Color Transfer Between Images

Mini Project: Team "nqb"

Bora Buyukturk, Hakki Can Karaimer, Nguyen Q. Binh

What We Solved

- We solved color transfer between images using two different algorithms.
- To apply different transformations to different parts of image, we used three different approach.

The Goal

- Alter an image’s color
- Example: removing a dominant and undesirable yellow in a photo taken under incandescent illumination
- Borrow one image’s color characteristics and look and feel from another
- Modify all color channels in tandem and changing pixels’ color in a coherent way

Color Transfer between Images (2001)

- Statistics and color correction
- Choosing a suitable color space to human vision and natural scenes
- Minimum correlation between channels
- lαβ orthogonal perception-based color space of Ruderman et al
- Independent operations in color channels

```
# scale by the standard deviations
l = (lStdSrc / lStdTar) * l
a = (aStdSrc / aStdTar) * a
b = (bStdSrc / bStdTar) * b
```

Initial code: http://www.pyimagesearch.com/2014/06/30/super-fast-color-transfer-images/

LEVEL-1: Color transfer by matching variances

Decorrelation Operations

- RGB -> XYZ device-independent tristimulus values

  XYZitu601-1 (D65) standard conversion matrix

  $M_{itu}x = (111)^T$ -> solve for x
- XYZ ->LMS cone space normalization
- Logarithmic space to eliminate skewness on the data
- LMS -> lαβ maximal decorrelation using PCA
- Variant of opponent-color models
- An achromatic channel (I), chromatic yellow–blue (α) and red–green (β) opponent channels
- Achromatic axis is orthogonal to the equiluminar

$$M_{itu} = \begin{bmatrix} 0.4306 & 0.3415 & 0.1784 \\ 0.2220 & 0.7067 & 0.0713 \\ 0.0202 & 0.1295 & 0.9394 \end{bmatrix}$$

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.5141 & 0.3239 & 0.1604 \\ 0.2651 & 0.6702 & 0.0641 \\ 0.0241 & 0.1228 & 0.8444 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 0.3897 & 0.6890 & -0.0787 \\ -0.2298 & 1.1834 & 0.0464 \\ 0.0000 & 0.0000 & 1.0000 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 0.3811 & 0.5783 & 0.0402 \\ 0.1967 & 0.7244 & 0.0782 \\ 0.0241 & 0.1288 & 0.8444 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$\begin{bmatrix} l \\ \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{3}} & 0 & 0 \\ 0 & \frac{1}{\sqrt{6}} & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -2 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{L} \\ \mathbf{M} \\ \mathbf{S} \end{bmatrix} \leftarrow \begin{aligned} \boldsymbol{L} &= \log L \\ \boldsymbol{M} &= \log M \\ \boldsymbol{S} &= \log S \end{aligned}$$

$$\begin{aligned} \text{Achromatic} &\propto r + g + b \\ \text{Yellow–blue} &\propto r + g - b \\ \text{Red–green} &\propto r - g \end{aligned}$$

LEVEL-1: Color transfer by matching variances

Color processing, color transfer Operations [1]

along each of the three axes:

- Subtract μ source from the source data points

  $l^* = ls - \mu ls$, $\alpha^* = \alpha s - \mu \alpha s$, $\beta^* = \beta s - \mu \beta s$

- scale the data points by respective standard deviations $\sigma t/\sigma s$

  $l' = l^*.\sigma t/\sigma s$, $\alpha' = \alpha^*.\sigma t/\sigma s$, $\beta' = \beta^*.\sigma t/\sigma s$

- Add μ target

  $L^{**} = l' + \mu lt$, $\alpha^{**} = \alpha' + \mu \alpha t$, $\beta^{**} = \beta' + \mu \beta t$

Convert back to RGB

- $l\alpha\beta$ -> $\log^{-1}$ LMS (10^x) -> LMS -> XYZ -> RGB

$$l^* = l - \langle l \rangle \qquad l' = \frac{\sigma_t^l}{\sigma_s^l} l^* \qquad l^{**} = l' + \langle l \rangle$$

$$\alpha^* = \alpha - \langle \alpha \rangle \qquad \alpha' = \frac{\sigma_t^\alpha}{\sigma_s^\alpha} \alpha^* \qquad \alpha^{**} = \alpha' + \langle \alpha \rangle$$

$$\beta^* = \beta - \langle \beta \rangle \qquad \beta' = \frac{\sigma_t^\beta}{\sigma_s^\beta} \beta^* \qquad \beta^{**} = \beta' + \langle \beta \rangle$$

$$\begin{bmatrix} \mathbf{L} \\ \mathbf{M} \\ \mathbf{S} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -2 & 0 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{3}}{3} & 0 & 0 \\ 0 & \frac{\sqrt{6}}{6} & 0 \\ 0 & 0 & \frac{\sqrt{2}}{2} \end{bmatrix} \begin{bmatrix} l \\ \alpha \\ \beta \end{bmatrix}$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 4.4679 & -3.5873 & 0.1193 \\ -1.2186 & 2.3809 & -0.1624 \\ 0.0497 & -0.2439 & 1.2045 \end{bmatrix} \begin{bmatrix} L \\ M \\ S \end{bmatrix} \leftarrow \quad \boldsymbol{L} = 10^L \quad \boldsymbol{M} = 10^M \quad \boldsymbol{S} = 10^S$$

[1] E. Reinhard, M. Ashikhmin, B. Gooch and P. Shirley, "Color Transfer between Images," *IEEE Computer Graphics and Applications,* vol. 21, no. 5, pp. 34-41, 9 2001.

