Assignment 2

Mehmet Giray Nacakcı, 21989009
Department of Computer Engineering
Hacettepe University
Ankara, Turkey
b21989009@cs.hacettepe.edu.tr

November 16, 2022

1 Introduction

Color transfer between images might sound complicated, yet there is a simple statistical method developed by Reinhard et al. [1]. This method will be directly applied in part 1. In part 2, unlike part 1, this method will be applied to not images as a whole, but to each local regions in the images, for the sake of better color statistics and transfer in local areas in the images.

2 Experiment

2.1 Part 1

Aim is to obtain a new version of a Source image that looks like the Target image's colors. This Result image should have the structure of Source image and colors of Target image. The color transfer is done by simple statistical calculation steps. In Reinhard's algorithm, Calculations are carried out in not [r,g,b] but [l,a,b] color space.

$$I_k = \frac{\sigma_s^k}{\sigma_t^k}(S^k - mean(S^k)) + mean(T^k) \ , k = (\ell, \alpha, \beta)$$

Figure 1: To obtain Resulting image (Ik), Subtract its pixel-wise means from Source image, scale by standard deviations of Source/Target, add means of Target image.

2.2 Part 2

In part 2, color transfer is done from equal rectangular regions of Target to Source images. Same algorithm as Part 1, now applied to not images as a whole, but separately for each individual divided small region.

The advantage is that, means and standard deviations of a local Target image region is now NOT biasing/skewing the whole image's colors, it is being applied to only the assigned

Source image region. This approach is expected to transform colors more accurately locally and on the whole.

Although the instructions suggested 1, my experimentation has lead me to implement two approaches for this part.

2.3 Part 2.1 : Approach 1: SSD

Color transfer is done between "similar" regions. For each rectangular region of a Source image, the most similar Target image region (to get colors from) is found by SSD (Sum of Squared Differences) measure.

This approach is the suggested approach, since similar structures are expected to transform color to similar structures. Such as image subject to image subject, background to background.

Example: 3x3 divided regions,
pairing between "similar" regions,
can be at different positions,
a Target region can be sampled by Source more than once

Source image				Target image		
A'	A'	D'		A	B	C
F'	H'	D'		D	E	F
B'	E'	J'		G	H	J

SSD Input Domain

sumOfSquareDifferences = sum((sourceRegion - targetRegion) ** 2)

In terms of what to input as sourceRegion and targetRegion into the equation,

I tested 4 variations with heightDivide = 10 and widthDivide = 15 parameters:

a) Grayscale : Best version, industry standard.

In RGB space, before calculating SDD, R G B channels are combined into a gray intensity channel. Experimentally, this improves performance and color accuracy, since we are already looking for structurally similar regions, not color.

b) R G B separately

c) L A B separately

d) L (only intensity) channel of LAB

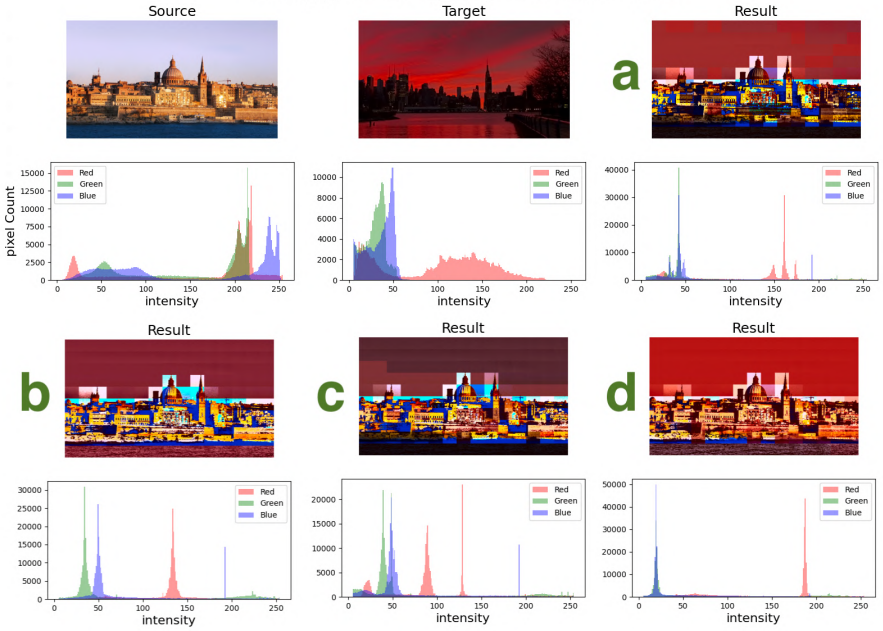
Let MxN be region count in an image. Option b and c, having 3xMxN complexity (since same calculations for 3 channels), performed worse than (20 vs. 8 seconds) option a and d (having MxN complexity).

