IMPROVED DC ESTIMATION FOR JPEG COMPRESSION VIA CONVEX RELAXATION

Jianghui Zhang[1], Bin Chen[2], Yujun Huang[1], Han Qiu[3], Zhi Wang[1], Shu-Tao Xia[1]

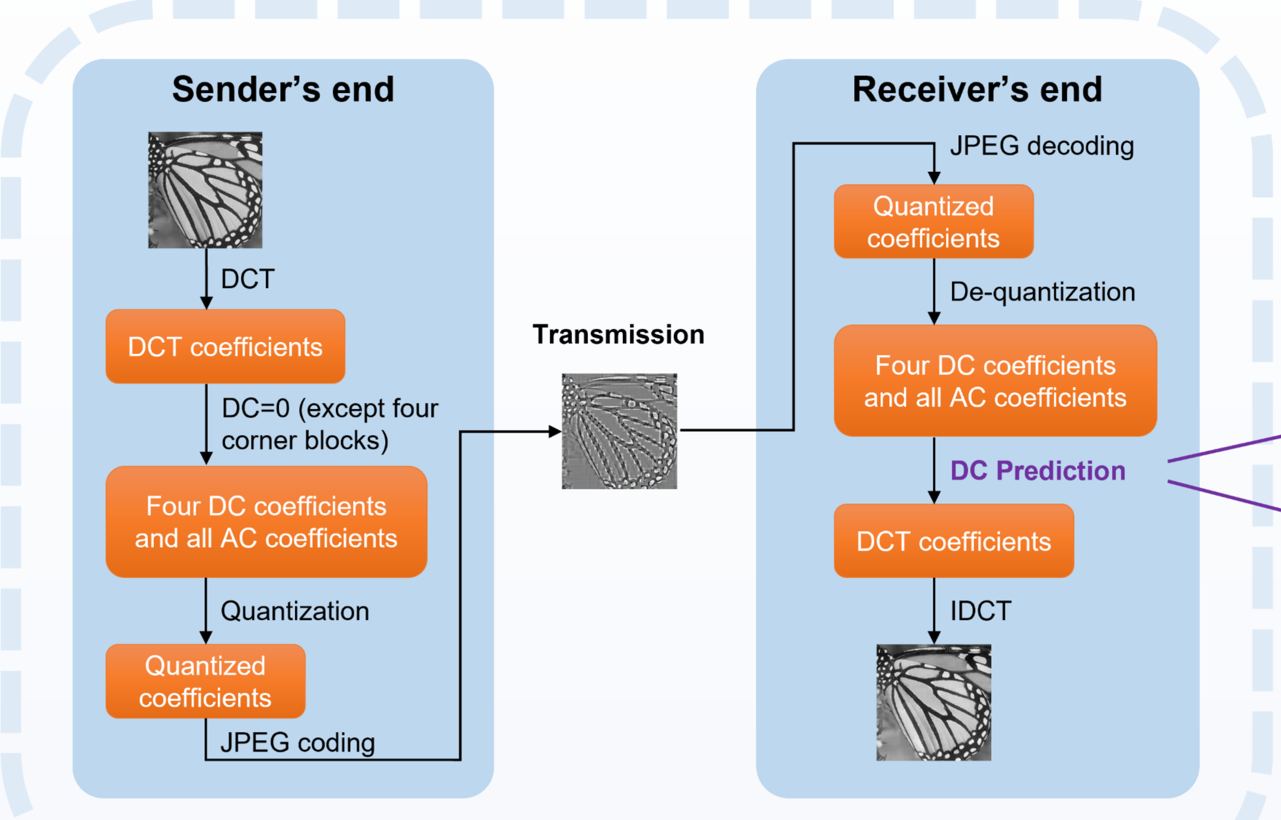
[1]Tsinghua Shenzhen International Graduate School, Tsinghua University

