Chroma Prediction for Low-Complexity Distributed Video Encoding

Kai Langen
Department of Computer Science
University of Saskatchewan
Saskatoon, SK, CANADA, S7N 3C9
Email: kai.langen@usask.ca

Dwight Makaroff
Department of Computer Science
University of Saskatchewan
Saskatoon, SK, CANADA, S7N 3C9
Email: makaroff@usask.ca

Ketan Mayer-Patel
Department of Computer Science
University of North Carolina
Chapel Hill, NC, USA 27599–3175
Email: kmp@cs.unc.edu

***Abstract*—Inter-frame encoding is an integral technology in modern video codec design, used to achieve high compression efficiency by reducing temporal redundancy. An important step in inter-frame encoding is the temporal prediction loop, involving computationally expensive block-matching algorithms. When the video server is concerned with customer Quality of Service for on-demand or live-streaming, the encoder is the appropriate location for most of the compression computation to enable low client decoding time.**

However, emerging use-cases such as Wireless Video Sensor Networks (WVSNs) and remote surveillance present new technical challenges in compression. For these scenarios, the video capture device may be power-constrained with limited resources, and it is better that the computational burden fall on resource-rich decoders/transcoders available at data centres.

In this paper, we present a video coding scheme with low encoding complexity that passes the computational burden to the decoder. Bit-rate reduction is achieved by removing Chroma information from specific frames and then re-colouring them at the decoder using a technique called Motion Compensated Recolouring (MCR). MCR can reduce bit-rate by as much 16% compared to Intra-only video coding, with only a slight drop in objective quality. We also demonstrate that MCR out-performs other Chroma prediction techniques at higher quality and for videos with complex motion.

***Index Terms*—colour reproduction, video coding, wireless video sensor networks**

I. Introduction

Conventional predictive video codecs, such as H.264/AVC (Advanced Video Coding) address a *downlink* model of video communication, where a sequence is encoded at a high-powered encoder device and broadcast to a number of decoders [1]. The downlink model benefits from an architecture where the majority of computation occurs at the encoder, which is typically 10 times more complex than its matching decoder [2]. More modern codecs, such as H.265/HEVC (High Efficiency Video Coding) provide further advances in compression efficiency, offering 50% bit-rate reduction for the same perceptual quality as H.264. This improved rate-distortion performance arrives at the cost increased computational demand on the encoder, such that H.265 encoders are expected to be several times more complex than those of H.264 [3].

There are emerging applications, however, where encoder resources are scarce. Some examples of *uplink* model applications include wireless video sensor networks, Internet-of-Things (IoT) video, agricultural monitoring, and capsule endoscopy. Conventional video codecs are ill-matched to this class of applications; instead, new video coding solutions must be considered that can balance encoder and decoder complexity based on the system's available hardware and power resources. A major source of encoder complexity is the inter-frame prediction loop, which can constitute up to 70% of encoding time [4]. Intra-frame coding requires less computation, but cannot achieve the same levels of Rate-Distortion (RD) efficiency as inter-frame coding. Uplink-model codecs move this inter-frame prediction to the decoder to reduce encoder complexity while still improving RD efficiency over Intra-only coding schemes.

In this paper, we present an uplink-model codec that uses motion-based video recolouring to lower bit-rate. The encoder separates frames into a series of "Colour-Groups-of-Pictures" (CGOPs), consisting of a single full-colour video frame followed by N "grayscale" frames with the Chroma removed. At the decoder, missing Chroma is estimated using a bidirectional motion compensation technique. Our video recolouring codec lowers bit-rate by reducing the quantity of Chroma coefficients transmitted, but incurs only a minor decrease in the video's average Peak-Signal-to-Noise-Ratio (PSNR). Furthermore, we propose a new Chroma estimation technique called Motion Compensated Recolouring (MCR), which improves upon prior, related work.

The rest of this paper is organized as follows. Section II describes existing research in the field and contrasts it against our own design. Section III presents our proposed implementation. Section IV outlines the experimental parameters and metrics used to evaluate the resulting system. Section V presents our findings, and finally, Section VI concludes with the implications of using video recolouring with MCR, along with an outline for future work.

II. Related Work

A. Distributed Video Coding

Distributed Video Coding (DVC) is one field of research that addresses the *uplink* model of video coding. Similar to our own work, DVC codecs seek to move the computational complexity from encoder to decoder. The DISCOVER Architecture and

its descendants represent the leading DVC codecs [1], [2], [5], [6]; they operate by splitting video frames into two groups: key-frames, and "Wyner-Ziv" (WZ) frames. Key-frames are encoded using conventional Intra-coding techniques and transmitted to the decoder first. Intra coding for these frames is generally performed using a standard video codec such as JM H.264 [5].

The remaining WZ-frames are coded with a separate pipeline. First, a frame residual is created by subtracting the current WZ-frame by one or more key-frames. The residuals are then channel encoded and the result is stored at the encoder. These channel codes, also referred to as "error-correction codes", are used in telecommunication to improve transmission reliability over a noisy communication channel [7].

The decoder does not receive WZ-frames directly, instead it receives a subset of the channel codes for each WZ-frame; a prediction of the WZ-frame must be generated by applying motion-compensation to its neighbouring key-frames. These predictions are treated as a "noisy" representation of the true frame data, and the motion-estimation model as a "virtual dependency channel" [8]. Channel codes are then used at the decoder to correct this channel noise and recover the original WZ-frames.

The channel decoding process is slow and iterative. If the frame prediction is not accurate enough to reconstruct the original frame from the channel codes available, additional codes can be requested from the encoder via a feedback loop. The frame estimate is then iteratively refined; more channel codes are requested until the WZ-frame is fully reconstructed.

Requesting additional channel codes increases both the total transmission rate and decoding time of the video. More accurate initial predictions result in fewer channel code requests, and so research has focused on producing high-quality frame predictions at the decoder. The leading prediction technique is called Motion Compensated Interpolation (MCI), depicted in Figure 1.