From images and histograms, what is seen is that, b and c created farther from target color results than a and d.

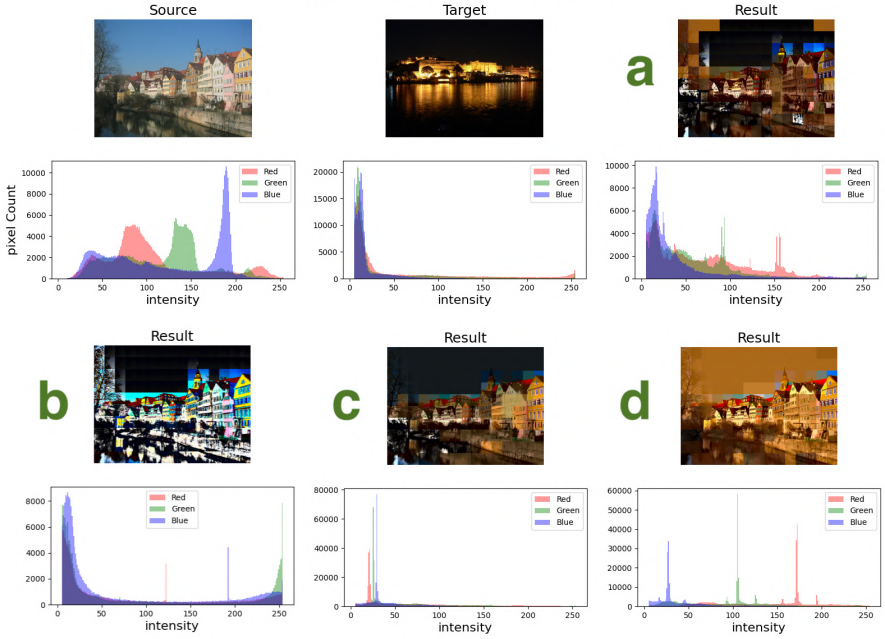
None of b,c,d quite replicated Target image colors as good as option a. As expected and widely used, option a: grayscale is a safe bet.

Therefore, in the next chapters, all (approach 2.1) SSD calculations will be carried out in Grayscale.

PART 2_approach1 : in_01.png Color Transfer Results



PART 2_approach1 : in_24.png Color Transfer Results



2.4 Part 2.2 : Approach 2: Same Position

Color transfer is done between same position regions. For each rectangular region of a Source image, the Target image region (to get colors from) is the same position region in Target image.

In my opinion, this approach enabled me to transform colors more meaningfully since the Resulting image's spacial color distribution looks much more like the Target image, compared to Approach 1.

Example: 3x3 divided regions,
one-to-one pairing between same position regions

Source image				Target image		
A'	B'	C'		A	B	C
D'	E'	F'		D	E	F
G'	H'	J'		G	H	J

3 Comparison of Color Transfer Results

A. Part1(global)

B. Part2.1 Approach1 (local, SSD)

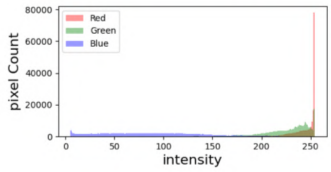
C. Part2.2 Approach2 (local, same position)

and Comparing Different Division counts for B and C

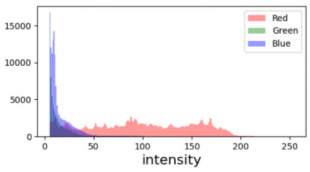
(division count = heightDivide x widthDivide)

