PERCEPTION PRESERVING DECOLORIZATION

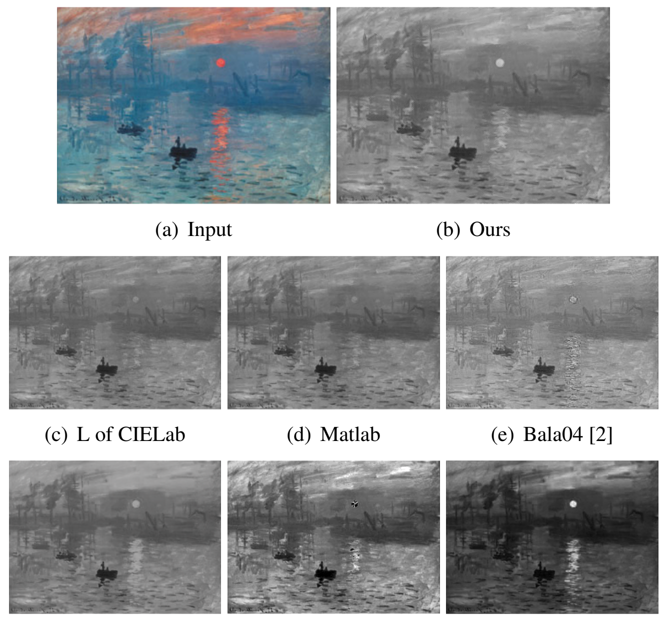
Bolun Cai, Xiangmin Xu, Xiaofen Xing*

South China University of Technology, Guangzhou, China
caibolun@gmail.com, {xmxu, xfxing}@scut.edu.cn

ABSTRACT

Decolorization is a basic tool to transform a color image into a grayscale image, which is used in digital printing, stylized black-and-white photography, and in many single-channel image processing applications. While recent researches focus on retaining as much as possible meaningful visual features and color contrast. In this paper, we explore how to use deep neural networks for decolorization, and propose an optimization approach aiming at perception preserving. The system uses deep representations to extract content information based on human visual perception, and automatically selects suitable grayscale for decolorization. The evaluation experiments show the effectiveness of the proposed method.

Index Terms— Color-to-gray conversion, perception preserving, deep neural networks.



(a) Input (b) Ours

(c) L of CIELab (d) Matlab (e) Bala04 [2]

(f) Color2Gray [3] (g) Rasche05 [4] (h) Lu12 [5]

Fig. 1. Color-to-gray conversion. We use deep neural networks to preserve perception for color-to-gray conversion.

1. INTRODUCTION

Image decolorization is the process of transforming color image to a grayscale one, which is required in many single-channel image processing applications, such as digital printing, photograph rendering, and image debluring. The general goal is to use the limited range in grayscale to preserve as much as possible the color information.

Color-to-gray conversion is to extract a single-channel representation from the three-channel color space, which inevitably suffers from information loss. The intuitive methods would easily lose salient structures and important information, such as the lightness (L) channel in the CIELab [1] color space shown in Fig. 6(a). Therefore, there are many decolorization methods proposed to address this problem. Depending on the solution space, they can be broadly divided into local adjustment and global mapping.

Local adjustment methods take account of independent or neighboring pixels in the color image, and usually rely on the local chrominance edges for enhancement. The classical color-to-gray model (Matlab built-in $rgb2gray()$ function) assumes that the grayscale image (Fig. 6(c)) is a constrained linear combination of color channels (R, G and B), where the weights are fixed as 0.2989, 0.5870, 0.1140, respectively. As shown in Fig. 1(e), Bala *et al.* [2] added high-frequency components to enhance chrominance edges. Although local adjustment methods preserve local features well, they have the shortcoming that may lose color contrast (Fig. 6(c)) or produce halo artifacts (Fig. 1(e)).

Global mapping methods try to minimize the objective functions, which requires structure descriptors extracted from the grayscale image are close to the original color image. As shown in Fig. 1(f), Color2Gray [3] enforced color contrast optimization to preserve saliency between pixel pairs. Rasche *et al.* [4] directly constrained the color difference and constructed a quadratic function to optimize the grayscale image shown in Fig. 1(g). In Fig. 1(h), Lu *et al.* [5] employed bimodal energy function to maximally preserve the color contrast. However, global mapping methods ignore local lightness perception, which will produce inconsistent result to hu-

*Xiangmin Xu is the corresponding author. This work is supported by Natural Science Foundation of China (No.61702192, No.61751202 and U1636218) and Fundamental Research Funds of Central Universities (2017MS045).