MCI can use both key-frames and previously-decoded WZ-frames as references in its prediction model. The algorithm begins with forward Motion-Estimation (ME) between the previous and future reference frames, X_{n-1} and X_{n+1}, to create a set of initial motion vectors. Bi-directional ME is then used to refine the motion vectors from the previous step, with the added constraints that each selected motion-vector follows a linear trajectory between X_{n-1} and X_{n+1} and passes closest to the centre of the blocks in the interpolated frame [9]. Next, a spatial motion smoothing filter is applied to the resulting motion field to increase the spatial coherence between each motion vector and its neighbours [10], [11]. Finally, a weighted bi-directional Motion Compensation (MC) is applied to the motion-field to create the frame prediction, or "Side Information".

B. Chroma Estimation

A major influence on our work is the "Colour Frame Reproduction" codec of Hasan *et al.* [12]. To our knowledge, this

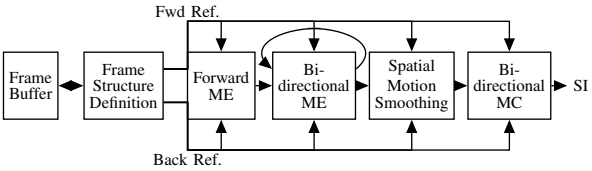


Fig. 1: Motion Compensated Interpolation

is the first codec to use Chroma trimming at the encoder and motion-based recolouring at the decoder to improve compression efficiency. In their decoding scheme, motion vectors are calculated between the coincident Luma in the current frame and the Luma in the previous colour frame. These motion vectors are then applied to motion-compensate the Chroma planes and recolour the current frame. A skip block technique is also performed to reduce computation at the decoder. For each block candidate, the Absolute Mean Difference (AMD) is calculated using the following equation:

$$AMD = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} |x_i - y_j|, \tag{1}$$

where x_i and y_i represent the pixels in current and previous frame Luma blocks, respectively, and n is the total number of pixels per block. If the result falls below a given threshold, the block is considered a "Skip" block and the previous Chroma values are used directly without motion compensation.

While this codec provides a good proof-of-concept for motion-based video recolouring, there is room for further improvement. For instance, the Hasan codec only considered the YUV444 video format, where all colour planes are in the same resolution. However, most modern video codecs use YUV420 format, where Chroma resolution is reduced by half in both the vertical and horizontal directions [13], [14]. This down-sampling removes a significant amount of redundant Chroma information, which would reduce the effectiveness of the Hasan recolouring technique.

A more complex prediction structure could also help to improve the codec. Motion-vectors in the Hasan scheme are always computed from the frame that directly precedes the current grayscale frame. This motion model is simplistic, using only a single reference frame, and does a poor job at predicting periodic motion, lighting changes, and occluded background regions [15]. As well, when the previous frame's Chroma was itself predicted with video recolouring, this can result in a phenomenon called "drift error propagation" [16], where distortion in a reference frame leads to degradation in the quality of future frames. Without error-correction, drift error can reduce video quality when the CGOP is large. The MCR codec avoids drift error by using only the original full-coloured key-frames as references in its motion-prediction model.

III. Implementation

A high-level block diagram of the proposed video recolouring framework is shown in Figure 2. At the encoder, frames

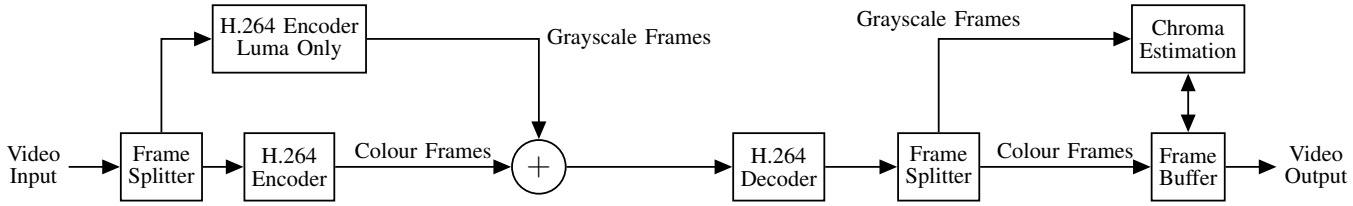


Fig. 2: Recolouring Codec

are Intra-coded, and Chroma coefficients are removed from a subset of the coded frames, reducing the total bit-rate. Video frames are ordered such that each colour frame is followed by one or more grayscale frames before the next colour key-frame in the sequence. Three codecs have been implemented following this framework:

1) **Modified Hasan** - an implementation of the Hasan *et al.* recolouring codec [12], modified to encode YUV420 input videos.
2) **MCI** - the Motion Compensated Interpolation technique used in the DISCOVER architecture [9], implemented using OpenDVC [6], an open-source version of the DISCOVER codec. We modified this design to interpolate Chroma planes rather than Luma.
3) **MCR** - a new Motion Compensated Recolouring scheme implemented that combines aspects of the first two strategies. Like the Hasan codec, MCR performs ME in the Luma plane and applies this motion field to the Chroma plane, however, many of the prediction strategies used within this codec were borrowed from the MCI design. As such, this design was also developed using OpenDVC as a reference.

For each codec, Intra coding is done using an H.264 video codec, namely, the JM 18.5 reference codec, a commonly-used benchmark in video coding research.[1] As explained in Section I, more modern codecs such as H.265/HEVC (High Efficiency Video Coding) increase the encoder complexity, which contradicts our goal of optimizing RD efficiency for low-complexity encoding.

The encoder is the same for all three codecs; the difference lies in the decoding process, where different algorithms are applied to estimate the Chroma planes and recolour the grayscale frames. The following subsections describe each of the three codecs, including the design decisions and assumptions that were made.

A. Hasan Codec Implementation

In the Hasan codec, ME occurs in the Luma plane while motion compensation is performed in the Chroma planes. The original use-case relied on all three colour channels having the same resolution, however, our application is complicated by Chroma planes having a quarter the resolution of Luma, which means that motion vector precision differs between the estimation and compensation steps. A translation scheme is required to use the vectors at a smaller resolution.