LEVEL-2: Color transfer by matching distributions[2]

Convert from RGB to *l*αβ

- RGB -> XYZ -> LMS -> $\log^{-1}$ LMS (10^x) -> *l*αβ

X $\in [0, 255]$

CDF_s (Xs) = CDF_t (Xt)

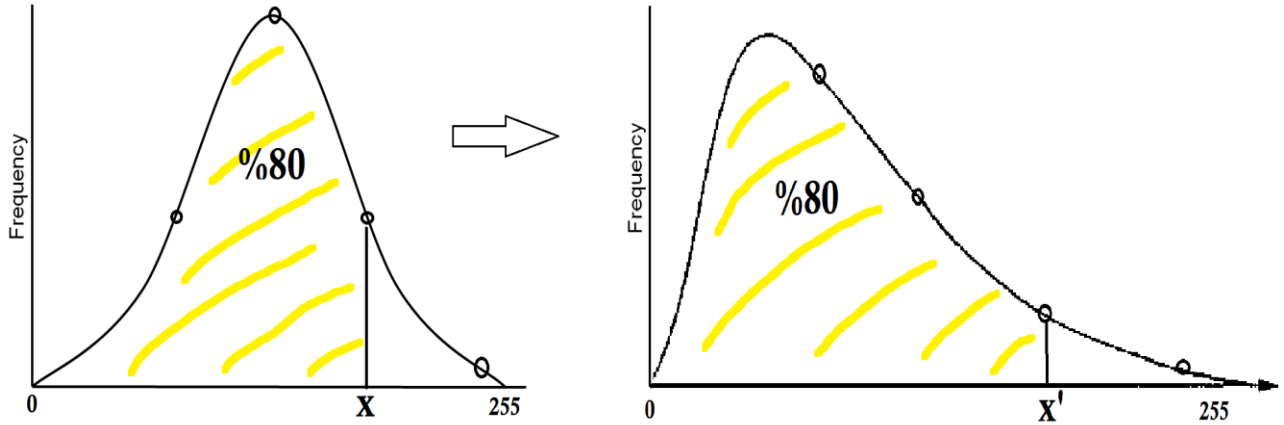
Mapping function

Simple solution : $t(x) = C^{-1}_Y (C_X (x))$ [2]

- **$F(l) = CDF_{lt}^{-1} (CDF_{ls} (l))$**
- **$F(\alpha) = CDF_{\alpha t}^{-1} (CDF_{\alpha s} (\alpha))$**
- **$F(\beta) = CDF_{\beta t}^{-1} (CDF_{\beta s} (\beta))$**

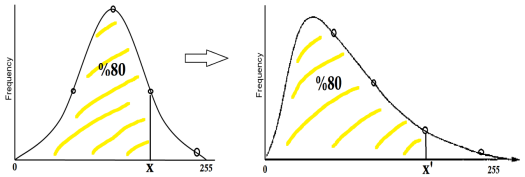
Convert back to RGB from *l*αβ

- *l*αβ -> $\log^{-1}$ LMS (10^x) -> LMS -> XYZ -> RGB



[2] F. Pitie, A. Kokaram and R. Dahy, "N-Dimensional Probability Density Function Transfer and its Application to Colour Transfer," in *CCV'05 Tenth IEEE International Conference on Computer Vision* , Beijing, China, 2005.

LEVEL-2: Implementation details



Swapping values simple technique

D1= [… 230,…,22,..113,…22,…55, … 167, …, 12,…25,…]

D2= [… 33,…,173,…212,…111,… 17, …, 212,…,172,…34]

Sorted D1= […12,…,22,22,22,…,25,..,55,…,113,…,167,…,243]

Sorted D2= […,17,..,33,…,34,..,111,…,172,173,…,212, 212,…]

. . . .

Cautions!

- What if # of pixels are not equal?
  * We populated missing values at equidistance points
- During swapping, what if two different values get the same value! Theoretically not proper!
  * Histogram matching with high discretization resolution. It spreads values better instead of populating.

Swapping values simple technique

D1= [… 230,…,22,..113,…22,…55, … 167, …, 12,…25,…]

D2= [… 33,…,173,…212,…111,… 17, …, 212,…,172,…34]

Sorted D1= […12,…,22,22,22,…,25,..,55,…,113,…,167,…,243]

Value 22: 52th in 700pixels -> 100000*(52/700)= 74285th in 100000 pixels, AUC ~ 0.074285

Value 22: 53th in 700pixels -> 100000*(53/700)= 75714th in 100000 pixels, AUC ~ 0.075714