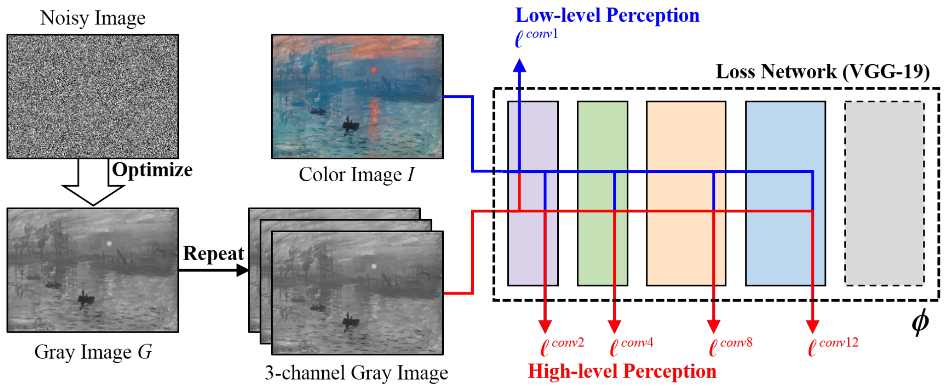


Fig. 2. Perception preserving decolorization. We use a loss network (VGG-19) pretrained for object categorization to define multi-level perceptual loss functions, which measure perceptual differences between the grayscale and color images. The loss network remains fixed during the optimization process.

man vision, such as the sun in Fig. 1(g) and the sea-surface in Fig. 1(h).

Inspired by image style transfer [6], we propose a **perception preserving** decolorization method focusing on both **local** and **global** information. In this work, we show how to use the deep features to minimize perceptual loss between grayscale images and color images.

2. PERCEPTION PRESERVING DECOLORIZATION

To transfer the color image I to a grayscale image G, we generate a grayscale image that matches the perception representation of original image. As shown in Fig. 2, we optimize a single-channel noisy image to minimize the representation distance with the three-channel color image in a number of layers of the convolutional neural network.

2.1. Deep Perceptual Loss

We define a perceptual loss function that measures low-level and high-level perceptual differences between color and gray images. The perceptual measures are generated on a VGG network [7], which was pretrained for object categorization [8]. In this paper, the perceptual feature is provided by a normalised version of the 16 convolutional and 5 pooling layers of the 19-layer VGG network (VGG-19).

Generally each layer in the network includes a linear filter (*conv*) and non-linear activation function (*relu*), corresponding to dense perception and sparse perception, respectively. The dense perception describes the convolutional output for high-order cognition, and the sparse perception describes the convolutional input for feature selection. The perceptual loss encourages the color image I and the gray image G having similar perceptual representations computed by the loss network ϕ. Let $\phi^j_{conv}(\cdot)$ and $\phi^j_{relu}(\cdot)$ respectively be the dense and sparse activations in the j-th layer, where $\phi^j(\cdot)$ are the feature maps with the shape $C_j \times H_j \times W_j$. The perceptual loss is defined as the squared-error distance:

$$\ell^j(G,I) = \frac{1}{C_j H_j W_j}\left(\left\|\phi^j_{conv}(G) - \phi^j_{conv}(I)\right\|_2^2 + \left\|\phi^j_{relu}(G) - \phi^j_{relu}(I)\right\|_2^2\right). \tag{1}$$

2.2. Color-to-Gray Conversion

For color-to-gray conversion, we synthesise a grayscale image that matches the perceptual representation of the color image. Given a color image I and a layer set J to perform perception reconstruction, a grayscale image G is generated by solving the problem

$$\arg\min_G \sum_{j\in J} \lambda_j \ell^j(G,I), \tag{2}$$

where λ_j are the weighting factors for each layer, G is initialized with white noise. Here we use ADAM optimizer [9], which works best and converges fastest for image decolorization. The derivative of perceptual loss ℓ with respect to the activations in the j-th layer equals

$$\frac{\partial \ell^j}{\partial \phi^j(G)} = \frac{2}{C_j H_j W_j}\Big(\big(\phi^j(G) - \phi^j(I)\big) + \big(\max\left(\phi^j(G),0\right)\left(\phi^j(G) - \max\left(\phi^j(I),0\right)\right)\big)\Big), \tag{3}$$

where $\phi^j(\cdot)$ is the terse expression of convolutional output $\phi^j_{conv}(\cdot)$.

