Neural Image Recolorization for Creative Domains

Boyi Li
Cornell University

Serge Belongie
University of Copenhagen

Ser-nam Lim
Meta AI

Abe Davis
Cornell University

Abstract

We present a self-supervised approach to recolorization of images from design-oriented domains. Our approach can recolor images based on image exemplars or target color palettes provided by a user. In contrast with previous approaches, our method can reproduce color palettes with luminance distributions that differ significantly from input, and our method is the first palette-based approach to distinguish between recolorings that match reflectance and those that match illumination, making it particularly well-suited to visualizing different aesthetic decisions in design applications. The key to our approach is first to learn latent representations for texture and color in a setting where self-supervision is especially straightforward, and then to learn a mapping to our color representation from input color palettes and scene illumination, which offers a more intuitive space for controlling and exploring recolorization.

1. Introduction

In visual art, the concept of a *color palette* has its origins in the painter's mixing plate (or "palette"). To use a mixing plate, the artist first deposits a small number of primary pigments around the plate's edge. The center of the plate is then used to mix pigments in different ratios, creating different combinations of the chosen primaries. Over time, the term "palette" came to describe not just the physical plate, but also the range of colors that it caries—and, eventually, the artistic decision to limit that range in a creative work. In the digital age, this modern notion of a palette often distinguishes random or natural images from those that are the product of a creative process. While the colors of a random scene may be arbitrarily distributed, creative works often adhere to a more structured palette. For example, when adding furniture to a home, decorators coordinate the selection and placement of items based on the colors and textures present in each room.

While recolorization is not a new problem, our approach offers several advantages over previous work. Most of these advantages build on two basic observations. First, that adherence to aesthetic principles creates a lower-dimensional space of color distributions in images from design-oriented domains, which we can leverage through adversarial losses to improve recolorization. And second, that we can think of color palettes as human-understandable parameterizations of this space. Putting these observations together, we take a two-stage approach to recolorization. In the first stage we cast exemplar-based recolorization as a style transfer problem to learn a latent representation for the constrained distribution of colors in a target domain. Then, in the second stage, we learn a mapping to our latent color distribution space from simple human-interpretable color palettes. We call our approach *ColorHouse* in reference to the domain of images we first used when developing it (interior design) and show that it offers a flexible and efficient way to explore recolorization for creative applications.

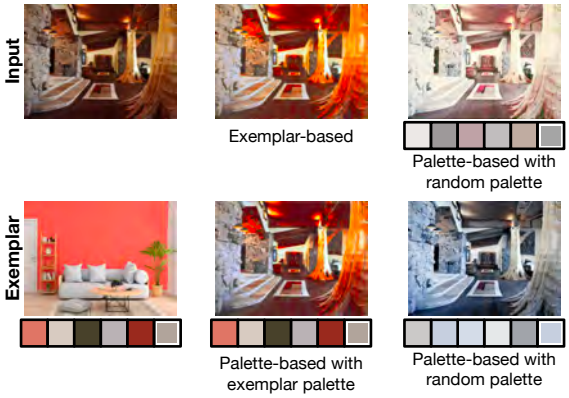

Figure 1: ColorHouse recolorization. Exemplar-based: recolorization by selected exemplar. Palette-based with exemplar palette: recolorization by corresponding palette (the last item is the corresponding illumination) extracted from the exemplar. Palette-based with random palette: recolorization by random input palette. It is obvious that ColorHouse could recolorize the inputs to the corresponding desired color styles with very natural outputs based on the color of the exemplar and ensure the reflectances based on the given color palette, which has not be seriously considered by previous recolorization or style-transfer approaches.

