Adaptively Hashing 3DLUTs for Lightweight Real-time Image Enhancement

Fengyi Zhang, *Lin Zhang**, *Tianjun Zhang*, *Dongqing Wang*

School of Software Engineering, Tongji University, Shanghai, China

***Abstract*—Image enhancement is an essential and long-standing task in computer vision, in which the 3D lookup table (3DLUT) is widely used due to its powerful mapping capability and high time efficiency. However, the 3DLUT generally requires a large parameter amount since it caches the mapping results for all colors of the entire discrete color space with a 3D array. As a result, standard 3DLUT-based enhancement methods suffer from heavy memory footprints, limiting their practical applications. Based on the analyses of the inherent low grid utilization rate of 3DLUT, we propose HashLUT, an efficient hash form of the standard 3DLUT, and further build a lightweight real-time image enhancement network that adaptively learns HashLUTs and handles hash collisions end-to-end. Experiments on two benchmarks demonstrate that our model achieves comparable enhancement performance to the state-of-the-art methods with significantly fewer parameters. Source codes are available at https://github.com/Xian-Bei.**

***Index Terms*—Image enhancement, photo retouching, 3-dimensional lookup table (3DLUT), hashing, compression**

I. Introduction

Image enhancement techniques are widely applied in the digital imaging pipeline and retouching tools to improve the image quality. This domain has witnessed the rapid development of learning-based automatic enhancement methods since they reduce the expensive manual work and further boost the performance. The 3-Dimensional LookUp Table (3DLUT), one of the most popular image enhancement operators, has also been explored to be combined with deep-learning, achieving state-of-the-art (SOTA) overall performance [1]. 3DLUT directly caches the mapping outputs for all colors of the discrete color space with a 3D lattice in memory, instead of computing them only for input colors in runtime using a series of cascade modules, such as adjustment of exposure, white balance, hue and saturation. Consequently, for 3DLUT, on the one hand, only highly parallelizable lookup and interpolation operations are needed for the enhancing procedure, which brings great superiority in time efficiency. On the other hand, however, its space complexity grows cubically with the resolution which controls the mapping fineness, resulting in massive parameters to learn and store in 3DLUT-based models [1]–[6] which typically contain several to dozens of basis 3DLUTs. Such a large parameter amount not only hinders their practical deployment but also gives rise to the training difficulty.

Focusing on this problem, we conduct in-depth analyses on 3DLUT and observe its extremely low grid utilization rate, which could be further improved to obtain a much more compact form without affecting its mapping capabilities. Specifically, the colors appearing in a single input image generally only utilize a small part of grids of a 3DLUT, as intuitively and quantitatively demonstrated in Fig. 1 (a) and (b), respectively. Therefore, given an input image, most grids of 3DLUT are redundant and contribute nothing to the enhanced result apart from increasing the memory usage. Inspired by the success of sophisticated hash techniques in compressing neural networks [7], [8], signed distance functions (SDF) [9]–[11] and explicit density fields [11], we propose to adaptively hash 3DLUT to increase its grid utilization rate and construct a lightweight real-time image enhancement network based on it. Our main contributions are summarized as follows:

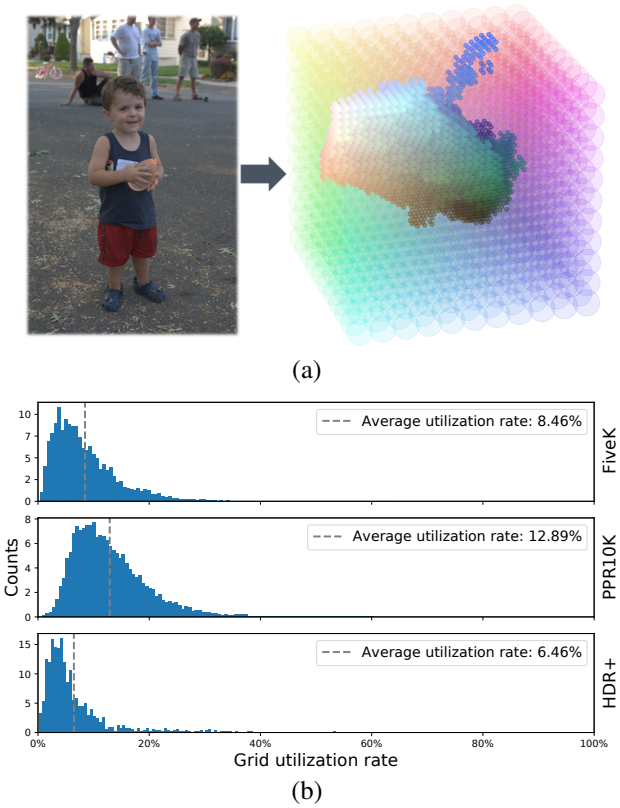

Fig. 1. (a) The utilized grids when applying a 3DLUT of $D = 33$ to the given input image only occupy a small part of the 3DLUT's entire mapping space. (b) The distribution histograms of the grid utilization rates when applying a 3DLUT of $D = 33$ to each input image of three benchmark datasets. The horizontal axis represents the utilization rate while the vertical axis represents the number of images that fall into the same utilization rate interval.

