University of Applied Sciences
Department of
Design, Computer Science, Media
Prof. Dr. D. Richter

Transilvania University Brasov
Faculty of
Mathematics and Informatics
Dr. I. Plajer

ISIP 2019
International School on Image Processing
July 15th - 26th, 2019

detlef.richter@HS-RM.de

University of Applied Sciences
Department of
Design, Computer Science, Media
Prof. Dr. D. Richter
detlef.richter@HS-RM.de

Lectures in Image Processing

Chapter 12B
Color Image Processing

12.b Color Image Processing

12.b Color Image Processing, Notations

Not discussed considerations:

- Does it matter which color space is used for color image processing?
- Are results of color image processing independent on the color space used?
- Exist „ideal“ color spaces for certain color image operations?

The following discussion is based on RGB space.

12.b Color Image Processing, Notations

Notations used in this chapter depending on context:

Color vector at position (x,y)	cImg(x,y)	c(x,y)	f(x,y)	pix(x,y)
Red component at position (x,y)	rImg(x,y)	c_r(x,y)	r(x,y)	pix_r(x,y)
Green component at position (x,y)	gImg(x,y)	c_g(x,y)	g(x,y)	pix_g(x,y)
Blue component at position (x,y)	bImg(x,y)	c_b(x,y)	b(x,y)	pix_b(x,y)

12.b Color Image Processing, 12.1 Grey Levels vs. Colors

Grey Level Images	*Color Images*
2 image coordinates (x,y) + 1 grey intensity = 3D-problem	2 image coordinates (x,y) + 3 colors = 5D-problem
mean grey level variance noise reduction of intensity edge detection of intensity segmentation: algorithm of Otsu k-means cluster	mean color vector covariance matrix noise reduction in RGB etc. edge detection in RGB etc. segmentation: supervised or unsuperviced learning k-means cluster

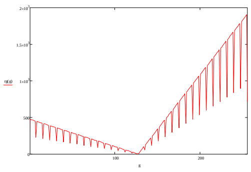
12.b Color Image Processing, 12.3 Synthetic Images

Synthetic image for algorithm testing

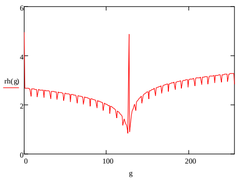
- Red, yellow. Green, cyan, blue, magenta, black, white
- Intensity of outer zone: full saturation (255)
- Intensity in center area: 128 (127) depending on rounding
- Uniform grey 127

12.b Color Image Processing, 12.3 Synthetic Images

How will look like the histogram of the colored square (without grey value fields) ?
How will look like the histogram of the total image (including grey value fields) ?



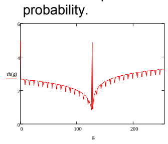
Zero values deleted, minimum at 127/128



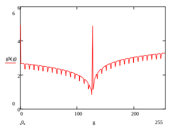
Local maxima at 0 and 127, log. Representation, incl. grey

12.b Color Image Processing, 12.3 Synthetic Images

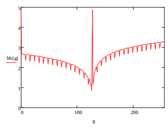
All color components have the same probability.



Logarithmic Histogram Red Channel



Logarithmic Histogram Blue Channel

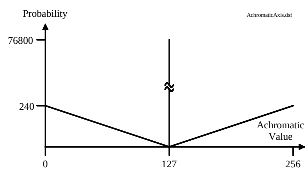


Logarithmic Histogram Green Channel

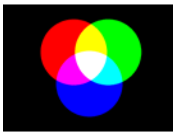
12.b Color Image Processing, 12.3 Synthetic Images

How will look like the probability of the achromatic values?

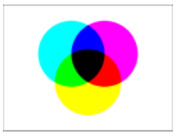
12.b Color Image Processing, 12.3 Synthetic Images



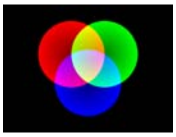
12.b Color Image Processing, 12.3 Synthetic Images



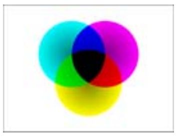
Upper left:
Additive
color mixing



Upper right:
Subtractive
color mixing

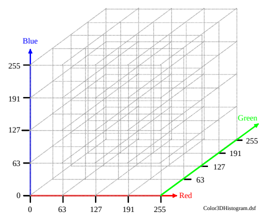


Lower left:
Color value
outside = 255,
center = 128



Lower right:
Color value
outside = 255,
center = 128

12.b Color Image Processing, 12.4 Color Histogram



3D Color Histogram

- Color space divided into four clusters {opt. 8, …} for each primary vector.
- ⇒ 64 color cubes {opt. 512, …}
- Each cube contains the number of respective color pixels of the image.
- Representation:
  - Standard size sphere of resp. color at resp. position for each count > threshold
  - Size of sphere related to number of counts