Input 9: Color Transfer Results

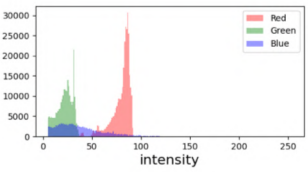
Source



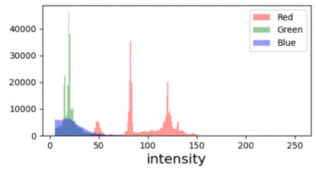
Target



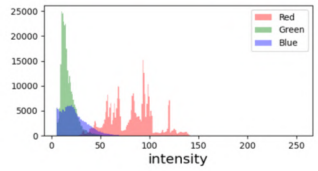
Part 1



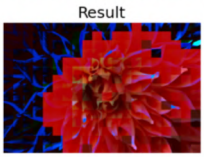
Part 2: Aprroach 1, 4x7 divide

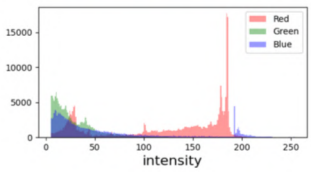


Part 2: Aprroach 2, 4x7 divide

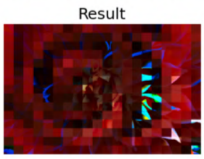


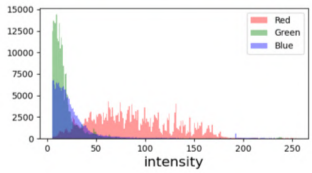
Part 2: Aprroach 1, 15x20 divide



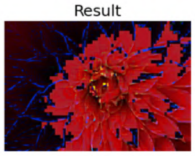


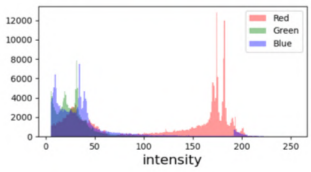
Part 2: Aprroach 2, 15x20 divide



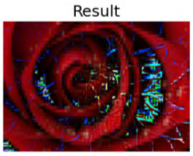


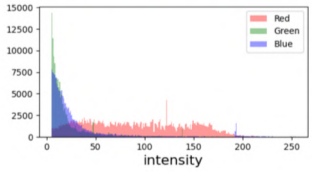
Part 2: Aprroach 1, 40x70 divide



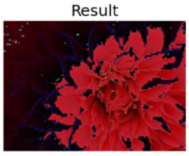


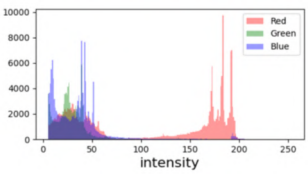
Part 2: Aprroach 2, 40x70 divide



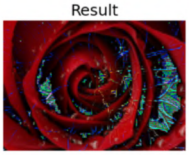


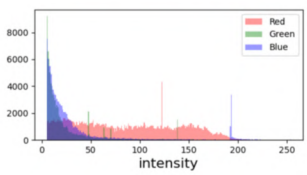
Part 2: Aprroach 1, 80x140 divide





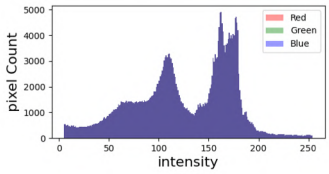
Part 2: Aprroach 2, 80x140 divide



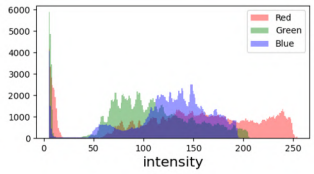


Input 8: Color Transfer Results

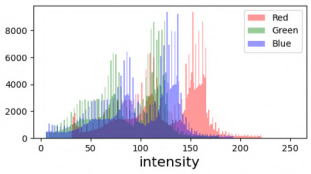
Source



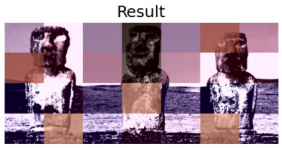
Target

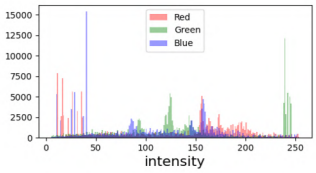


Part 1

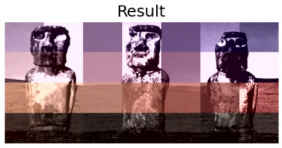


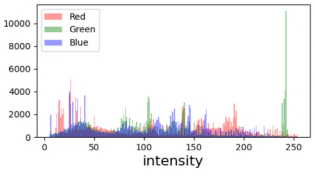
Part 2: Aprroach 1, 4x7 divide



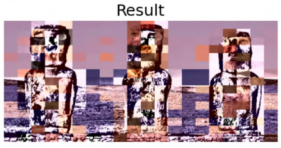


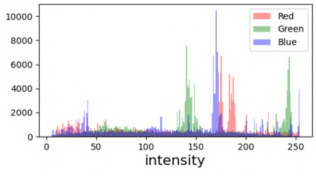
Part 2: Aprroach 2, 4x7 divide



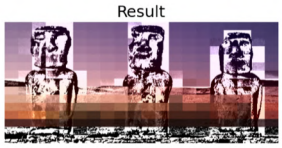


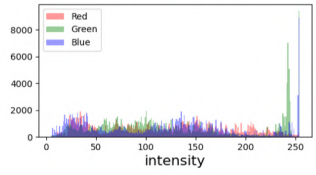
Part 2: Aprroach 1, 15x20 divide



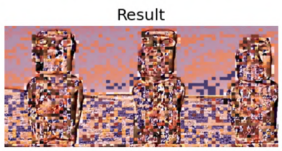


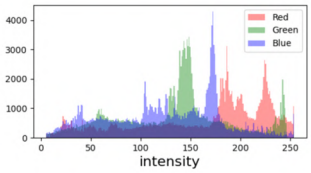
Part 2: Aprroach 2, 15x20 divide



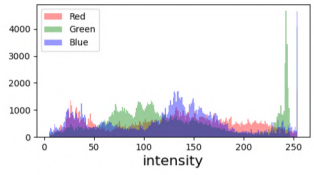


Part 2: Aprroach 1, 40x70 divide

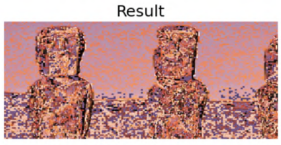


