Color2Gray: Salience-Preserving Color Removal

Amy A. Gooch Sven C. Olsen Jack Tumblin Bruce Gooch

- Visually important image features often disappear when color images are converted to grayscale.

The painting “Impression, Sunrise” by Monet.

Isoluminant Colors

Color

Grayscale

(a) CIECAM97 Lum

(b) $L^*a^*b^*$ Lum.

(c) XYZ Lum.

(d) YCrCb Lum.

(e) Auto Contrast

(f) Color2Gray

Traditional Methods

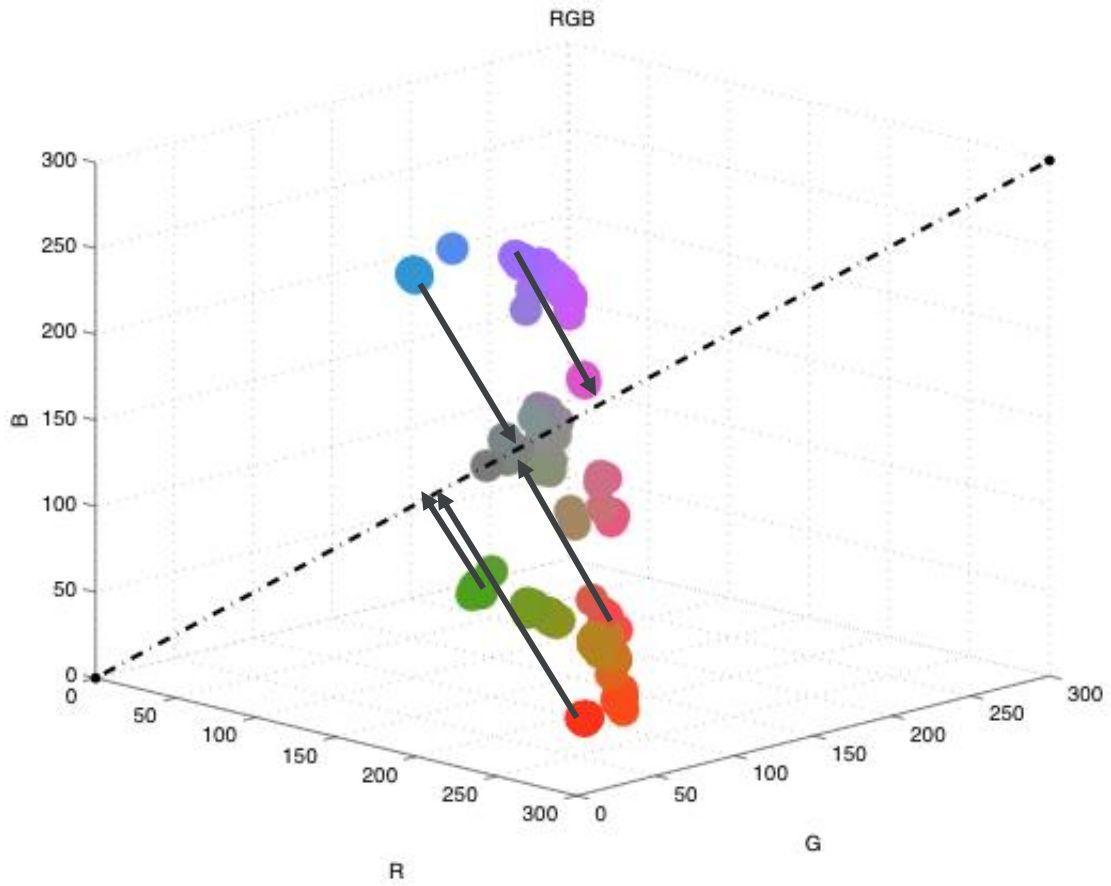
- Contrast enhancement & Gamma Correction
  - Doesn't help with isoluminant values

Photoshop Grayscale

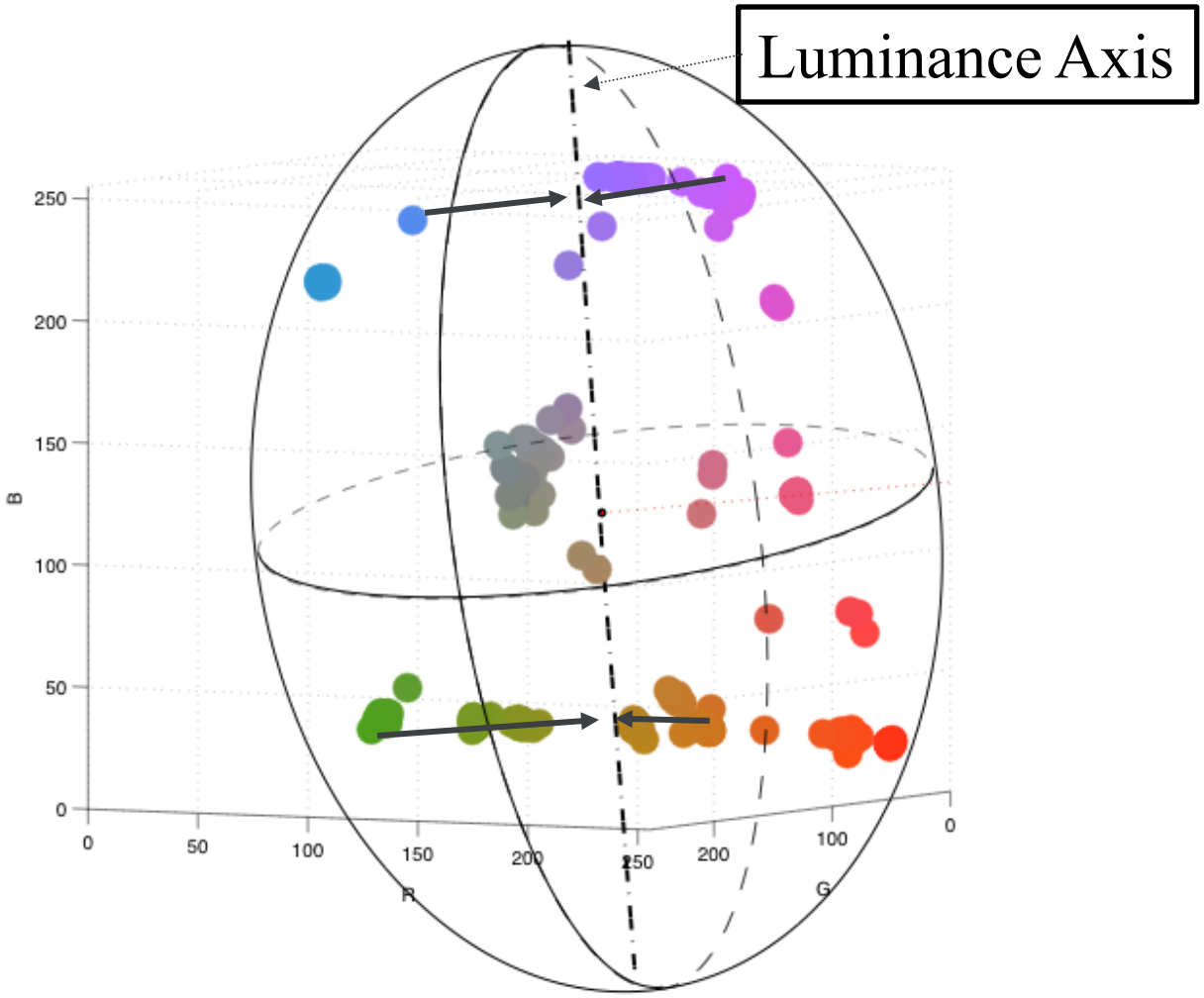
PSGray + Auto Contrast

New Algorithm

Simple Linear Mapping

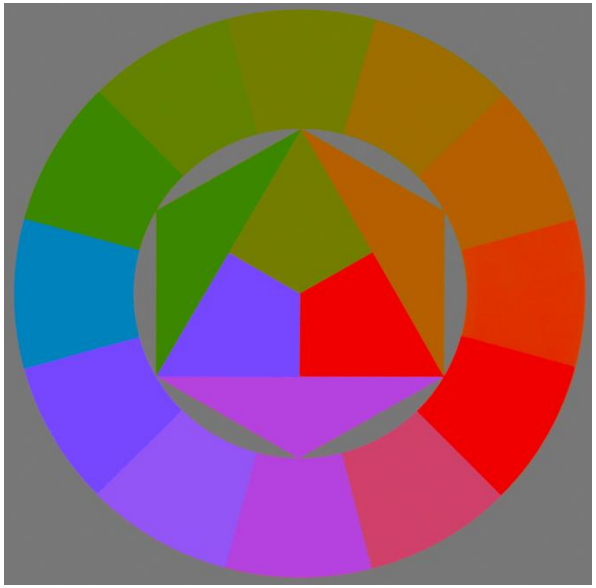


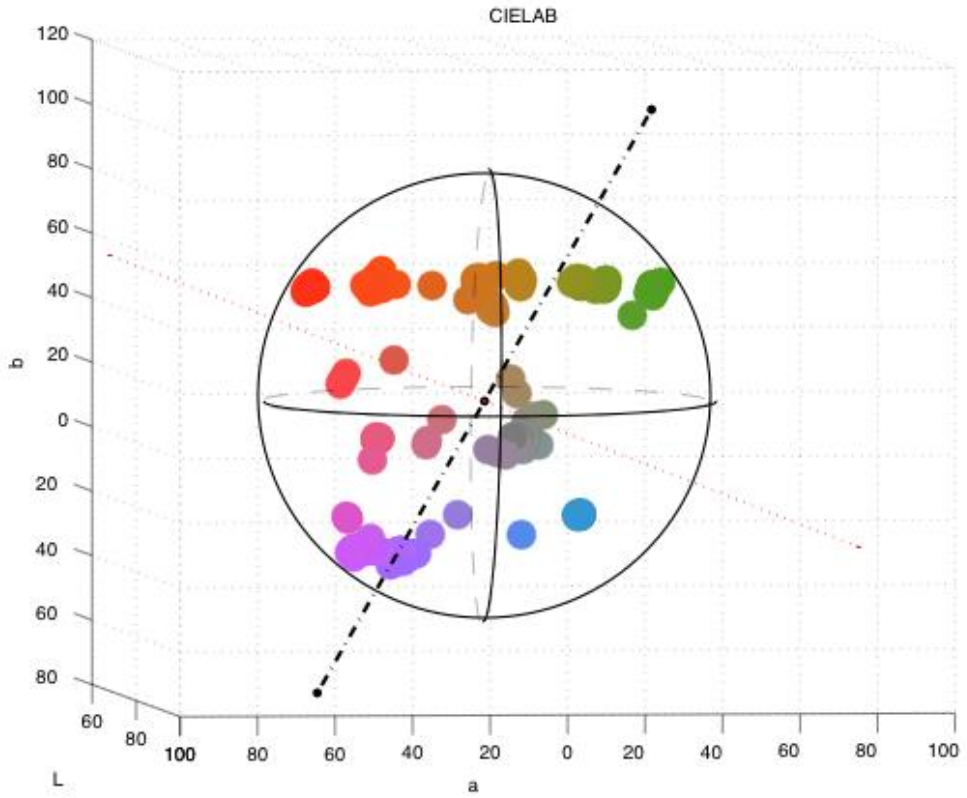
Principal Component Analysis (PCA)



