Convolution Based Fast Lossless Compression Algorithm for Color Filter Arrays

Chiranjeev Bhaya, Manish Kumar, Chandra Mohan Velpula and Praveen Bangre Prabhakar Rao
System Intelligence Team, Samsung R&D Institute Bangalore, India
Email: (c.bhaya, manish.kmr, chandra.mv, bp.praveen)@samsung.com

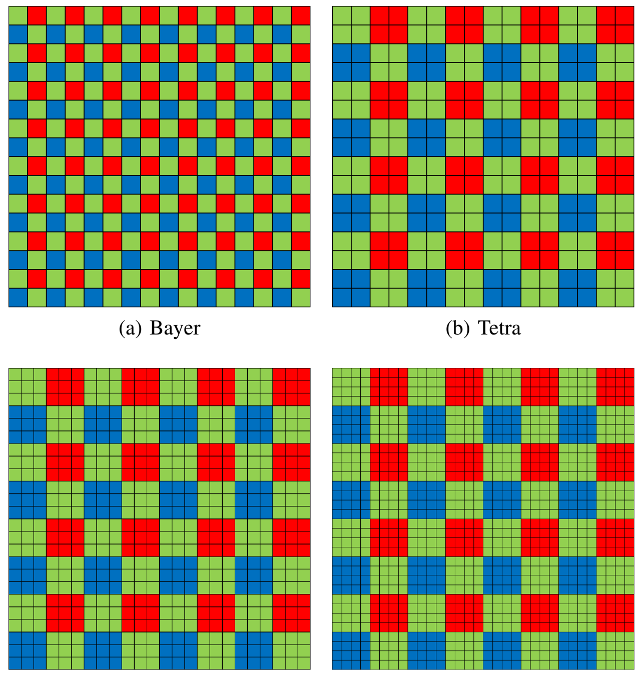
***Abstract*—Data from image sensors are acquired in the form of *Color Filter Arrays* (CFAs), which are two-dimensional matrices representing the intensity of one of the primary colors in every pixel. These CFAs are unprocessed and linear, and they undergo various image processing algorithms to generate color images. However, as the size of the sensor grows, the amount of memory required to store these CFAs increases. In this paper, a convolution-based fast lossless compression method for CFAs is proposed. In this method, every pixel value is predicted using a convolution operation based on the previously traversed pixel in raster order. The predicted value, being very close to the actual value, has a small error. This error, which has low entropy, is compressed and stored using a suitable entropy coding algorithm, such as Huffman coding. The error value is sufficient to losslessly decompress the original CFA values. Unlike conventional algorithms that perform context modeling at every pixel value, simple convolution is used in the proposed algorithm, making it fast. Experimental results show that the proposed algorithm achieves a good trade-off between time and compression ratio, making it suitable for real-time applications such as image processing pipelines.**

***Index Terms*—Color Filter Array, Convolution, Entropy Coding, Lossless Compression**

I. Introduction

In today's era, where data is growing exponentially, memory is becoming an increasingly expensive resource. Especially with the growth in the resolution of camera sensors, the memory required to store the CFAs captured by them is also increasing. For example, a 12 MP, 50 MP, 108 MP, and 200 MP sensor generates CFA frames of sizes 17.93 MB, 59.57 MB, 205.99 MB, and 381.01 MB, respectively. Moreover, with the increase in resolution, the CFA patterns are also changing. Conventional Bayer pattern CFAs [1], as shown in Fig. 1a, had alternate arrangements of the primary colors. However, modern high-resolution CFAs, such as tetra, nona, and hexadeca [2], as shown in Fig. 1b, 1c, and 1d, respectively, include more color information of the same color in adjacent pixels. The increase in the complexity of these patterns, combined with the increase in memory requirements, necessitates efficient compression algorithms that are fast and scalable to meet the demands.

