Differential Evolutionary (DE) Based Interactive Recoloring Based on YUV Based Edge Detection for Interior Design

[1]Fitria Wulandari Ramlan
School of Electronics Engineering
Kyungpook National University
Daegu, Republic of Korea, 702 701
rwfitri8@gmail.com

[2]Vikas Palakonda
School of Electronics Engineering
Kyungpook National University
Daegu, Republic of Korea, 702 701
vikas11475@gmail.com

[3]Rammohan Mallipeddi
School of Electronics Engineering
Kyungpook National University
Daegu, Republic of Korea, 702 701
mallipeddi.ram@gmail.com

***Abstract*— In this paper, we present a technique to the real-time recoloring images for use as what users prefer based YUV Algorithm and Differential Evolutionary (DE) Algorithm. An approachment from a combination of several techniques that have been proven for real-time recoloring interior wall paintings. The performance measures of these techniques are good enough for real-time applications. The structural and functional operations of the YUV and DE algorithm are similar to the natural neurons which are completely computational in nature. We apply this technique for the wall house painting, and we validate our approach with a user study and compare against the state of the art, where we show significant improvements. Furthermore, we demonstrate our method extensively on many different types of images interior house and show realistic recoloring.**

Keywords— Recolorization, YUV Algorithm, Differential Evolutionary Algorithm, Image Processing, Computational Image

I. Introduction

Computational applications for image recolorization is one of the most popular discussions for image editing. The complexity and the practical difficulties in the applications involved in this application are very high. Our approach uses a combination of several techniques that have been proven is being developed to recoloring interior wall paintings to make them suitable for a real-time before applying the real color in the wall house. Choosing the right color when a painting is crucially important. However, for selecting the color is pretty daunting. The reason is not the only paradox of preferences, but also because of the inability visualize several color effects on the real walls.

Several approaches to colors description, or some other empirical organizations give rise to various color spaces. Red, Green, and Blue (RGB) are the primary colors of visible light and RGB colors are often appear brighter and more vivid specifically because the light is being projected directly into the eyes of the viewer [1]. The color which we see is the continuous signal of wavelength of electromagnetic radiations. The visible range lies within the 380 to 780 nm [2]. RGB color spaces seems unrealistic for man-machine communications since its metrics does not represent color differences in a uniform scale and the colors are not organized in a manner intuitive for human observers [3]. At some stages one color spaces must be translated in to other color spaces. For example there is a need of printing an image in a printing press if do with RGB to CMYK (Cyan, Magenta, Yellow, and Key Black) Conversion of images. CMYK itself is a subtractive color model that used in color printing [4].

Providing a good colors in a visualizer is a challenging process. Digital painting is different from physical painting which raises some technical difficulties, such as separating objects from walls, identifying edges and corners of the room in images because sometimes there are some images where the wall has irregular edges, and handles the diversity of image attributes caused by various types of cameras and room lighting, such as rooms that vary in style and lighting or images have low resolution. The convenience of image processing application is also very important. Users are prefer with easy, useful and user friendly to be used.

Using YUV algorithm for transferring color into the part of object color image is significantly well, but there are some drawbacks with the representation. For example, the brightness of a pixel may not be changed easily, as it is must be computed from the RGB components. An alternative technique that can be utilized, we use optimization problem that can be solved using Differential Evolution (DE) at real-world [5-8], an Evolutionary Algorithm which is used to eliminate the irrelevant pixel while transferring the color, can significantly improve the quality of the outputs.

On this paper, we propose a new technique to automatically recoloring wall house image as user preference using optimal problem-solving techniques. Among the problem-solving techniques, we highly prefer to use machine learning algorithms because it is significantly high accuracy. The techniques of incorporating intelligence in machine learning algorithms have produced significantly superior results.

II. Related Work

A. Interactive Evolutionary Computation (IEC)

Interactive evolutionary computation (IEC) is an optimization method that adopts Evolutionary Computation (EC) among system optimization based on subjective human evaluation [9, 10]. Interactive Evolutionary Computation offers significant benefits over non-interactive approaches, as it removes the necessity to find a mathematical solution for the fitness function. Simply stated, the interactive evolutionary computation is an EC whose fitness function is replaced by a human [11, 12].