*Corresponding author: cslinzhang@tongji.edu.cn.

- To the best of our knowledge, we are the first to introduce hash techniques to the image enhancement task, and propose an efficient hash form of 3DLUT, namely HashLUT, which enjoys all its advantages while has orders of magnitude fewer parameters.
- A lightweight and real-time image enhancement network is constructed based on HashLUT, which adaptively learns progressive basis HashLUTs and handles hash collisions in an end-to-end manner.
- Experiments conducted on two benchmarks verify the efficacy of our proposed method which presented comparable image enhancement performance to the SOTA models but with significantly fewer parameters.

II. Related Work

A. Image enhancement

Recent learning-based enhancement methods could be roughly classified into two paradigms. The first paradigm [12]–[14] leverages image translation networks to predict the enhancement results directly. Since the convolution computational burden increases rapidly with the input resolution, methods belonging to this paradigm usually struggle to meet the practical applications. The second paradigm employs neural networks only to predict [15]–[19] or simulate [20] certain manual-designed enhancement operators. Such a paradigm generally possesses high feasibility as usually only lightweight networks are employed to work on down-sampled input images and most of the manual-designed operators enjoy fast processing speed even on high-resolution images.

B. 3DLUT-based methods

Zeng *et al*. [1] first embedded 3DLUT into the learning-based image enhancement framework and achieved powerful enhancement capability and high time efficiency. Following their strategy, Liang *et al*. [2] collected a large-scale portrait dataset and applied 3DLUT to the portrait photo retouching (PPR) task. Wang *et al*. [3] extended the network of [1] to achieve spatial-aware enhancement effects by predicting pixel-level weights. Yang *et al*. [5] proposed to compensate 3DLUT with 1DLUT to improve its efficiency. Yang *et al*. [6] learned the non-uniform sampling intervals in the 3D color space to achieve a more flexible sampling point allocation.

C. Hash-based compression

There exists a wide range of studies focusing on data compression, where hash-based methods rearrange the parameters into a more compact space by allowing different indexes to share the memory. HashedNets [7] and follow-up work like [8] achieved drastic parameter reduction of neural networks by exploiting their inherent redundancy through hashing. VoxelHashing [9] and ChunkFusion [10] applied hashing techniques on SDF to achieve scalable volumetric representation. Instant-NGP [11] constructed an effective hash encoding for representing images, SDF and NeRF [21]. In this work, we deeply analyzed 3DLUT and observed that its inherent low grid utilization properties are naturally suitable for hashing to rearrange the parameter space.

III. Method

A. Grid utilization of 3DLUT

A standard 3DLUT is commonly formulated as a 3D cubic lattice $\mathbf{\Phi} \in \mathbb{R}^{3\times D\times D\times D}$ containing D^3 three-channel elements where D is the resolution controlling the mapping fineness. Each element is denoted by $\mathbf{\Phi}_{[i,j,k]} \in \mathbb{R}^3$ and defines a color mapping as $(i,j,k) \rightarrow \mathbf{\Phi}_{[i,j,k]}$ where $i,j,k = 1,...,D$. Given an input color (r_{in}, g_{in}, b_{in}) in which each element is distributed in $[0,1)$, its corresponding mapping output in a 3DLUT $\mathbf{\Phi}$ is calculated as,

$$(r_{out}, g_{out}, b_{out}) = t(l(r_{in}, g_{in}, b_{in}, \mathbf{\Phi})), \tag{1}$$

which contains a lookup operation l followed by a trilinear interpolation t. Specifically, the input color is first scaled by D as $(\tilde{r}_{in}, \tilde{g}_{in}, \tilde{b}_{in}) = (r_{in}\cdot D, g_{in}\cdot D, b_{in}\cdot D)$ to fall in $[0, D)$ and then its eight surrounding elements are found as,

$$l(r_{in}, g_{in}, b_{in}, \mathbf{\Phi}) = \{\mathbf{\Phi}_{[\tilde{r},\tilde{g},\tilde{b}]}\}, \tag{2}$$

where $\tilde{r} \in \{\lfloor\tilde{r}_{in}\rfloor, \lceil\tilde{r}_{in}\rceil\}$, $\tilde{g} \in \{\lfloor\tilde{g}_{in}\rfloor, \lceil\tilde{g}_{in}\rceil\}$, and $\tilde{b} \in \{\lfloor\tilde{b}_{in}\rfloor, \lceil\tilde{b}_{in}\rceil\}$. Then the eight surrounding elements are trilinearly interpolated to get the mapping output. This procedure is intuitively illustrated in the left part of Fig. 2. Given an input image $\boldsymbol{I}$, each of its pixels is processed in parallel following (1) to get the enhanced image $\boldsymbol{O}$ denoted by $\boldsymbol{O} = \mathbf{\Phi}(\boldsymbol{I})$.