[1] http://iphome.hhi.de/suehring/tml/

In keeping with the original codec's goal of both fast encoding and decoding, a simple scheme was chosen to minimize computation time. Motion vectors calculated from the Luma plane were halved and truncated, allowing them to be used at the lower resolution. This relationship is given by the following equation:

$$mvX_C = \lfloor \frac{mvX_L}{2} \rfloor \tag{2}$$

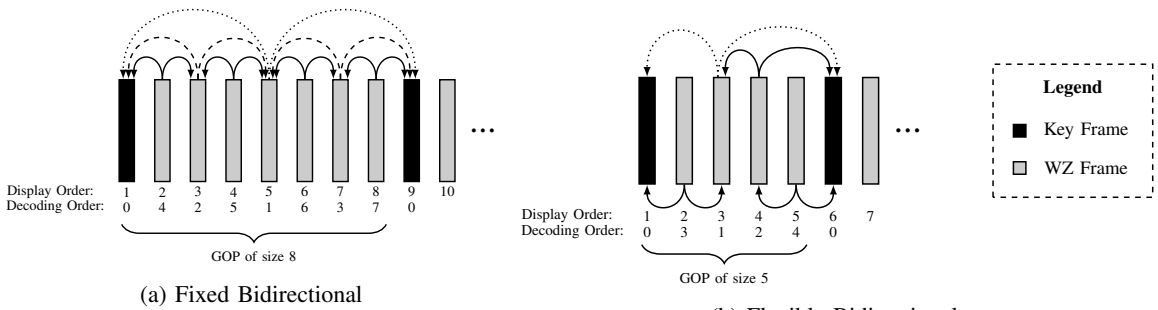
$$mvY_C = \lfloor \frac{mvY_L}{2} \rfloor, \tag{3}$$

where mvX_C and mvY_C are horizontal and vertical offsets of a Chroma block and mvX_L and mvY_L are the offsets of the corresponding Luma block. As in the original design, our implementation used the Three Step Search (TSS) algorithm [17] to perform block-motion search. TSS is a computationally efficient block-motion search algorithm that uses a coarse-to-fine search strategy. Unlike Exhaustive Search, TSS is not guaranteed to find the optimal block match, however, it significantly reduces the computational complexity. Finally, an AMD threshold calculation was included to label each Macroblock as either a "Motion" or "Skip" block.

B. MCI Modified for Chroma Prediction

MCI was originally designed for use in a DVC codec, to interpolate a missing Luma frame at the decoder. We have modified this design to interpolate missing Chroma information, using two separate MCI pipelines: one for the C_b and C_r Chroma planes. These MCI pipelines follow the same design shown in Figure 1, and motion fields for both colour planes are calculated and applied independently. Interpolation occurs as if the coincident Luma planes are absent at the decoder, and no Luma information is used to aid in motion-estimation.

While the actual DISCOVER codec allows for a flexible CGOP size (Figure 3b) [9], OpenDVC has been designed with a fixed bidirectional prediction structure (Figure 3a), such that only CGOPs of power two can be selected. We maintain this same fixed prediction structure in our MCI recolouring scheme. Similar to the Hasan codec, the bidirectional prediction structure used in MCI has the potential to introduce drift error, because frames recoloured earlier in the CGOP are referenced by future frames. A true DVC codec will prevent this drift error through error-correction codes. However, in our video recolouring use-case where no error correction is used, drift error will tend to worsen for larger CGOP sizes.



(a) Fixed Bidirectional

(b) Flexible Bidirectional

Fig. 3: Prediction Structures

C. Motion Compensation Recolouring

Like the Hasan codec, our MCR codec computes all motion vectors in the Luma plane, and applies motion compensation to the Chroma planes. At the same time, algorithms from MCI have been introduced to improve RD efficiency. Ones example is the Spatial Motion Smoothing (SMS) module, which is applied after ME. Generally, ME is performed separately for each block without considering the motion of neighbouring blocks. These neighbours often contain parts of the same object and may have correlated motion. In SMS, a weighted vector median filter is applied to each neighbourhood of a block, with weights determined by each block's matching success [10]. This filtering increases spatial coherence, and helps remove motion outliers [11].

Another technique added to MCR is Bi-Directional Motion Estimation and Compensation (BDME / BDMC). BDME uses multiple references frames, which improves prediction accuracy compared to a single-reference frame model [15]. MCR uses two references frames, the adjacent full-colour frames on either side of the CGOP, referred to as "key-frames". Weighted BDMC is used to combine the motion-vector candidates from these two reference frames to create the final prediction result.

MCR also up-samples the Chroma planes so that their resolution matches that of the Luma plane prior to motion compensation. This enables the BDMC to use the motion vectors directly, without any scaling or truncation. MCR is described below in Algorithm 1, and is depicted as a block-diagram Figure 4.

Algorithm 1 MCR Recolouring Algorithm

$Key_0 \leftarrow$ initial key-frame
$PrevKey_L \leftarrow Key_0$ Luma
$PrevKey_C \leftarrow$ upsample(Key_0 Chroma)
for CGOP in video **do**
  $NextKey \leftarrow$ next key-frame
  $NextKey_L \leftarrow NextKey$ Luma
  $NextKey_C \leftarrow$ upsample(Key_N Chroma)
  for Luma frame, $Curr_L$, in CGOP **do**
    $MV_P \leftarrow$ ME($PrevKey_L$, $Curr_L$)
    $MV_P \leftarrow$ SMS($PrevKey_L$, $Curr_L$, MV_P)
    $MV_N \leftarrow$ ME($NextKey_L$, $Curr_L$)
    $MV_N \leftarrow$ SMS($NextKey_L$, $Curr_L$, MV_N)
    $Curr_C \leftarrow$ WeightedBDMC ($PrevKey_C$, $NextKey_C$, MV_P, MV_N)
    Add $Curr_C$ to frame
  end for
  $PrevKey_L \leftarrow NextKey_L$
  $PrevKey_C \leftarrow NextKey_C$
end for

IV. Experimental Setup

For our evaluation, five standard video benchmarks were used from the Xiph.org Video Test Media collection:[2] "Akiyo", "Flower", "Mobile", "Foreman", and "Football". A custom video source was also used, depicting a time-lapse video of a Canola test plot ("Canola). The resolution of all test videos was 352x288 or Common Intermediate Format (CIF), sampled at a rate of 30 frames/s.