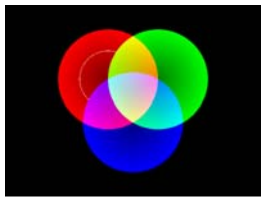
12.b Color Image Processing, 12.5 Discrete Colors

Search for discrete colors:

example:

r = 200
g = 200
b = 0

12.b Color Image Processing, 12.5 Discrete Colors

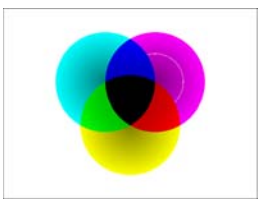


Search for discrete colors:

example:

r = 200
g = 0
b = 0

12.b Color Image Processing, 12.5 Discrete Colors

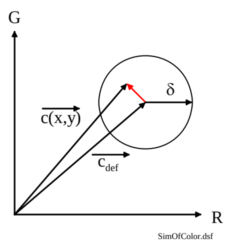


Search for discrete colors:

example:

r = 200
g = 0
b = 200

12.b Color Image Processing, 12.6 Similar Colors



defining 3D color vector c_{def}

actual 3D color vector c(x,y)

Similarity exists for a given δ, if:

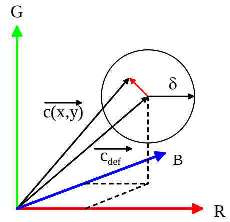
$$\|c_{def} - c(x,y)\| \le \delta$$

$$\|c_{def} - c(x,y)\| =$$

$$\left[(c_{def} - c(x,y))^T (c_{def} - c(x,y))\right]^{1/2} =$$

$$\left[(c_{def,R} - c_R(x,y))^2 + (c_{def,G} - c_G(x,y))^2 + (c_{def,B} - c_B(x,y))^2\right]^{1/2}$$

12.b Color Image Processing, 12.6 *Similar Colors*



SimOfColor.dsf

defining 3D color vector c_{def}

actual 3D color vector c(x,y)

Similarity exists for a given δ, if:

$$\left\| c_{def} - c(x,y) \right\| \le \delta$$

$$\left\| c_{def} - c(x,y) \right\| =$$

$$\left[\left(c_{def} - c(x,y) \right)^T \left(c_{def} - c(x,y) \right) \right]^{1/2} =$$

$$\left[\left(c_{def,R} - c_R(x,y) \right)^2 + \left(c_{def,G} - c_G(x,y) \right)^2 + \left(c_{def,B} - c_B(x,y) \right)^2 \right]^{1/2}$$

12.b Color Image Processing, 12.6 *Similar Colors*

Example for light blue:
r = 80, g = 120, b = 165
δ = 30

if $\left\| c_{def} - c(x,y) \right\| \le \delta$

c(x,y) = 255

12.b Color Image Processing, 12.6 *Similar Colors*

Example for light green:
r = 145, g = 200, b = 70
δ = 40

if $\left\| c_{def} - c(x,y) \right\| \le \delta$

c(x,y) = 255

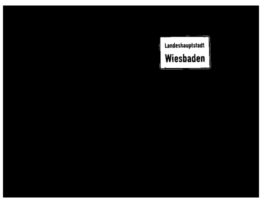
12.b Color Image Processing, 12.6 *Similar Colors, Application*

Red: r=194, g=0, b=41, δ=45

Segmented image:

12.b Color Image Processing, 12.6 *Similar Colors, Application*

Example for yellow sign: r = 221, g = 138, b = 3, δ = 40



if $\left\| c_{def} - c(x,y) \right\| \le \delta$: c(x,y) = 255, else c(x,y) = 0

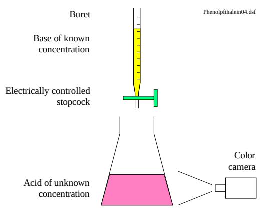
12.b Color Image Processing, 12.6 *Similar Colors,* Chem. Process Control

Phenolphthalein

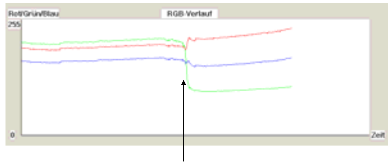
- Chemical pH Indicator
- pH < 8.2: transparent liquid (acid)
- pH > 8.2: magenta (basic, alkaline)
- pH = 7.0: neutral solution

pH ~ -$\log_{10}(H^+)$

12.b Color Image Processing, 12.6 *Similar Colors,* Chem. Process Control



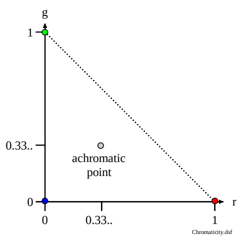
12.b Color Image Processing, 12.6 *Similar Colors,* Chem. Process Control



Color change from transparent to magenta.
Unknown acid concentration is calculated from known base concentration and time of color change