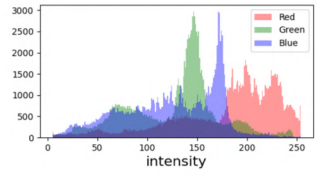


Part 2: Aprroach 2, 40x70 divide

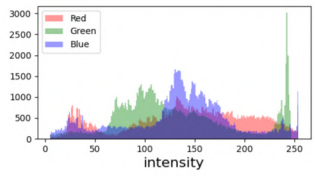


Part 2: Aprroach 1, 80x140 divide



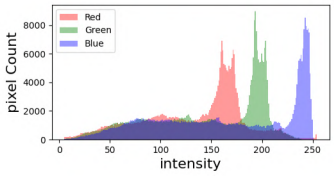


Part 2: Aprroach 2, 80x140 divide

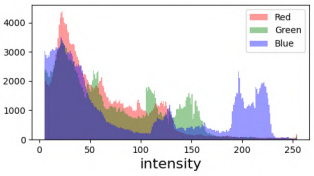


Input 3: Color Transfer Results

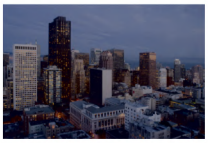
Source

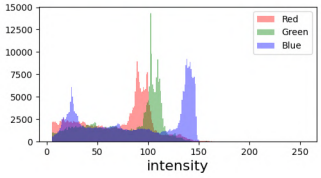


Target

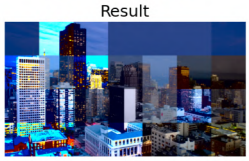


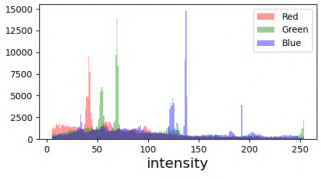
Part 1



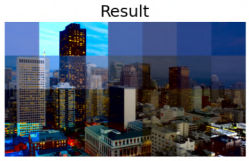


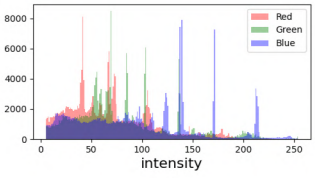
Part 2: Aprroach 1, 4x7 divide



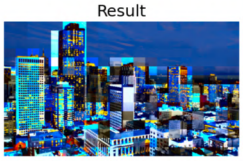


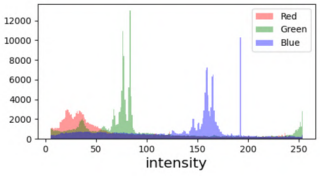
Part 2: Aprroach 2, 4x7 divide



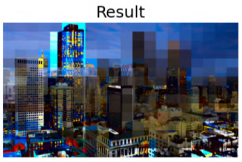


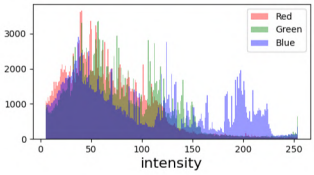
Part 2: Aprroach 1, 15x20 divide



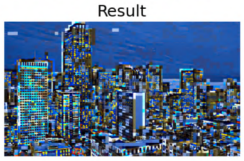


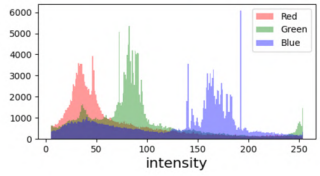
Part 2: Aprroach 2, 15x20 divide



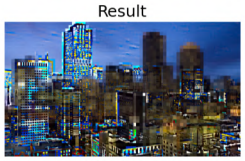


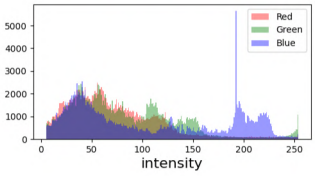
Part 2: Aprroach 1, 40x70 divide



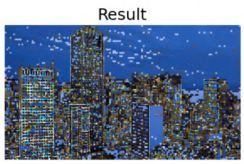


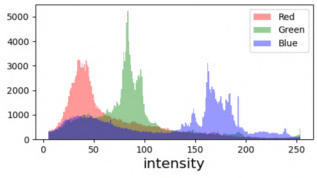
Part 2: Aprroach 2, 40x70 divide



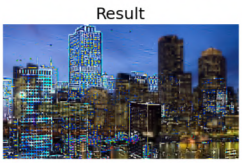


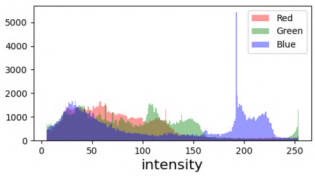
Part 2: Aprroach 1, 80x140 divide





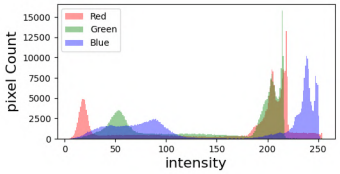
Part 2: Aprroach 2, 80x140 divide



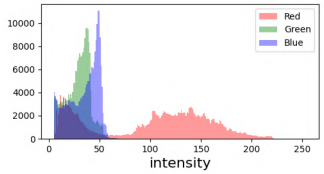


Input 1: Color Transfer Results

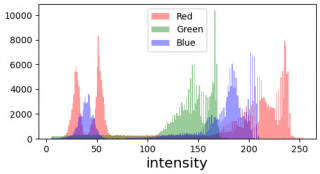
Source



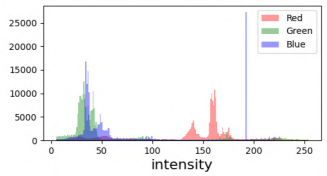
Target



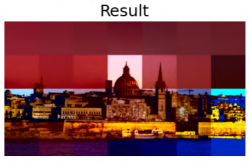
Part 1

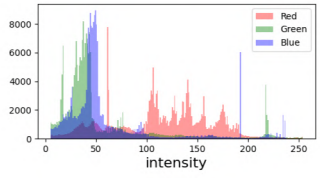


Part 2: Aprroach 1, 4x7 divide

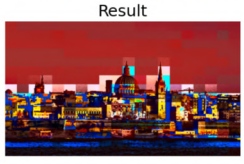


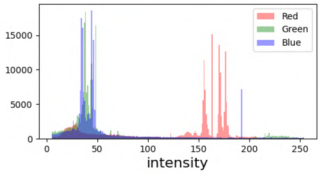