To ensure that this data-set covered a range of content and motion-levels, we analyzed each video according to two metrics: Spatial Perceptual Information (SI) and Temporal Perceptual Information (TI), defined in the International Telecommunication Union (ITU) Recommendation P.910 [18]. SI is calculated by applying a Sobel filter to the Luma plane of each frame (F_n). Next, the standard deviation across all filtered Luma pixels is computed, and the max result across the entire time series is returned as the value for SI: $SI = max_{time}\{std_{space}\,[Sobel(F_n)]\}$.

TI is based on the frame delta, acquired by subtracting pixels in the current frame, F_n, from those in the previous frame, F_{n-1}. The value for TI is gained by computing the standard deviation across all pixels in each difference frame, and returning the maximum result across the video sequence: $TI = max_{time}\{std_{space}[F_n(i,j) - F_{n-1}(i,j)]\}$.

Higher SI values mean that a video is more spatially complex, while higher TI indicates that there is a greater degree of motion in the video. The SI and TI values of all

[2]https://media.xiph.org/video/derf/

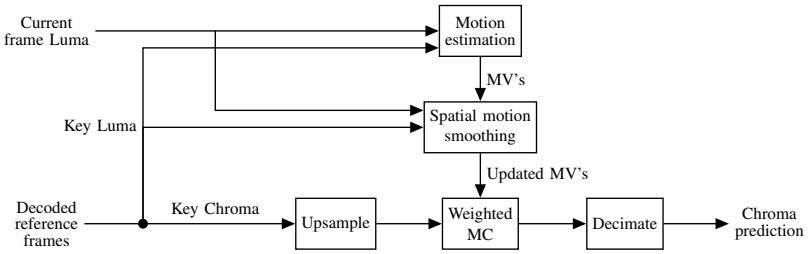


Fig. 4: MCR Chroma Prediction Strategy

selected videos are depicted in Table I.

Video	SI	TI
Akiyo	66.9	5.9
Canola	86.9	22.8
Foreman	89.7	36.3
Football	124.3	40.2
Flower	161.6	39.2
Mobile	173.7	33.0

TABLE I: Spatial and Temporal Information Measurements

Two parameters were used during the recolouring analysis, the JM I-frame Quantization Parameter (QP), and the recolouring codec Group-of-Picture size (CGOP). First, each video was Intra-coded using the JM reference codec. Rate distortion graphs were generated by varying the I-frame QP over the range 22-34 (even). For each of these QP values, a "key-frame video" was also created by setting the "FrameSkip" parameter equal to $CGOP-1$. Both the full-colour video and key-frame video were fed into the video recolouring framework, and each of the presented techniques was used to recolour the remaining non-key-frames. Finally, the Luma from the baseline Intra video and the recoloured Chroma were multiplexed together to create the final videos.

Two metrics were used to compare the different Chroma estimation schemes: Peak-Signal-to-Noise-Ratio (PSNR) and average bit-rate. Intra PSNR was obtained directly from the output statistics file generated by the JM codec, while PSNR for both planes of the recoloured Chroma was calculated via the standard equation:

$$PSNR = 10 \cdot log_{10}(\frac{255^2}{MSE}), \quad (4)$$

where MSE is the average Mean Squared Error. A single, combined PSNR value was used to represent all three colour channel, and was calculated as a 6:1:1 weighted average of Luma and Chroma PSNR, following convention [19], [20]. This equation is given below:

$$PSNR_{Avg} = \frac{6 \cdot PSNR_L + PSNR_{Cb} + PSNR_{Cr}}{8}, \quad (5)$$

where $PSNR_L$, $PSNR_{Cb}$, and $PSNR_{Cr}$ are the corresponding PSNR values for each of the YC_bC_r planes.

Rather than altering the encoded bit-stream, we instead used a mathematical model to determine the effect of removing Chroma coefficients at the encoder. Our intent with this paper is to compare the efficacy of different motion-based Chroma estimation techniques, and so implementation effort was invested into designing the decoder and re-colouring algorithms, rather than modifying or redesigning the JM codec for now.

All relevant bit-rate information was obtained from the JM codec's encoding statistics file, "stats.dat". This file includes both the total bit-rate as well as a breakdown of the bit coefficients by component (Luma or Chroma). The following equation was derived to estimate the average bit-rate that could be achieved using a recolouring codec for a given CGOP:

$$BR_{Avg} = (BR_{Tot} - BR_C \cdot \frac{CGOP-1}{CGOP}) \cdot FR, \quad (6)$$

where BR_{Avg} is the calculated bits-per-second, BR_{Tot} is the average total bits-per-frame, BR_C is the average Chroma bits-per-frame, and FR is the frames per second.

For each video, CGOP, and QP, we calculated the theoretical maximum rate-distortion efficiency that could be achieved using video recolouring. This theoretical max assumes all of the improvements to bit-rate that arise from trimming Chroma, but maintains the same PSNR as the original Intra-coded video. This provides a useful upper-bound on the potential for video recolouring as a technique.

V. Evaluation

A. Decoding Time

During recolouring analysis, the average decoding times per frame were measured across all videos and CGOP sizes. The results are depicted as bar graphs in Figure 5, with standard deviation shown as vertical error lines. From these error lines, it is clear that the decoding time of each recolouring scheme does not vary significantly depending on the video. The Hasan codec is orders of magnitude faster than the other two recolouring schemes due to its efficient block matching algorithm and "Skip block" technique. MCI is also faster than MCR because its motion estimation is done at a smaller resolution (MCI motion vectors are calculated in the Chroma plane rather than Luma plane).