Since only lookup and interpolation operations are involved, 3DLUT enjoys extremely high time efficiency that it could easily enhance 4K video streams in real-time on a laptop device. However, 3DLUT possesses a space complexity of $O(D^3)$ which grows cubically with D, resulting in massive parameters to learn and store in 3DLUT-based models.

To reduce the parameters of 3DLUT while maintaining its superiority, we analyzed its intrinsic mapping space and

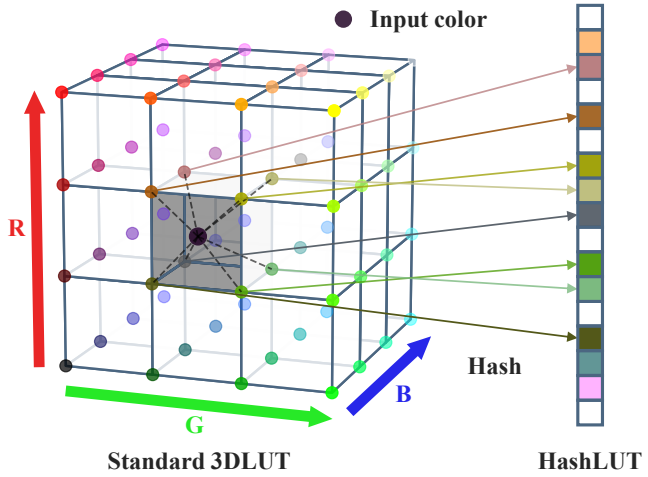

Fig. 2. Illustration of the standard mapping process of 3DLUT and the additional hash operation of HashLUT. Given an input color, both 3DLUT and HashLUT calculate the indexes of its nearest eight neighbors, trilinearly interpolating the neighbors' values as the mapping output. The only difference is that the actual storage locations of the eight neighbors are the indexes themselves for 3DLUT, but calculated by querying the indexes through a hash function for HashLUT.

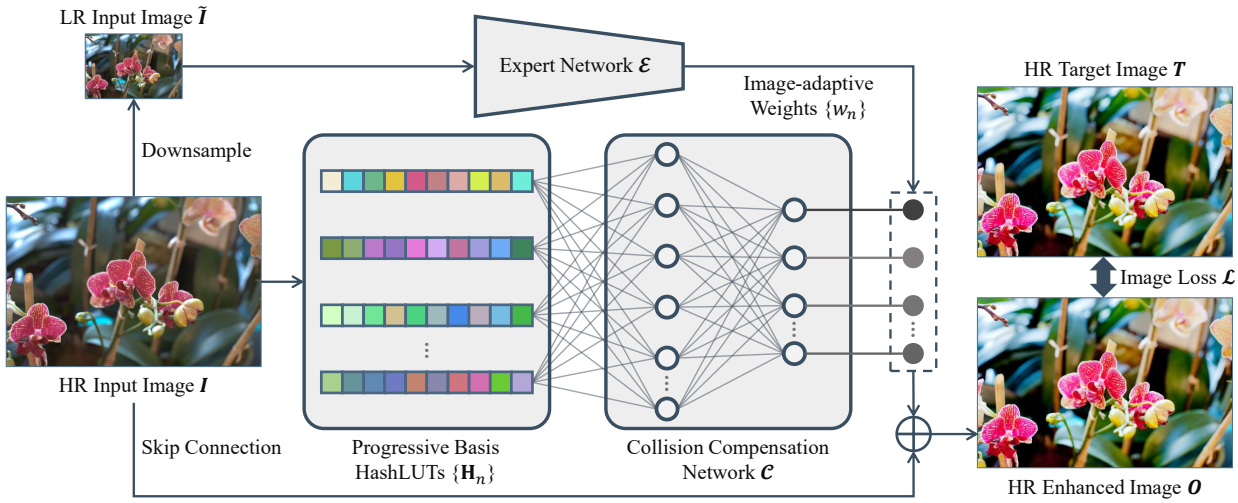

Fig. 3. Framework of our proposed HashLUT-based image enhancement network which contains N progressive basis HashLUTs, a collision-compensation network and an expert network. The basis HashLUTs not only cover the enhancement effects required by different scenes but also are set with progressive resolutions, cooperating with the collision-compensation network to largely mitigate the impact of hash conflict. The expert network selectively fuses the results of different HashLUTs by predicting image-adaptive weights based on the understanding of the characteristics of the input image.