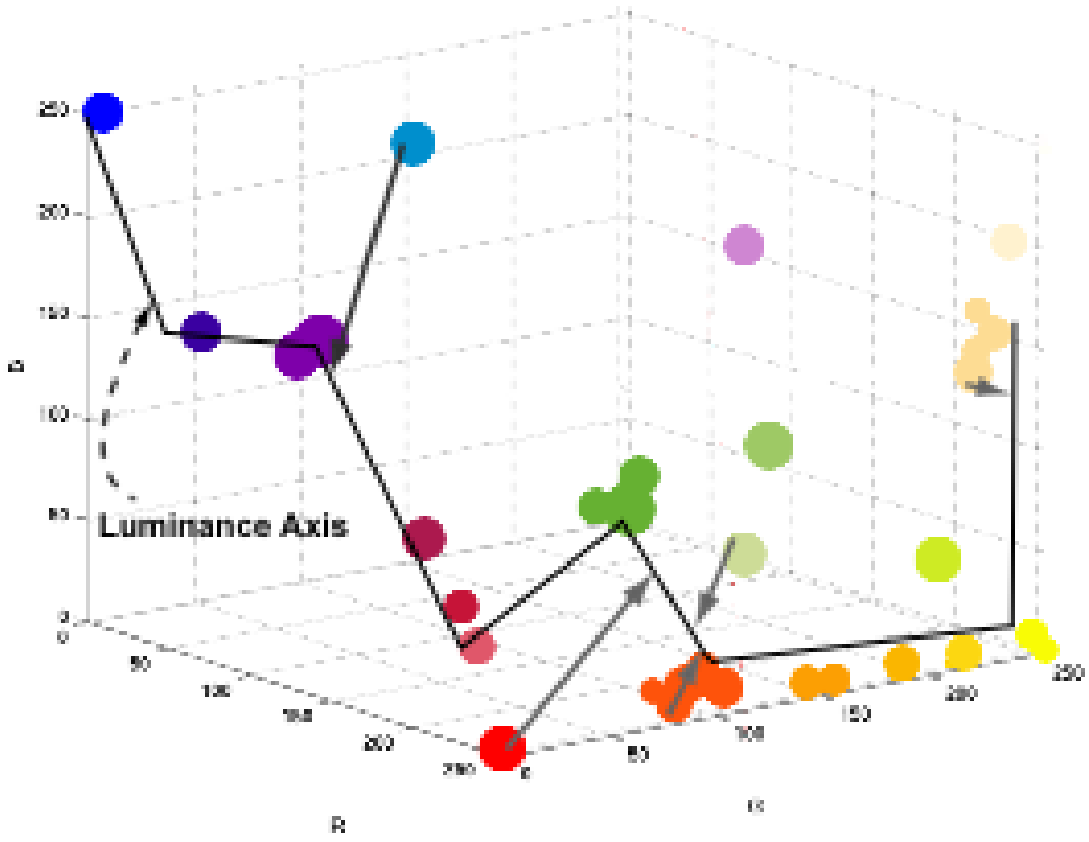
Problem with PCA

Worst case:
Isoluminant Colorwheel





Non-linear mapping



- we want digital images to preserve a meaningful visual experience, even in grayscale.

Figure 2: Isoluminant changes are not preserved with traditional color to grayscale conversion. Converting an image of a reddish square whose luminance matches that of the blue background (Left) to grayscale (Middle) results in a featureless gray image. The Color2Gray algorithm incorporates chrominance changes (Right).

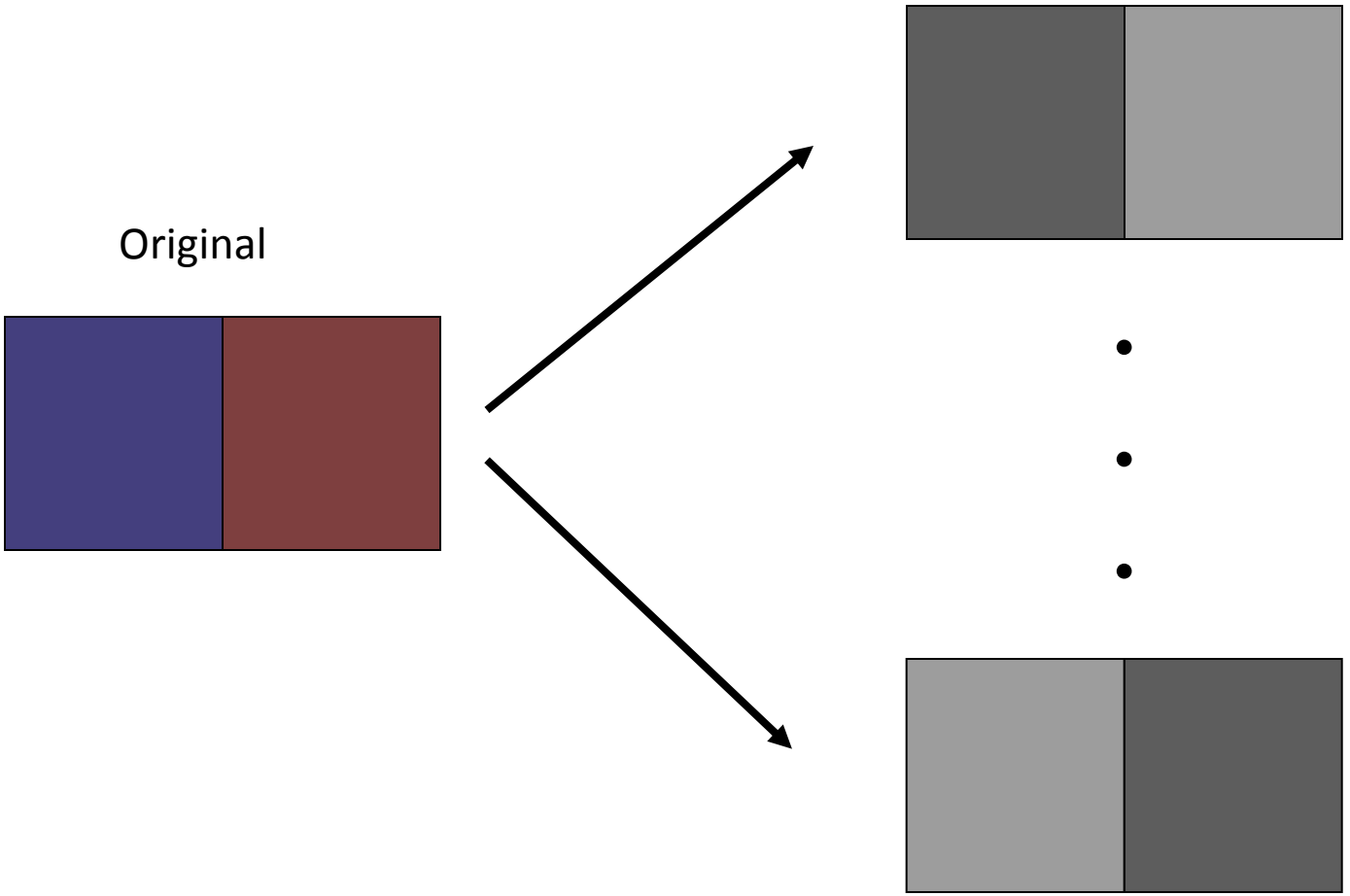
Goals

- Dimensionality Reduction
  - From tristimulus values to single channel

    Loss of information

- Maintain salient features in color image
  - Human perception

Challenge:
Many Color2Gray Solutions



preserve the salient features of the color image

The algorithm introduced here reduces such losses by attempting to preserve the salient features of the color image.

The *Color2Gray* algorithm is a 3-step process:

1. convert RGB inputs to a perceptually uniform CIE $L*a*b*$ color space,
2. use chrominance and luminance differences to create grayscale target differences between nearby image pixels, and
3. solve an optimization problem designed to selectively modulate the grayscale representation as a function of the chroma variation of the source image.

The *Color2Gray* results offer viewers salient information missing from previous grayscale image creation methods.

Parameters

- ϑ: controls whether chromatic differences are mapped to increases or decreases in luminance value (Figure 5).
- α: determines how much chromatic variation is allowed to change the source luminance value (Figure 6).
- μ: sets the neighborhood size used for chrominance estimation and luminance gradients (Figure 7).

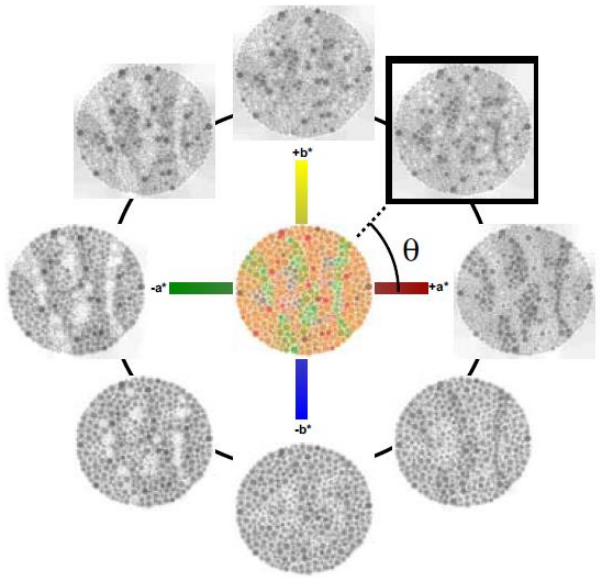
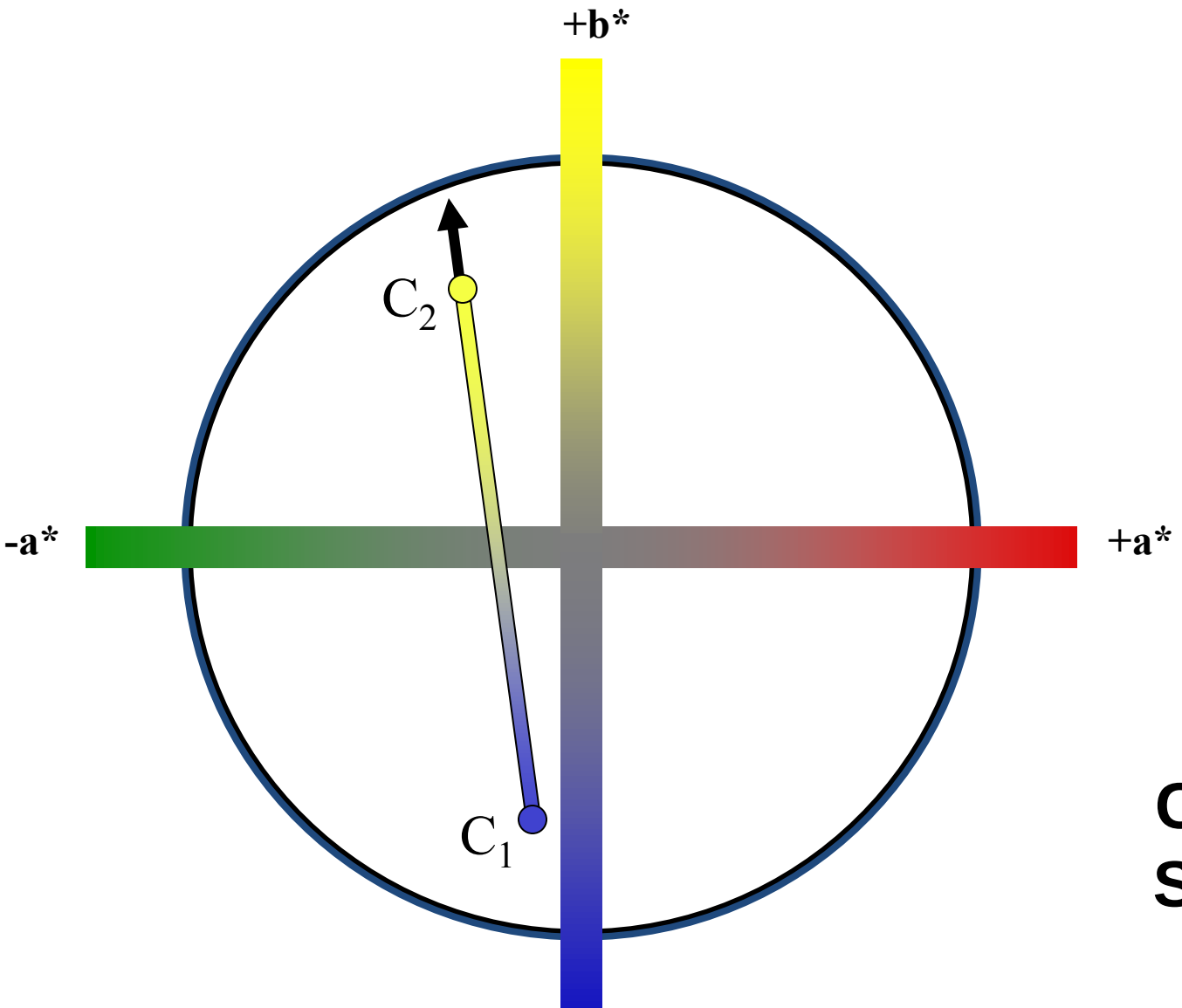


Figure 5: Color2Gray results for the color image in the center of the figure over several values of the parameter θ, which divides the chrominance plane. Note, for this colorblind image, you should not be able to see the 45 made of dots in the source color image unless you are colorblind. *Original image copyright Jay Neitz.*

Figure 6: Changing the α parameter: $\alpha = 5$, 10, 25, respectively. Increasing alpha increases the contribution from chrominance differences but may cause dynamic range mapping problems ($\theta = 45°$).