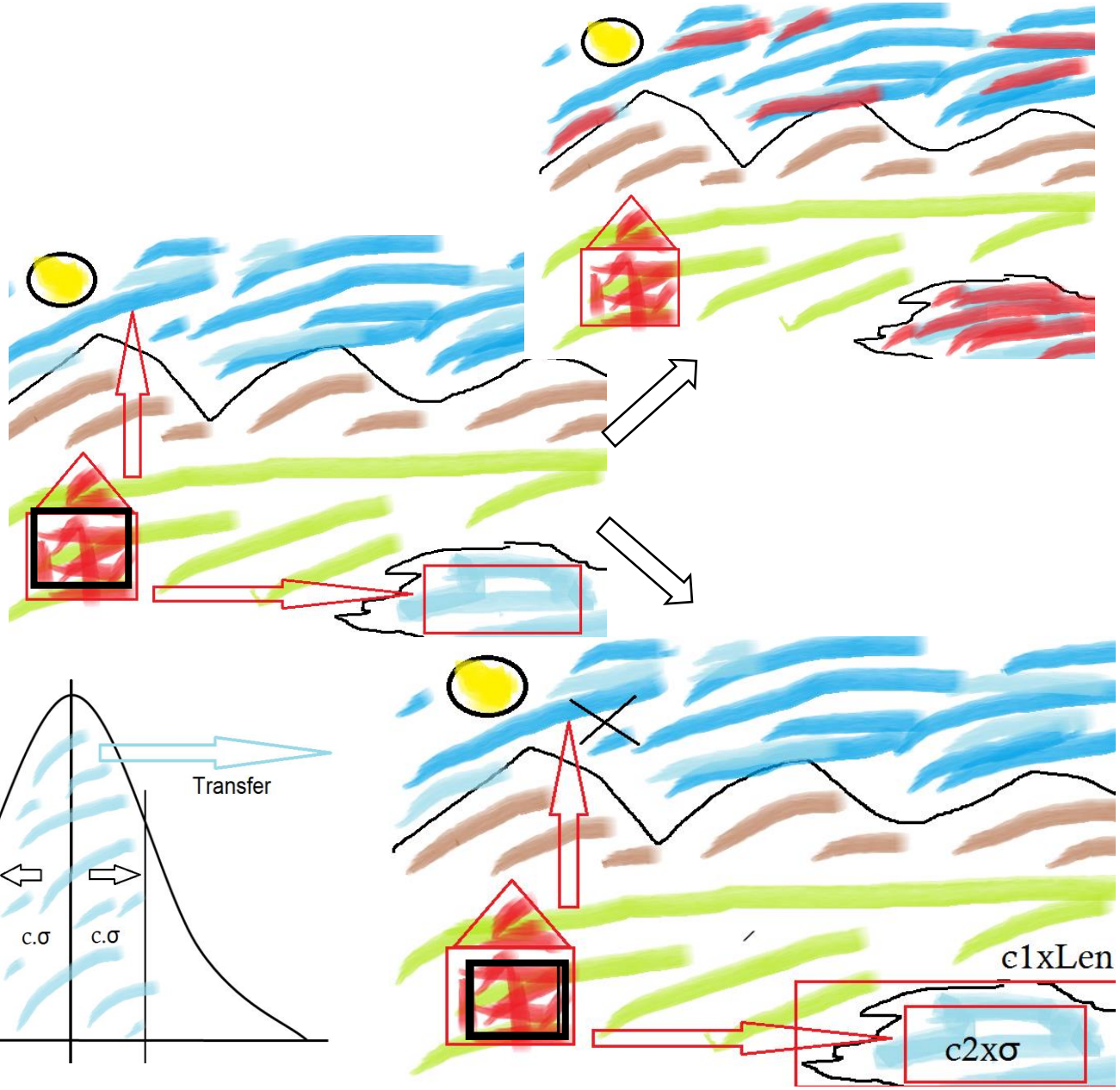
Value 22: 54th in 700pixels -> 100000*(54/700)= 77142th in 100000 pixels, AUC ~ 0.077142

Lookup 74285th element or closest one to it in sorted D2 dictionary.

It may correspond to 74444th key in D2 dictionary as 67th of 900.

LEVEL-3:

- Apply techniques of level 1 and 2
- Selected a boundary to calculate statistics μ & σ
- Boundary creates an inner rectangle
- For alternating edges inner rectangle is expanded with a coefficient to an outer rectangle
- Idea: Transfer to all pixels of bigger rectangle which are similar to the inner rectangle's statistics.
- Advanced selection algorithms will output more precise since rectangular selection is limited
- Trade of between rectangle selection (c1) and statistical stringency (c2) coefficients!
- More stringent c2 will miss closer colors, less stringent c2 will cause a line appearance.