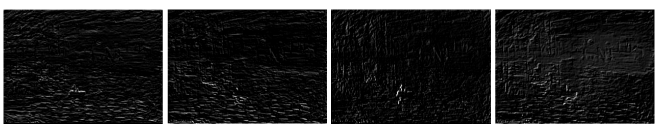
We find that unconstrained optimization of Equation (2) typically results in grayscale images whose pixels fall outside the range $[0,255]$. To avoid irrational values and reduce the solution space, we minimize Equation (2) using projected ADAM by clipping the grayscale image G to the range $[0,255]$ at each iteration.

2.3. Analysis

We discuss how the proposed method is related to ideas in traditional image decoloration, including edge preserving and saliency preserving.

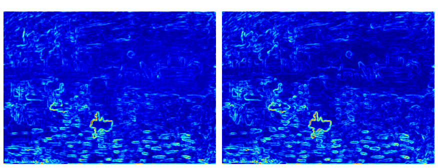
2.3.1. Low-level Perception and Edge Preserving

A common problem of color-to-gray conversion is that the distinction between two different colors with similar lightness is lost. In [2], edge preserving is introducing to distinguish adjacent colors and extract high-frequency information into the lightness channel. For deep neural networks, the chrominance edge information is picked up by the shallow neurons, e.g. the first convolutional layer *conv*1. Therefore, minimizing the low-level perceptual loss is equivalent to edge preserving. In Fig. 3, we compare the output of *conv*1 in VGG-19 and Sobel [10] edge detector. We select some representative feature maps (Fig. 3(a)), and overlay them to produce a similar edge map (Fig. 3(b)). As we can see, the low-level perception effectively extracts high-frequency components of the chrominance edges.



Horizontal edges Vertical edges

(a) Representative low-level features of Fig. 1(a), from left to right, #11, #27, #52, and #24 feature map extracted by *conv*1 in the VGG-19.

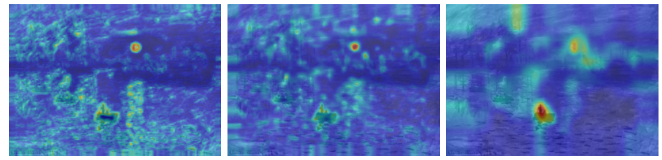


(b) Overlay of Fig. 3(a) (c) Sobel [10]

Fig. 3. Low-level perception and edge preserving.

2.3.2. High-level Perception and Saliency Preserving

Due to the center-surround visual cells [11], human vision system does not perceive absolute values, and instead chrominance and luminance perception are based upon relative assessments. Color2Gray [3] maintains the saliency by mapping chrominance changes in the color image to lightness changes in the grayscale image. Deep neural networks as biologically inspired models have the similar ability of saliency preserving. In [12], fully-convolutional neural networks trained on object categorization is proven effective for saliency detection. As shown in Fig. 4, the average activation of sparse perceptions, including *relu*4, *relu*8, and *relu*12, is extracted by VGG-19. Based on the saliency of visual attention, the sun and boats are located by the high-level perception.



(a) *relu*4 (b) *relu*8 (c) *relu*12

Fig. 4. High-level perception and saliency preserving.

3. EXPERIMENTS

3.1. Implementation Details

For all decolorization experiments, we compute feature reconstruction loss at low-level perceptual layer (the 1-th layer) and high-level perceptual layers (the 2,4,8,12-th layer). Here these high-level perceptual layers are the feature layers before the pooling layers in VGG-19. The weighting factor λ of low-level perception is set to 1, and the factors of high-level perception decrease by half from 1 to 0.125 along with network depth. Initial learning rate is set to 0.5, and final learning rate is set to 0.0001. The learning rate is updated by half when the improvement of the loss function is smaller than a threshold $\mu = 10^{-4}$. Based on the parameters above, our implementation uses PyTorch [13], and optimizing a grayscale image with the 256×256 size takes roughly 30s on a single Nvidia GeForce GTX 1080 GPU.

3.2. Comparison with other methods

Fig. 5 (e) shows the decolorization results[1] with perception preserving. For qualitative comparison, this paper focuses on 4 identified challenging images collected in [14]. We compare ours with 3 representative methods: Bala04 [2] and Color2Gray [3] respectively are classical local and global methods, and Lu12 [5] is a recent state-of-the-art decolorization. Bala05 [2] method can preserve the color differences, but produces some unnatural textures (the highlight edges in image *text*) and amplifies noise (the backlit region in image *impatient*). For Color2Gray [3], the point and detail features perceived in a color image cannot be seen in the converted grayscale image, such as the spots on pear skin in image *fruits*. Lu12 [5] method reduces strict order constraint and maximally preserves the contrast, which results strange color mapping, e.g. the red region in image *impatient* and *tulips*.