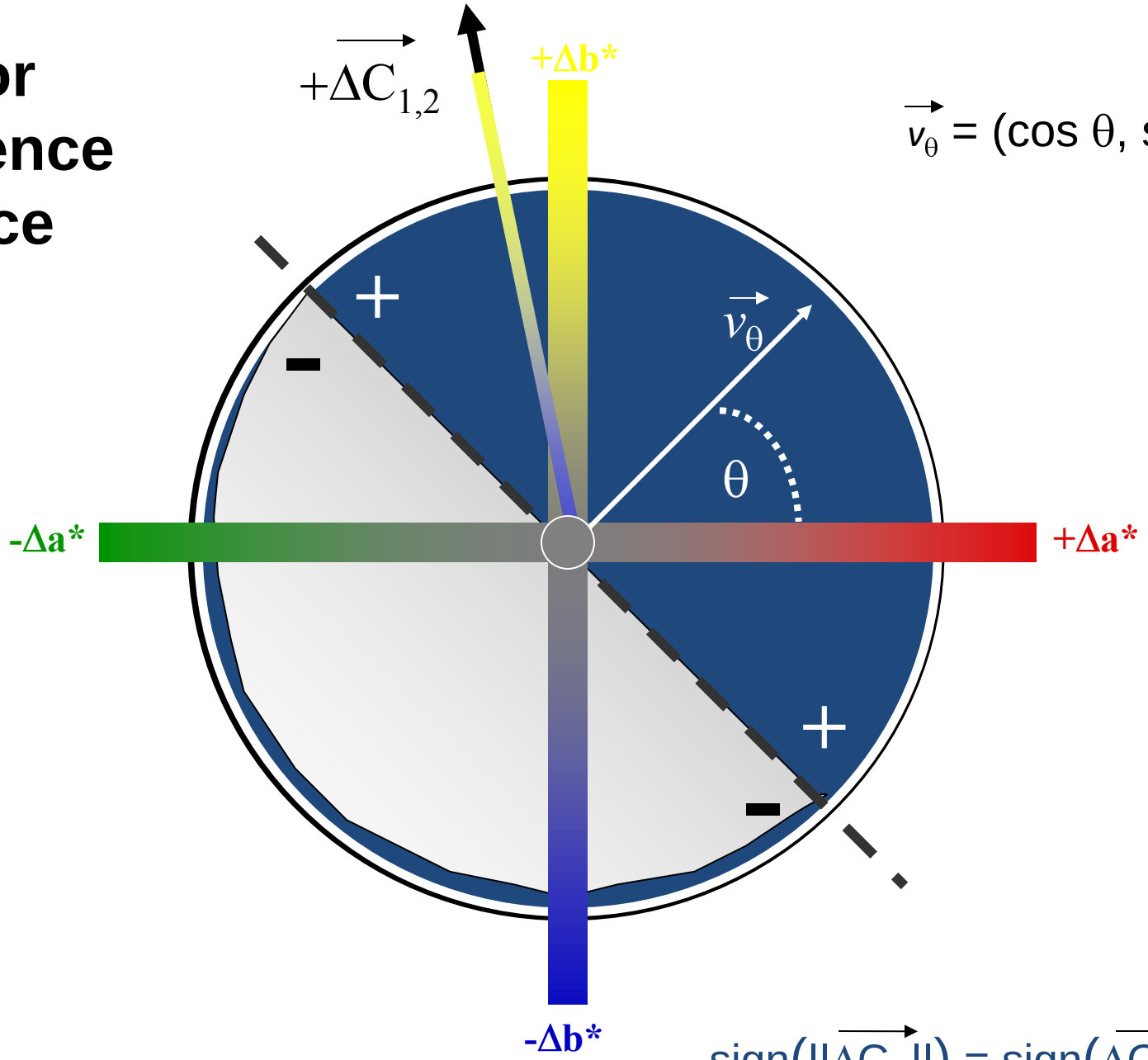
Figure 7: *Left:* Equiluminant fade from gray to blue on a background with the same luminance. *Middle:* Color2Gray result with $\mu = 9$ neighborhood. *Right:* Color2Gray result with a full neighborhood ($\theta = 270°$; $\alpha = 8.$). Using a small neighborhood produces a dark band through the middle of the fade and produces artifacts where the edge of the fade meets the background; whereas using the entire set of pixels as the neighborhood (full neighborhood) preserves the horizontal fade with respect to its background.

Map chromatic difference to increases or decreases in luminance values



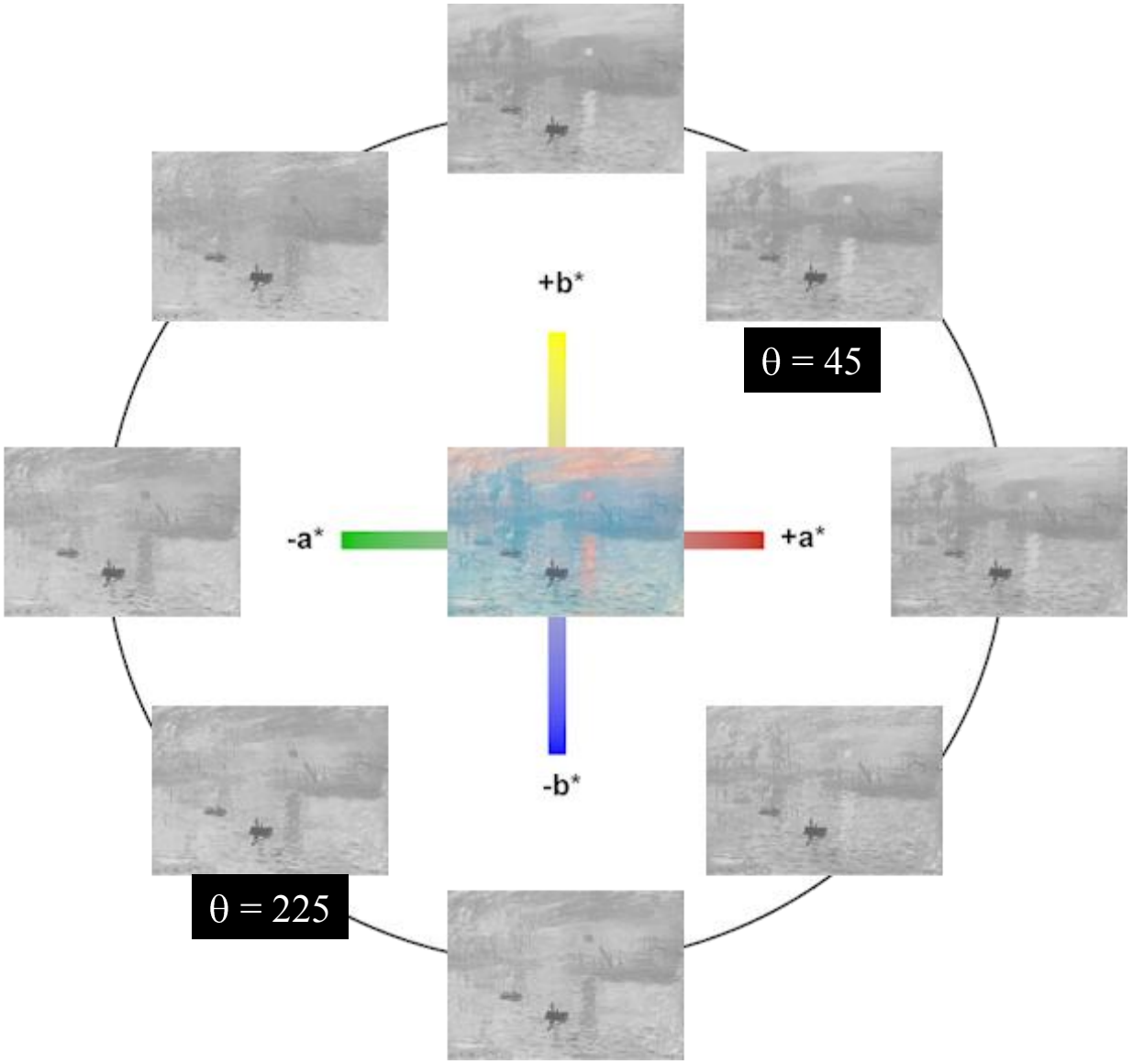
Color Space

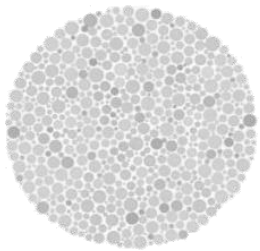
Color Difference Space



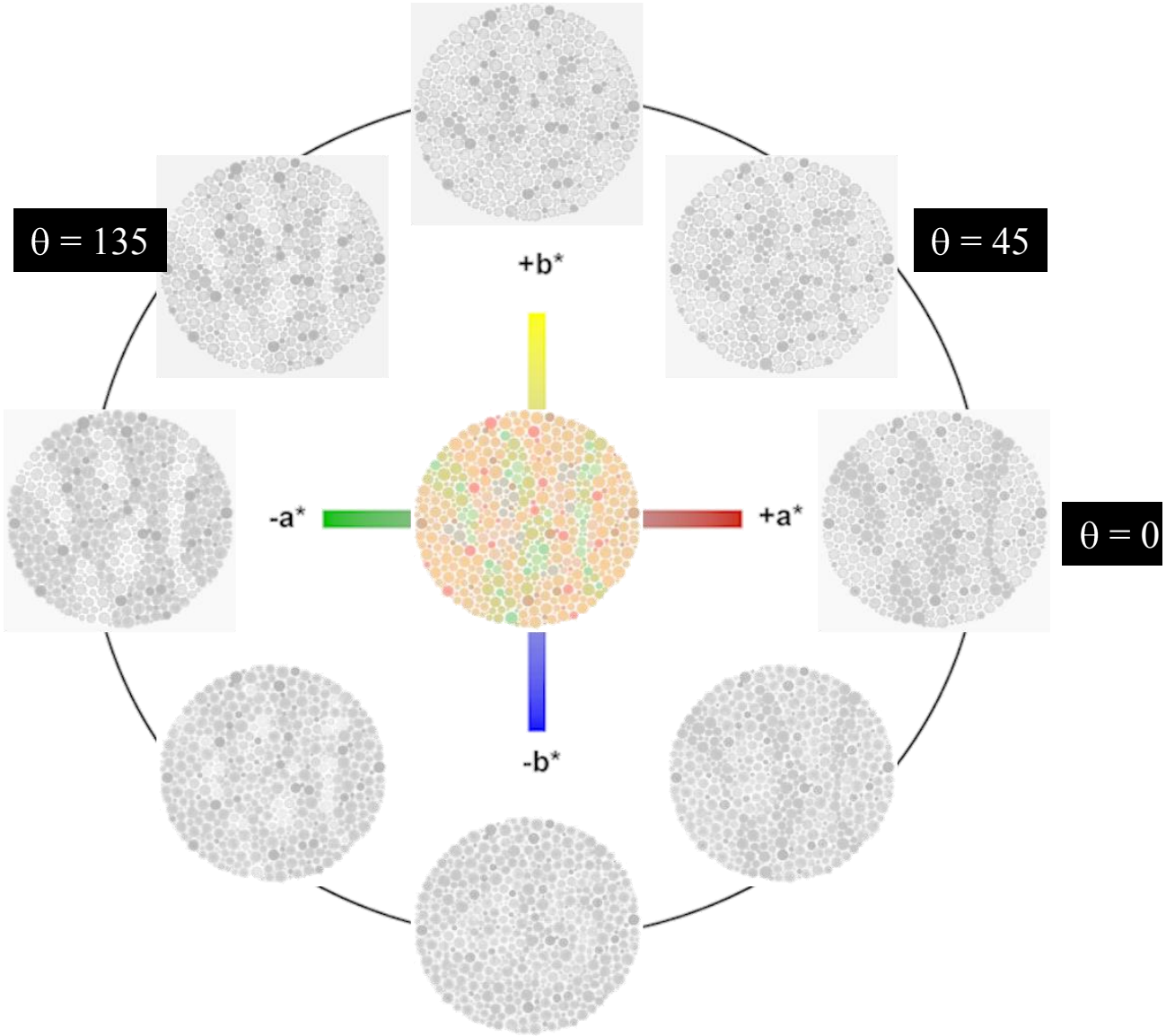
$$\text{sign}(||\Delta C_{i,j}||) = \text{sign}(\Delta C_{i,j} \cdot \vec{v}_\theta)$$