Level-1 RESULTS: **Original images** **Matching variances**

Level-2 RESULTS: matching distributions

swap ranked channel values

Original images

histogram matching

RESULTS

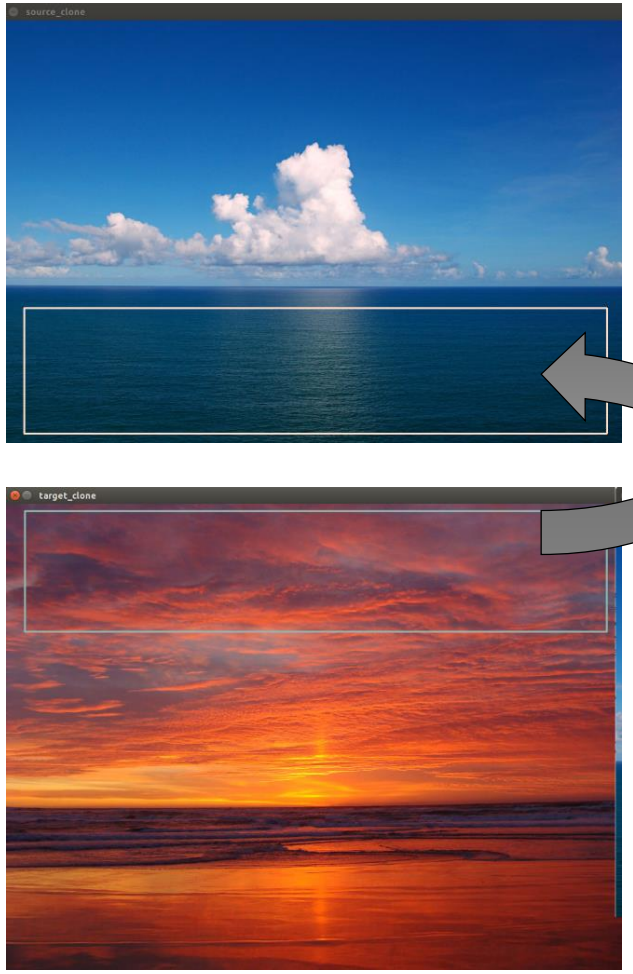
Level-2 swap ranked channel values

Level-1 matching variances

Level-2 histogram matching

LEVEL-3: Partial Color Transfer

Image composition is an effective factor for higher success



Level-1 RESULTS: **Original images** **Matching variances**

Level-2 RESULTS: matching distributions

swap ranked channel values

Original images

histogram matching

RESULTS

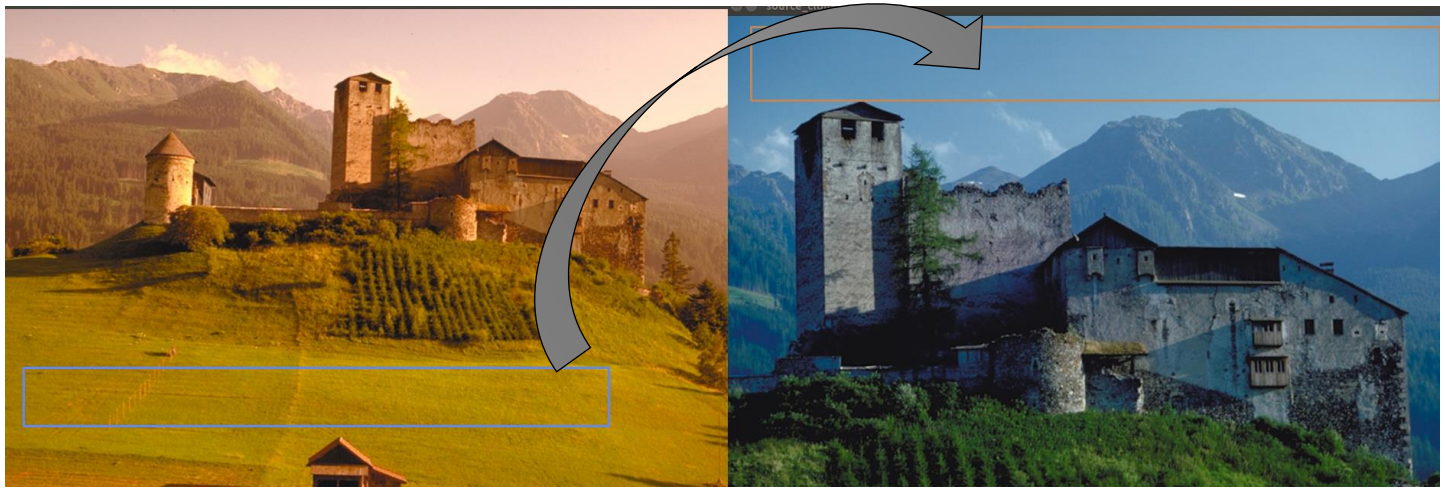
Level-2 swap ranked channel values

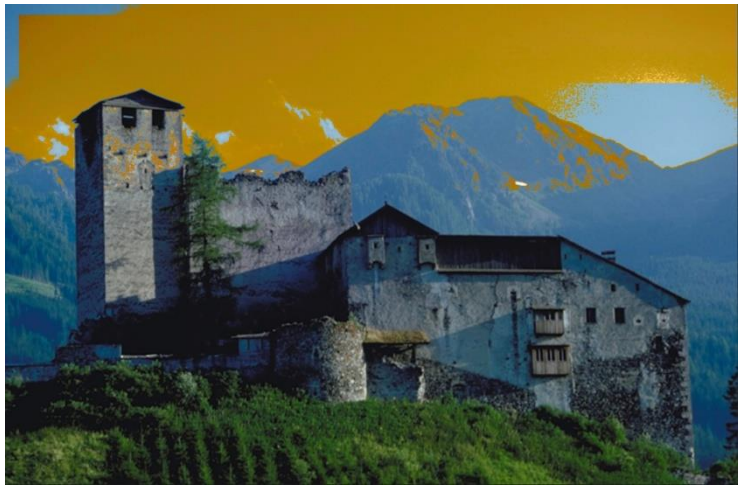
Level-1 matching variances

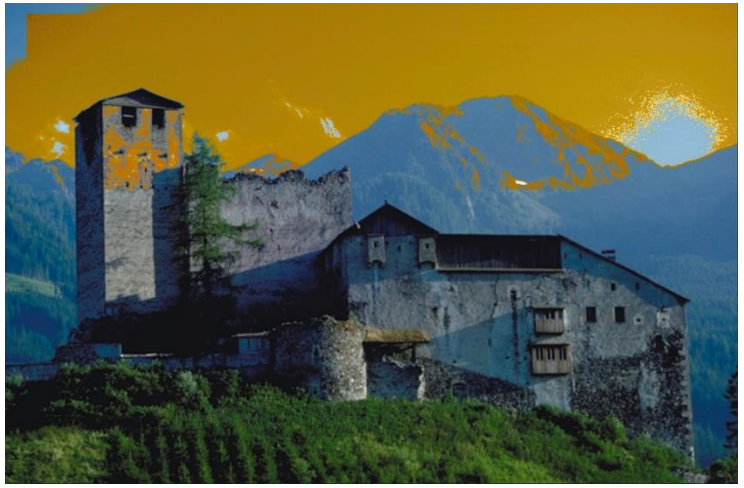
Level-2 histogram matching

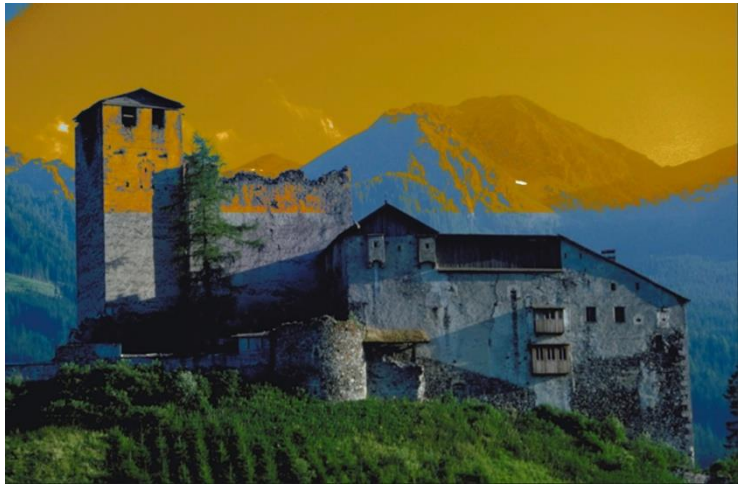
LEVEL-3: Partial Color Transfer

Positional sensitivity and local calculations are necessary!









Trade of is too obvious