Numerous conventional lossy image and CFA compression algorithms exist in the literature [3]. However, for this use case, data loss would lead to deterioration in the quality of the final image generated after post-processing the CFAs. Most lossless compression algorithms aim to reduce the entropy of the original image or CFAs through suitable context modeling algorithms, and then compress the reduced-entropy data using entropy coding, such as Huffman [4], arithmetic [5], or Golomb-Rice [6] codes. Context modeling algorithms involve calculations based on image regions, as seen in CALIC [7] and JPEG 2000 [8]. Such techniques have also been adopted in numerous lossless compression methods for CFAs, which are summarized in TABLE I.



(a) Bayer (b) Tetra (c) Nona (d) Hexadeca

Fig. 1: Different CFA patterns.

In this paper, a convolution-based fast lossless compression method for CFAs is proposed. The convolution is computed based on the premise that a pixel value is a linear combination of its neighboring pixels due to inter-pixel correlation. The prediction of pixel values is framed as an optimization problem, in which the mean-squared ($\mathcal{L}_2$) loss between the current pixel and its weighted neighbors is minimized to obtain an optimal convolution filter corresponding to different color positions in the CFAs. The obtained weights of the convolution are used in the compression algorithm to generate a predicted matrix of the CFA. The error between the predicted and original values

TABLE I: State-of-the-art lossless compression algorithms for Bayer CFAs

Method	Description	Limitations
Chung and Chan [9]	• Split the CFA into green and non-green channels. • Perform green sample prediction. • Perform non-green sample prediction using color difference optimization. • Compress the error residues using *adaptive Rice codes*.	• Prediction method is two pass - separate for green and non-green channels. • Prediction of non-green channels can be done only after green, hence cannot be parallelized. • High complexity in prediction resulting in an increase in the time complexity.
Kim and Cho [10]	• Hierarchical prediction by sub-sampling Bayer CFA into four color channels. • Encode the first green sub-sample using grayscale coding • Perform prediction on the second green subsample from the first sub sample. • Obtain red and blue sub-samples using interpolations and predictions form the green subsamples. • Encode the predicted errors using context adaptive arithmetic coding.	• Bayer image needs to be divided into four sub-images corresponding to the colors and positions. • The prediction has high complexity, and is dependent on multiple hierarchical levels, hence cannot be parallelized. • High time for computation.
Hernandez-Cabronero *et al.* [11]	• Decompose the Bayer CFA into four color sub-samples. • Compress each CFA using JPEG-lossless.	• Inter-pixel correlation among different colors are not utilized due to decomposition. • High execution time of JPEG lossless.

is compressed using entropy coding, such as *Huffman* [4]. The same convolution can be employed during decompression to losslessly recover the original CFA.

II. Proposed Method

The proposed method consists of three distinct parts. The first part involves obtaining the convolution coefficients by minimizing the $\mathcal{L}_2$ loss between the current pixel and its weighted neighbors. The second part involves using the obtained convolution for entropy reduction and compression. Finally, during decompression, the same convolution is employed to obtain the losslessly decompressed CFA.

The detailed steps of the proposed method are presented as follows.

A. Obtaining the Convolution Coefficients

CFA frames, just like images, have a high pixel intensity correlation among its neighborhoods. A pixel at a position (i, j) in a CFA I with intensity $I(i, j)$, can therefore be predicted as a linear combination of its neighborhood pixels in a local $k \times k$ kernel. The predicted value, $I'(i, j)$ can be written as

$$I'(i,j) = \sum_{i'=-\lfloor \frac{k}{2} \rfloor}^{\lfloor \frac{k}{2} \rfloor} \sum_{j'=-\lfloor \frac{k}{2} \rfloor}^{\lfloor \frac{k}{2} \rfloor} \hat{w}\left(i' + \left\lfloor \frac{k}{2} \right\rfloor, j' + \left\lfloor \frac{k}{2} \right\rfloor\right) I(i+i', j+j') \quad (1)$$