Measurement time: 300 s
Time intervals: 500 ms

12.b Color Image Processing, 12.7 *Chromatic Map* *(Farbarten)*



2D Chromaticity Plane

- RGB color system:
- Reduction of 3D color space into 2D color space with chromaticity independent of luminance

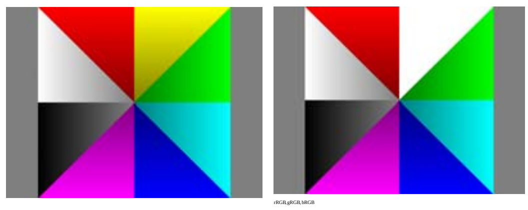
$$r(x,y) = \frac{rImg(x,y)}{rImg(x,y) + gImg(x,y) + bImg(x,y)}$$

$$g(x,y) = \frac{gImg(x,y)}{rImg(x,y) + gImg(x,y) + bImg(x,y)}$$

$$b(x,y) = 1 - (r(x,y) + g(x,y)) \quad \text{Eq.(1)}$$

- r(x,y), g(x,y), b(x,y) are normalized intensities
- Eq. (1): b = 1 - (r+g) is redundant

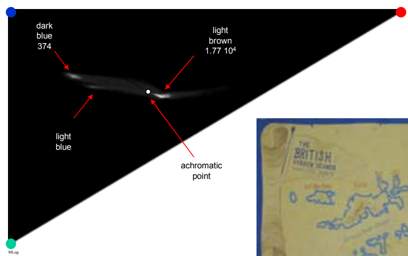
12.b Color Image Processing, 12.7 *Chromatic Map* *(Farbarten)*



Deleting chromatic colors of r = 200, g = 200, b = 0

12.b Color Image Processing, 12.7 *Chromatic Map* *(Farbarten)*

12.b Color Image Processing, 12.7 *Chromatic Map* *(Farbarten)*



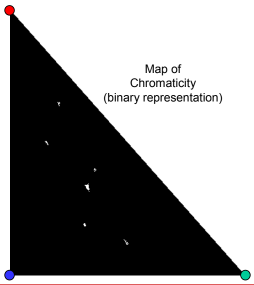
Example of natural image and chromaticity map (logarithmic representation)

12.b Color Image Processing, 12.7 Segmentation by *Chromatic Map*

Colored Balloons

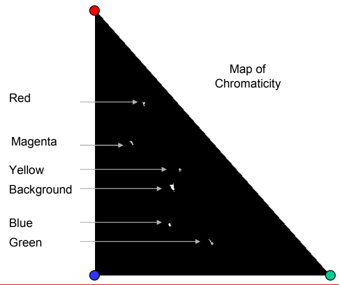
Image©Schwanecke, HS-RM

Segmentation of Colors
using real image including
shades, shading and noise

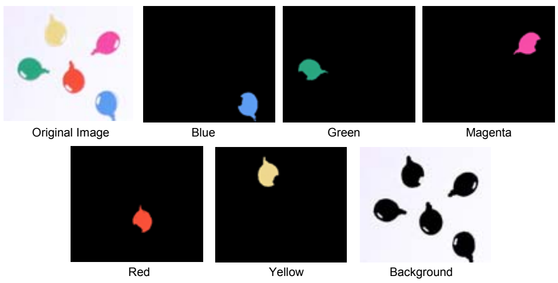


12.b Color Image Processing, 12.7 Segmentation by *Chromatic Map*

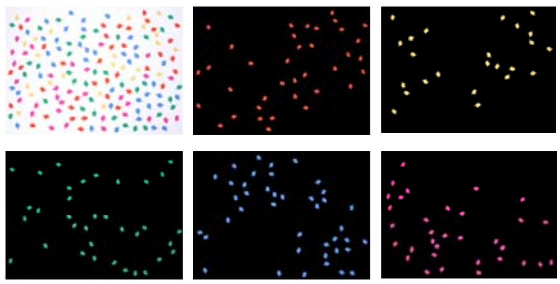
- Analyzing chromaticity of image
- Application of global threshold for chromaticity (thr=40) and binarization
- Defining positions of chromaticity
- Checking original image for defined chromaticity within given 3D-surrounding (δ=30)
- Reconstructing the resulting segmentation



12.b Color Image Processing, 12.7 Segmentation by *Chromatic Map*



12.b Color Image Processing, 12.7 Segmentation by *Chromatic Map*



12.b Color Image Processing, 12.7 Segmentation by *Chromatic Map*

R = 182, G = 147, B = 122, δ = 30

Source: P. Macho, HS-RM, WS2016/17, 27 Fehler

12.b Color Image Processing, 12.8 *Covariance Matrix*

Assume to have a RGB-image with color vectors for each pixel $\overrightarrow{\text{cImg}(x,y)} = \begin{pmatrix} \text{rImg}(x,y) \\ \text{gImg}(x,y) \\ \text{bImg}(x,y \end{pmatrix}$