B. Chroma Estimation Methods

The RD performance of the three recolouring schemes were compared against the Intra coded baseline as well as the Theoretical Max (T_{max}). Figures 7 and 8 depict these RD graphs, dividing the videos into low and high spatial

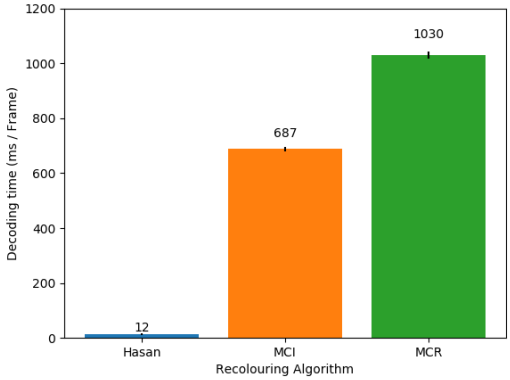


Fig. 5: Decoding / recolouring time across all videos

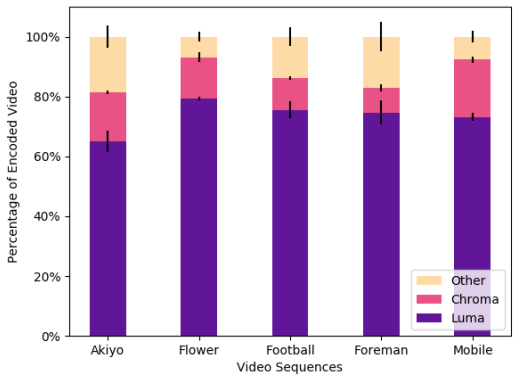


Fig. 6: Proportion of video bit-rate by category

complexity groupings. The T_{max} curve represents the highest RD improvement possible using video recolouring, and is highly dependent on the video, CGOP, and QP. These T_{max} curves show that recolouring offers greater potential rate savings when CGOP and video quality is high.

For many videos, MCR approaches the RD efficiency of the T_{max} curves, especially for smaller CGOP. MCR out-performs the Hasan codec in almost every instance. The difference in prediction quality is particularly apparent for Flower and Mobile, where MCR's bidirectional motion estimation enables it to better predict objects that are occluded or enter from out-of-frame. MCR and MCI have nearly identical performance for many of the videos, though MCR sees a slight increase in prediction accuracy at higher video quality and larger CGOP sizes. With the Foreman video in particular, we see that the MCR suffers less degradation as CGOP increases. MCR's better performance in these instances is likely because it uses only the key-frames as references in its motion-estimation model, preventing drift error propagation.

From the videos observed, there is not a strong correlation between the efficacy of video recolouring and the values for SI and TI calculated in Section IV. MCR is effective for videos where SI/TI is low (Akiyo) and high (Mobile). Instead, a better predictor for the efficacy of video recolouring is the percentage of Intra encoded bit-stream taken up by Chroma coefficients. Figure 6 shows a breakdown of the components of the video frames by category averaged across QP values. Standard deviation is represented using black error lines.

The Football video has a small Chroma-to-Total (C/T) bitrate ratio, with Chroma averaging 11% of the total rate across all QP values. Thus, removing Chroma from even a high number of frames results in only a minor bit-rate improvement. MCR applied to Football with a CGOP of 16 and a QP of 22 lowers the bit-rate by 11% compared to Intra coding, but also reduces PSNR by 2.4 dB, which brings its RD-curve below that of Intra-only coding.

Alternatively the Mobile video, with an average C/T of 19%, has a much greater potential to benefit from our codec. In particular, MCR applied with a CGOP of 16 and QP of 22 results in a full 20% bit-rate reduction compared to Intra-only coding, with only a 0.65 dB drop in PSNR. The difference in prediction accuracy between Football and Mobile may also be explained by the type of motion they contain (erratic vs. smooth motion).

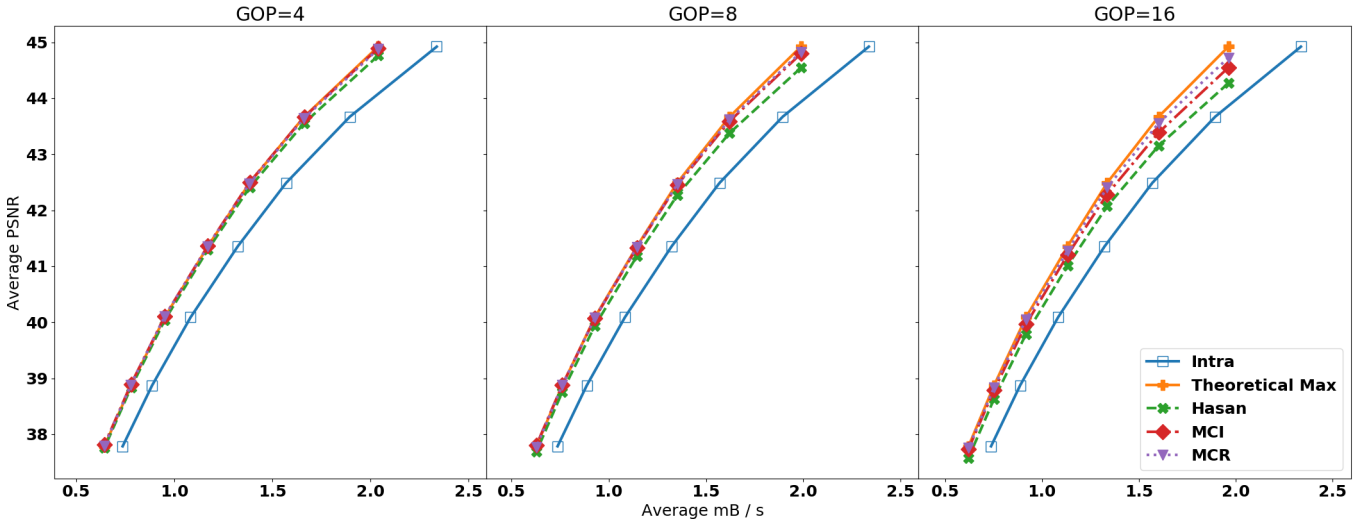
VI. Conclusions and Future Work

In this paper, we presented a video codec, MCR, that predicts Chroma information at the decoder to increase coding efficiency while keeping encoder complexity low. MCR was compared against two similar recolouring codecs, MCI and Hasan, and a baseline Intra-only coding.