To quantitatively assess the decolorization methods, the color contrast preserving ratio (CCPR) [15] is calculated and listed below each subfigure in Fig. 5. CCPR measures that the color difference in the CIELab space is smaller than a certain value τ. It is suggested in [16] that when $\tau = 6$, CCPR becomes nearly invisible in human vision. The higher CCPR represents a better color-to-gray quality. The proposed

[1] More results and comparisons can be found at `https://caibolun.github.io/deepdecolor/`

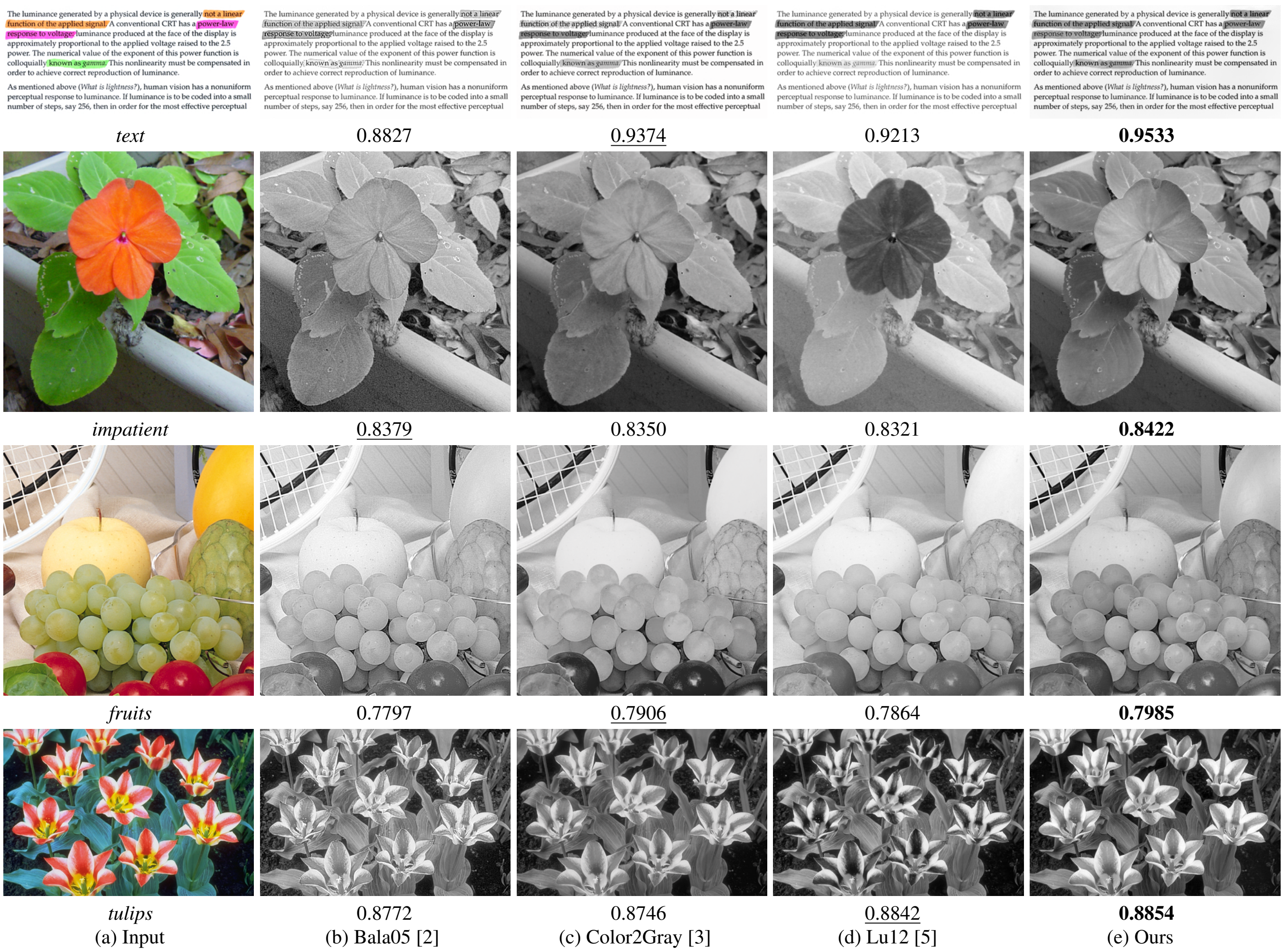


Fig. 5. Color-to-grayscale conversion comparison of 4 images collected from [14].

method achieves the best scores in agreement with our subjective experience.