RESULTS

Original images

Level-1 matching variances

Level-2 histogram matching

LEVEL-3: Partial Color Transfer

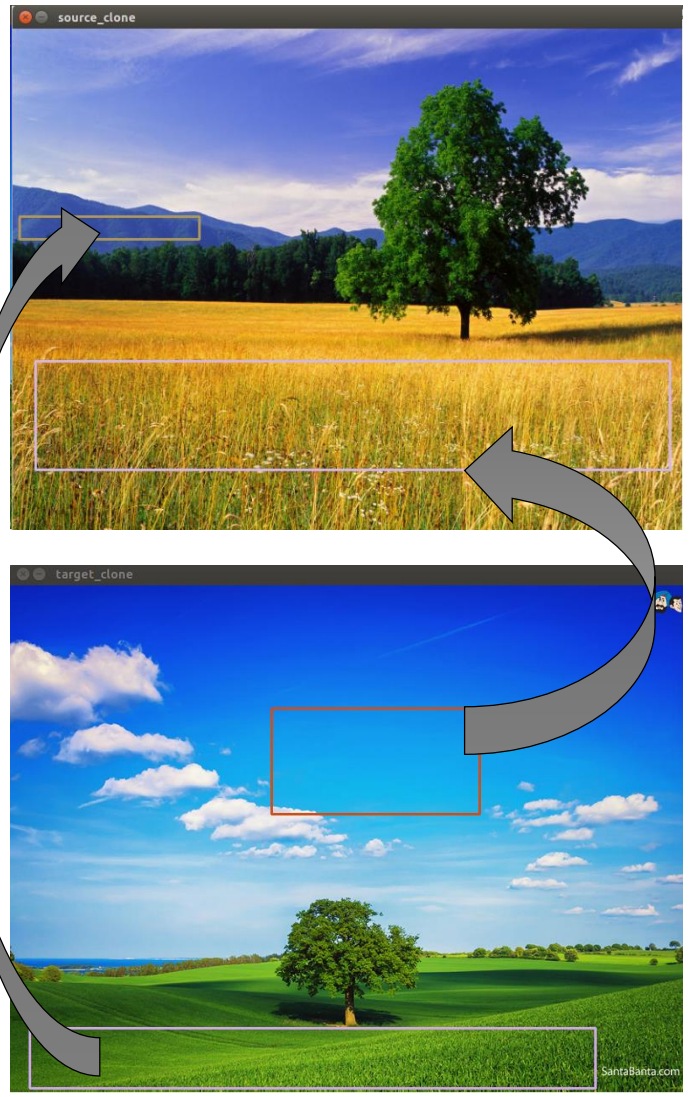
Trade of is not obvious.
SAFE!

LEVEL-3: Partial Color Transfer

Concurrent transfer

LEVEL-3: Partial Color Transfer

Concurrent transfer



LEVEL-3: Partial Color Transfer

Sensitive to sharp color differences (high c1)

Level 3 – Another Approach

- 1- White balance and color space transformation.
  - White balancing.
  - Consistent color space: sRGB.
- 2- Color transfer using the selected components of two scenes.
  - Iterative non-linear minimization using angular error (Cheng et. al., 2015) between two colors.
  - $\theta = \cos^{-1}\frac{a.b}{|a||b|}$, where a and b are source and target color vectors.
  - Our error function calculates Θ between source and target vector, at each iteration of quasi newton algorithm.
  - K-means clustering and watershed segmentation are applied on target image.