[2]Harbin Institute of Technology, Shenzhen [3]Tsinghua University

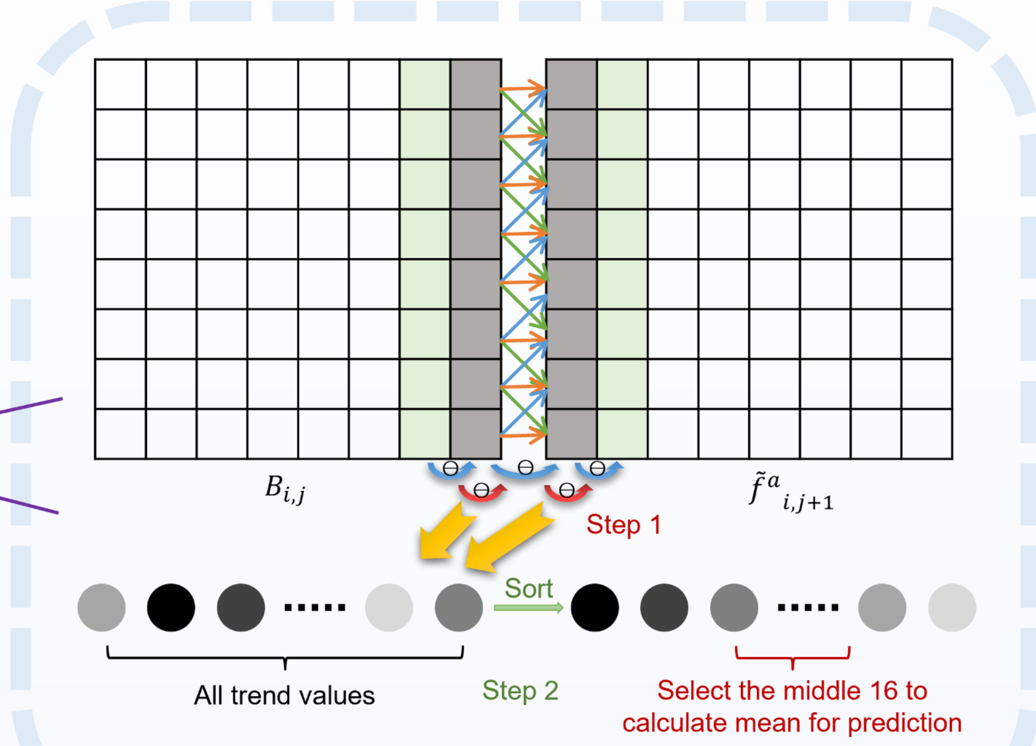
ABSTRACT

Mass image transmission has undergone an explosion of growth with the development of the internet, DCT-based lossy image compression like JPEG is pervasively conducted to save the transmission bandwidth. Recently, DCT-domain coefficient estimation approaches have been proposed to further improve the compression ratio by discarding DC coefficients at the sender's end while recovering them at the receiver's end via DC estimation. However, known DC estimation needs to enumerate all possible DC coefficients. Consequently, they are limited and resource-consuming due to the low delay requirements in real-time transmission. In this paper, we propose an improved DC estimation method via convex relaxation, which achieves state-of-the-art performance in terms of both recovery image quality and time complexity. Extensive experiments across various data sets demonstrate the advantages of our method.

ARCHITECTURE



(a) Pipeline of enhanced JPEG compression.



(b) Improved DC prediction method.

Pipeline:

1. At the sender's end, compress the image with the standard JPEG algorithm while discarding DC coefficients of all 8x8 blocks except four corner blocks.
2. At the receiver's end,

- Firstly dequantize the JPEG file into DCT coefficients.
- Then estimate all missing DC coefficients with our **DC estimation** method.
- Finally, convert DCT coefficients into RGB pixels with IDCT.

DC Estimation:

Estimate all missing DC coefficients from four corners, *i.e.*, from upper-left to bottom-right, from upper-right to bottom-left, from bottom right to upper-left and from bottom-left to bottom-right. Then drop the highest and lowest predicted value of each block and calculate the average of the remaining two values as the predicted values. For instance, the algorithm for estimating all missing DC coefficients from upper-left to bottom-right is shown as follows:

Notation	Definition
v_n	Number of 8×8 block in vertical direction
h_n	Number of 8×8 block in horizontal direction
$\tilde{f}^d_{i,j}$	DC component of block with location (i, j)
$\tilde{f}^a_{i,j}$	AC components of block with location (i, j)
$B_{i,j}$	pixel values of block with location (i, j), $B_{i,j}[x, y] = \tilde{f}^d_{i,j} + \tilde{f}^a_{i,j}[x, y]$ for $x, y \in [0, 7]$ according to equation (4)
trends	Calculate all trend values of each pixel at the adjacent boundary columns of two neighbor blocks
sorted	Sorting operation
mean	Calculate mean value
middle16	Select the middle 16 from sequence

Algorithm 1: DC component recovery from upper-left corner to bottom-right corner.