3.3. Failure Cases

The proposed method has a common limitation for unnatural images shown in Fig. 6. Without rich content information in the first image, the deep neural network cannot extract effective perception to generate high-quality grayscale image. Therefore, it may exhibit artifacts near the edges. For the second image, colour-blindness test chart challenges human vision system, and the proposed method is also hard to perceive it. Even so, the proposed method is still better than the lightness channel in the CIELab color space.

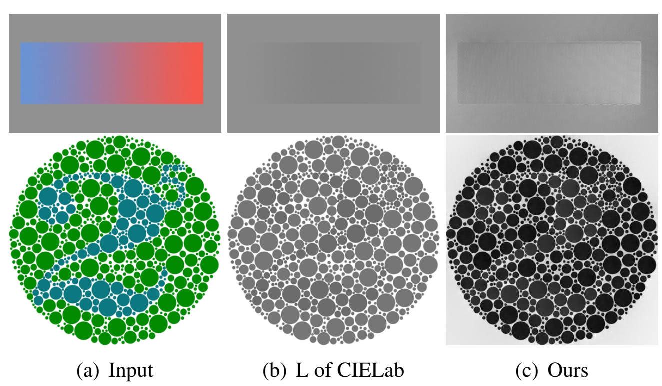


Fig. 6. Failure cases.

4. CONCLUSION

In this paper, we have presented a novel optimization approach based on deep neural networks for image decolorization. In future work, we will combine the benefits of feed-forward image conversion [17] and optimization-based methods for image generation by training feed-forward decolorization networks with perception preserving.

5. REFERENCES

[1] Richard S. Hunter, "Photoelectric color difference meter," *J. Opt. Soc. America*, vol. 48, no. 12, pp. 985–993, 1958.

[2] Raja Bala and Reiner Eschbach, "Spatial color-to-grayscale transform preserving chrominance edge information," in *Color and Imaging Conference*, 2004, pp. 82–86.

[3] Amy A. Gooch, Sven C. Olsen, Jack Tumblin, and Bruce Gooch, "Color2gray: salience-preserving color removal," in *ACM SIGGRAPH*, 2005, pp. 634–639.

[4] Karl Rasche, Robert Geist, and James Westall , "Recoloring images for gamuts of lower dimension," *Computer Graphics Forum*, vol. 24, no. 3, pp. 423432, 2005.

[5] Cewu Lu, Li Xu, and Jiaya Jia, "Contrast preserving decolorization," in *IEEE International Conference on Computational Photography*, 2012, pp. 1 – 7.

[6] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge, "Image style transfer using convolutional neural networks," in *Computer Vision and Pattern Recognition*, 2016, pp. 2414–2423.

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:11409.1556*, 2014.

[8] Jia Deng, Wei Dong, Richard Socher, Li Jia Li, Kai Li, and Fei Fei Li, "Imagenet: A large-scale hierarchical image database," *Proc. of IEEE Computer Vision and Pattern Recognition, 2009*, pp. 248–255, 2009.

[9] Diederik Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[10] Duda Ro and Hart Pe, "Pattern classification and scene analysis," 1973.

[11] R. Beau Lotto and Dale Purves, "The effects of color on brightness," *Nature Neuroscience*, vol. 2, no. 11, pp. 1010–1014, 1999.

[12] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba, "Learning deep features for discriminative localization," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2921–2929.

[13] Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan, "Pytorch: Tensors and dynamic neural networks in python with strong gpu acceleration," 2017.

[14] M. Cadik, "Perceptual evaluation of colortograyscale image conversions," *Computer Graphics Forum*, vol. 27, no. 7, pp. 1745–1754, 2010.

[15] Cewu Lu, Li Xu, and Jiaya Jia, "Contrast preserving decolorization with perception-based quality metrics," *International Journal of Computer Vision*, vol. 110, no. 2, pp. 222–239, 2014.

[16] Hsin Chia Chen and Sheng Jyh Wang, "The use of visible color difference in the quantitative evaluation of color image segmentation," in *IEEE International Conference on Acoustics, Speech, and Signal Processing, 2004. Proceedings*, 2004, pp. iii–593–6 vol.3.

[17] Justin Johnson, Alexandre Alahi, and Fei Fei Li, "Perceptual losses for real-time style transfer and superresolution," in *European Conference on Computer Vision*, 2016, pp. 694–711.