where $\hat{w}$ is a $k \times k$ matrix denoting the weight coefficients and k is taken as odd. Since the same prediction needs to be done for decompression, a well-defined ordering and traversal is required. In this algorithm, raster order traversal has been followed, therefore, the weights of the positions after the pixel of interests are not considered. Mathematically, $\hat{w}(i, j) = 0$ if $i > \lfloor k/2 \rfloor$ or $i = \lfloor k/2 \rfloor \wedge j \geq \lfloor k/2 \rfloor$, as shown in Fig. 2. In all other cases, $\hat{w}$ is the minimization of the set of weights w having the least squared error between current pixel intensity and its weighted neighbors. Mathematically,

$$\hat{w} = \arg\min_{w} \sum_{\mathbf{r}} \left(I(\mathbf{r}) - \sum_{\mathbf{s}} w_{\mathbf{rs}} \cdot I(\mathbf{s}) \right)^2 \quad (2)$$

where $\mathbf{r}$ is the position of the current pixel and $\mathbf{s}$ is the kernel neighborhood. Based on the structure of CFA pattern, different set of weight coefficients are required based on the relative position of pixel in CFA pattern. For example, as shown in the Fig. 2, three set of weight coefficients $\hat{w}_g, \hat{w}_r, \hat{w}_b$ have been determined for the Bayer CFA pattern depending on the color intensity it denotes.

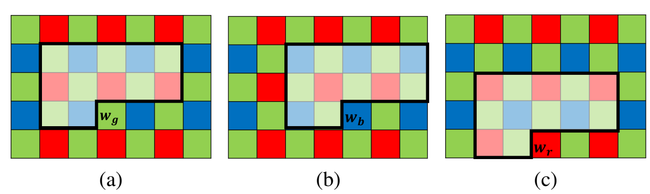


Fig. 2: Calculation of the weighted coefficients based on color positions in a Bayer CFA considering a 5×5 kernel.

B. Compression

The overview of the compression algorithm is shown in Fig. 3. Once the weight coefficients are determined, the obtained convolution filter corresponding to the pixel positions are run in raster order to obtain a predicted image I' according to Eq. 1. For border pixels where the convolution is not possible, the original CFA values are copied, *i.e.*, $I' = I$. This predicted image is very close to the original image. To enable lossless decompression, the difference between the predicted and the original image is stored. For every pixel (i, j) of the CFA, the difference $e = I(i,j) - I'(i,j)$ is converted to unsigned integer u, using the relation

$$u = \begin{cases} 2e, & e \geq 0 \\ -2e - 1, & e < 0 \end{cases} \tag{3}$$

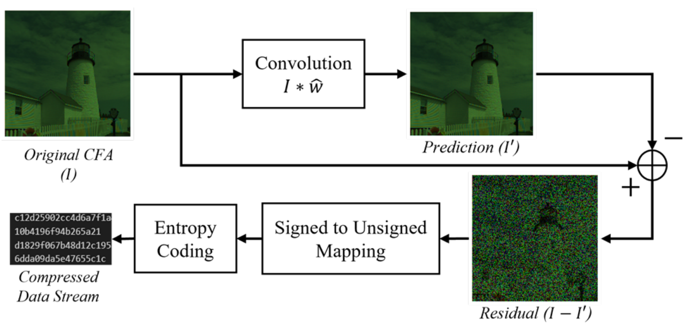


Fig. 3: Overview of compression algorithm

The resultant values have lower entropy as compared to original image data distribution. They can be efficiently compressed using suitable entropy coding algorithm. In this paper, *Huffman* coding [12] has been used for compression. The compressed values, along with the Huffman tree used for encoding are stored as compressed data stream. This occupies much lesser space than the original CFA, and can be losslessly decompressed using the decompression algorithm.

