A Better Color Space Conversion Based on Learned Variances For Image Compression

Yundong Zhang*[1], Ming Li*[2], Changsheng Xia[2], Zhangming Huang[2],
Jianhua Hu[2],Dekai Chen[2], Jinwen Zan[1], Guoxin Li[1], Jing Nie[2]

[1]Vimicro AI Chip Technology Corporation
The National Key Laboratory of Digital Multimedia Chip Technology
Room607,6/F, Shining Tower, No.35 Xueyuan Road, Haidian District, Beijing 100191 China
[2]Guangdong Vimicro Microelectronics Corporation
Building 16,Hengqin Financial District, Hengqin New Area, Zhuhai City, Guangdong Province, P.R.C
li.ming@zxelec.com

Abstract

Modern image coders, especially the lossy ones, encode the YCbCr channels separately.Processing the Y channel is always much more sophisticated than the Cb/Cr. The raw image retrieved from the camera sensor is of Bayer-RGB[1] or 3-color RGB[2] format, and the conversion between RGB and YCbCr format normally follows the ITU-R BT.601[4] standard, which essentially defines a fixed 3x3 space conversion matrix with offsets. The algorithm presented in this paper, however, learns a better color space conversion algorithm tailored for each image, squeezing more information into the Y channel before encoding. In order to achieve this goal, the principle component analysis (PCA)[6] algorithm has been trained, to find the image's primary axes giving the highest variance. The PCA algorithm is carried out onto the AC values of each 16x16 pixel block (RGB values minus the block DC). During decoding, the least square method (LSM) is proposed, to estimate the optimal inverse conversion and to compensate for the coding noise. Overhead of the proposed algorithm is negligible 12 coefficients per image only, around 0.00019 bit per pixel for an image of size 2M bytes. The image after PCA conversion is coded by the latest H.266 codec running in INTRA mode, with a binary arithmetic coding engine as the entropy coder. Experiments on the CLIC2019's valid dataset has shown a significant RGB-PSNR performance boost: ***0.26db*** *or* ***7.4%*** *bitrate save@****0.145bpp****, and* ***1.2db/22.5%@1.0bpp****. The choice on Cb/Cr axis and the channel range are also studied. The proposed algorithm also outperforms the YCoCg[5] conversion algorithm, and is more robust than the YCoCg/BT.601 algorithm.*

*The first two authors share first-authorship

1. Introduction

Many advanced lossy image compression draft/standard had been published in the recent years, such as HEVC, H266, AVS 2.0, SVAC 2.0 mainly for video and JPEG, JPEG2000, BPG for still image. Image encoders based on these standards are all processing Y/Cb/Cr channels separately and using a fixed 3x3 RGB-YUV coversion matrix and 3x1 offset to convert the raw image's RGB into the required YCbCr format. The widely used covert formula comes from ITU-R BT.601, the digital version of BT.601 defines (Y is within [16,235], Cb/Cr is within [16,240]) as eq.1. We define T_{601} and $Offset_{601}$ as the 3x3 matrix and 3x1 vector in this eq:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.257 & 0.504 & 0.098 \\ -0.148 & -0.291 & 0.439 \\ 0.439 & -0.368 & -0.071 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} \tag{1}$$

Eq.1 is induced from a biological perspective that human's eyes are much more sensitive to lumina than color. Meanwhile green light contributes more to lumina than blue and red components. This observation make the Y has more weight for G and Cb for B, Cr for R.

As a result of above, modern image encoder will put more resources on the Y processing pipeline than the Cr/Cb's and Y/Cb/Cr channels' processes are independent, although recent H266 adopted some techniques such CCLM[3] to reduce the coherence between Y and CbCr components, it's still far from enough. So it would be helpful if more original image's RGB information is squeezed into the Y channel's processing, and the rest left to Cb/Cr channels. Obviously eq.1 can not guarantee this, because firstly it's fixed, not optimal for a specific image, and secondly each row of the matrix is not even orthogonal, meaning the result Y/Cb/Cr are some how coherent.