Part 2: Aprroach 2, 4x7 divide



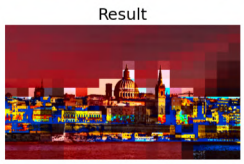


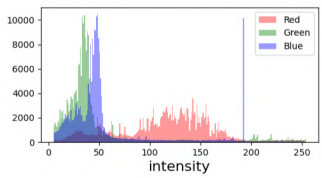
Part 2: Aprroach 1, 15x20 divide



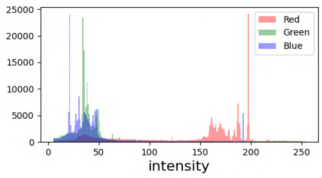


Part 2: Aprroach 2, 15x20 divide

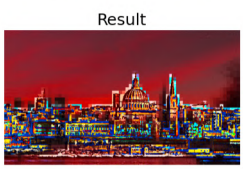


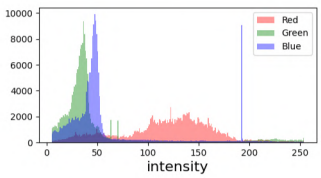


Part 2: Aprroach 1, 40x70 divide

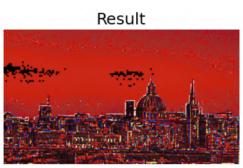


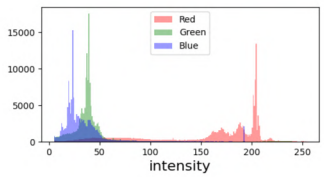
Part 2: Aprroach 2, 40x70 divide



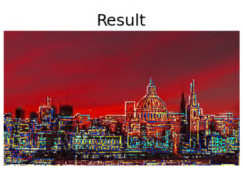


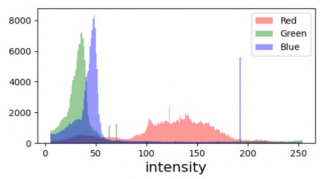
Part 2: Aprroach 1, 80x140 divide





Part 2: Aprroach 2, 80x140 divide





Performance

Table: Algorithm Run times (seconds) per Result image
(a pair of Source-Target image is processed.).

Division count	not-divided	4x7	15x20	40x80	80x140
Part 1	5	-	-	-	-
Part 2 Approach 1	-	7	12	180	2500
Part 2 Approach 2	-	6	7	8	12

Histograms proved the theory better than images themselves.

We expect a Result image's histogram to look much more like Target image than Source image, since we desire such color distribution (color palette). Although Resulting images were not always visually appealing or looking like good color transfers, Histograms mostly agreed with the theory and demonstrated desired successful color distributions, in **all** variants of the Algorithm.

Approach 1 of Part 2

For comparing regions for "similar" structures, SSD algorithm does not give guaranteed perfect-match, but more like heuristic; since Source and Target images have different sized and shaped structures.

Since all divided Regions gets transferred the color of a Target region, independently of each other, a checkered/pixelated pattern occurs. An extra filtering step could be added to smooth out the edges of these patterns, yet, no such processing was applied in this experiment.