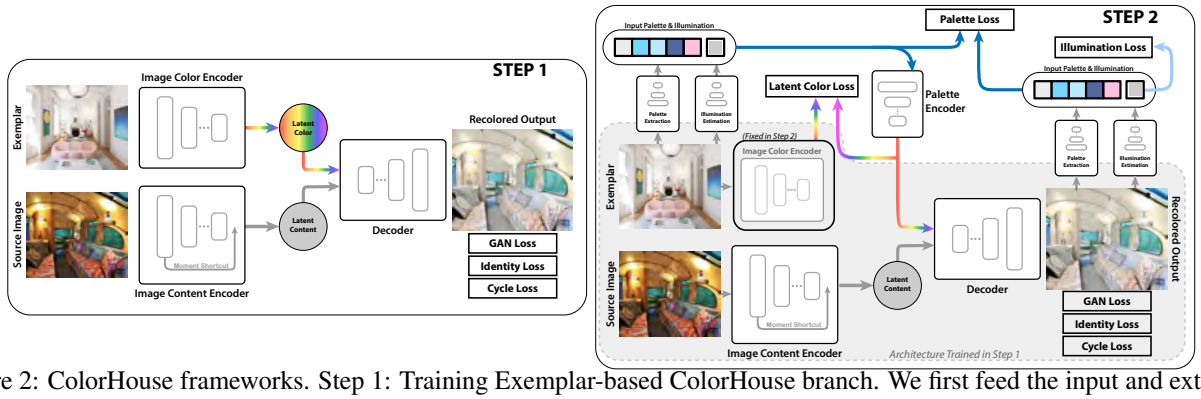

Figure 2: ColorHouse frameworks. Step 1: Training Exemplar-based ColorHouse branch. We first feed the input and extracted color latent vector into the model to get an output. Step 2: Training Color palette-based ColorHouse branch. We fix the exemplar based vector extractor. We feed the input, palette, and the corresponding illumination into the model. In contrast to previous methods, we push the color latent vector toward that of the exemplar-based extractor.

2. Related Work

Exemplar-based Color Manipulation. Previous approaches have explored different ways to map the distribution of colors in an input image to that of a target for different applications. For example, several approaches to color grading [4, 17, 20, 22] estimate constrained mappings from the colors in an input video to the distribution of colors in a target. These approaches typically work well for more subtle tonal changes, but are not designed for the dramatic sort of recoloring addressed in our work.

Perhaps the most widely-used approach to recolorization is Histogram matching [18], which seeks to adjust the color histogram of a source image to match a target. One improvement on this approach is the Monge-Kantorovich solution [19] which is derived from the Monge-Kantorovicth theory of mass transportation. These approaches work impressively well for hueristics but have no way to learn from or adapt to specific image domains.

Other work focuses on conditioning image synthesis on a color objective while using an adversarial loss to prevent a network from generating implausible images. For example, HistoGAN [1] uses a color histogram-based objective to control the color of GAN-generated images. Neural Style Transfer [8, 9] separates and recombines the image content and style of natural images which can produce plausible recolored images in the style of a target. ColorMatcher [10] is a recent approach that does impressively well on color grading applications. These approaches are perhaps most similar to our own in design, but focus on more general and less dramatic recolorization. Also, to our knowledge, ours is the first of this kind to explicitly target reflectance in recolorization without first needing to solve for an intrinsic decomposition of the scene. This trait helps us move away from color grading-like results toward explorations that reflect different design decisions in domains like interior decorating.

Palette-based Color Manipulation. Many works [2, 7, 15, 25, 26] have used color palettes to guide the editing of color. Like exemplar-based work, some approaches achieve impressive results in limited scenarios by relying on hueristics. For example, palette-based photo recoloring [5] introduces a simple interactive tool for recoloring images with color palettes. The palette-based image decomposition algorithm of [23, 24] decomposes the image into a set of additive mixing layers, each of which corresponds to a palette color applied with varying weight. As with their exemplar-based counterparts, these approaches are limited in their ability to adapt to different domains.

3. Method

ColorHouse comprises two network-based recoloring paths. The first gets its latent representation of color from an exemplar. The second conditions recoloring on a target palette expressed as five RGB colors and a target illumination in the form of one additional RGB color.

3.1. Architecture

In Fig 2, we illustrate the system pipeline. In a notable departure from most previous neural approaches to recolorization, our system both conditions on and generates full RGB images, rather than conditioning on the L channel and generating the AB channels in CIE LAB space. This makes our task even less constrained, but lets us explore a wider range of recolorings, which is especially useful for design applications.

The exemplar-based ColorHouse generator is trained in the first stage, shown on the left side of Fig 2, as a style transfer task. Here, unlike most style transfer applications, we want to bias our style representation toward capturing color without explicitly constraining output luminance. We do this by providing the content encoder (which encodes our input image) a distinct advantage over the color encoder

when representing image structure. We create this advantage by adding a moment shortcut [14] to the content encoder, which passes on structural information directly from the early layers. The moment shortcut extracts the mean and standard deviation based on positional normalization for use in the later layers; the extracted information captures structural information in the input and this connection can boost the training performance while preserving the structure. We then feed the input into the recolorization backbone and the color latent vectors into the intermediate layers. Finally, we feed our latent content vector along with the latent color representation from our color encoder to a decoder network for synthesis.