In this paper, we manage to find a optimal RGB-YCbCr

convert matrix for each image, as it's eq.2, also we define T_{enc} and $Offset_{enc}$

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} x1 & x2 & x3 \\ y1 & y2 & y3 \\ z1 & z2 & z3 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} Y\_offset \\ Cb\_offset \\ Cr\_offset \end{bmatrix} \quad (2)$$

Each row in T_{enc} is the direction of the new YCbCr space's base axis, they should be orthogonal and should ensure the resulting YCbCr are within valid range.

The first goal is to find a new primary new base axis to replace matrix's first row in eq.1. The original picture is of RGB 3-dimension. We assume each channel's entropy is proportional to their variances. To find a new axis that present the most entropy, the PCA is natural choice since it's simple and effective. Data samples are pixels' RGB minus corresponding averages. The PCA also produce the second/third base axes as well, which can be used as directions of base axes for Cb/Cr directly.

Many researches of PCA used in color-to-gray image conversion are published such as [7],[8],[9]. They are welled concentrated in improving better and faster conversion ,but few is related with the later image compression. We focus on improving image compression performance in this paper.

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = T_{enc}^{-1} \cdot (\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} - Offset_{enc}) \quad (3)$$

When decoding, the usual way to do the YCbCr-RGB conversion is simply use the inverse T_{enc} and $Offset_{enc}$ like eq.3. This reversion is good but not optimal, because YCbCr are lossy and somewhat distorted from the uncompressed version. We use LSM to estimate the optimal T_{dec} and $Offset_{dec}$, and it help to further increase the RGB-PSNR.

Further experiment are done to try to clearify:

1. Using different Cb/Cr axes instead of directly using the second/third axes from PCA, does not help.
2. Using different valid range for Y/Cb/Cr, would influence the bitstream rate, but does not help to improve bd-rate.

We will discuss these in later sections.

2. Details of our approach

2.1. Conversion Range

We assume the whole system is of 8bit quantization. R/G/B is within [0,255], Y is within [16,235] and CbCr is within [16,240]. So the eq.2 must keep the Y/Cb/Cr within the valid range after conversion. For a natural image, its R/G/B are strongly positive coherent. It always keep the elements of primary axis's vector is all positive(or all negative if revert the axis, we use the positive as default), that is $x1, x2, x3 > 0$, and let eq.4 be the restrict for Y.

$$x1 + x2 + x3 = (235 - 16)/255 \quad (4)$$

The Second/third axis derived from PCA is always orthogonal to the primary axis, since all element in the primary vector is positive, there must always be positive and negative elements in [y1,y2,y3] and [z1,z2,z3]. Second axis's range restriction should be:

$$\begin{aligned} |y1| + |y2| + |y3| &= (240 - 16)/255 \\ |z1| + |z2| + |z3| &= (240 - 16)/255 \end{aligned} \quad (5)$$

2.2. Find the primary axis using PCA

The primary axis in the new space should produce the most variance, and the converted data is as the input to Y channel pipeline. PCA data samples are as eq.6:

$$\begin{aligned} R_{sample} &= each\_pixel'R - average\_R \\ G_{sample} &= each\_pixel'G - average\_G \\ B_{sample} &= each\_pixel'B - average\_B \end{aligned} \quad (6)$$

Dimension of the sample set is [h*w, 3], h and w is height and width of the image. The normalization is dropped here because it would make R/G/B's average variance equal to 1, eliminating the variance(entropy) difference between R/G/B, which is a key information for our method.

Assuming a channel's entropy is proportional to variances is good for single-centered distributed samples, but bad for multi-centered's. It is obvious not feasible to gather our sample using R/G/B minus the whole image's average. We divide the whole image in to 16x16 grids, each sample minus the average within the same grid, not within the whole image. The division help to further improve the accuracy of the covariance matrix and is a key difference to previous works.