D. Cheng, B. Price, S. Cohen, and M. S. Brown. Effective learning-based illuminant estimation using simple features. In CVPR, 2015

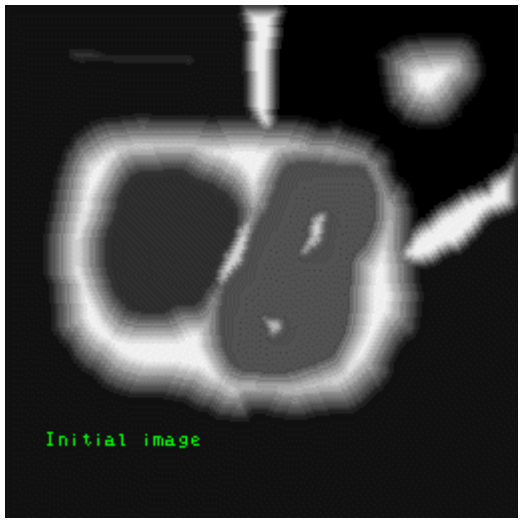
Segmentation with K-Means Algorithm

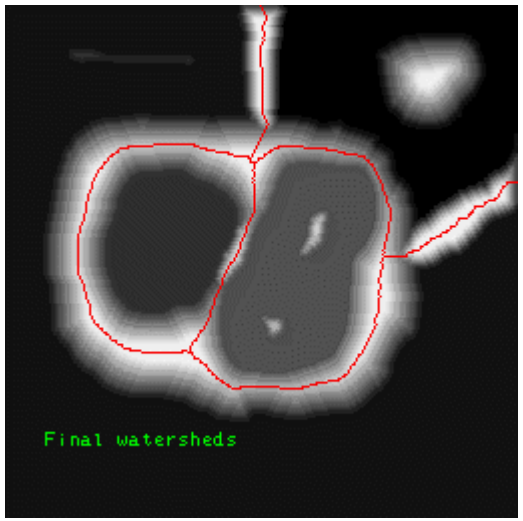
- Step 1: Convert image's color space to LAB space.
  - L: Luminosity layer. A: Chromaticity layer where color falls along the red-green axis. B: Chromaticity layer where the color falls along the blue-yellow axis.
- Step 2: We used A and B, since only A and B have color information. Then we also add 2D position information to our data to achieve better results. This way, K-Means finds partitions such that objects within each cluster are as close to each other as possible, and as far from objects in other clusters as possible.
- Step3: Choose K parameter ,and run the algorithm 3 times.

Segmentation with K-Means Algorithm: http://www.mathworks.com/help/images/examples/color-based-segmentation-using-k-means-clustering.html?prodcode=IP&language=en , Accessed: 3.10.2015

Segmentation with Marker-Controlled Watershed Algorithm

- Any greytone image can be considered as a topographic surface.
- If we flood this surface from its minima and, if we prevent the merging of the waters coming from different sources, we partition the image into two different sets: the catchment basins and the watershed lines.

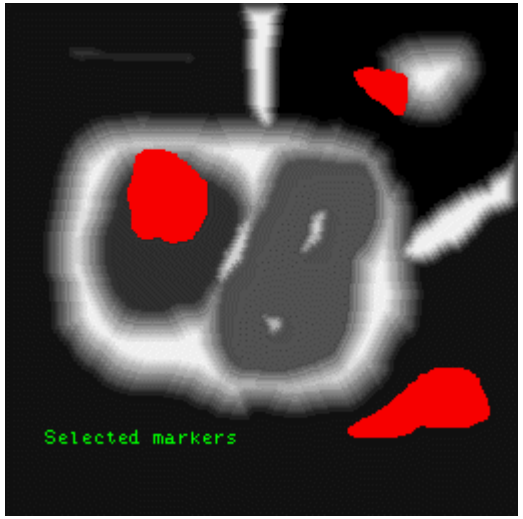


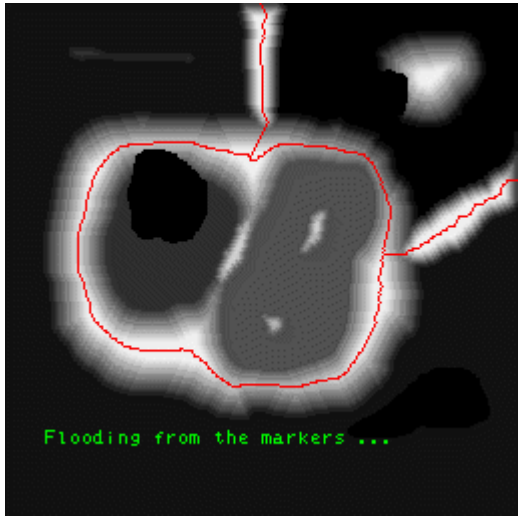


Marker-controlled watershed: http://cmm.ensmp.fr/~beucher/wtshed.html, Accessed: 10.10.2015