In step 2 of our approach we learn a mapping from vectors consisting of 5 palette colors and one illumination color to the latent color space learned in step 1. For this we need loss functions that can relate palettes to images. In other words, we need a differentiable means of estimating the color palette and illumination of a given image.

Color Palette Extraction. There is a popular and widely-used tool for estimating color palettes in images, called Color Thief [6]. However, Color Thief itself is based on a clustering algorithm and is not differentiable. We address this by training a simple network, illustrated in appendix, to act as a differentiable approximation of Color Thief.[1] The palette extractor consists of two 3×3 stride-1 convolutional layers. After each layer, we apply an adaptive pooling layer to downsample the features. In the end, we apply two linear layers to model interdependencies between channels and learn channel-wise feature responses adaptively [11]. We train on ImageNet [13] for generality and use the L_1 distance between palettes as the loss function [12]. Empirically, we find the model works quite well and produces reasonable color palettes for arbitrary inputs.

Illumination. Previous palette-based methods do not distinguish between solutions that change reflectance and solutions that change illumination. To disambiguate these solutions, we condition our palette-based generator on a target illumination color. In general this could create a conflict between the latent palette and illumination colors—for example, if any palette color is orthogonal to the illumination. However, in this case, our end goal is not to use our latent vector to explore different illumination colors—after all, that is easy to achieve with simple color transformations—but to ensure that our network does not use illumination as a shortcut to achieve the target palette. Employing a strategy similar to the one we used for palette extraction, we use a differentiable network to predict the illumination. We use the network from [3, 16] to estimate illumination, and augment pre-training with the illumination augmentation strategy from the same work. At test time, the default latent illumination color is set to mid-gray, pushing the network to generate white-balanced outputs.

[1] https://github.com/fengsp/color-thief-py

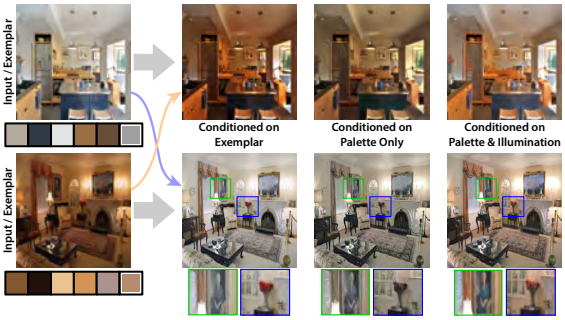

Figure 3: Here we compare our exemplar-based approach with what happens if we extract a palette and illumination from the exemplar and feed it to our palette-based approach with and without the illumination loss. The task here is applying the color from each of the images on the left to the other (i.e., swapping their color schemes). The exemplar approach, which receives the most direct information from the exemplar, is able to match the exemplar's overall visual feel most closely. The palette-only approach often results in color grading-like results satisfy the palette objective by affecting overall tone. Adding our illumination loss pushes this solution toward images that modify reflectance.

Once we have our networks for extracting palette and illumination, we have a differentiable way to self-supervise mapping input image palettes and illumination to the latent color space we learned in step 1. We use this to train the palette encoder shown in step 2 of Fig. 2, which provides a path from color palettes and illumination to the decoder for synthesis.

Also, we add illumination adjustment, aiming to push the neural networks to interpret and control the reflection and illumination separately. To justify this point, we display an example to compare our examplar-based approach to our palette-based approach with and without the illumination loss in Figure 3. We explain the detailed loss function in appendix.

4. Experiments

4.1. Comparison with Other Methods

We now compare ColorHouse with several widely-used and/or recent methods.

Comparison with Exemplar-based Methods We randomly select interior design photos from the web and compare our method with several competitive baselines: Neural Style Transfer [9], ColorMatcher [10] and its included color grading algorithms Multi-Variate Gaussian Distribution (MVGD) transfer [17], Monge-Kantorovich solution [19], and classical histogram matching [18]. We randomly select 500 test images and compare the average quantitative results using Inception Score [21]. Inception Score is a widely used objective metric for evaluating the quality of

Figure 4: Comparison with exemplar-based methods on interior design photos. Please zoom in for details.