observed its property of sparse grid utilization. Specifically, we conducted statistics on the 5,000 input images of FiveK [22], 111,61 ones of PPR10K [2], and 922 ones of HDR+ [23]. The distribution histograms of the grid utilization rates when applying a 3DLUT with a classical setting of $D = 33$ to each input image are demonstrated in Fig. 1 (b). It can be seen that the average grid utilization rate across all three benchmarks is around 10%, which indicates the huge redundant space of 3DLUT that could be leveraged for lossless compression. In other words, given any input image, there exists a compact representation of 3DLUT that could produce the same enhancement effect as the standard representation does, while the average parameter amount of the former is only about 10% of that of the latter.

B. Adaptively hashing 3DLUTs

Motivated by the aforementioned observation, we propose an efficient hash form of 3DLUT, namely HashLUT, denoted by $\mathbf{H} \in \mathbb{R}^{3\times T}$ containing T three-channel elements where T balances the mapping capacity and the parameter amount. Instead of representing the mapping space explicitly as a 3D array, HashLUT represents it as a 1D hash table by mapping 3D indexes into 1D. Specifically, given an index $[i, j, k]$, $\mathbf{H}_{[i,j,k]}$ is actually stored in $\mathbf{H}_{[h(i,j,k)]}$ where h denotes a hash function of $\mathbb{R}^3 \to \mathbb{R}^1$. This procedure is illustrated in the right part of Fig. 2. In our work, h is implemented as a classical spatial hash function [24] which is widely used [9], [11] and enjoys a time complexity of $O(1)$, resulting in little extra time costs. Consequentially, equation (2) is changed to,

$$l(r_{in}, g_{in}, b_{in}, \mathbf{H}) = \{\mathbf{H}_{[\tilde{r},\tilde{g},\tilde{b}]}\}, \tag{3}$$

and the result of enhancing a given input image $\boldsymbol{I}$ with a HashLUT $\mathbf{H}$ is denoted by $\boldsymbol{O} = \mathbf{H}(\boldsymbol{I})$.

As illustrated in Fig. 3, we construct a HashLUT-based image enhancement network which comprises N basis HashLUTs denoted by $\{\mathbf{H}_n\}_{n=1,\dots,N}$, a collision-compensation network $\mathcal{C}$ and an expert network $\mathcal{E}$. The role of the basis HashLUTs lies in twofold. On the one hand, different HashLUTs cover the enhancement effects required by various scenes as the basis 3DLUTs do in the classical 3DLUT-based models [1]. On the other hand, they are set with progressive resolutions, cooperating with the collision-compensation network $\mathcal{C}$ to largely mitigate the impact of hash conflicts, which would be analyzed in detail in Sect. III-C. The expert network $\mathcal{E}$ selectively fuses the results of different HashLUTs by predicting image-adaptive weights based on the understanding of the characteristics of the input image such as environment, lightness and tones. Since such global information preserves across the image resolution, it is a common practice [1]–[3], [5], [6], [16], [25] to let $\mathcal{E}$ only work on the down-sampled input image to greatly save the computational cost.

Specifically, given a high-resolution (HR) input image $\boldsymbol{I}$, it is first enhanced by N basis HashLUTs in parallel as $\{\boldsymbol{O}_n = \mathbf{H}_n(\boldsymbol{I})\}_{n=1,\dots,N}$ which are then refined by $\mathcal{C}$ as $\{\tilde{\boldsymbol{O}}_n\} = \mathcal{C}(\{\boldsymbol{O}_n\})$. At the same time, $\boldsymbol{I}$ is downsampled to a low-resolution (LR) version $\tilde{\boldsymbol{I}}$ and then $\tilde{\boldsymbol{I}}$ is fed into $\mathcal{E}$ to get the N image-adaptive weights $\{w_n\}_{n=1,\dots,N} = \mathcal{E}(\tilde{\boldsymbol{I}})$. Finally, the enhanced image $\boldsymbol{O}$ is generated as,

$$\boldsymbol{O} = \sum_{n=1}^{N} w_n \tilde{\boldsymbol{O}}_n + \boldsymbol{I}, \tag{4}$$

where a residual strategy is applied for the stability of training by combining the mapping output with $\boldsymbol{I}$. Given P pairs of input and target images denoted by $\{(\boldsymbol{I}^p, \boldsymbol{T}^p)\}_{p=1,\dots,P}$, the associated optimization problem for training our model is formulated as,

$$\underset{(\{\mathbf{H}_n\},\mathcal{E},\mathcal{C})}{\arg\min} \sum_{p=1}^{P} \mathcal{L}(\boldsymbol{T}^p, \boldsymbol{O}^p), \tag{5}$$

where $\mathcal{L}$ denotes the loss function which in our implementation is defined as the L_1 distance for simplicity.