Segmentation with Marker-Controlled Watershed Algorithm

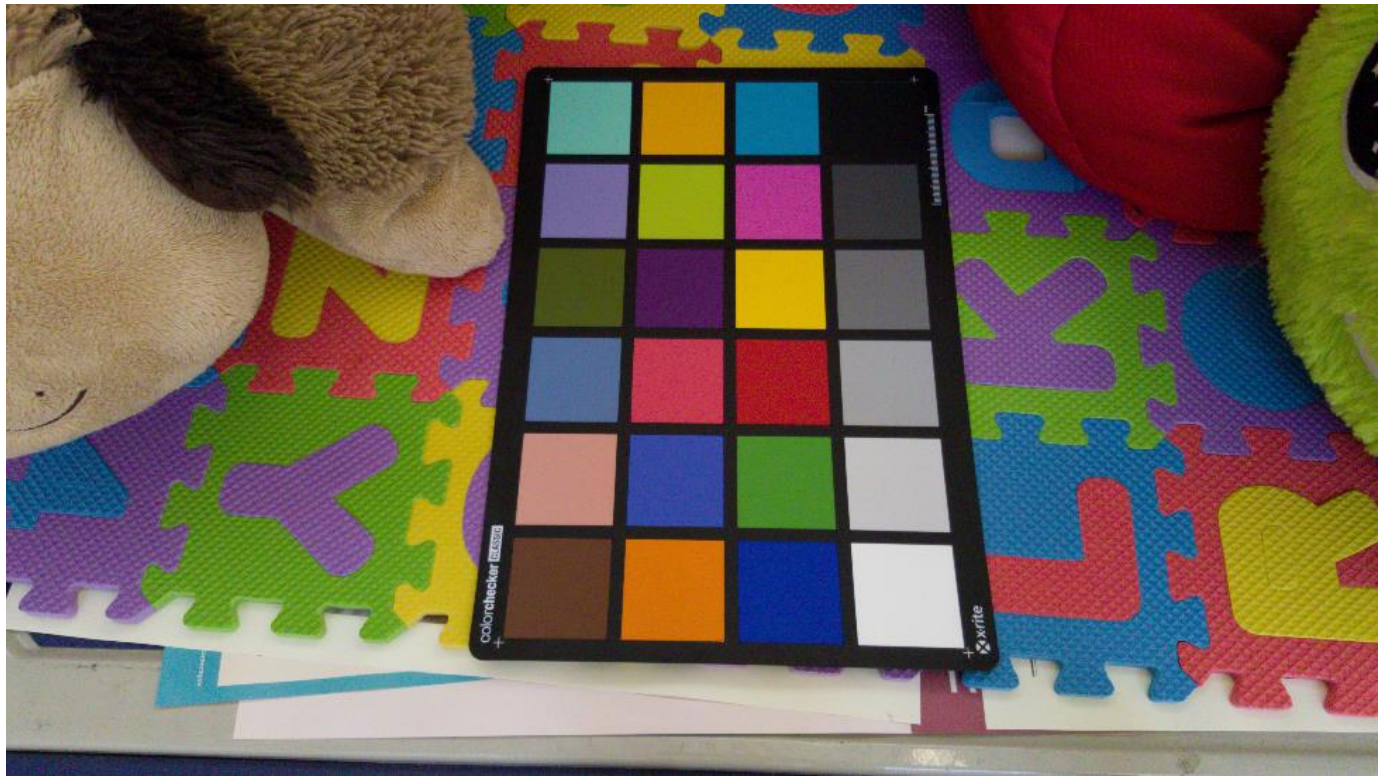
- Watershed algorithm can cause over segmentation problem, since natural images have noise and local irregularities. Solution is: Marker controlled watershed algorithm.