Define the sample as S, and $S^T \times$ S is the covariance matrix COV of size 3×3. PCA method decompose matrix COV to find eigenvalues and eigenvectors. Stack the eigenvectors along column to form transformation matrix T_{pca}.Note that T_{pca} is identity orthogonal matrix, $T_{pca}^{-1} = T_{pca}^{T}$

$$T_{pca} = \begin{bmatrix} x_p1 & x_p2 & x_p3 \\ y_p1 & y_p2 & y_p3 \\ z_p1 & z_p2 & z_p3 \end{bmatrix}$$

And finally according to YCbCr's range restrictions eq.4 and eq.5:

$$[x1, x2, x3] = L1\_normalize([x_p1, x_p2, x_p3]) * 219/255$$
$$Scale_{Cb} := 224/255/(|y_p1| + |y_p2| + |y_p3|)$$
$$[y1, y2, y3] = [y_p1, y_p2, y_p3] * Scale_{Cb}$$

The same scaling is done with the Cr axis. Offset$_{enc}$ is modified as eq.7

$$\begin{aligned} Y\_offset &= 16 \\ Cb\_offset &= -1 * sum\_neg(y1, y2, y3) * 255 + 16 \\ Cr\_offset &= -1 * sum\_neg(z1, z2, z3) * 255 + 16 \end{aligned} \quad (7)$$

sum_neg() is the function of summing up all the negative elements.

2.3. Discussion about Cb/Cr axes

As discussed above, we directly use the [y_p1 , y_p2 ,y_p3] and [z_p1 , z_p2 ,z_p3] as the new second/third axes' direction. PCA always arrange the order of axes with sorted eigenvalues from large to small. We rotate the Cb/Cr axes(0∼90 degree) in the plane P_t which is perpendicular to the Y axis, to explore the impact on the final PSNR.

While rotating, we also calculate the Cb/Cr variance, the experiment is done on some specific images(evaluation on the whole dataset is a huge task).

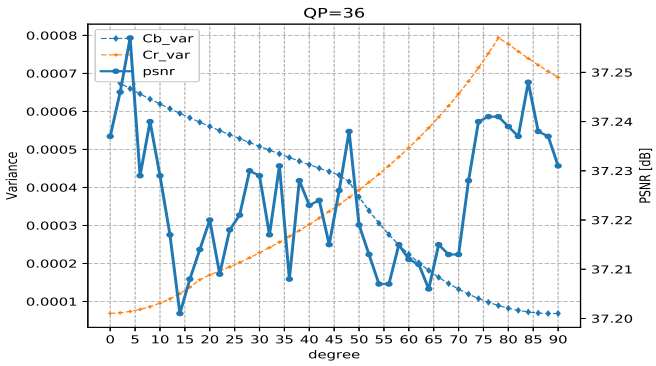

Figure 1: relation ship of psnr with rotate angle, Cb/Cr variances

A typical one can be seen on fig.1, since bitstreams' sizes result from different angles are similar, we only observe the PSNR. PSNR varies along rotate angles. A week trend is observed that PSNR is better when Cb/Cr variance difference is large. Since 0 degree is the difference max case(PCA's Properties), we don't need to rotate the Cb/Cr axes.

2.4. Discussion about valid range

As the eg.1 show that Y do not use the whole 0∼255 range. We decide to find whether performance is related to enlarging or shrinking the range. The experiment is done with different new Y ranges as [250,219,200,160,80,80] and QP[34∼38], and the result is show as fig.2

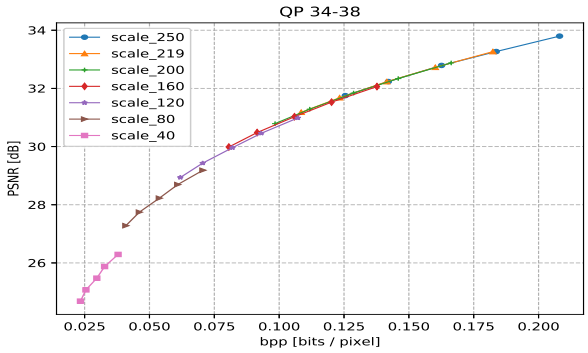

Figure 2: relationship of psnr with different range, qp

Shrinking the range would decrease the bitstream while also degrade the PSNR. At the same bpp, range 250/219/200/160 perform almost the same. So the range select do not impact the bd-rate.