C. Decompression

The steps for decompression are reverse than that of the compression, and are shown in Fig. 4. Firstly, the residual matrix is obtained by the corresponding decompression algorithm of the entropy encoder used. This matrix contains the unsigned values u obtained from the compression algorithm, which can be converted to the corresponding signed error values e using the following equations.

$$e = \begin{cases} u/2, & u \equiv 0 (\mathrm{mod} 2) \\ -(u+1)/2, & u \equiv 1 (\mathrm{mod} 2) \end{cases} \tag{4}$$

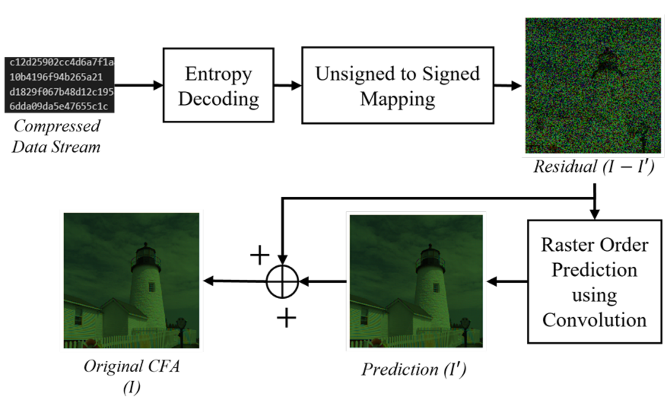


Fig. 4: Overview of the decompression algorithm

The obtained signed values denote the border pixel values and the errors that were obtained after prediction during compression. From this matrix, the same convolution filters can be applied and scanned in raster to obtain the predicted matrix I'. This matrix can be added to the residual matrix to obtain the original CFA I.

III. Experimental Results and Analysis

The proposed algorithm has been tested several CFAs directly captured from several mobile devices. These CFAs vary in type and resolution, and are unprocessed and linear. The detailed experimental results are presented as follows.

A. Experimental Environment

All the results presented in this section have been obtained by running the algorithm on a mobile device. The device has a 10-core CPU processor with a 64-bit arm architecture, and a RAM memory of 12 GB. The codes have been written in C and compiled using clang 17.0.2 compiler in Android 15 operating system.

B. Obtaining the Convolution Parameters

In this experiment, kernels of size 5×5 have been used. For obtaining the filter values, images from the Kodak dataset [13] have been sub-sampled into Bayer and tetra patterns, and have been obtained by solving the optimization as described earlier in Section II-A. The obtained coefficient values used in this paper are shown in Fig. 5. In the original calculation, the values are obtained by dividing with the sum of the values mentioned in the figures to ensure that the sum of the coefficients is equal to one.

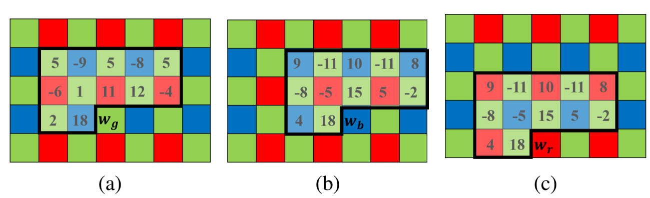


Fig. 5: Coefficients of the convolution kernel for Bayer CFA.

C. Compression Performance Analysis of the Proposed Algorithm

The proposed algorithm enhances the compression of the CFA by reducing its entropy. This is seen from the graph presented in Fig. 7, where a set of 100 Bayer CFAs of dimension 4080×3060 with a bit-depth of 12-bits-per-pixel have been compressed using the proposed algorithm. The proposed algorithm reduces the entropy from 7.8501 bits/pixel to 6.5635 bits/pixel on an average which helps in compression.