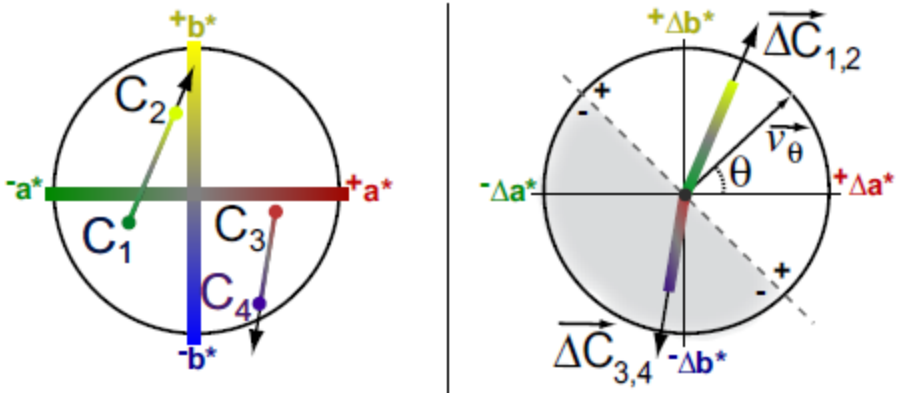
Photoshop Grayscale





Grayscale





(a) Color Space (b) Color Difference Space

Figure 8: Signed chrominance distance between pixels. Given (a) pixel pairs (i.e. $(C_i, C_j) = (C_1, C_2)$ or (C_3, C_4)), and (b) their chromatic difference, $\overrightarrow{\Delta C_{ij}}$, we divide the space of color differences into positive and negative halves based upon the parameter θ. We set the sign of the chromatic difference to be the same as the sign of $(\overrightarrow{\Delta C_{ij}} \cdot \overrightarrow{v_\theta})$, where $\overrightarrow{v_\theta}$ is a normalized vector defined by θ relative to the Δa^* axis, as illustrated in (b).

Finally, we define the target differences, δ_{ij}. If the absolute luminance difference is smaller than the chrominance difference, then we set δ_{ij} to be a measure of the chromatic differences; otherwise, δ_{ij} is set to the luminance differences. Formally, δ_{ij} is defined as follows.

$$\begin{aligned} \text{Given:} \qquad \text{crunch}(x) &= \alpha * \tanh(x/\alpha) \\ \overrightarrow{v_\theta} &= (\cos\theta, \sin\theta) \end{aligned}$$

then:

$$\delta(\alpha,\theta)_{ij} = \begin{cases} \Delta L_{ij} & \text{if } |\Delta L_{ij}| > \text{crunch}(||\overrightarrow{\Delta C_{ij}}||) \\ \text{crunch}(||\overrightarrow{\Delta C_{ij}}||) & \text{if } \overrightarrow{\Delta C_{ij}} \cdot \overrightarrow{v_\theta} \geq 0 \\ \text{crunch}(-||\overrightarrow{\Delta C_{ij}}||) & \text{otherwise.} \end{cases}$$

How to Combine Chrominance and Luminance

$$\delta_{ij} = \Delta L_{ij}$$ (Luminance)

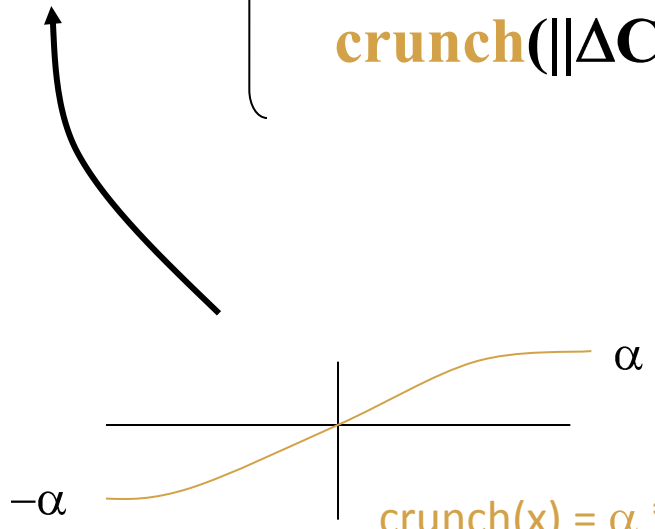
How to Combine Chrominance and Luminance

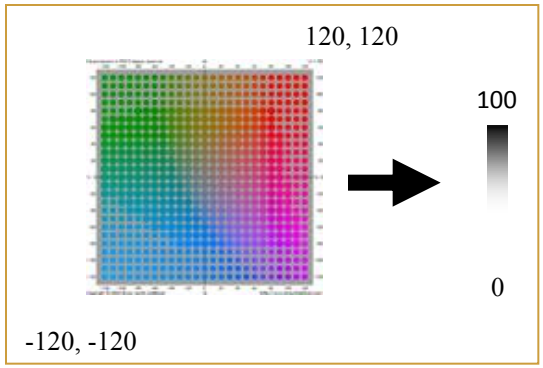
$$\delta_{ij} = \begin{cases} \Delta L_{ij} & \text{(Luminance)} \quad \text{if } |\Delta L_{ij}| > ||\Delta C_{ij}|| \\ ||\Delta C_{ij}|| & \text{(Chrominance)} \end{cases}$$

How to Combine Chrominance and Luminance

$$\delta(\alpha)_{ij} = \begin{cases} \Delta L_{ij} \\ \text{crunch}(\|\Delta C_{ij}\|) \end{cases}$$

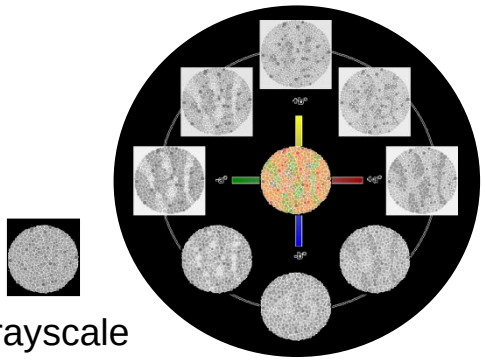
if $|\Delta L_{ij}| > \text{crunch}(||\Delta C_{ij}||)$



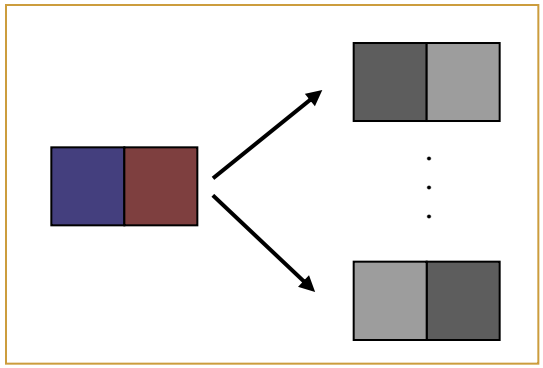


How to Combine Chrominance and Luminance

$$\delta(\alpha,\theta)_{ij} = \begin{cases} \Delta L_{ij} & \text{if } |\Delta L_{ij}| > \text{crunch}(||\Delta C_{ij}||) \\ \text{crunch}(||\Delta C_{ij}||) & \text{if } \Delta C_{ij} \cdot \nu_\theta \geq 0 \\ \text{crunch}(-||\Delta C_{ij}||) & \text{otherwise} \end{cases}$$



Grayscale



Color2Grey Algorithm

Optimization:

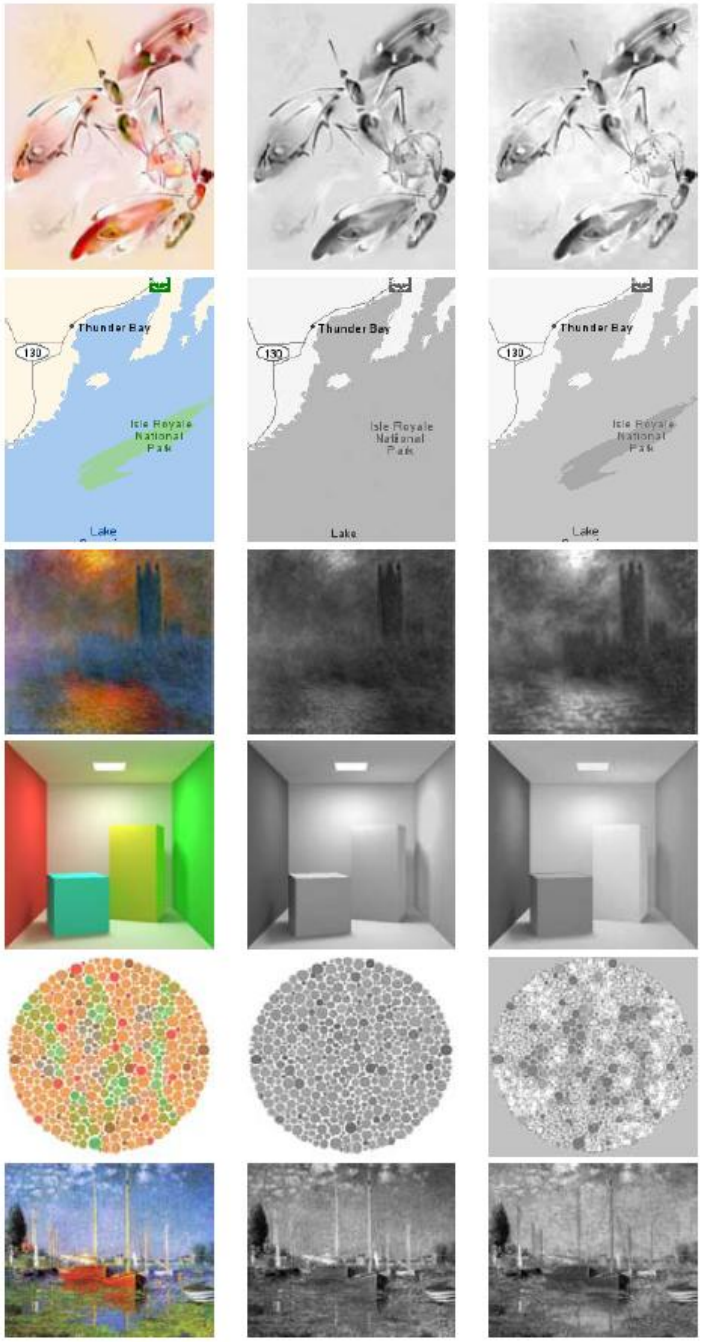
$$\min \sum_{i} \sum_{j=i-\mu}^{i+\mu} \left((g_i - g_j) - \delta_{i,j} \right)^2$$