Method	Inception Score ↑
Neural Style Transfer	2.17 ± 0.5
Histogram Matching	2.38 ± 0.3
MVGD	2.27 ± 0.6
Monge-Kantorovich	2.22 ± 0.2
ColorMatcher	2.32 ± 0.2
ColorHouse	2.49 ± 0.1

Table 1: Inception Score comparison between ColorHouse and several competitive baselines. The higher the better.

generated images that correlates with human judgment.[2] We show the results in Table 1, we could notice that ColorHouse achieves better performance which justify it effectiveness in generating natural images that humans find visually realistic. We display the qualitative comparison in Fig 4, it could be observed that Neural Style Transfer is able to transfer the color style but brings lots of artificial textures that largely affect the generated results and makes the outputs very unnatural given an natural photo. MVGD algorithm causes serious global color artifacts and pushes the output to an incorrect color pattern. Monge-Kantorovich, Histogram Matching and default ColorMatcher works well, however, they either generate distorted color or fail to translate the exact color style from the reference image to the input. For example, in the first row, Histogram Matching causes color distortion and unmatched color region(the floor turns to be red), leading to unnatural outputs. And in the third row, histogram matching recolorizes the wall as yellow, Monge-Kantorovich and default ColorMatcher generates dark outputs, which are inconsistent with reference images. Furthermore, to justify the efficiency of ColorHouse, we also conduct a comparison on BAM dataset, we display the results in Fig 5, and observe similar advantages of ColorHouse in generating natural-looking output that meet the objective.

Comparison with Palette-based Methods Color palette has been an very useful tool to change the color. However, it is challenging to generate natural images and avoid color distortion using an arbitrary color palette. ColorHouse benefits from the GAN framework so it is able to push the outputs to be natural and interprets the new color pattern very well based on the given color palette. In Fig 6, we

[2]We refer to https://github.com/sbarratt/inception-score-pytorch for Inception Score implementation.

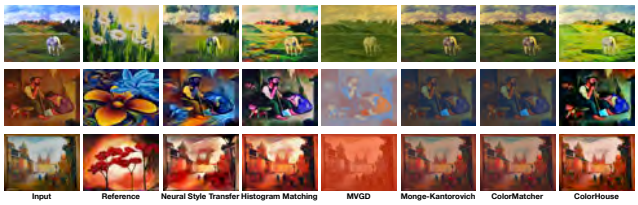

Figure 5: Comparison with exemplar-based methods on BAM dataset. Please zoom in for details.

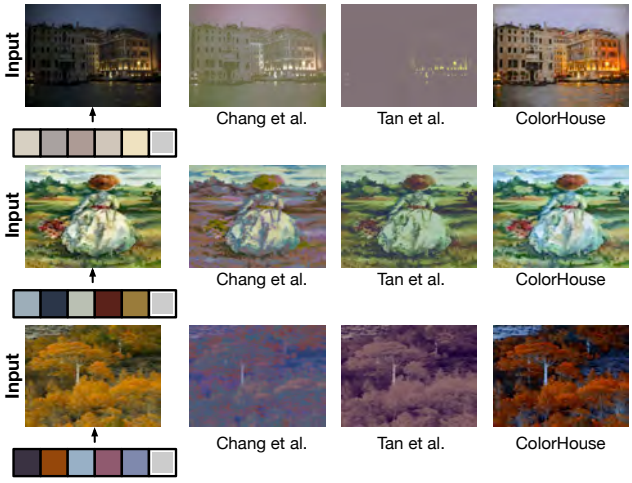

Figure 6: Comparison with palette-based methods.

compare ColorHouse with several most widely-used baselines [5, 23]. Since the provided color palettes contains two totally different color, it is very hard for baselines to generate a promising output. While for ColorHouse, we observe that it is able to achieve natural looking results that match different color palettes. ColorHouse offers clear advantages in these comparisons as well.

User study. In addition to the above mentioned metrics, we ran a user study with 50 particpants and 100 randomly selected sets of examples. Each example is generated by our method with and without the illumination loss (see Fig. 3 for demo). 91% of the time, participants responded that the output produced by our system is the most faithful to the corresponding exemplar reflectance. The results suggest that most users chose images by our method with illumination loss over those from the baselines by a large margin.