2.5. Optimization of the reverse conversion

In decoding, instead of using the T_{enc}^{-1} and Offset$_{enc}$, we optimize the T_{dec} and Offset$_{dec}$ to replace the them in eq.3. But eq.3 is not a normal form for LSM, we rewrite a equivalent formula as eq.8

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = T_{dec} \cdot \begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} + Offset_{dec} \quad (8)$$

Optimizing the T_{dec} and Offset$_{dec}$ is to minimize the mean square error between eq.8's resulting RGB and the uncompressed RGB. Each row of T_{dec} and Offset$_{dec}$ are estimated within one LSM process. The T_{dec} should be very closed to T_{enc}^{-1}, but is more optimal, we can see the improvement in the experiment section.

3. Experiment result and discussion

3.1. Analysis of PSNR-boost

The latest H.266(VTM 4.0) with all tools enable(SAO,CCLM,ALF,....) is used as the backbone YCbCr codec, YCbCr 4:2:0 format is used. We wrap it up with different encoding/decoding color space conversion, Cb/Cr are scaled up and down with normal bi-cubic interpolation. To verify the method we discussed, we divide the tests in to 4 group.

1. Use the usual BT.601 conversion for coding and decoding. It is the baseline of the experiment.

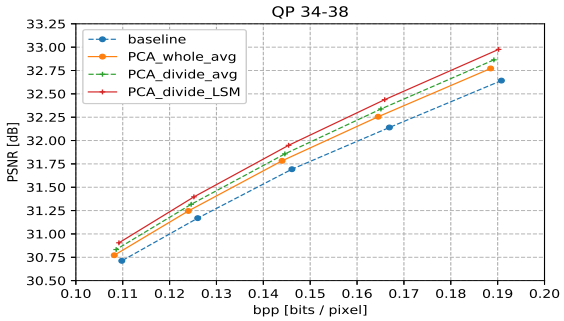

Figure 3: RGB PSNR at high QP34∼38

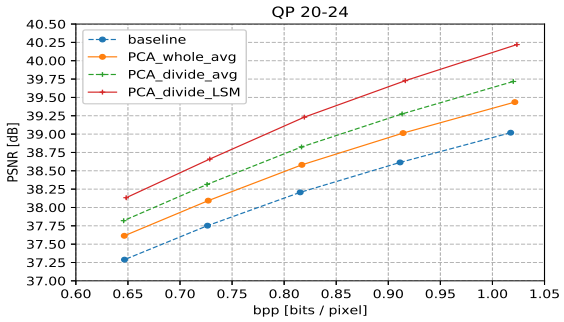

Figure 4: RGB PSNR at low QP20∼24

2. Use PCA to optimize the T_{enc} and $Offset_{enc}$, and T_{enc}^{-1} for decoding. Average RGB use whole image's
3. Same as 2, except using 16x16 grid's averages
4. Same as 3, except decoding with the optimized T_{dec} and $Offset_{dec}$.

RGB-PSNR is tested at QP 34∼38 and QP 20∼24, on the CLIC2019 valid dataset, and the bd-rate curves are shown in fig.3 and fig.4:

Group 2,3,4 show significant improvements of **-4%,-5.3%,-7.4% @0.15bpp** than the baseline, and even huge boost of **-10.0%,-16.5%,-22.5% @1.0bpp**. Obviously the proposed method performes much better at lower QP.