If δ_{ij} == ΔL then ideal image is g
Otherwise, selectively modulated by ΔC_{ij}

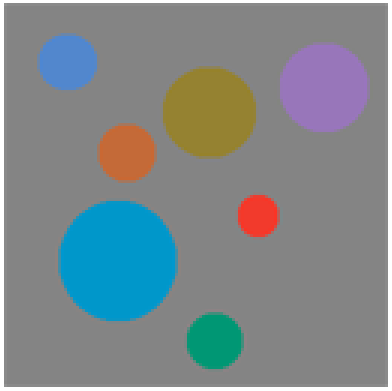
3.3 Solving the Optimization

Given a set of desired signed differences δ_{ij}, we find a gray image g that minimizes the following objective function, $f(g)$, where $\mathcal{K}$ is a set of ordered pixel pairs (i, j):

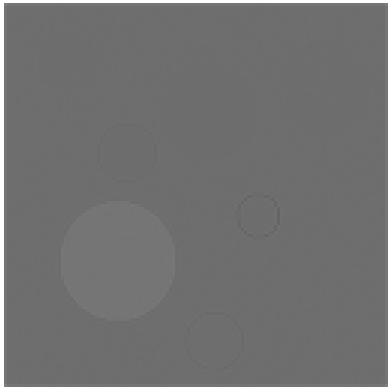
$$f(g) = \sum_{(i,j)\in\mathcal{K}} ((g_i - g_j) - \delta_{ij})^2 \quad (1)$$


Thunder Bay
130
Isle Royale National Park
Lake

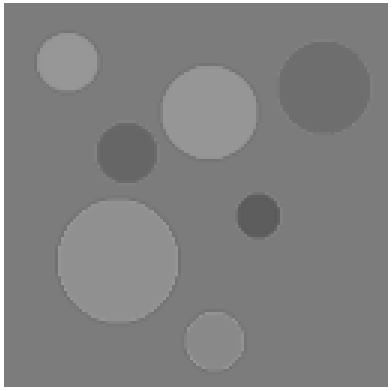
Validate "Salience Preserving"



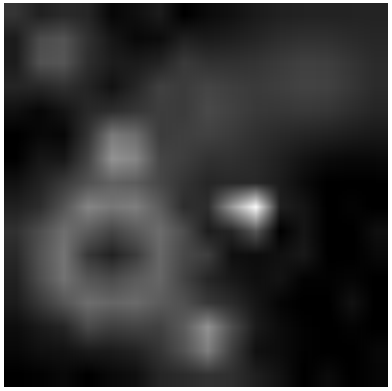
Original

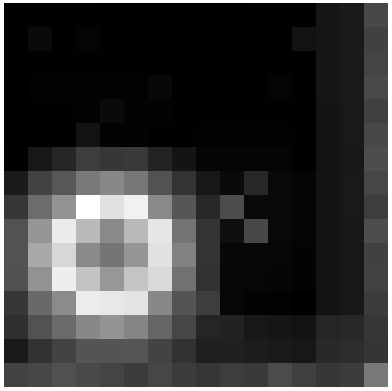


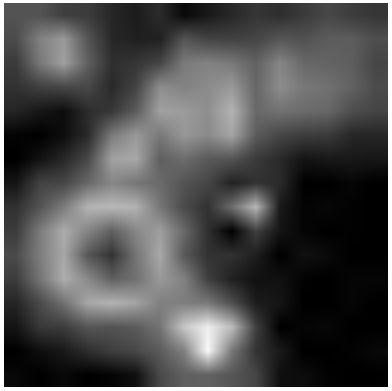
PhotoshopGrey



Color2Grey







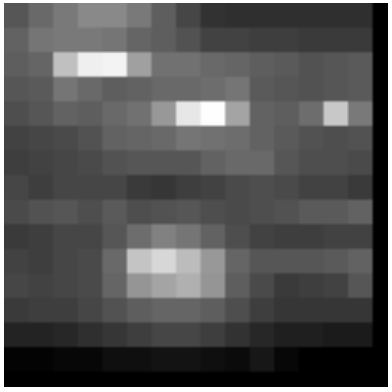
Apply Contrast Attention model by Ma and Zhang 2003

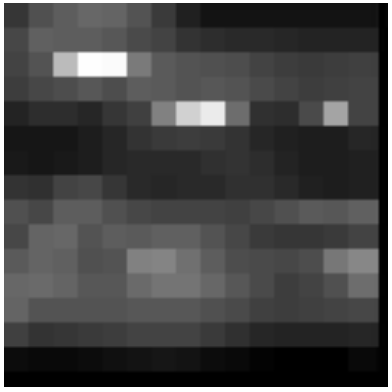
Validate "Salience Preserving"

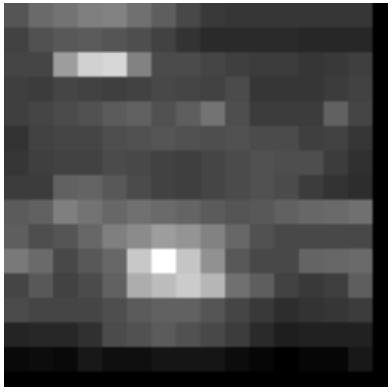
Original

PhotoshopGrey

Color2Grey





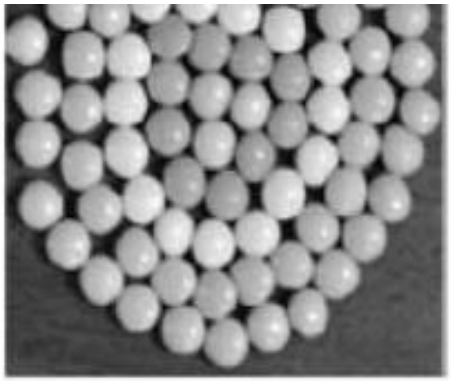


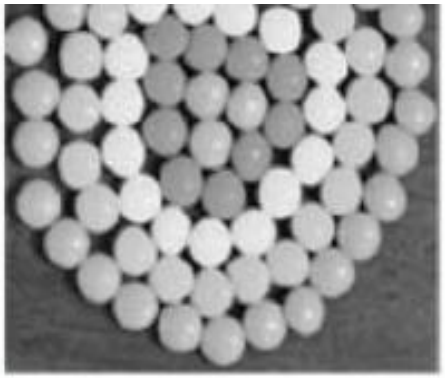
IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 32, NO. 9, SEPTEMBER 2010

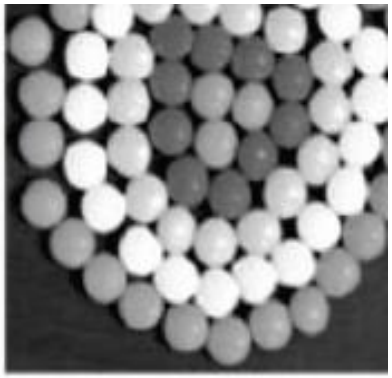
Color to Gray: Visual Cue Preservation

Mingli Song, Member, IEEE, Dacheng Tao, Member, IEEE, Chun Chen,
Xuelong Li, Senior Member, IEEE, and Chang Wen Chen, Fellow, IEEE

- algorithm based on a probabilistic graphical model with the assumption that the image is defined over a Markov random field.
- color to gray procedure can be regarded as a **labeling process** to preserve the newly well-defined visual cues of a color image in the transformed gray-scale image
- **three cues** are defined namely, color spatial consistency, image structure information, and color channel perception priority
- visual cue preservation procedure based on a probabilistic graphical model and optimize the model based on an integral minimization problem.

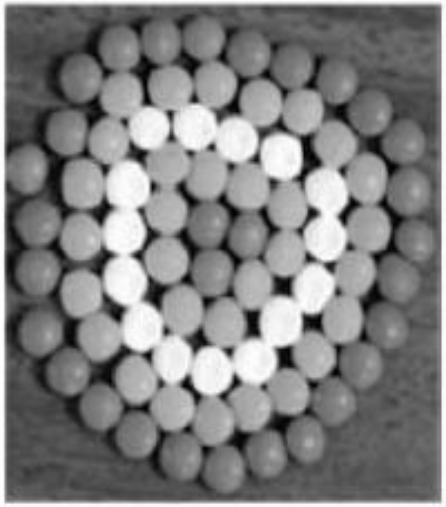






KPCA

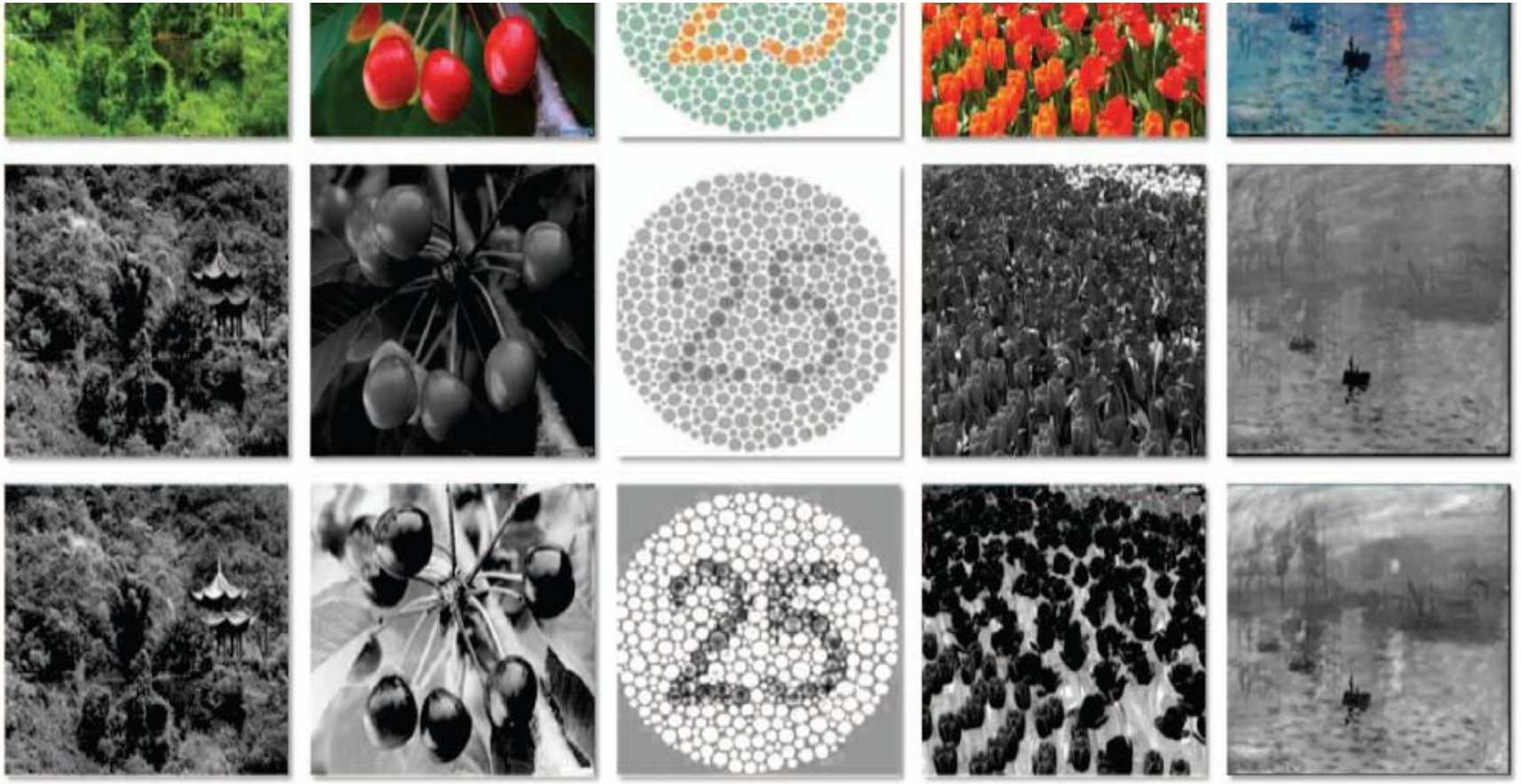
VCP



Preference Matrix

	PS	ICG	MDS	KPCA	VCP	Scor
PS	--	3	0	0	0	3
ICG	28	--	3	2	3	36
MDS	31	28	--	15	15	89
KPCA	31	29	16	--	15	91
VCP	31	28	16	16	--	91

or this case, KPCA and the proposed VCP achieved the best performance, and MDS performed comparably to the KPC