5. Conclusion

In this paper, we presented an efficient deep learning framework called ColorHouse that enables fast and flexible neural recolorization. ColorHouse provides a general solution both for exemplar based and color palette-based recolorization, as well as illumination adjustment for arbitrary inputs.[3]

[3]Acknowledgement: SJB's work was supported in part by the Pioneer Centre for AI, DNRF grant number P1. SJB and BL's work were supported in part by Meta.

References

[1] Mahmoud Afifi, Marcus A. Brubaker, and Michael S. Brown. Histogan: Controlling colors of gan-generated and real images via color histograms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2

[2] Hyojin Bahng, Seungjoo Yoo, Wonwoong Cho, David Keetae Park, Ziming Wu, Xiaojuan Ma, and Jaegul Choo. Coloring with words: Guiding image colorization through text-based palette generation. In *Proceedings of the european conference on computer vision (eccv)*, pages 431–447, 2018. 2

[3] Simone Bianco and Claudio Cusano. Quasi-unsupervised color constancy. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12212–12221, 2019. 3

[4] Nicolas Bonneel, Kalyan Sunkavalli, Sylvain Paris, and Hanspeter Pfister. Example-based video color grading. *ACM Trans. Graph.*, 32(4):39–1, 2013. 2

[5] Huiwen Chang, Ohad Fried, Yiming Liu, Stephen DiVerdi, and Adam Finkelstein. Palette-based photo recoloring. *ACM Trans. Graph.*, 34(4):139–1, 2015. 2, 4

[6] Lokesh Dhakar. Color thief, 2015. 3

[7] Chie Furusawa, Kazuyuki Hiroshiba, Keisuke Ogaki, and Yuri Odagiri. Comicolorization: semi-automatic manga colorization. In *SIGGRAPH Asia 2017 Technical Briefs*, pages 1–4. 2017. 2

[8] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015. 2

[9] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016. 2, 3

[10] Christopher Hahne and Amar Aggoun. Plenopticam v1.0: A light-field imaging framework, 2020. 2, 3

[11] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 3

[12] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 3

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012. 3

[14] Boyi Li, Felix Wu, Kilian Q Weinberger, and Serge Belongie. Positional normalization. In *Advances in Neural Information Processing Systems*, pages 1620–1632, 2019. 3

[15] Jingbei Li, Huaxin Xiao, Diaoyin Tan, Maojun Zhang, and Yu Liu. Image colorization based on texture by using of cnn. In *2019 IEEE 4th International Conference on Image, Vision and Computing (ICIVC)*, pages 167–171. IEEE, 2019. 2

[16] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 3

[17] Craig A Mertler and Rachel Vannatta Reinhart. *Advanced and multivariate statistical methods: Practical application and interpretation*. Taylor & Francis, 2016. 2, 3

[18] Laszlo Neumann and Attila Neumann. Color style transfer techniques using hue, lightness and saturation histogram matching. In *Computational Aesthetics*, pages 111–122, 2005. 2, 3

[19] François Pitié and Anil Kokaram. The linear monge-kantorovitch linear colour mapping for example-based colour transfer. 2007. 2, 3

[20] Erik Reinhard, Michael Adhikhmin, Bruce Gooch, and Peter Shirley. Color transfer between images. *IEEE Computer graphics and applications*, 21(5):34–41, 2001. 2

[21] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *arXiv preprint arXiv:1606.03498*, 2016. 3

[22] Haleh H Shenas and Victoria Interrante. Compositing color with texture for multi-variate visualization. In *Proceedings of the 3rd international conference on computer graphics and interactive techniques in Australasia and South East Asia*, pages 443–446, 2005. 2

[23] Jianchao Tan, Jose Echevarria, and Yotam Gingold. Efficient palette-based decomposition and recoloring of images via rgbxy-space geometry. *ACM Transactions on Graphics (TOG)*, 37(6):1–10, 2018. 2, 4

[24] Jianchao Tan, Jyh-Ming Lien, and Yotam Gingold. Decomposing images into layers via rgb-space geometry. *ACM Transactions on Graphics (TOG)*, 36(1):1–14, 2016. 2

[25] Qing Zhang, Chunxia Xiao, Hanqiu Sun, and Feng Tang. Palette-based image recoloring using color decomposition optimization. *IEEE Transactions on Image Processing*, 26(4):1952–1964, 2017. 2

[26] Changqing Zou, Haoran Mo, Chengying Gao, Ruofei Du, and Hongbo Fu. Language-based colorization of scene sketches. *ACM Transactions on Graphics (TOG)*, 38(6):1–16, 2019. 2