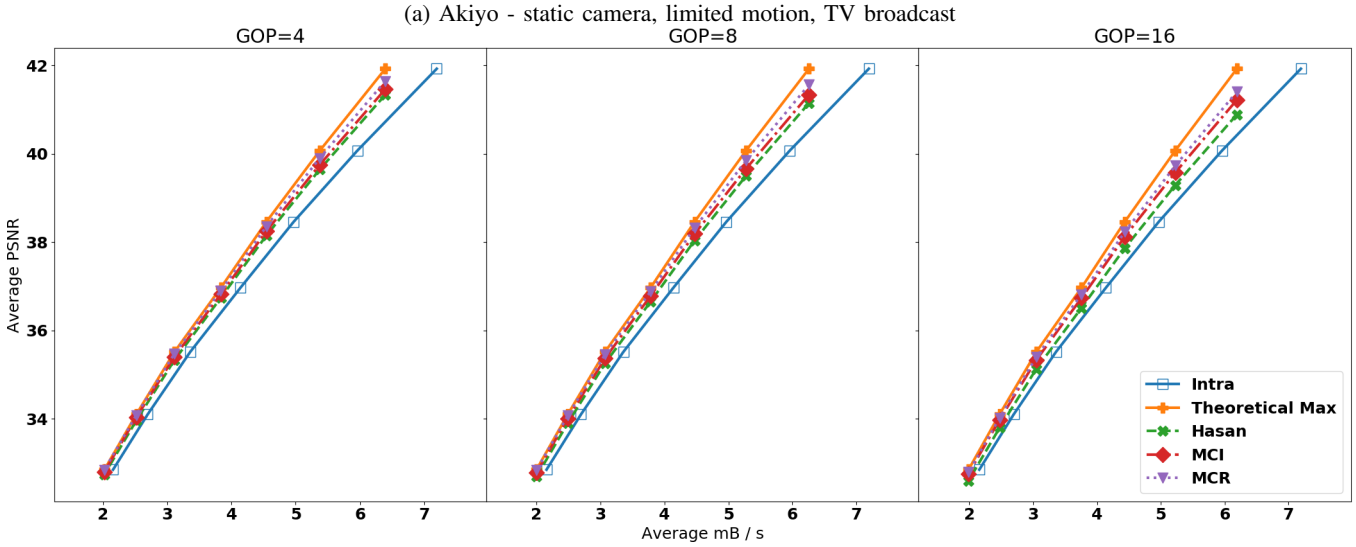
MCR outperformed the Intra-only baseline for videos with both low and high SI/TI values, though its performance proved worse for videos where Chroma made up only a small portion of the encoded video. MCR was also able to achieve better RD efficiency than the prior Hasan method for all videos, QP values, and CGOP sizes, as well as MCI at higher video quality and larger CGOP.

Compared with DVC codecs such as DISCOVER, our MCR codec offers a simple and practical solution, with the ability to produce full-colour video and no feedback loop requirements. We have also demonstrated that MCI, the frame interpolation method used in DVC, can be used effectively within a video recolouring framework.

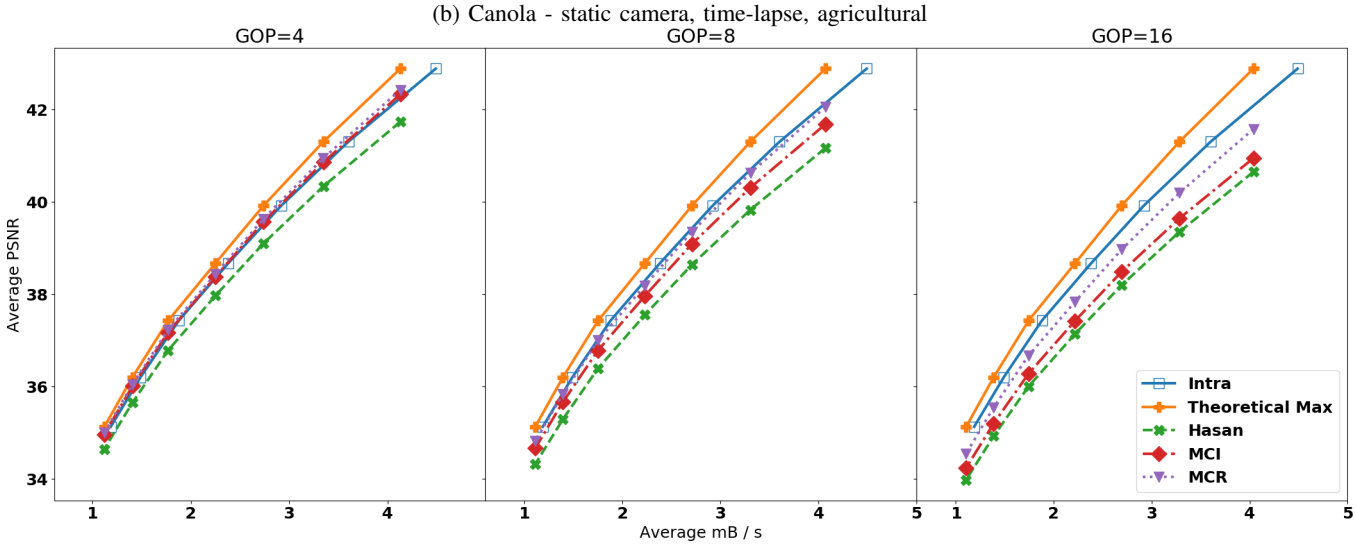
Future work will be to implement a standalone video recolouring codec that trims Chroma information from the encoded bitstream and uses the MCR at the decoder, rather than simulating its effect. Extra analysis can be performed using additional video quality metrics such as the Structural Similarity Index Measure [21] (SSIM) and a larger video dataset. Another future work will be to implement an adaptive CGOP size within this framework. From the preliminary results, we have observed a high degree of variability in rate-distortion performance across different video types and CGOP sizes. A codec that can adapt CGOP size depending on the degree of motion contained in the video would be able to take full advantage of this technique without the risk of degrading RD efficiency below that of Intra-only coding.