C. Analyses of the hash collision

Hashing technology reduces storage overhead by mapping different indexes to the same grid, which is called the hash collision. Given an input image, if a collision happens between one of its utilized grids and other unused ones, this collision avoids the space consumption of the latter without harming the enhancement effects. In fact, we rely on such collisions to achieve the compression of 3DLUT. However, if a collision happens among utilized grids, it may affect the enhancement results since different input colors would be mapped to the same one, determined by the training gradients. In the following discussion, we will only focus on the latter type of collisions and ignore the former one.

Previous approaches like [9] explicitly handled hash collisions by typical means such as probing, bucketing and chaining which are nontrivial and may increase the model complexity and introduce additional costs. Unlikely, inspired by the successful practice of recent multi-resolution hash encoding [11], we leverage multiple hash tables with progressive resolutions followed by a refine network and rely on the gradient-based optimization to handle collisions adaptively under the end-to-end learning framework.

As aforementioned, the basis HashLUTs work in two ways. Apart from adapting to various scenes, they are set with different resolutions denoted by $\{D_n\}_{n=1,\dots,N}$ to statistically avoid collisions occurring in the same places simultaneously. Take one color channel as an example for simplicity, given an input color c_{in}, the indexes of its surrounding grids are $\lfloor c_{in} \cdot D_n \rfloor$ and $\lceil c_{in} \cdot D_n \rceil$ according to the description in Sect. III-A. If $\{D_n\}$ share the same value, the input color would utilize grids with the same indexes across all the HashLUTs, meaning collisions happen in either none of HashLUTs or the same places of all of them. Instead, we implement $\{D_n\}$ as a geometric progression, leading to utilized grids with different indexes across the progressive HashLUTs so that hash collisions occur approximately randomly in time and space.

Thanks to the the progressive resolutions, given an input image, the N basis HashLUTs representing different scenes would statistically have collisions with different degrees and space distributions and enjoy the potential to complement each other. We thus propose to consider the mapping results of all basis HashLUTs comprehensively and refine each of them. Specifically, a lightweight collision-compensation network is added following the progressive HashLUTs, cooperating with them to reduce the impact of hash conflicts. The efficacy of such collision handling strategies is verified in Sect. IV-B1.

IV. Experiments

A. Experimental setup

Our model was implemented based on PyTorch and the hash encoding of Tiny-CUDA-NN. Since our primary objective lies in hashing 3DLUT for lightweight image enhancement, we simply implemented the expert network as a five-layer CNN and the collision-compensation network as a one-layer MLP instead of more sophisticated architectures. A batch size of 1 and the Adam [26] optimizer was employed for training. Experiments were conducted on two benchmark datasets, namely MIT-Adobe FiveK [22] and PPR10K [2], and quantitative enhancement performance was measured according to three metrics, namely PSNR, SSIM [27], and the L_2 distance in CIELAB color space (ΔE) which is proven consistent to human perception [28].

B. Ablation study

Ablation studies were conducted on FiveK [22] to verify the efficacy of our adaptive collision-handling strategy and demonstrate the effects of the hyper-parameters in our model.

1) Efficacy of the adaptive collision-handling strategy: We adaptively handle hash collisions by employing progressive basis HashLUTs and the followed collision-compensation network. To verify their efficacy, we compared our model with three baselines on FiveK [22] dataset. All the experimental settings were the same except for the collision-handling strategy as presented below: (1) "**Single**", Single-resolution basis HashLUTs without the collision-compensation network; (2) "**Single+$\mathcal{C}$**", Single-resolution basis HashLUTs with the collision-compensation network; (3) "**Progressive**", Progressive-resolution basis HashLUTs without the collision-compensation network; and (4) "**Progressive+$\mathcal{C}$** (Ours)", Progressive-resolution basis HashLUTs with the collision-compensation network.

TABLE I
QUANTITATIVE COMPARISON OF DIFFERENT BASELINES AND OUR MODEL ON FIVEK [22] DATASET.

Method	PSNR ↑	SSIM ↑	ΔE ↓
Single	24.88	0.853	9.80
Single+$\mathcal{C}$	25.24	0.912	8.43
Progressive	25.22	0.908	8.49
Progressive+$\mathcal{C}$ (Ours)	**25.50**	**0.926**	**7.46**

As shown in Table I, both **Single+$\mathcal{C}$** and **Progressive** enjoyed a notable improvement in terms of all three metrics compared to **Single**, which fully verifies the effectiveness of the two parts of our adaptive collision-handling strategy, i.e. the progressive-resolution HashLUTs and the collision-compensation network. Besides, by combining them, our model **Progressive+$\mathcal{C}$** achieved the best overall performance compared to **Single+$\mathcal{C}$** and **Progressive**, which indicates that these two parts are able to cooperate well with each other to further improve performance. It is in line with our analyses since the progressive resolutions statistically reduce the probability that collisions happened at the same time and place across the basis HashLUTs, providing the potential to let them complement each other and compensate for collision.