The extent to which a data can be compressed depends on its entropy. From Shannon's source coding theorem [14], for a CFA with entropy $H(I)$, the average length of pixels $\bar{R}$ when compressed using entropy coders are bound by

$$H(I) \leq \bar{R} \tag{5}$$

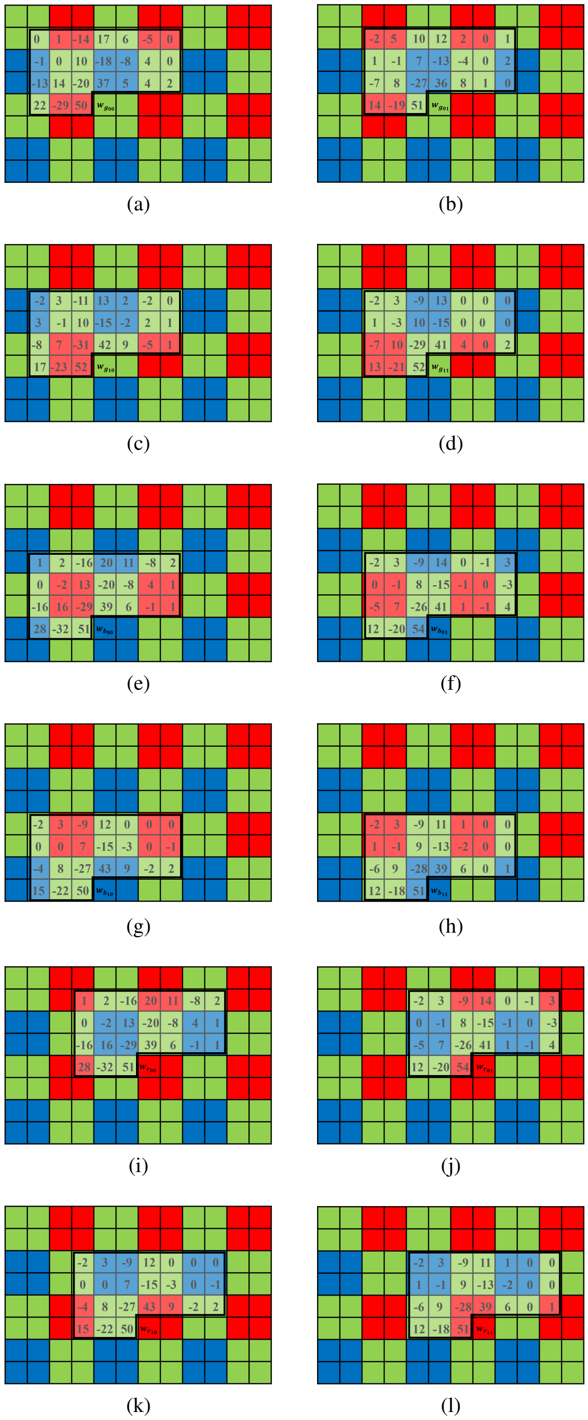


Fig. 6: Coefficients of the convolution kernel for Tetra CFA.

Since the entropy of the residual matrix is lesser than the original CFA, the proposed method requires lesser number of bits per pixel when compressed with Huffman codes compared to the number it would have taken if the original CFA was compressed. In the same dataset of 100 Bayer CFAs, the proposed method has an average $\bar{R} = 6.298$ bits/pixel, which is reduced from 7.6247 bits/pixel if the original CFA was

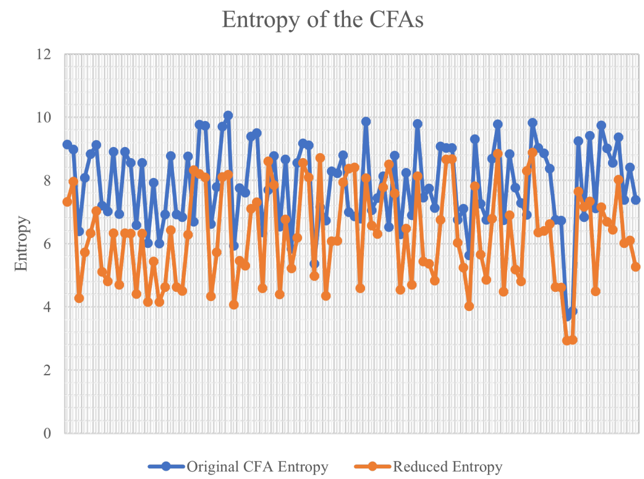


Fig. 7: Entropy of the original Bayer CFA vs residual matrix. Average entropy of the CFAs = 7.8501 bits/pixel. Average entropy of the residual matrix = 6.5635 bits/pixel. Testing done on a set of 100 Bayer CFAs of dimension 4080×3060 with a bit-depth of 12-bits-per-pixel.

compressed.