Colorization: History

- Hand tinting
  - http://en.wikipedia.org/wiki/Hand-colouring

Colorization: History

- Film colorization
  - http://en.wikipedia.org/wiki/Film_colorization

Colorization in 1986

Colorization in 2004

- Colorization using “scribbles”

Colorization by example

R.Irony, D.Cohen-Or, D.Lischinski

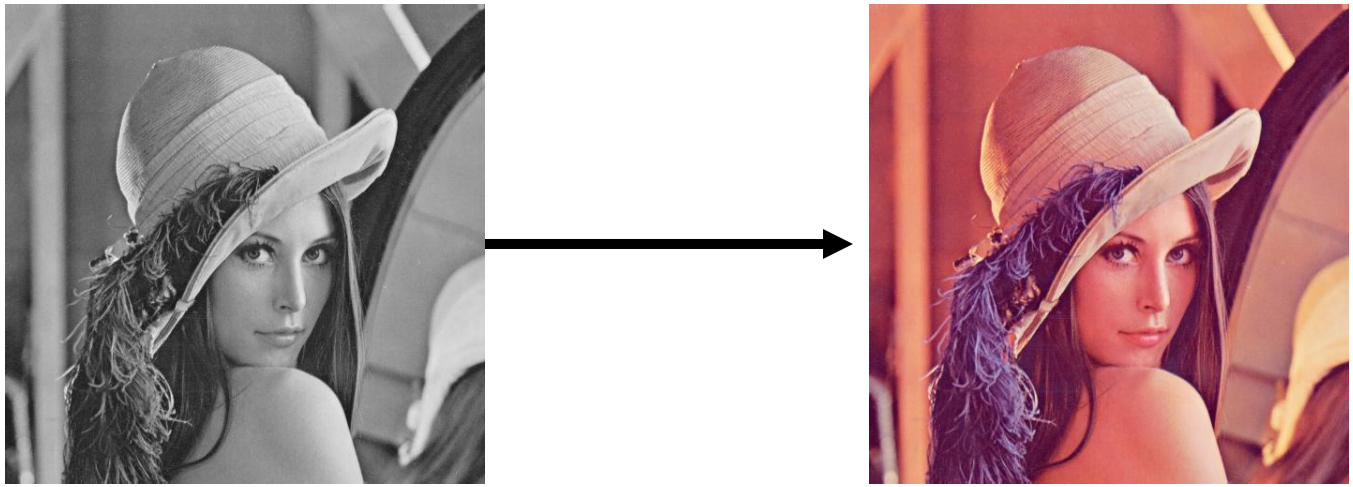
Eurographics Symposium on Rendering (2005)

Motivation

■**Colorization, is the process of adding color to monochrome images and video.**

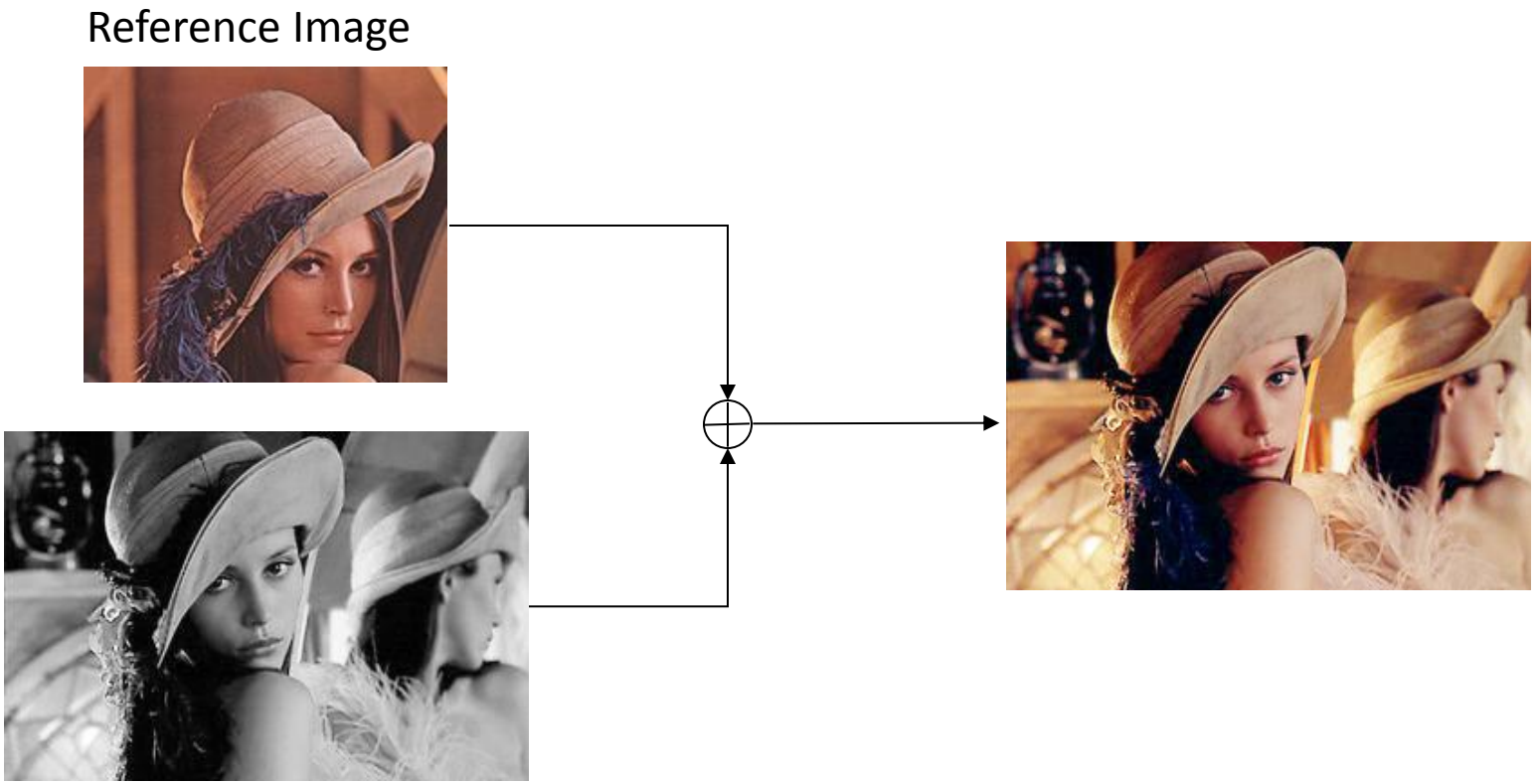
■**Colorization typically involves segmentation + tracking regions across frames**
- neither can be done reliably – user intervention required – expensive + time consuming

■ **Colorization by example – no need for accurate segmentation/region tracking**



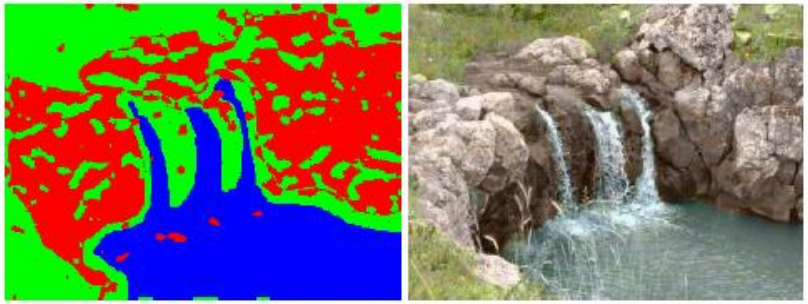
The method

- Colorize a grayscale image based on a user provided reference.



(a) Levin *et al.*'s colorization. Left: dozens of user drawn scribbles (some very small). Right: resulting colorization.

(b) Reference image along with a partial segmentation.



(c) Our classification and resulting colorization.

Figure 2: *(a) The method of Levin et al. might require the user to carefully place a multitude of appropriately colored scribbles. (b) Our approach requires an example image with a few user-marked or automatically segmented regions, and produces a comparable colorization (c).*

Naïve Method

Transferring color to grayscale images [Walsh, Ashikhmin, Mueller 2002]

- Find a good match between a pixel and its neighborhood in a grayscale image and in a reference image.

By Example Method

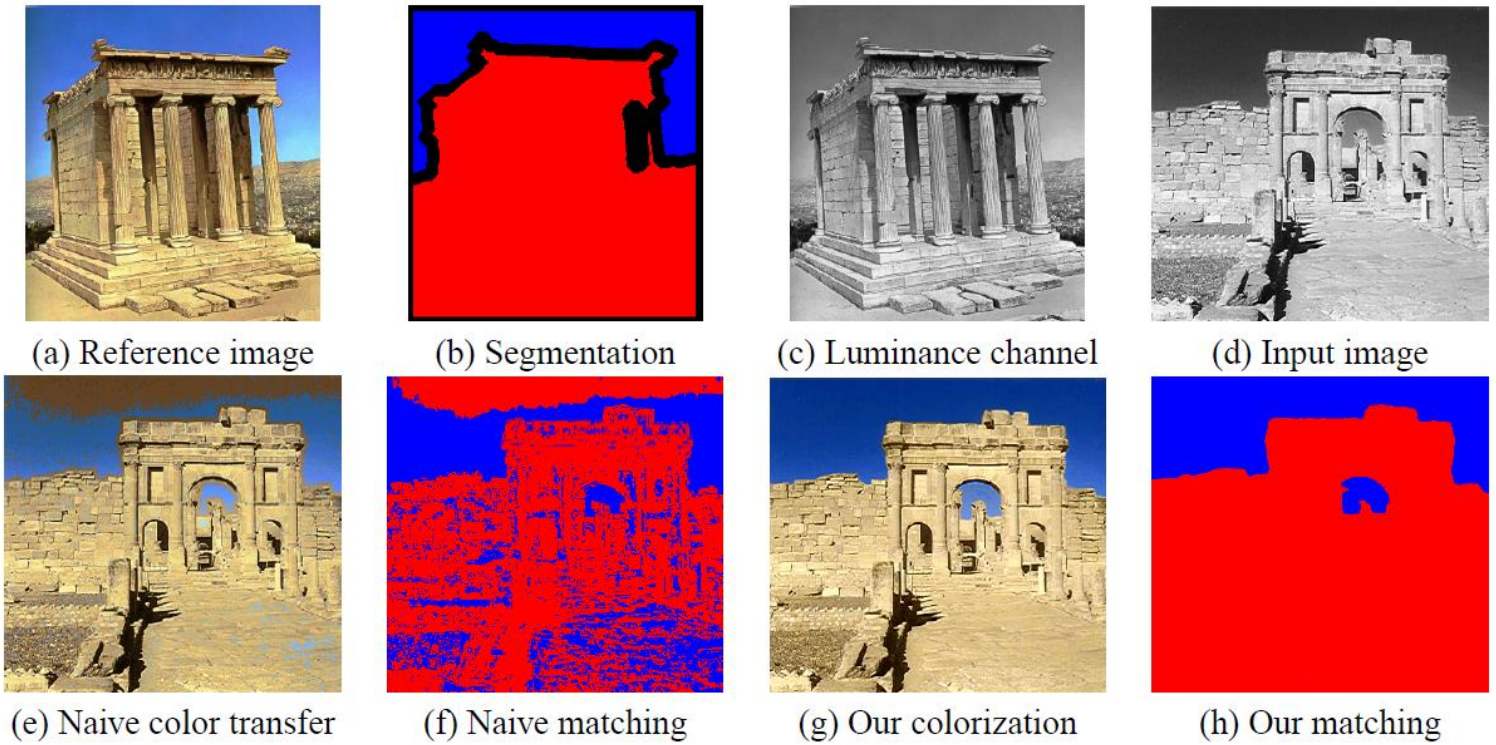
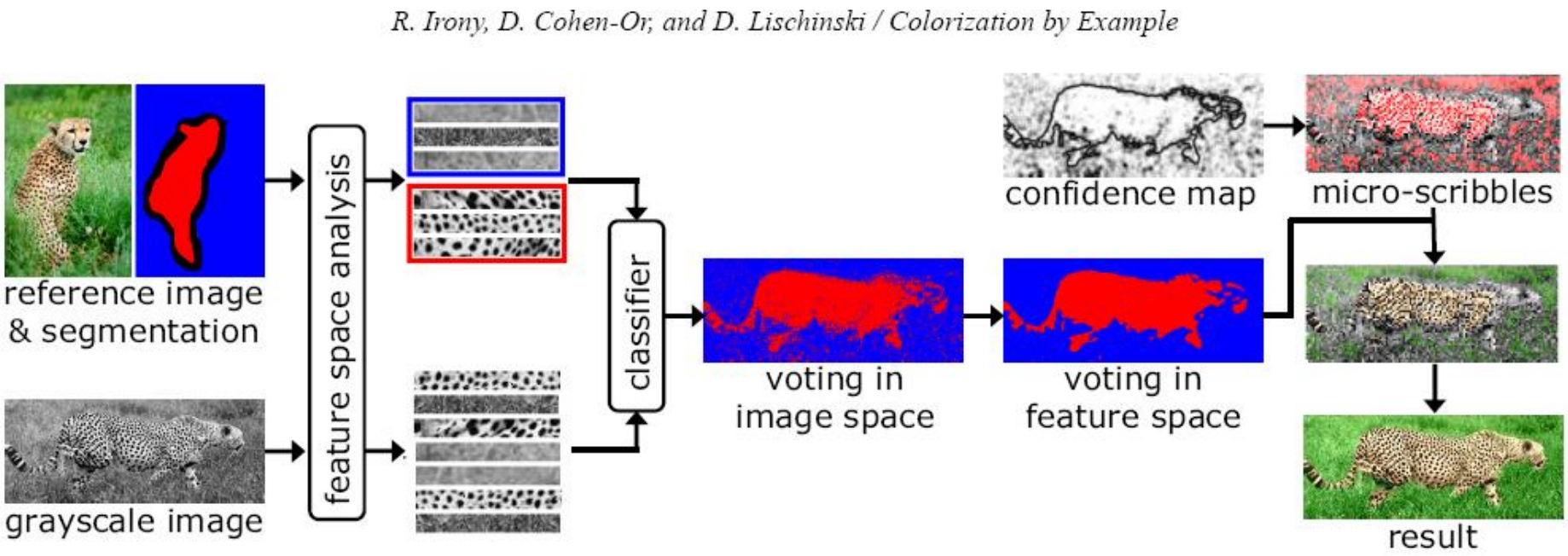