B. RGB and YUV Color Spaces

RGB and YUV color spaces are both based upon the perceptual capabilities of human eye. The RGB color space is plainly based on the acquisition capabilities of cone cells in retina, which are able to react to different wavelength [13]. In the YUV color space, the black and white information is separated from the color information. Primarily, YUV was used in analog television standards, when color information was added to the existing luminance channel. To enable backward compatibility for black-and-white transceivers, the chrominance channels were added in a separate subcarrier.

Using a YUV color space, also usually involves loss of information, but for a different reason than in RGB color space. In analog YUV it is popular to use interlacing in chrominance channels (the contrast in luminance channel is more significant information for a human eye than color in chrominance channels). In digital YUV, the signal is usually converted from RGB acquisition hardware, which involves a lossy conversion from RGB to YUV. Therefore, the YUV color space is also a compromise of perceptually-reasoned loss of information [13].

C. YUV Formats and Conversion from RGB

The term *YUV* was used for analog encoding. Nowadays this term is frequently used for analog and digital encoding as well. where *Y* is the monochromatic luminance channel, which we will refer to simply as intensity, while *U* and *V* are the chrominance channels, encoding the color [14]. There are many formulas to convert from *RGB* to *YUV*. Color spaces *YUV*, *YIQ*, YC_bC_r, and YP_bP_r all belong to the *YUV* family. If the *RGB* data has a range of 0-255 (black-white), as is commonly found in PCs, the following equations should be used to maintain the correct black and white levels:

$$\begin{pmatrix} Y' \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 0.257 & 0.504 & 0.098 \\ -0.148 & -0.291 & 0.439 \\ 0.439 & -0.368 & -0.071 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} 16 \\ 128 \\ 128 \end{pmatrix};$$

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1.164 & 0 & 1.596 \\ 1.164 & -0.813 & -0.392 \\ 1.164 & 2.017 & 0 \end{pmatrix} \begin{pmatrix} Y' - 16 \\ Cb - 128 \\ Cr - 128 \end{pmatrix} \quad (1)$$

YC_bC_r is a scaled and offset version of the *YUV* color space. *Y* has an 8-bit range from 16 to 235. C_b and C_r have a range of 16-240. YC_bC_r is used for JPEG compression. Where *Rd, Gd, Bd* represent 8-bit values for red, green and blue color channels.

The basic equations to convert between gamma-corrected RGB and YUV are:

Y = 0.299R' + 0.587G' + 0.114B'
U = -0.147R' – 0.289G' + 0.436B' = 0.492 (B' – Y)
V = 0.615R' – 0.515G' – 0.100B' = 0.877(R' – Y)
R' = Y + 1.140V
G' = Y – 0.395U – 0.581V
B' = Y + 2.032U

D. Review of Differenctial Evolutionary (DE) Algorithm

Differential Evolution (DE) is a population-based, efficient, robust, and direct search method [15-19]. The DE algorithm arguably one of the most powerful stochastic real-parameter optimization algorithms in current use [7]. DE operates through similar computational steps as employed by a standard evolutionary algorithm (EA) [20, 21]. We selected DE since it offers fast convergence rate and capability of working directly with real numbers (threshoding levels).

Like other evolutionary algorithms, DE starts with an initial population vector which randomly generated when no preliminary knowledge is available [5, 6, 22]. Let assume that $X_{i,G}(i = \{1, 2, ..., Np\}$ are Np *Nv*-dimensional parameter vectors of generation G (Np is a constant number which represents the population size) [19, 23]. In order to generate a new population of vectors, for each target vector in population three vectors randomly selected, and weighted difference of two of them is added to the third one. For classical DE, the mutation, crossover, and selection have straightforward procedures as follows [23].

