. 12, pp. 1855–1875, 2000.

[38] Xilinx, http://www.xilinx.com/support/documentation/ip_documentation/fir_compiler_ds534.pdf

[39] x264, http://www.videolan.org/developers/x264.html

[40] Z. Yang, B. Yu, W. Wu, K. Nahrstedt, R. Diankov, R. Bajscy, "A study of collaborative dancing in tele-immersive environments," in *Proc. 8th IEEE Int'l Symp. Multimedia*, 2006, pp. 177–184.

[41] J. Zhang, T. Tan, "Brief review of invariant texture analysis methods," *Pattern Recognition*, vol. 35, no. 3, pp. 735–747, 2002.