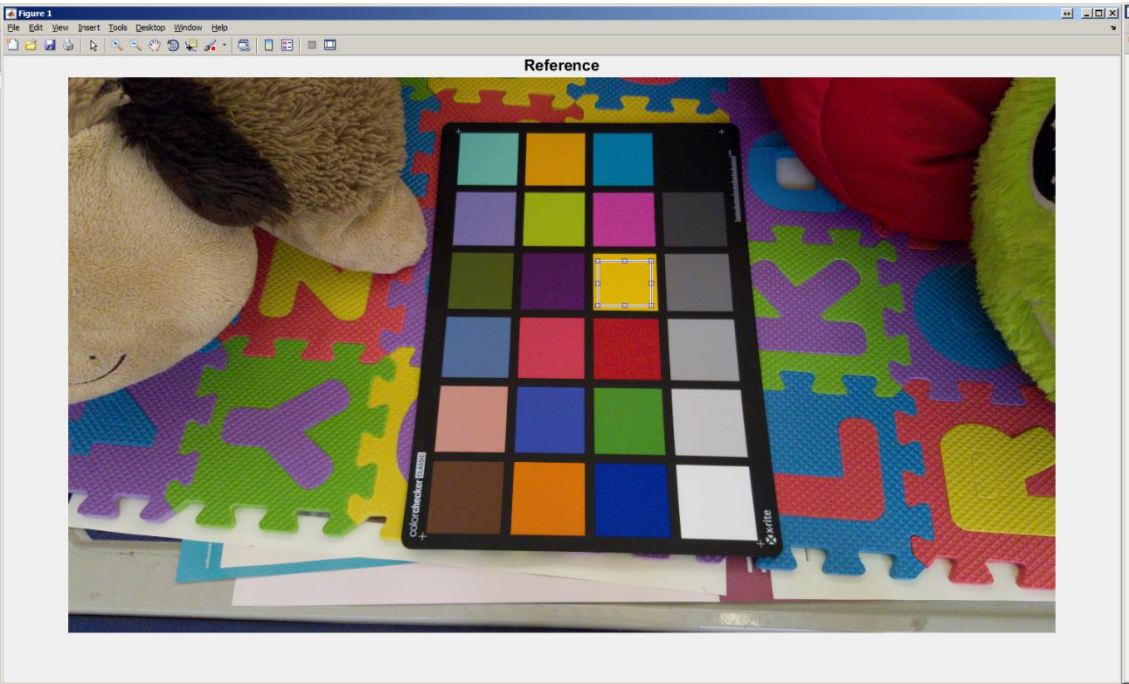


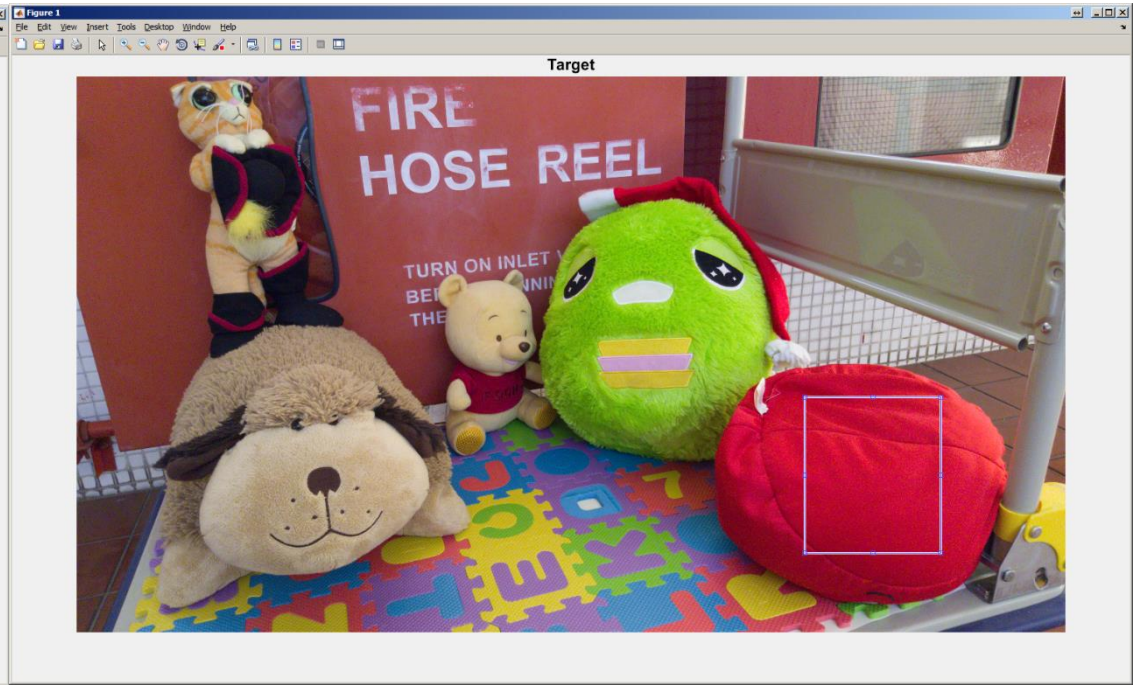
Marker-controlled watershed: http://cmm.ensmp.fr/~beucher/wtshed.html, Accessed: 10.10.2015

- Reference image:



- GUI for selection of color patches.
  - We calculate average of each patch to find color vector.
  - We can add more than one reference and target color.





Sample 1

- Reference: Orange patch in color chart.
- Target: Red of pillow in scene.

Before

After

Sample 2

- Reference: Red patch in color chart.
- Target: Green face toy in scene.

Before

After

Sample 3

- Reference: Blue patch in color chart.
- Target: Red of pillow is in scene.

Before

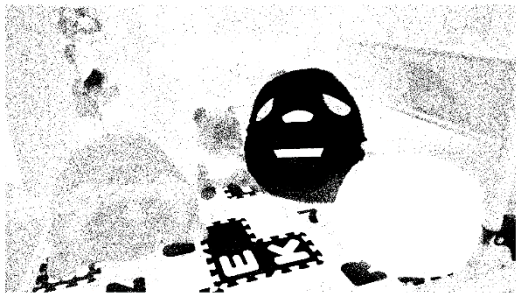
After

Sample 4

- Reference: Orange patch in color chart.
- Target: Green face toy in scene.

Before

After



Sample 5

- Reference: Orange patch in color chart.
- Target: Green face toy in scene.
- Reference: Pink patch in color chart.
- Target: Red of pillow in scene.

Before

After

Sample 6

- Reference: Orange patch in color chart.
- Target: Green face toy in scene.
- Reference: Purple patch in color chart.
- Target: Red of pillow in scene.

Before

After

Sample 7

- Reference: Pink patch in color chart.
- Target: Red of pillow in scene.

Before

After

Sample 8

- Reference: Purple patch in color chart.
- Target: Red of pillow in scene.

Before

After

Sample 9

- Reference: Blue patch in color chart.
- Target: Red of pillow in scene.
- Reference: Orange patch in color chart.
- Target: Green face toy in scene.

Before

After

Thank you.