2) Effects of hyper-parameters: To quantitatively demonstrate the effects of the hyper-parameters T and N in our model, we conducted experiments on FiveK [22] dataset with $T = \{2^{10}, 2^{11}, 2^{12}, 2^{13}, 2^{14}, 2^{15}\}$ and $N = \{3, 5, 7, 10, 15, 20\}$.

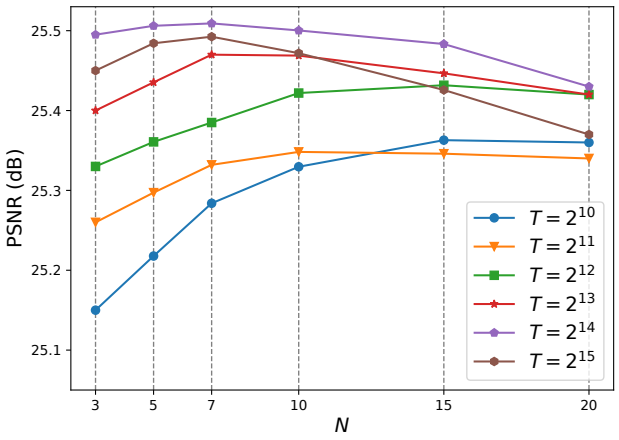

Fig. 4. Effects of hyper-parameters T and N on the enhancement performance on FiveK [22] dataset.

As shown in Fig 4, the enhancement performance increases with both T and N before meeting the saturation points, and then stagnates or even starts to decrease slightly. It is reasonable since T balances the hash collision degree and space utilization, and N determines the number of basis HashLUTs which not only controls the coverage of various scenes but also affects the ability to handle collisions. We attribute the stagnation and declination to the overfitting of excessive parameters on the training set. Overall, we set $T = 14$ and $N = 3$ for a good balance between the enhancement performance and model complexity.

C. Comparison with SOTA

We compared our model with eight SOTA learning-based counterparts, including UPE [14], DPE [13], HDRNet [18], CSRNet [20], 3DLUT [1], SALUT [3], SepLUT [5] and AdaInt [6]. The quantitative experimental results obtained on the two benchmarks are reported in Table II and Table III, respectively. It needs to be noticed that following the practice of [2], UPE [14] and DPE [13] were not experimented on PPR10K [2] due to their heavy computational burden. As highlighted in red, our model achieved the best performance on most metrics of each dataset. Most importantly, compared with the standard [1] and the follow-up [3], [5], [6] 3DLUT-based methods, our model achieved comparable or even better enhancement performance but with significantly fewer parameters. Although CSRNet [20] enjoys an extremely small model size, our method outperformed it on each dataset in terms of all three metrics by a large margin.

TABLE II
QUANTITATIVE COMPARISON ON FIVEK [22] DATASET.

Method	PSNR ↑	SSIM ↑	ΔE ↓	Param. (K)
UPE [14]	21.88	0.853	10.80	999
DPE [13]	23.75	0.908	9.34	5,750
HDRNet [18]	24.32	0.912	8.49	482
CSRNet [20]	25.23	0.923	7.70	37
3DLUT [1]	25.23	0.912	7.60	592
SALUT [3]	25.40	0.925	7.46	4,155
SepLUT [5]	25.47	0.921	7.54	120
AdaInt [6]	25.49	0.926	7.47	620
Ours	25.50	0.926	7.46	114

TABLE III
QUANTITATIVE COMPARISON ON PPR10K [2] DATASET.

Dataset	Method	PSNR↑	ΔE ↓	Param. (K)
PPR-a	HDRNet [18]	23.93	8.70	482
	CSRNet [20]	22.72	9.75	37
	SALUT [3]	25.85	6.84	4,155
	3DLUT+HRP [2]	25.99	6.76	11,716
	SepLUT [5]	26.28	6.59	120
	AdaInt [6]	26.33	6.56	620
	Ours	26.34	6.56	114
PPR-b	HDRNet [18]	23.96	8.84	482
	CSRNet [20]	23.76	8.77	37
	SALUT [3]	25.01	7.67	4,155
	3DLUT+HRP [2]	25.06	7.51	11,716
	SepLUT [5]	25.23	7.49	120
	AdaInt [6]	25.40	7.33	620
	Ours	25.42	7.40	114
PPR-c	HDRNet [18]	24.08	8.87	482
	CSRNet [20]	23.17	9.45	37
	SALUT [3]	25.36	7.54	4,155
	3DLUT+HRP [2]	25.46	7.43	11,716
	SepLUT [5]	25.59	7.51	120
	AdaInt [6]	25.68	7.31	620
	Ours	25.65	7.30	114

The qualitative comparison with SOTA competitors is presented in Fig. 5. It can be seen that compared with the other methods, the enhancement effects of our model are visually the most pleasing and closest to the ground truth. Overall, by adaptively learning progressive basis HashLUTs and handling hash collisions end-to-end, our method improves the enhancement performance of the standard 3DLUT-based methods both quantitatively and qualitatively with a significantly smaller parameter amount, achieving SOTA on consideration of both the enhancement performance and the model complexity.