The size of the compressed CFAs obtained after applying the proposed method on the dataset of 100 Bayer CFAs is shown in Fig. 8. It can be seen that the proposed method reduces the CFA size to a great extent. On an average, the size of the compressed CFA reduces to 9.37 MB from the original 17.93 MB. The proposed algorithm, therefore, reduces the CFA size to around two times.

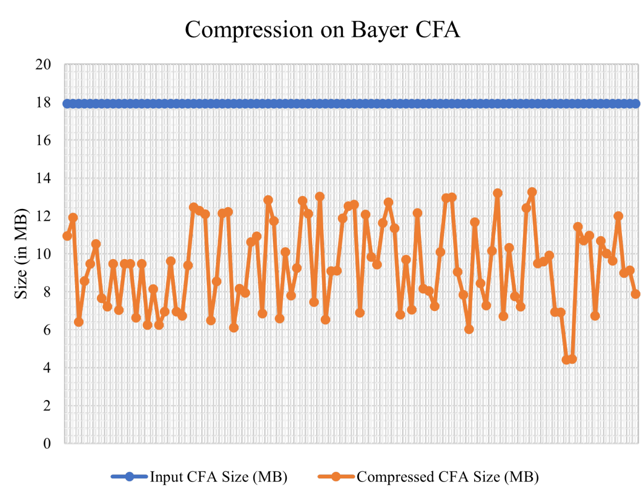


Fig. 8: Compressed CFA size using the proposed algorithm. Size of original CFA buffer = 17.93 MB. Average size of compressed CFAs = 9.37 MB Testing done on a set of 100 Bayer CFAs of dimension 4080×3060 with a bit-depth of 12-bits-per-pixel.

The proposed algorithm can also be extended to higher resolution, non-Bayer CFAs. Results on compressing such

TABLE II: Compression ratio of the proposed algorithm against different CFA buffers.

CFA Type	Bit-Depth (bits/pixel)	Resolution (pixels)	Input Size (MB)	Compression Ratio ↑ Best	Worst	Avg.
Bayer	12	4080 × 3060	17.930	4.062	1.351	2.038
Tetra	10	8160 × 6120	59.579	2.806	1.423	1.950
Nona	10	12000 × 9000	205.994	1.634	1.540	1.560
Hexadeca	10	16320 × 12240	381.006	3.275	2.401	2.942

CFAs have been shown in TABLE II. The ratio between the original CFA size to the compressed CFA size has been tabulated as the compression ratio. It can be seen that on an average, the proposed algorithm has a compression ratio of nearly two, thereby reducing the memory of the CFAs by around 50%.

D. Comparative Analysis

The performance of the proposed method has been compared with some of the existing lossless compression methods for Bayer CFAs, results for which have been shown in TABLE III. In terms of compression ratio, the proposed method is much better compared to Huffman compression of the pixels, and is very close to the state of the art ones. However, in terms of speed, it is very fast. The speed of compression is close to that of Huffman compression with a much higher compression ratio. It is much faster than the state-of-the-art lossless compression algorithms with a close compression ratio to them. Therefore, the proposed algorithm provides a trade-off between compression speed and compression ratio, and can be used in real-time use cases such as image processing pipelines.

E. Ablation Study

Experimentally, it has be observed that the average compression ratio increases with increasing the kernel size till $k = 5$ and then starts decreasing, as shown in Fig. 9. Moreover, increasing the convolution kernel size leads to increased CPU execution time. Therefore, choosing $k = 5$ gives a good trade-off between average compression ratio and execution time.