Mutation - For each vector $V_{i,G}$ from generation *G* a mutant vector $V_{i,G}$ is defined by

$$V_{i,G} = X_{r1,G} + F(X_{r2,G} - X_{r3,G}) \quad (2)$$

where $i = \{1, 2, ..., Np\}$ and r_1, r_2, and r_3 are mutually different random integer indices selected from {1, 2, ..., Np}. Further, i, r_1, r_2, and r_3 are different such that $n \geq 4$. $F \in$ is a real constant which determines the amplification of the added differential variation of $(Xr_{2,G} - Xr_{3,G})$ [6, 24]. Larger values for *F* result in higher diversity in the generated population and the lower values in faster convergence.

Crossover - DE utilizes crossover operation to increase diversity of the population. It defines following trial vector:

$$U_{i,G} = (U_{1i,G}, U_{2i,G}, \ldots, U_{Nvi,G}), \quad (3)$$

Where $j = 1, 2, \ldots, N$, and

$$U j_{i,G} = \begin{cases} V_{ji,G} & \text{if } rand_j\,(0, 1) \leq Cr \quad (2) \\ X_{ji,G} & \text{otherwise} \quad (2) \end{cases}$$

$Cr \in (0, 1)$ is predefined crossover constant; $rand_j\,(0, 1)$ is $j^{th} \in [0, 1]$ evaluation of uniform random generator. Most popular value for *Cr* is in the range of (0.4, 1) [25].

Selection - Now it must be decided which vector ($U_{i,G}$ or $X_{i,G}$) should be a member of new generation, *G + 1*. Vector with the higher fitness value is chosen. There are other variations of DE [18, 19].

The structural and functional operations of the YUV and DE algorithm are similar to the natural neurons which completely computational in nature. In the machine vision and computer vision mostly are dominated by the RGB color space. Red, Green and Blue optical filters, combined with Active-Pixel sensors comprise the simplest and most popular color vision acquisition systems. YUV was primarily introduced to add the color information to existing monochromatic channel, it turned out that YUV is also in a way similar to a human vision – the "black and white" information has more impact on the image for human eye than the color information [13]. The combination between this both techniques is a rule based system which ultimately improves the accuracy of the output.

E. Palette Based Recoloring

A method for coloring vector art by palettes based on a probabilistic model [26]. They learn and predict the distribution of properties such as saturation, lightness and contrast for individual regions and adjacent regions, and use the predicted distributions and color compatibility model by [27] to score pattern colorings. adapt the edit propagation method in [28] to obtain a soft image segmentation and recolor an image [29]. While our method works for a pair of initial and final values for each palette entry, they only have the final palette colors. Thus the bulk of their method addresses how to associate pixels in the image with the final palette colors (which can be ill posed, and also leads to a complex and slow method). Intuitive and interactive tool that allows non-experts to recolor an image by editing via a color palette [30]. Their results creating a color palette from an image, and a novel color transfer algorithm that recolors the image based on a user-modified palette.

III. Approach And Methods

Our approach for this experiment is to recoloring the paint color of the interior house using combination techniques like YUV and Differential Evolution (DE) algorithm. We showing proven of the technique of merging between YUV and Different Evolution (DE) algorithm proved to give a significant high result.

Our coloring method is using palette color to choosing which color that user prefer for their interior house. Our network is formed by several subcomponents that form a Directed Acyclic Graph (DAG) and contain important discrepancies with widely-used standard models for the recoloring paint interior house which is when recoloring paint interior, we need to detect the edge. In particular, our model:

- Can process images of any resolution,
- Can directly changes the color style of an image into the recoloring of another image.

For the algorithm *YUV* that we use is given as input an intensity volume *Y (x,y,t)* and outputs two color volumes *U (x,y,t)* and *V (x,y,t)*. To simplify notation we coloring the object by clicking and drawing object with edge detection to make the difference between each color.

Algorithm

Input input image A and mask with color
Process transfering
Output output image C

Begin
Insert the image into layer;
Select the color from palette color;
Applying to the object by click and draw in the part of image that would be recoloring;
/* During recoloring some of pixel on image might will not be selected*/
While (satisfying termination criteria)
/*DE algorithm steps (mutation, crossover, and selection) to reduce an error results*/
Mutation; **/*Eq. 2*/**
Crossover; **/*Eq. 3*/**
Selection; **/*Eq. 4*/**
Calculate fitness value for each individual in the population;
While end
Transferring the color
End

A. Color Palettes for Data Visualization

Color is pre-attentively observed and used to segment the visual environments into objects. This characteristic makes it particularly effective in coding qualitative information. In nominal color coding all colors should be distinguishable, the palette should display no clear ordering and all colors should be perceived as equally important. There should be no perceptual ordering in the representation. The number of colors used to represent nominal data should be restricted to seven or less. The problem is that we have often more items to represent than easily discrimination of information items, but also to convey high-level information at local and/or global level as well.