Input: AC components $\tilde{f}^a_{i,j}$ with $(i, j) \in \{h_n, v_n\}$, DC component of upper-left corner $\tilde{f}^d_{i,j}$

Output: Recovered DC component $\tilde{DC}_{i,j}$

```
for i ← 1 to h_n do
    for j ← 1 to v_n do
        Tlist_1 = sorted(trends(B_{i,j-1}, f̃^a_{i,j}))
        Tlist_2 = sorted(trends(B_{i-1,j}, f̃^a_{i,j}))
        f̃^d_{i,j} = 1/2 [mean(middle16(Tlist_1)) +
            mean(middle16(Tlist_2))]
    end
end
```

Where all trends between $B_{\{i,j\}}$ and $\tilde{f}^a_{\{i,j+1\}}$ are defined as follows:

$$T_1 = (B_{\{i,j\}}[k,6] - B_{\{i,j\}}[k,7]) - (B_{\{i,j\}}[k,7] - \tilde{f}^a_{\{i,j+1\}}[k,0])$$

$$T_2 = (B_{\{i,j\}}[k,6] - B_{\{i,j\}}[k+1,7]) - (B_{\{i,j\}}[k,7] - \tilde{f}^a_{\{i,j+1\}}[k+1,0])$$

$$T_3 = (B_{\{i,j\}}[k,6] - B_{\{i,j\}}[k-1,7]) - (B_{\{i,j\}}[k,7] - \tilde{f}^a_{\{i,j+1\}}[k-1,0])$$

$$T_4 = (B_{\{i,j\}}[k,7] - \tilde{f}^a_{\{i,j+1\}}[k,0]) - (\tilde{f}^a_{\{i,j+1\}}[k,0] - \tilde{f}^a_{\{i,j+1\}}[k,1])$$

$$T_5 = (B_{\{i,j\}}[k,7] - \tilde{f}^a_{\{i,j+1\}}[k+1,0]) - (\tilde{f}^a_{\{i,j+1\}}[k,0] - \tilde{f}^a_{\{i,j+1\}}[k+1,1])$$

$$T_6 = (B_{\{i,j\}}[k,7] - \tilde{f}^a_{\{i,j+1\}}[k-1,0]) - (\tilde{f}^a_{\{i,j+1\}}[k,0] - \tilde{f}^a_{\{i,j+1\}}[k-1,1])$$

And the calculation of $\tilde{f}^d_{\{i,j\}}$ in Algorithm 1 is based on our formulation via convex relaxation, more details refer to our paper.

EXPERIMENT

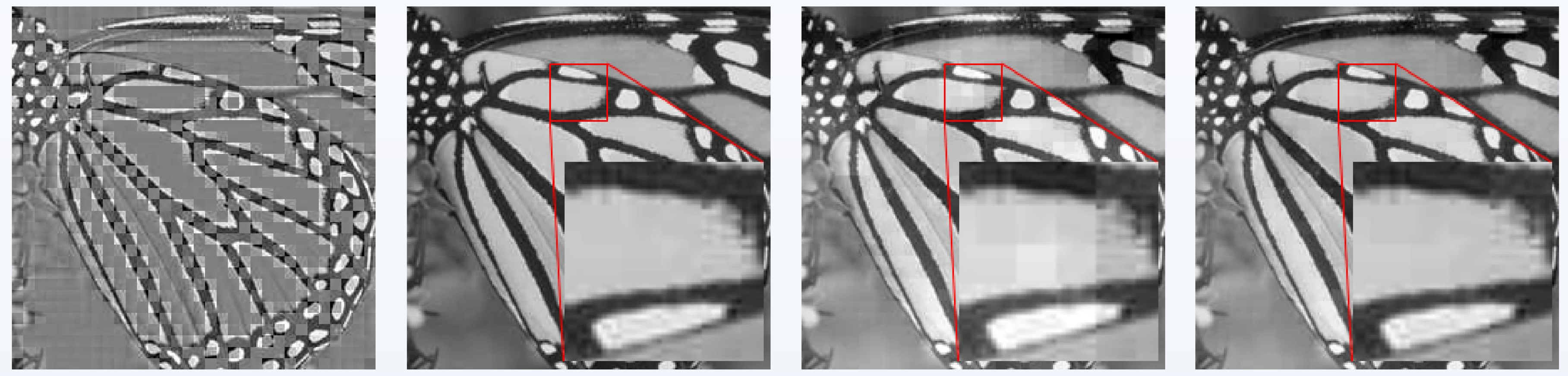
Experimental Platform: Our experiments are implemented on *Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz* with one single thread. And we use python to simulate the JPEG algorithm with standard Q50 quantization table.

Experiment Setup: To evaluate the efficacy and efficiency of our method, we compare the recovery image quality and recovery time cost on several public datasets. To be fair to compare, we compress the grayscale of the image in each dataset with the standard Q50 quantization table and then discard quantized DC coefficients of all 8 × 8 image blocks except for the four corner blocks. After that, we separately use our method and the method of [2] to recover those DC-free images. We compare two methods in terms of PSNR, SSIM, MS-SSIM and time cost.

Performance Evaluation: The results are shown as follow, where the boldface denotes results based on our method, the parenthesis denotes results based on the method in [2] and the red font denotes the increment of our method compared to the method in [2]. As shown, our proposed DC estimation method outperforms the method in [2] both in image quality and recovery time cost. The PSNR is 1.8 ~ 3.1dB higher, while the recovery time cost is only 1% of their method.