V. CONCLUSION

In this paper, we introduced hash techniques to the image enhancement task to reduce the space complexity of 3DLUT-based models, where 3DLUT is a powerful enhancement operator but suffers a large parameter amount. Specifically, in-depth analyses are conducted on the inherent low grid utilization rate of 3DLUT and an efficient hash form of 3DLUT is proposed, based on which a lightweight real-time image enhancement network is further constructed. Extensive experiments demonstrate that our model achieves SOTA enhancement performance with significantly fewer parameters.

VI. ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China under Grant 62272343, Grant 61973235, and Grant 61936014; in part by the Shuguang Program of Shanghai Education Development Foundation and Shanghai Municipal Education Commission under Grant 21SG23; and in part by the Fundamental Research Funds for the Central Universities.

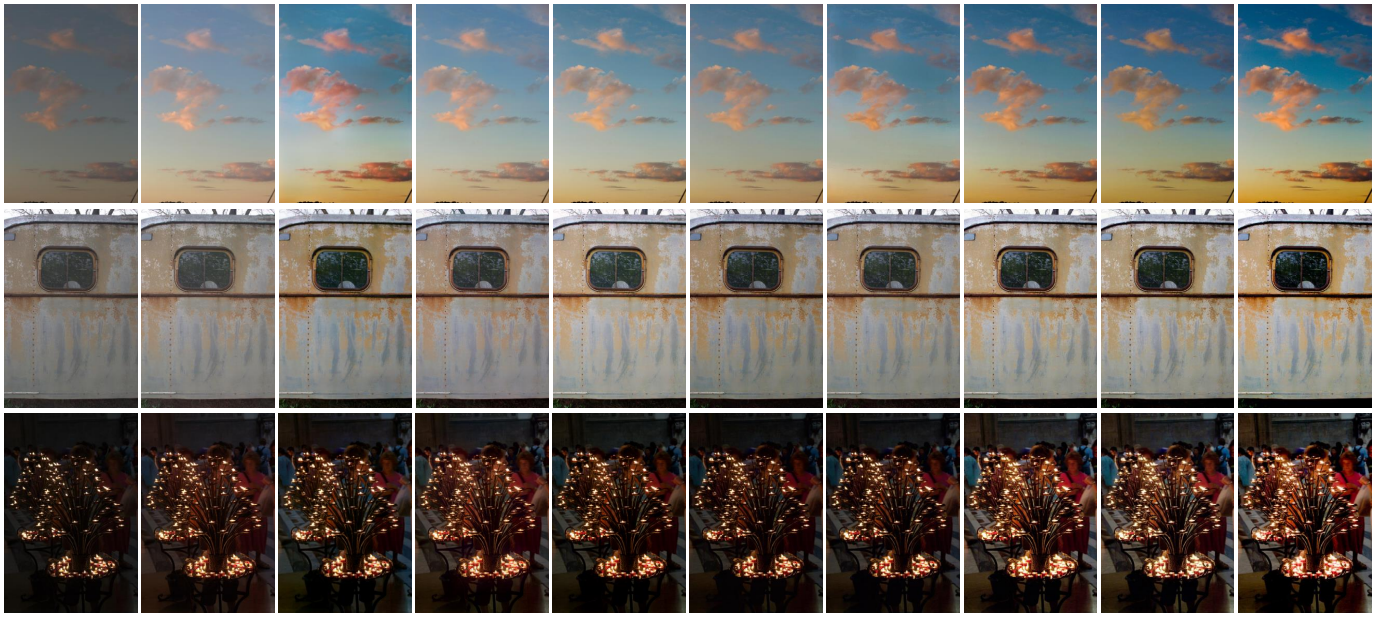

Fig. 5. Qualitative comparison on FiveK [22] dataset.

REFERENCES

[1] H. Zeng, J. Cai, L. Li, Z. Cao, and L. Zhang, "Learning image-adaptive 3D lookup tables for high performance photo enhancement in real-time," *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 44, no. 4, pp. 2058–2073, 2022.

[2] J. Liang, H. Zeng, M. Cui, X. Xie, and L. Zhang, "PPR10K: A large-scale portrait photo retouching dataset with human-region mask and group-level consistency," in *CVPR*, 2021, pp. 653–661.