Figure 1: *Our approach vs. color transfer. A reference color image (a) was automatically segmented into two major regions (b). Transferring color to a grayscale image (d) by matching means and variances of pixel neighborhoods (as described in [WAM02]) produces a poor result in this case (e), since pixels in (d) are matched to pixels in (c) in an incoherent manner, as visualized in (f). Our approach produces a much better result (g), since it matches pixels in a more contiguous manner (h).*

Overview

1. training
2. classification
3. color transfer



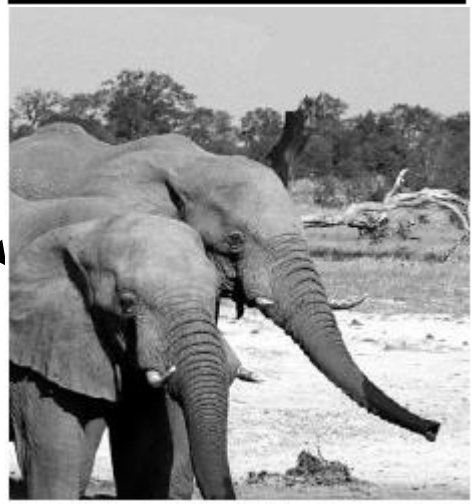
Training stage

Input:

1. The luminance channel of the reference image
2. The accompanying partial segmentation

Construct a low dimensional feature space in which it is easy to discriminate between pixels belonging to differently labeled regions, based on a small (grayscale) neighborhood around each pixel.

Training stage



Create Feature Space
(get DCT cefficients)

Classification stage

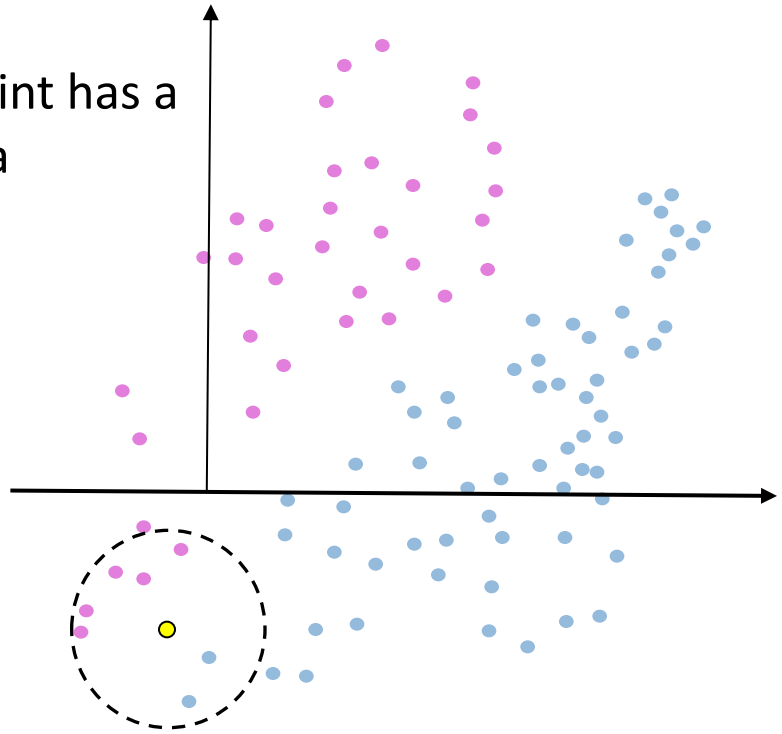
For each grayscale image pixel, determine which region should be used as a color reference for this pixel.

One way: K-Nearest –Neighbor Rule

Better way: KNN in discriminating subspace.

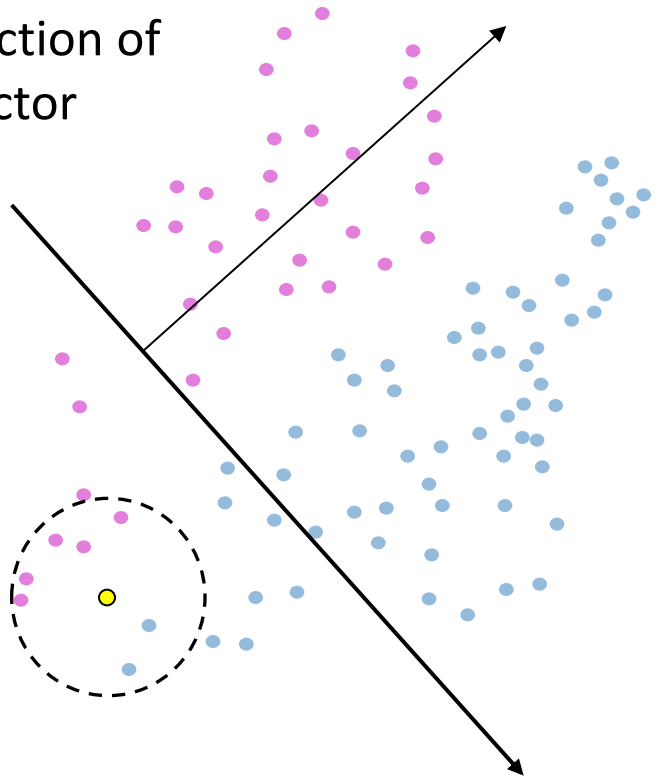
KNN in discriminating subspace

Originaly sample point has a majority of Magenta



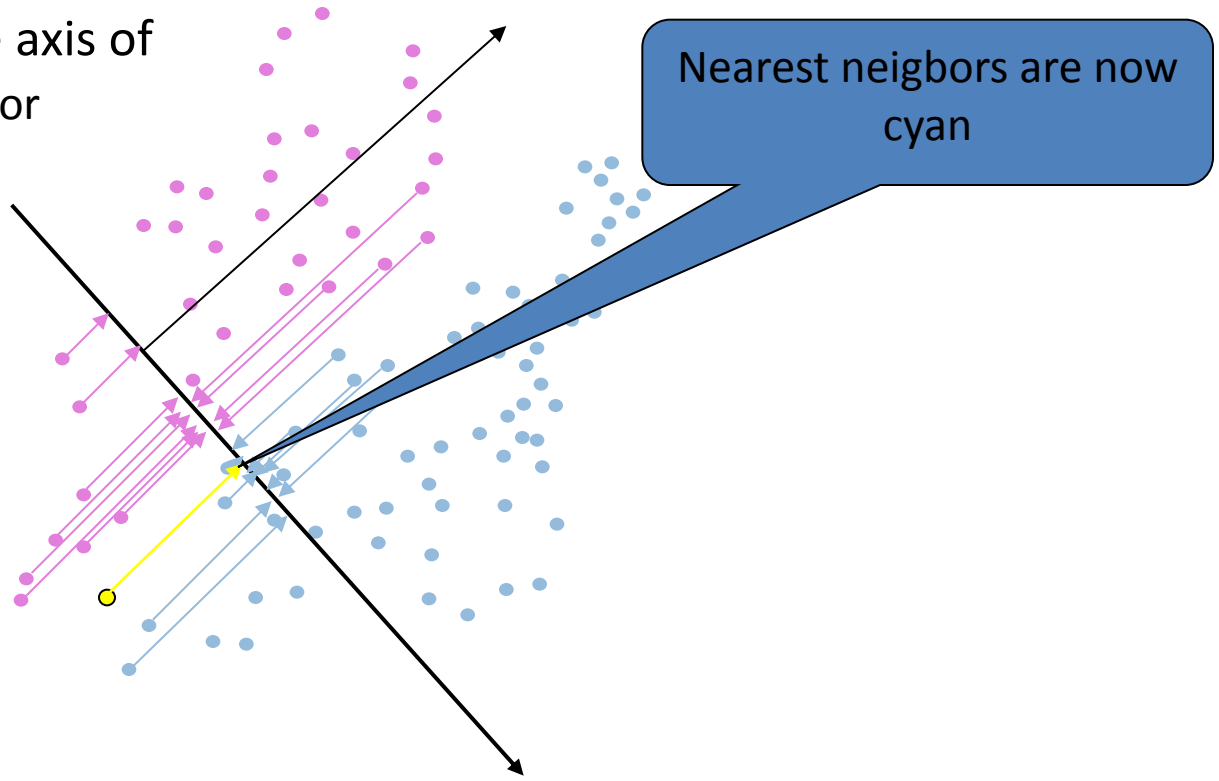
KNN in discriminating subspace

Rotate Axes in the direction of the intra-difference vector



KNN in discriminating subspace

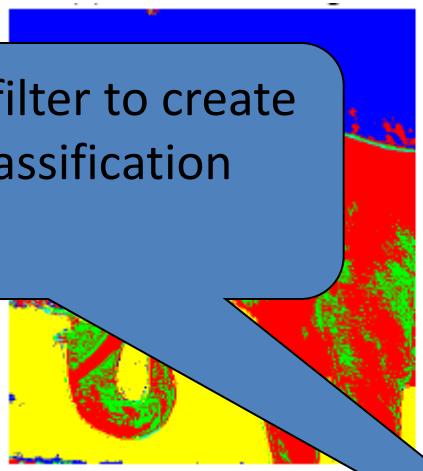
Project Points onto the axis of the inter-difference vector