Fig. 9: Ablation study showing the trade-off between average compression ratio and CPU execution time for choosing different convolution kernel sizes (k). Test done on set of 100 bayer CFAs of dimension 4080×3060 with bit-depth of 12-bits-per-pixel.

TABLE III: Comparative analysis of the compression ratio and CPU execution time (in ms) of the proposed algorithm against the state-of-the-art lossless compression algorithms on a set of 100 12-MP Bayer buffers of dimension 4080×3060 and a bit-depth of 12 bits/pixel.

Method	Compression Ratio ↑ Best	Worst	Avg.	CPU Exec. Time (ms) ↓ Best	Worst	Avg.
Huffman Compression [4]	3.210	1.194	1.580	96	147	109
Chung and Chan [9]	4.129	1.366	2.077	374	502	441
Kim and Cho [10]	4.165	1.134	2.078	1624	1982	1774
Hernandez-Cabronero *et al.* [11]	4.528	1.429	2.314	335	490	393
Proposed	**4.062**	**1.351**	**2.038**	**103**	**129**	**117**

IV. Conclusion

This paper presents a convolution-based fast lossless CFA compression algorithm. It predicts the pixel values based on its neighboring color values and stores the error values by compressing them using entropy encoders. The original CFA can be losslessly decompressed from the compressed stream. The compression is fast with a high efficiency. This method can be efficiently used to save memory of large size CFAs in real-time applications such as image processing pipelines.

References

[1] R. Lukac and K. N. Plataniotis, "Color filter arrays: Design and performance analysis," *IEEE Transactions on Consumer electronics*, vol. 51, no. 4, pp. 1260–1267, 2005.

[2] I. Kim, D. Lim, Y. Seo, J. Lee, Y. Choi, and S. Song, "On recent results in demosaicing of Samsung 108MP CMOS sensor using deep learning," in *2021 IEEE Region 10 Symposium (TENSYMP)*, 2021, pp. 1–4.

[3] X. Xie, G. Li, Z. Wang, C. Zhang, D. Li, and X. Li, "A novel method of lossy image compression for digital image sensors with bayer color filter arrays," in *2005 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2005, pp. 4995–4998.

[4] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, 1952.

[5] A. Said and K. Sayood, *Arithmetic coding*. Academic Press, San Diego, CA, 2003.

[6] R. Sugiura, Y. Kamamoto, N. Harada, and T. Moriya, "Optimal golomb-rice code extension for lossless coding of low-entropy exponentially distributed sources," *IEEE Transactions on Information Theory*, vol. 64, no. 4, pp. 3153–3161, 2018.

[7] X. Wu and N. Memon, "CALIC-a context based adaptive lossless image codec," in *1996 IEEE international conference on acoustics, speech, and signal processing conference proceedings*, vol. 4. IEEE, 1996, pp. 1890–1893.

[8] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The JPEG 2000 still image compression standard," *IEEE Signal processing magazine*, vol. 18, no. 5, pp. 36–58, 2002.

[9] K.-H. Chung and Y.-H. Chan, "A lossless compression scheme for bayer color filter array images," *IEEE Transactions on Image Processing*, vol. 17, no. 2, pp. 134–144, 2008.

[10] S. Kim and N. I. Cho, "Lossless compression of color filter array images by hierarchical prediction and context modeling," *IEEE transactions on circuits and systems for video technology*, vol. 24, no. 6, pp. 1040–1046, 2014.

[11] M. Hernandez-Cabronero, M. W. Marcellin, I. Blanes, and J. Serra-Sagrista, "Lossless compression of color filter array mosaic images with visualization via JPEG 2000," *IEEE Transactions on Multimedia*, vol. 20, no. 2, pp. 257–270, 2017.

[12] K. Sayood, *Introduction to data compression*. Morgan Kaufmann, 2017, ch. 3, p. 49.

[13] "Kodak dataset," [Online; accessed 2025-08-10]. [Online]. Available: https://www.kaggle.com/datasets/sherylmehta/kodak-dataset

[14] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.