[3] T. Wang, Y. Li, J. Peng, Y. Ma, X. Wang, F. Song, and Y. Yan, "Real-time image enhancer via learnable spatial-aware 3D lookup tables," in *ICCV*, 2021, pp. 2471–2480.

[4] W. Cong, X. Tao, L. Niu, J. Liang, X. Gao, Q. Sun, and L. Zhang, "High-resolution image harmonization via collaborative dual transformations," in *CVPR*, 2022, pp. 18449–18458.

[5] C. Yang, M. Jin, Y. Xu, R. Zhang, Y. Chen, and H. Liu, "SepLUT: Separable image-adaptive lookup tables for real-time image enhancement," in *ECCV*, 2022, pp. 201–217.

[6] C. Yang, M. Jin, X. Jia, Y. Xu, and Y. Chen, "AdaInt: Learning adaptive intervals for 3D lookup tables on real-time image enhancement," in *CVPR*, 2022, pp. 17522–17531.

[7] W. Chen, J. Wilson, S. Tyree, K. Weinberger, and Y. Chen, "Compressing neural networks with the hashing trick," in *ICML*, 2015, pp. 2285–2294.

[8] B. Reagan, U. Gupta, B. Adolf, M. Mitzenmacher, A. Rush, G. Wei, and D. Brooks, "Weightless: Lossy weight encoding for deep neural network compression," in *ICML*, 2018, pp. 4324–4333.

[9] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, "Real-time 3D reconstruction at scale using voxel hashing," *ACM Trans. Graph.*, vol. 32, no. 6, 2013.

[10] C. Guo, L. Zhang, Y. Shen, and Y. Zhou, "ChunkFusion: A learning-based RGB-D 3D reconstruction framework via chunk-wise integration," in *ICASSP*, 2022, pp. 3818–3822.

[11] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, vol. 41, no. 4, 2022.

[12] M. Afifi and M. Brown, "Deep white-balance editing," in *CVPR*, 2020, pp. 1397–1406.

[13] Y. Chen, Y. Wang, M. Kao, and Y. Chuang, "Deep photo enhancer: Unpaired learning for image enhancement from photographs with gans," in *CVPR*, 2018, pp. 6306–6314.

[14] R. Wang, Q. Zhang, C. Fu, X. Shen, W. Zheng, and J. Jia, "Underexposed photo enhancement using deep illumination estimation," in *CVPR*, 2019, pp. 6849–6857.

[15] M. Afifi, B. Price, S. Cohen, and M. Brown, "When color constancy goes wrong: Correcting improperly white-balanced images," in *CVPR*, 2019, pp. 1535–1544.

[16] E. Liu, S. Li, and S. Liu, "Color enhancement using global parameters and local features learning," in *ACCV*, 2020, pp. 202–216.

[17] S. Moran, S. McDonagh, and G. Slabaugh, "CURL: Neural curve layers for global image enhancement," in *ICPR*, 2021, pp. 9796–9803.

[18] M. Gharbi, J. Chen, J. Barron, S. Hasinoff, and F. Durand, "Deep bilateral learning for real-time image enhancement," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–12, 2017.

[19] S. Moran, P. Marza, S. McDonagh, S. Parisot, and G. Slabaugh, "DeepLPF: Deep local parametric filters for image enhancement," in *CVPR*, 2020, pp. 12823–12832.

[20] J. He, Y. Liu, Y. Qiao, and C. Dong, "Conditional sequential modulation for efficient global image retouching," in *ECCV*, 2020, pp. 679–695.

[21] B. Mildenhall, P. Srinivasan, M. Tancik, J. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *ECCV*, 2020, p. 405–421.

[22] V. Bychkovsky, S. Paris, E. Chan, and F. Durand, "Learning photographic global tonal adjustment with a database of input/output image pairs," in *CVPR*, 2011, pp. 97–104.

[23] S. Hasinoff, D. Sharlet, R. Geiss, A. Adams, J. Barron, F. Kainz, J. Chen, and M. Levoy, "Burst photography for high dynamic range and low-light imaging on mobile cameras," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 1–12, 2016.

[24] M.Teschner, B.Heidelberger, M. Müller, D. Pomeranets, and M. Gross, "Optimized spatial hashing for collision detection of deformable objects," in *VMV*, 2003, p. 47–54.

[25] H. Kim, Y. Koh, and C. Kim, "Global and local enhancement networks for paired and unpaired image enhancement," in *ECCV*, 2020, pp. 339–354.

[26] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015, pp. 1–15.

[27] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004.

[28] W. G. Backhaus, R. Kliegl, and J. S. Werner, *Color Vision: Perspectives From Different Disciplines*, De Gruyter, 1998.