(a) Akiyo - static camera, limited motion, TV broadcast

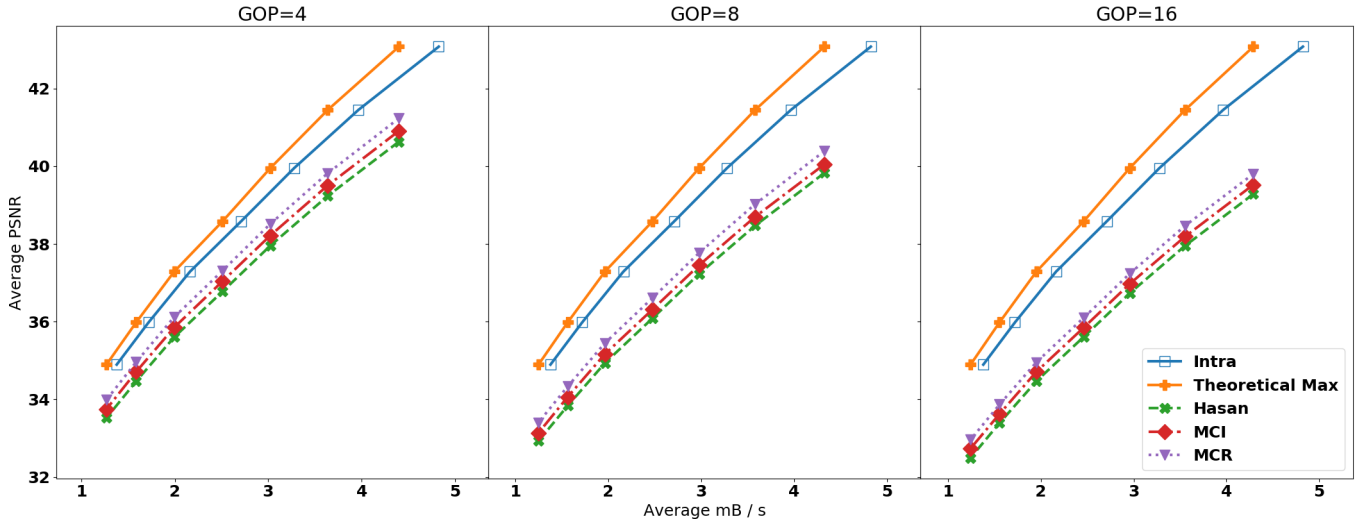


(b) Canola - static camera, time-lapse, agricultural

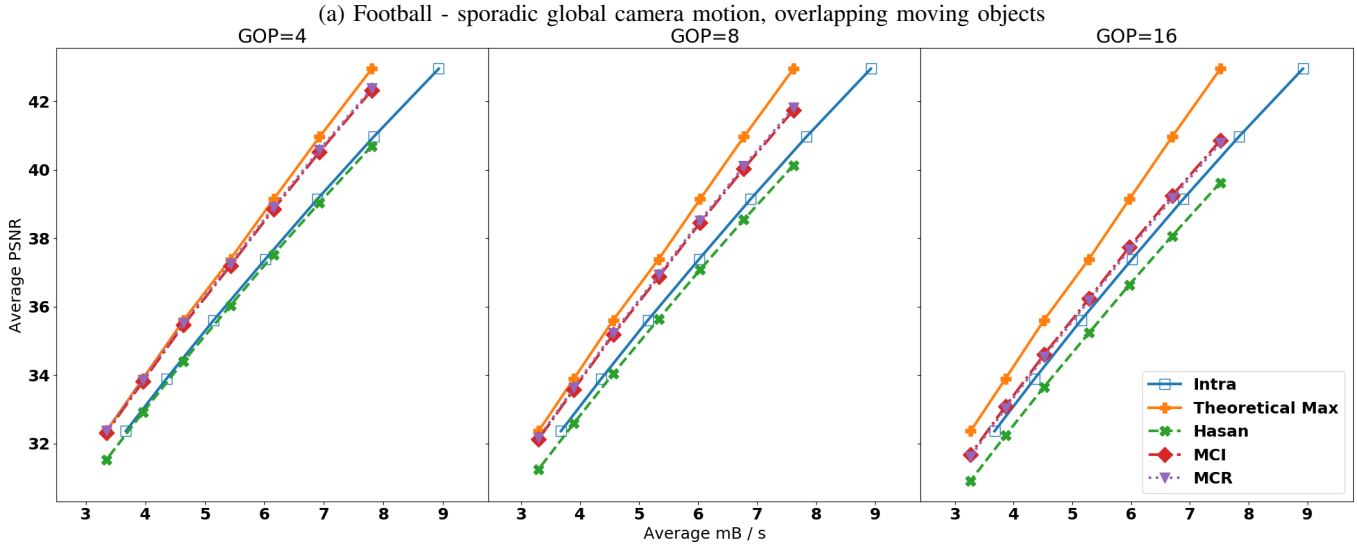


(c) Foreman - mixed static and dynamic camera motion, limited colour

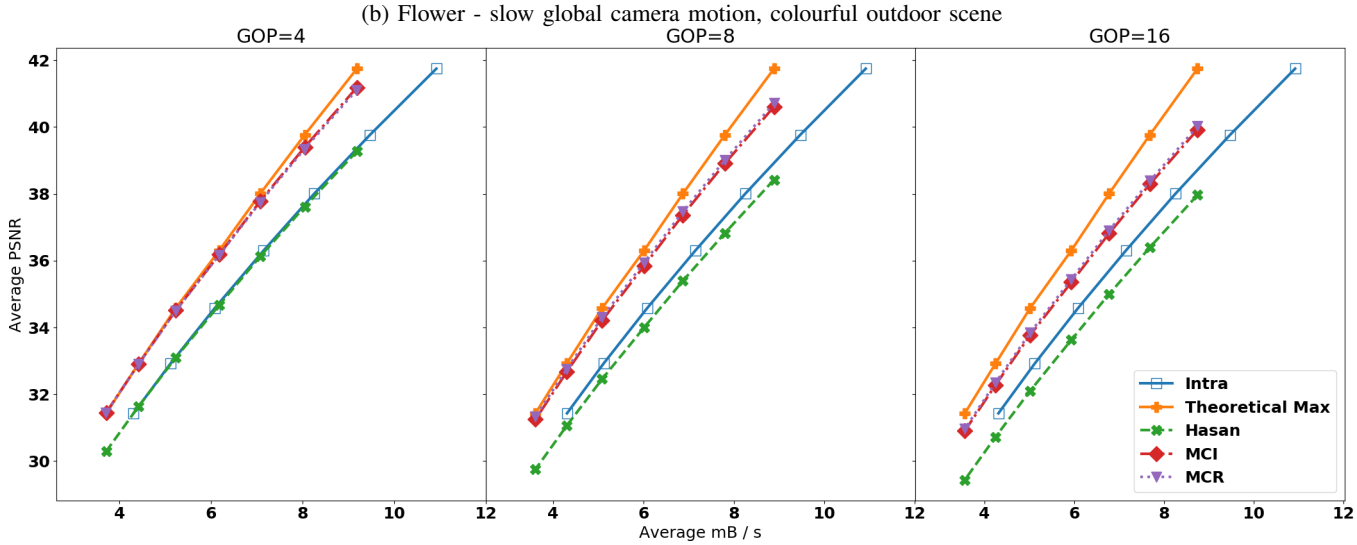
Fig. 7: Recolouring videos with low spatial complexity



(a) Football - sporadic global camera motion, overlapping moving objects



(b) Flower - slow global camera motion, colourful outoor scene



(c) Mobile - slow global camera motion, colourful indoor scene

Fig. 8: Recolouring videos with high spatial complexity

REFERENCES

[1] N. Imran, B. C. Seet, and A. C. M. Fong, "Distributed video coding for wireless video sensor networks: a review of the state-of-the-art architectures," *SpringerPlus*, vol. 4, no. 513, pp. 1–30, Sep. 2015.

[2] V. K. Kodavalla, "Challenges in practical deployment of distributed video coding," in *2016 International Conference on Microelectronics, Computing and Communications (MicroCom)*, Durgapur, India, Jan. 2016, pp. 1–6.

[3] F. Bossen, B. Bross, K. Suhring, and D. Flynn, "Hevc complexity and implementation analysis," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1685–1696, 2012.

[4] W. I. Choi, B. Jeon, and J. Jeong, "Fast motion estimation with modified diamond search for variable motion block sizes," in *IEEE International Conference on Image Processing*, Barcelona, Spain, Sep. 2003, pp. 371–374.

[5] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov, and M. Ouaret, "The DISCOVER codec: Architecture, techniques and evaluation," in *Picture Coding Symposium*, Lisboa, Portugal, Nov. 2007, pp. 1–4.

[6] S. Chien, T. Cheng, S. Ou, C. Chiu, C. Lee, V. S. Somayazulu, and Y. Chen, "Power consumption analysis for distributed video sensors in machine-to-machine networks," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 3, no. 1, pp. 55–64, Mar. 2013.

[7] S. Lin and D. J. Costello, *Error Control Coding, Second Edition*. Upper Saddle River, NJ: Prentice-Hall, Inc., 2004.

[8] R. P. Westerlaken, R. K. Gunnewiek, and R. L. Lagendijk, "The role of the virtual channel in distributed source coding of video," in *IEEE International Conference on Image Processing*, Genova, Italy, Sep. 2005, pp. I–581.

[9] P. L. Dragotti and M. Gastpar, Eds., *Distributed Source Coding: Theory, Algorithms and Applications*. Boston, MA: Academic Press, 2009.