3.2. Compare with YCoCg

Following the suggestions from reviewers, we add these following comparison with the YCoCg[5] with 420 and 444 format. YCoCg give a much suitable primary axis estimation than BT.601 which is Y=0.25R+0.5G+0.25B and PSNR is much better than BT.601. But our method still outperforms in all situations, as show in Fig.5 and Fig.6

3.3. Superb robustness

Our method also shows superb robustness than YCoCg/BT.601. Fig.7 is an special case of a purple filtered image. It's not the natural kind but can be often seen in

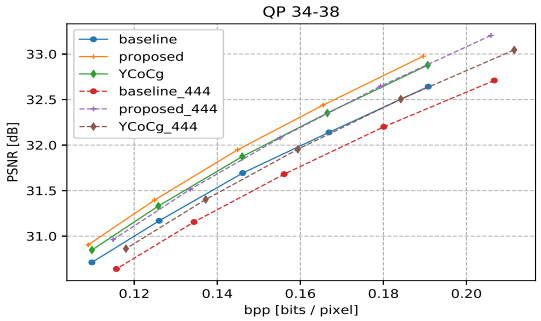

Figure 5: proposed/YCoCg/BT.601 with 420/444 format at QP34∼38

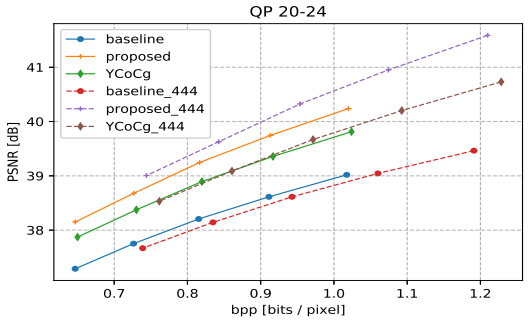

Figure 6: proposed/YCoCg/BT.601 with 420/444 format at QP20∼24

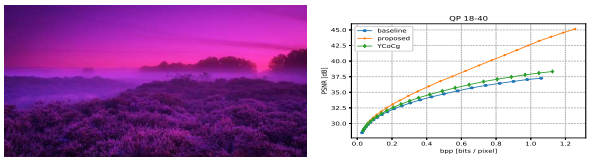

Figure 7: An image of purple style and comparison with proposed and YCoCg/BT.601

artist gallery, movies' special scene or so on. This picture's R/B are emphasized while G is surpressed which is a problem for YCoCg/BT.601 with fixed coefficients and emphasizing the G channel for converting the Y channel. We test this image with proposed/YCoCg/BT.601 with 420 format at different bitrates as Fig.7

The result shows that our method keeps increasing the RGB-PSNR constantly as the bitrate goes up, while PSNR increasing of YCoCg/BT.601 goes flat. The proposed method has over 40% performance gain than the YCoCg @1.0bpp. Our method is robust in all situations because the conversion is adaptive. The fixed conversions like YCoCg/BT.601 fail to keep constant performance in these special cases.

References

[1] Silicon Imaging *"RGB Bayer Color and MicroLenses"*

[2] Cliff Wootton *"A Practical Guide to Video and Audio Compression: From Sprockets and Rasters to Macroblocks"*, Elsevier. p. 137. ISBN 978-0-240-80630-3.

[3] Kai Zhang *"Multi-model Based Cross-component Linear Model Chroma Intra-prediction for Video Coding"*, VCIP 2017, Dec. 10 – 13, 2017, St Petersburg, U.S.A.

[4] International Telecommunication Union *"BT.601 : Studio encoding parameters of digital television for standard 4:3 and wide screen 16:9 aspect ratios"*

[5] *"YCoCg wikis"* https://en.wikipedia.org/wiki/YCoCg

[6] Jolliffe, I.T. *"Principal Component Analysis, second edition (Springer)."*,

[7] Arash Abadpour *"COLOR IMAGE PROCESSING USING PRINCIPAL COMPONENT ANALYSIS"*

[8] Ja-Won Seo *"NOVEL PCA-BASED COLOR-TO-GRAY IMAGE CONVERSION"*

[9] Esteban Vera *"Adaptive Color Space Transform using Independent Component Analysis"*, Proc Of SPIE-IS& T Electronics Imaging. 2007;6497:64970P–12.