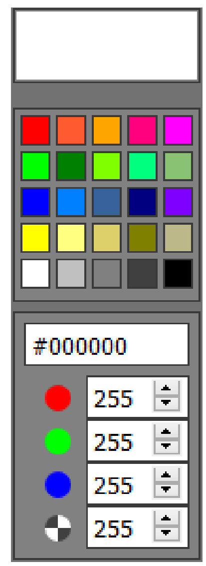

Fig. 1. Colors palette to changing the wall color of interior house. Our GUI shows a representative color palette. The user adjusts a selected palette color with basic colors and custom colors via YUV controller, and the image updates interactively

Overview GUI colors palette. We use a colors palette to changing the object which is in the gui that we can see from figure 1 is from basic colors that already shown after calculated by eq(1).

IV. Results And Discussion

We evaluate different variants of our model as well as comparing against the state of the art. Evaluation is done on a large set of diverse images for with full color of interior design. We also evaluate our model in a user study and find that the output of our model is considered “natural” 92.6% of the time. Furthermore, we also show how it was possible to do the style.

Some sample results of applying the proposed approach presented in figure 3 and the real images is come from figure 2. Note that these are all unconstrained and very challenging images to recoloring and all of these results are generating with user prefer.

Fig. 2. Some of the real images before recoloring proses. Note the diversity of scene types and images are interior house.

Fig.3. Results of our approach on some of the images after transferring color. The diversity of scene types and images are interior house. Note that all these results were obtained automatically and real time.

Conclusions

Translating the painting into digital paint is a challenge than applying in real life. Our approach to this method is working well. Particularly relevant to our application, we learned that the color picker was indeed a useful feature for inspecting selected colors to the interior room with painted walls and furniture, easy to help users to decide more easily on the best color match. Required for the parameter, we still need to increase while applying to the object because some of the pixels are not detected like in the edge of the wall. The experimental simulation showed a percentage of improvements in travel time between images considered around 92.6% of the time.

Acknowledgments

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (2016-0-00564, Development of Intelligent Interaction Technology Based on Context Awareness and Human Intention Understanding).

References

[1] K. Kulshreshtha, A. I. Niculescu, and B. Wadhwa, "On the design and evaluation of Nippon Paint Color Visualizer application–a case study," in *IFIP Conference on Human-Computer Interaction*, 2017, pp. 372-376.

[2] P. Koirala, "RGB Color Spaces," ed.

[3] S. Bianco, F. Gasparini, and R. Schettini, "Color coding for data visualization," in *Encyclopedia of Information Science and Technology, Third Edition*, ed: IGI Global, 2015, pp. 1682-1691.

[4] P. Nishad and R. M. Chezian, "Various colour spaces and colour space conversion algorithms," *Journal of Global Research in Computer Science,* vol. 4, pp. 44-48, 2013.

[5] R. Mallipeddi, P. N. Suganthan, Q.-K. Pan, and M. F. Tasgetiren, "Differential evolution algorithm with ensemble of parameters and mutation strategies," *Applied soft computing,* vol. 11, pp. 1679-1696, 2011.

[6] R. Mallipeddi and P. N. Suganthan, "Differential evolution algorithm with ensemble of parameters and mutation and crossover strategies," in *International conference on swarm, evolutionary, and memetic computing*, 2010, pp. 71-78.

[7] R. Mallipeddi, G. Iacca, P. N. Suganthan, F. Neri, and E. Mininno, "Ensemble strategies in compact differential evolution," in *2011 IEEE Congress of Evolutionary Computation (CEC)*, 2011, pp. 1972-1977.