Dataset		PSNR		SSIM		MS-SSIM		Time(s)
LFW	2.5 ↑	**30.4730** (27.9956)	0.018 ↑	**0.9572** (0.9391)	0.004 ↑	**0.9853** (0.9813)	96×	**0.30** (28.87)
Set5	3.1 ↑	**26.7766** (23.7161)	0.023 ↑	**0.9387** (0.9154)	0.004 ↑	**0.9773** (0.9729)	94×	**0.59** (55.62)
Set14	2.3 ↑	**25.5460** (23.2797)	0.012 ↑	**0.9462** (0.9343)	0.010 ↑	**0.9517** (0.9413)	100×	**1.12** (112.02)
Kodak	1.8 ↑	**24.7063** (22.8620)	0.015 ↑	**0.9368** (0.9219)	0.016 ↑	**0.9208** (0.9051)	91×	**1.93** (175.49)
DIV2K	1.8 ↑	**23.5728** (21.8036)	0.022 ↑	**0.9161** (0.8942)	0.012 ↑	**0.9424** (0.9309)	93×	**14.18** (1316.04)
Urban100	2.5 ↑	**22.7492** (20.2653)	0.026 ↑	**0.9176** (0.8916)	0.022 ↑	**0.9183** (0.8968)	97×	**3.91** (378.56)
BSDS100	2.0 ↑	**24.5026** (22.5262)	0.016 ↑	**0.9456** (0.9293)	0.013 ↑	**0.9311** (0.9180)	96×	**0.78** (74.73)
BSDS200	2.2 ↑	**25.0065** (22.8125)	0.018 ↑	**0.9485** (0.9301)	0.015 ↑	**0.9303** (0.9149)	98×	**0.77** (75.14)
Manga109	3.1 ↑	**24.8771** (21.7509)	0.020 ↑	**0.9568** (0.9370)	0.018 ↑	**0.9631** (0.9456)	98×	**4.98** (489.18)

Visual Evaluation: Example for visual compare is shown as follows. (a) presents the JPEG compressed image with only four corner DC coefficients. (b) presents the original standard JPEG image. (c) presents the image recovered from (a) based on the method in [2], while (d) presents the image recovered from (a) based on our method. There are some obvious blocking effects in (c), while it is hard to tell the difference between (d) and the original standard JPEG image (b).



(a) (b) (c) (d)

Code: We release our experimental code at https://github.com/jh-zhang21/DCE.

SUMMARY

1. We accelerate the DC coefficient estimation method proposed by Qiu *et al.* [2] via convex relaxation.
2. We further propose a novel prediction pattern for DC estimation, which significantly improves the recovered image quality.
3. Compared to [2], our DC estimation method is nearly 100 times faster and PSNR of the recovered image quality is 1.8 ~ 3.1dB higher.
4. With negligible computation overhead, our method can be naturally embedded into standard JPEG to reduce the transmission bandwidth.

REFERENCES

1. T. Uehara, R. Safavi-Naini, and P. Ogunbona, "Recovering dc coefficients in block-based dct," IEEE Transactions on Image Processing, vol. 15, no. 11, pp. 3592–3596, 2006.
2. Han Qiu, Qinkai Zheng, Gerard Memmi, Jialiang Lu, Meikang Qiu, and Bhavani Thuraisingham, "Deep residual learning-based enhanced jpeg compression in the internet of things," IEEE Transactions on Industrial Informatics, vol. 17, no. 3, pp. 2124–2133, 2021.
3. Gregory K Wallace, "The jpeg still picture compression standard," IEEE transactions on consumer electronics, vol. 38, no. 1, pp. xviii–xxxiv, 1992.
4. Nasir Ahmed, T Natarajan, and Kamisetty R Rao, "Discrete cosine transform," IEEE transactions on Computers, vol. 100, no. 1, pp. 90–93, 1974.
5. Shujun Li, Junaid Jameel Ahmad, Dietmar Saupe, and C-C Jay Kuo, "An improved dc recovery method from ac coefficients of dct-transformed images," in 2010 IEEE International Conference on Image Processing. IEEE, 2010, pp. 2085–2088.
6. Shujun Li, Andreas Karrenbauer, Dietmar Saupe, and C-C Jay Kuo, "Recovering missing coefficients in dct-transformed images," in 2011 18th IEEE International Conference on Image Processing. IEEE, 2011, pp. 1537–1540.
7. SimYing Ong, Shujun Li, KokSheik Wong, and KuanYew Tan, "Fast recovery of unknown coefficients in dct-transformed images," Signal Processing: Image Communication, vol. 58, pp. 1–13, 2017.