[10] L. Alparone, M. Barni, F. Bartolini, and V. Cappellini, "Adaptively weighted vector-median filters for motion-fields smoothing," in *IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, vol. 4, Atlanta, GA, May 1996, pp. 2267–2270.

[11] J. Ascenso, C. Brites, and F. Pereira, "Improving frame interpolation with spatial motion smoothing for pixel domain distributed video coding," in *EURASIP Conference on Speech and Image Processing, Multimedia Communications and Services*, Smolenice, Slovak Republic, Jan. 2005.

[12] R. Hasan, S. K. Mohammed, A. H. Khan, and K. A. Wahid, "A color frame reproduction technique for IoT-based video surveillance application," in *2017 IEEE International Symposium on Circuits and Systems*, Baltimore, MD, May 2017, pp. 1–4.

[13] C. Poynton, *A technical introduction to digital video*. New York: J. Wiley, 1996.

[14] M. Rapczynski, P. Werner, and A. Al-Hamadi, "Effects of video encoding on camera based heart rate estimation," *IEEE Transactions on Biomedical Engineering*, vol. PP, pp. 1–1, 03 2019.

[15] A. Leontaris, P. C. Cosman, and A. M. Tourapis, "Multiple reference motion compensation: A tutorial introduction and survey," *Foundations and Trends in Signal Processing*, vol. 2, no. 4, pp. 247–364, Apr. 2009.

[16] K. Cheng, N. Uchihara, and H. Kasai, "Simple drift error-resilient H.264/AVC encoder for fast video transcoding using DCT coefficients," *Computers & Mathematics with Applications*, vol. 64, no. 5, pp. 1420–1430, 2012.

[17] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," in *IEEE National Telecommunications Conference*, New Orleans, LA, Nov. - Dec. 1981, pp. C9.6.1–9.6.5.

[18] Rec. ITU-T P.910, *Subjective video quality assessment methods for multimedia applications*, International Telecommunication Union Standard, 2008.

[19] Y. Huang, H. Qi, B. Li, and J. Xu, "Adaptive weighted distortion optimization for video coding in RGB color space," in *IEEE International Conference on Image Processing*, Paris, France, Oct. 2014, pp. 3141–3145.

[20] G. J. Sullivan and J. R. Ohm, "Meeting report of the fourth meeting of the joint collaborative team on video coding (JCT-VC)," *Joint Collaborative Team on Video Coding (JCT-VC) JCTVC-D500*, pp. 20C–28, Jan. 2011.

[21] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, Apr. 2004.