[8] M. Krithikaa and R. Mallipeddi, "Differential evolution with an ensemble of low-quality surrogates for expensive optimization problems," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 78-85.

[9] H. Takagi, "Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation," *Proceedings of the IEEE,* vol. 89, pp. 1275-1296, 2001.

[10] J. Jung, Y. Matsuba, R. Mallipeddi, H. Funaya, K. Ikeda, and M. Lee, "Evolutionary programming based recommendation system for online shopping," in *2013 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, 2013, pp. 1-4.

[11] H. Takagi, "Interactive evolutionary computation: System optimization based on human subjective evaluation," in *IEEE Int. Conf. on Intelligent Engineering Systems (INES'98)*, 1998, pp. 17-19.

[12] J. Kruse, "Interactive Evolutionary Computation in design applications for virtual worlds," Auckland University of Technology, 2014.

[13] M. Podpora, G. P. Korbas, and A. Kawala-Janik, "YUV vs RGB-Choosing a Color Space for Human-Machine Interaction," in *FedCSIS Position Papers*, 2014, pp. 29-34.

[14] K. Jack, *Video demystified: a handbook for the digital engineer*: Elsevier, 2011.

[15] D. Corne, M. Dorigo, F. Glover, D. Dasgupta, P. Moscato, R. Poli, *et al.*, *New ideas in optimization*: McGraw-Hill Ltd., UK, 1999.

[16] K. Price, R. M. Storn, and J. A. Lampinen, *Differential evolution: a practical approach to global optimization*: Springer Science & Business Media, 2006.

[17] K. V. Price, "Differential evolution vs. the functions of the 2/sup nd/ICEO," in *Evolutionary Computation, 1997., IEEE International Conference on*, 1997, pp. 153-157.

[18] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of global optimization,* vol. 11, pp. 341-359, 1997.

[19] R. Mallipeddi and P. Suganthan, "Differential evolution algorithm with ensemble of populations for global numerical optimization," *Opsearch,* vol. 46, pp. 184-213, 2009.

[20] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE transactions on evolutionary computation,* vol. 15, pp. 4-31, 2011.

[21] R. Mallipeddi and P. N. Suganthan, "Improved adaptive differential evolution algorithm with external archive," in *International Conference on Swarm, Evolutionary, and Memetic Computing*, 2013, pp. 170-178.

[22] R. Tanabe and A. S. Fukunaga, "On the pathological behavior of adaptive differential evolution on hybrid objective functions," in *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, 2014, pp. 1335-1342.

[23] G. C. Onwubolu, "Differential evolution for the flow shop scheduling problem," in *New Optimization Techniques in Engineering*, ed: Springer, 2004, pp. 585-611.

[24] G. Wu, R. Mallipeddi, P. N. Suganthan, R. Wang, and H. Chen, "Differential evolution with multi-population based ensemble of mutation strategies," *Information Sciences,* vol. 329, pp. 329-345, 2016.

[25] S. Das, A. Konar, and U. K. Chakraborty, "Improved differential evolution algorithms for handling noisy optimization problems," in *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, 2005, pp. 1691-1698.

[26] S. Lin, D. Ritchie, M. Fisher, and P. Hanrahan, "Probabilistic color-by-numbers: Suggesting pattern colorizations using factor graphs," *ACM Transactions on Graphics (TOG),* vol. 32, p. 37, 2013.

[27] P. O'Donovan, A. Agarwala, and A. Hertzmann, "Color compatibility from large datasets," in *ACM Transactions on Graphics (TOG)*, 2011, p. 63.

[28] B. Wang, Y. Yu, T.-T. Wong, C. Chen, and Y.-Q. Xu, "Data-driven image color theme enhancement," in *ACM Transactions on Graphics (TOG)*, 2010, p. 146.

[29] X. An and F. Pellacini, "AppProp: all-pairs appearance-space edit propagation," in *ACM Transactions on Graphics (TOG)*, 2008, p. 40.

[30] Y. Chang, S. Saito, K. Uchikawa, and M. Nakajima, "Example-based color stylization of images," *ACM Transactions on Applied Perception,* vol. 2, pp. 322-345, 2006.