KNN Differences

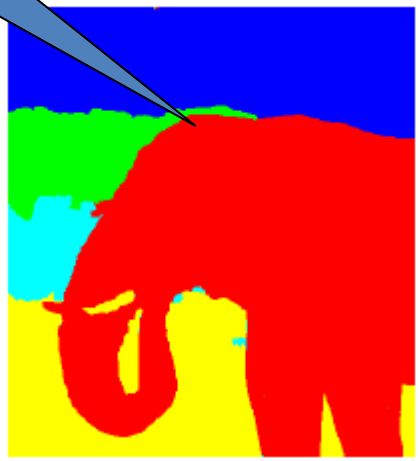
Matching

Classification



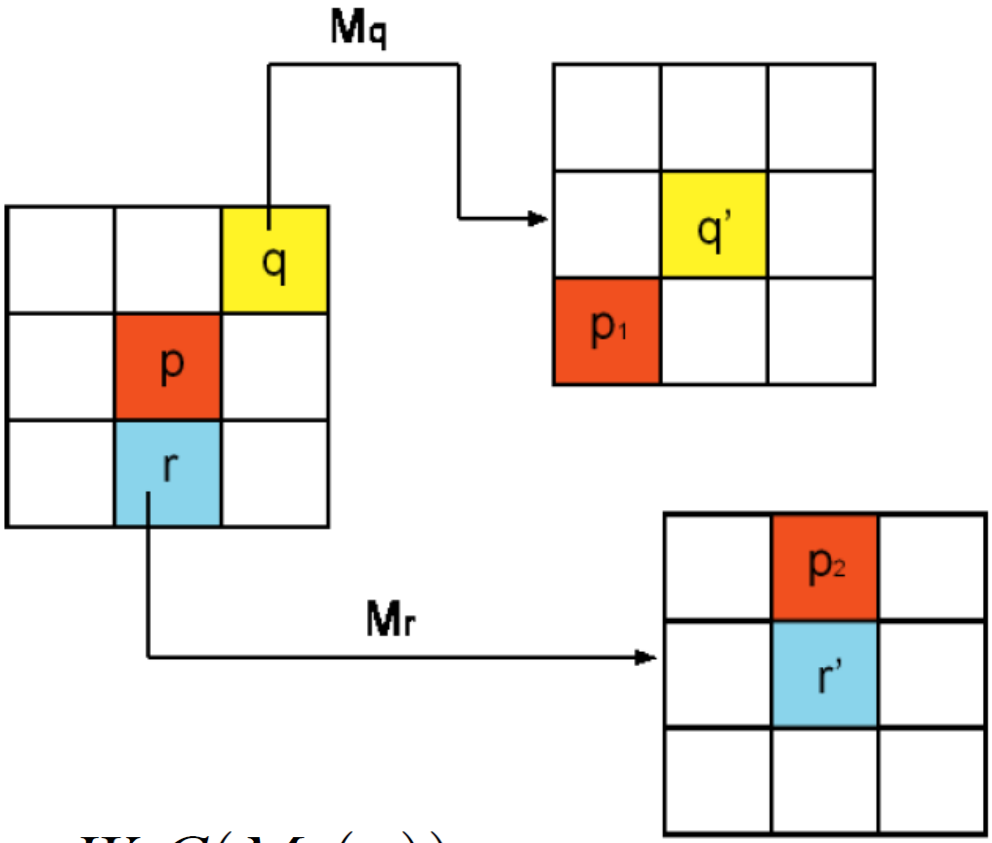
Use a median filter to create a cleaner classification

Discriminating KNN



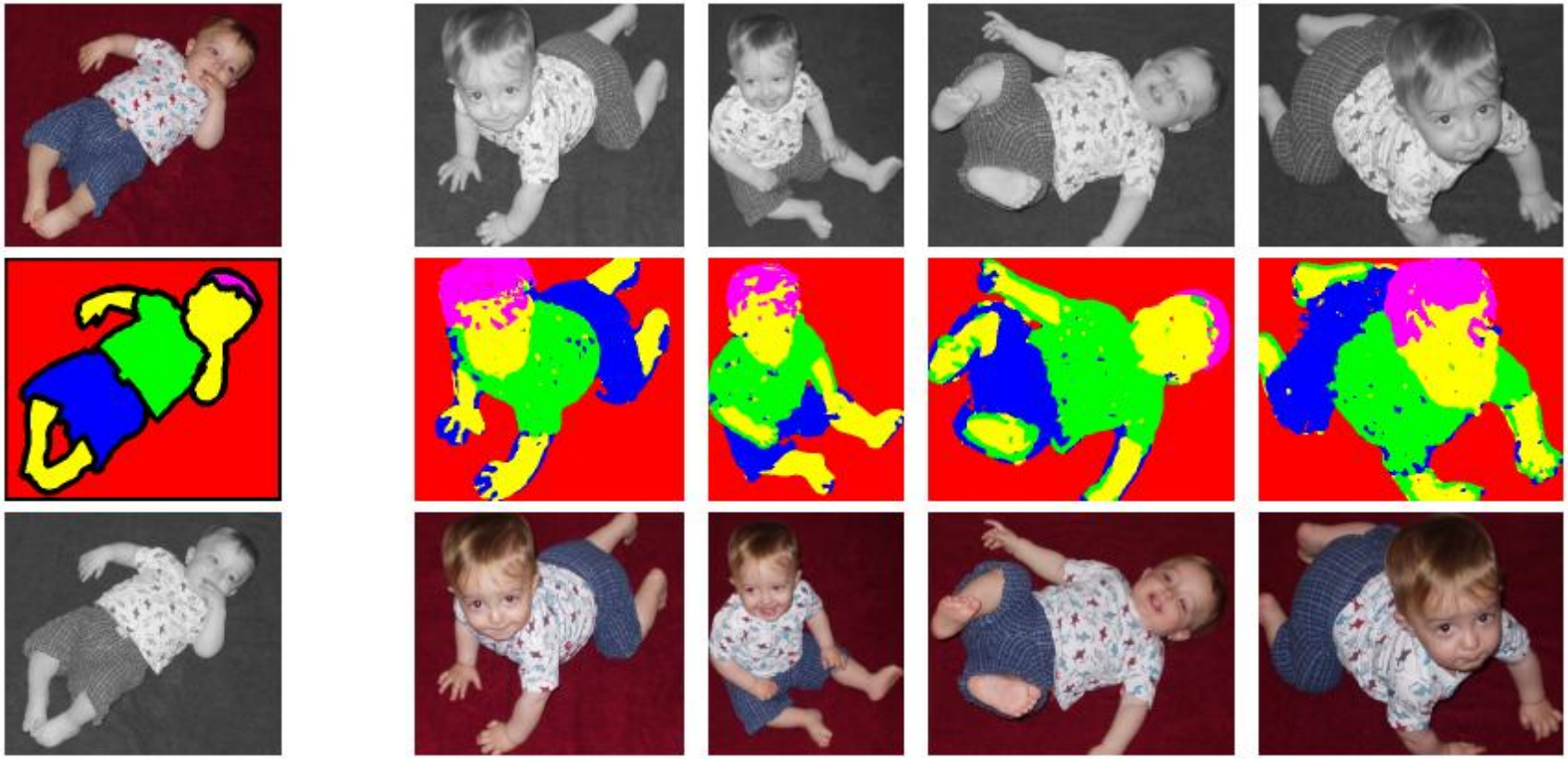
Color transfer

Using YUV color space



$$C(p) = \sum_{q \in N(p,\ell)} W_q C(M_q(p))$$

Final Results



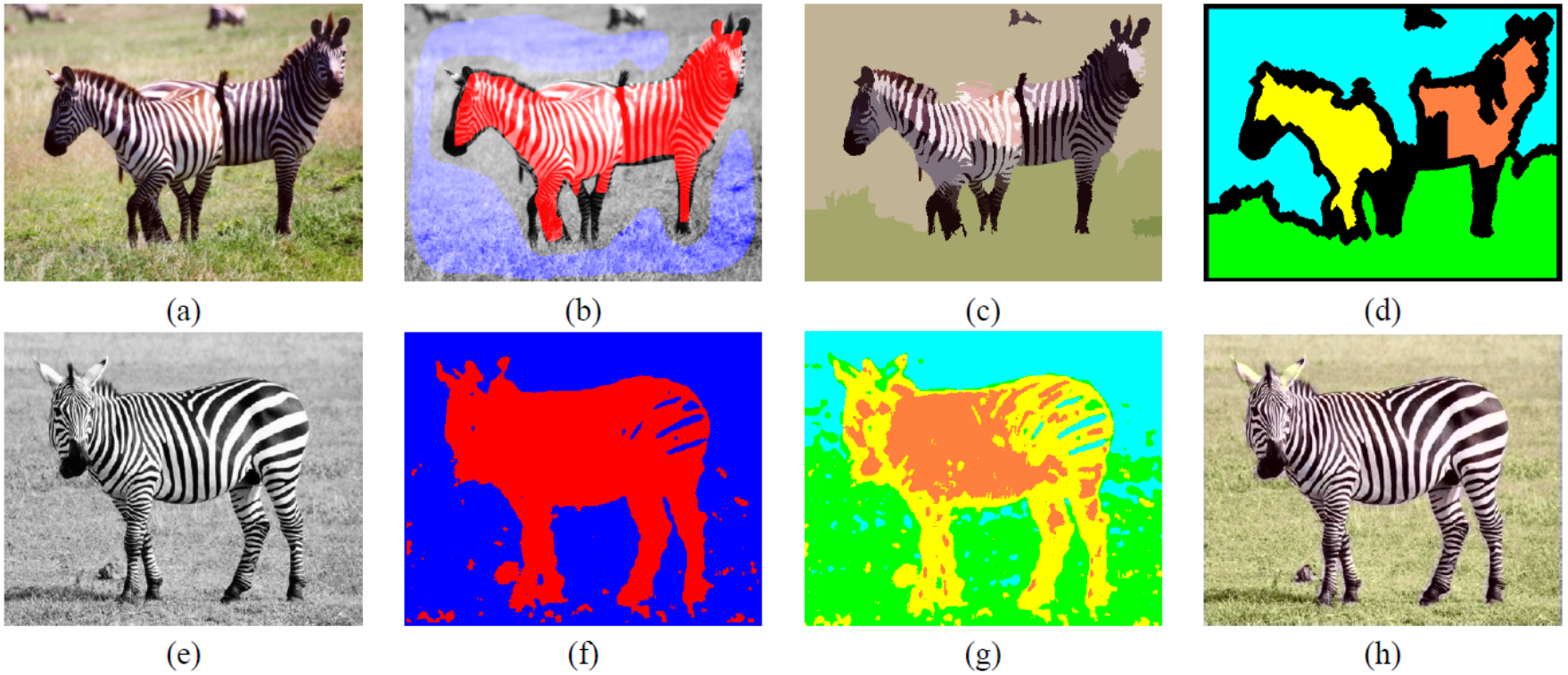


Figure 11: *Automatic region marking reduces human effort, while producing very similar results. (a) Reference image. (b) Manually marked regions: zebra (marked in red) and grass (marked in blue). (c) Automatic (mean shift) segmentation. (d) Regions obtained by merging and shrinking segments in (c). Zebra regions are now marked in orange and yellow, while grass regions are in green and cyan. (e) A new grayscale zebra image. (f) Classification using the manually marked regions in (b). (g) Classification using the automatic regions in (d). The union of the orange and yellow pixels is nearly identical to the red pixels in (f), meaning that both classification agree on which pixels best match the zebra region in the reference image, and produce the same colorization (h).*