Approach 2 of Part 2

My initial assumption with Approach 2 was wrong. When Target color is transferred to Source image, some of the structures(shapes) also got transferred naturally; as opposed to our aim. This Approach as a whole is a huge logical mistake of mine.

How many Divisions is better ?

I tried to optimize the **PART 2** Algorithms results with different region counts amounts, and above are the best result I am able to achieve at this moment.

The more divisions, the more precision. Yet this does not always guarantee better (accuracy) results; it might just make the image look pixelated like a chess board collage in low division counts, and look glitchy in high division counts. Also, more division means, Quadratic increase in execution time, considering complexity MxN = widthDivide * heightDivide.

Algorithm of Choice

Part 2 was an attempt to optimize Part1 and get better results, yet, even with high precision, experimental results suggest that, we choose **PART 1** algorithm as more robust and usable. Its results are not color-perfect but at least not pixelated but uniform.

4 Conclusion

Color Transfers in none of the algorithm variations were as satisfactory as the examples I have seen on the web or the example result files provided. In the code, during lab2rgb comparison, I get several "value out of range" warnings which I could not solve even though I tried many ways to change range and data type of pixel values and so on, no avail. This might have affected the results and might be the reason for the color inaccuracies. Yet, the results are satisfactory enough to prove the concept.

References

[1] Erik Reinhard, Michael Ashikhmin, Bruce Gooch and Peter Shirley, 'Color Transfer between Images', IEEE CGA special issue on Applied Perception, Vol 21, No 5, pp 34-41, September - October 2001.