Mean color vector is defined by $\overrightarrow{\text{cMean}} = \frac{1}{\text{Width x Height}} \sum_{x=0}^{\text{Width-1}} \sum_{y=0}^{\text{Height-1}} \overrightarrow{\text{cImg}(x,y)}$

Calculate covariance (CoV) matrix by

$$\text{CoV} = \frac{1}{\text{Width x Height}} * \sum_{x=0}^{\text{Width}-1} \sum_{y=0}^{\text{Height}-1} \begin{pmatrix} \text{rImg}(x,y) - \text{rMean} \\ \text{gImg}(x,y) - \text{gMean} \\ \text{bImg}(x,y) - \text{bMean} \end{pmatrix} * \begin{pmatrix} \text{rImg}(x,y) - \text{rMean} & \text{gImg}(x,y) - \text{gMean} & \text{bImg}(x,y) - \text{bMean} \end{pmatrix}$$

The diagonal elements of CoV show the variances of the rImg, gImg and bImg resp.

12.b Color Image Processing, 12.8 *Covariance Matrix*

Test the program by using a 1-bit 4 x 4 color image and calculate the correct values

$$\text{CoVBild} := \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

(red | green | blue)

$$\text{CoV} = \begin{pmatrix} 0.25 & 0 & 0.063 \\ 0 & 0.188 & 0.063 \\ 0.063 & 0.063 & 0.188 \end{pmatrix}$$

Specification:

red:
8 pixels 1, 8 pixels 0
rMean = 0.5

green:
4 pixels 1, 12 pixels 0
gMean = 4/16 = 0.25

blue:
12 pixels 1, 4 pixels 0
bMean = 12/16 = 0.75

12.b Color Image Processing, 12.8 *Covariance Matrix*

$$\text{CoV} = \begin{pmatrix} 39.21 & 50.412 & 19.043 \\ 50.412 & 69.255 & 24.336 \\ 19.043 & 24.336 & 13.621 \end{pmatrix}$$

$\sigma_{red} \approx 6.2$; $\sigma_{green} \approx 8.3$; $\sigma_{blue} \approx 3.6$

12.b Color Image Processing, 12.9 *Color Resolution*

This process results in $4^3 = 64$ different colors.

Alternative: six groups resulting in $6^3 = 216$ different colors.

These colors represent color palettes.

12.b Color Image Processing, 12.9 *Color Resolution*

Synthetic image

How will the image look like, if the colors of each color channel is divided into for groups?

2^{24} different colors

12.b Color Image Processing, 12.9 *Color Resolution*

Synthetic image

2^{24} different colors

4^3 different colors

12.b Color Image Processing, 12.9 *Color Resolution*

Natural image

2^{24} different colors

4^3 different colors

12.b Color Image Processing, 12.9 *Color Resolution*

Natural image

2^{24} different colors

6^3 different colors

12.b Color Image Processing, 12.10 Low Pass Filter

- There are several options to construct color Low Pass Filters
  1. Mean, Gauss or Binomial filter applied separately on each color channel, producing new color values.
  2. Median filter applied separately on each color channel, preserving the original color values in each color channel, but generating new color combinations in the color image.
  3. Median filter applied on color vectors, preserving the original color values.
  4. Median filter applied on color distances to the most similar color

12.b Color Image Processing, 12.10 Mean Low Pass Filter

1st Option:

- Decomposition of the color image (RGB) into three color channels.
  - Separate low pass filtering of each color channel (e.g. 3 x 3 Mean Filter, 5 x 5 Binomial Filter, σ-dependent Gauss Filter).
  - Composing the color image from the three color channels.
- These filters create new r, g, b values in the color channels and thus create new colors in the color image.

12.b Color Image Processing, 12.10 Median Low Pass Filter

2nd Option:

- Decomposition of the color image (RGB) into three color channels.
  - Separate low pass filtering of each color channel using median filter.
  - Composing the color image from the three color channels.
- This filter creates no new r, g, b values in the color channels, but creates new composition of the colors and thus creates also new colors in the color image. Resulting RGB combination of the pixel may not correspond to colors in the pixel neighborhood.
  - Consequently chromatic shifts may occur particularly in neighborhood of edge

12.b Color Image Processing, 12.10 Median Low Pass Filter

3rd Option:

- Apply Median Filter with 3 x 3 mask on the color image.
- Calculate for each mask element i the magnitude of the color vector f_i,

  i.e. $\left|\overrightarrow{f(x_i, y_i)}\right| = \sqrt{\sum_{r,g,b} [pix(x_i, y_i)]^2} = \sqrt{(pix_r(x_i, y_i))^2 + (pix_g(x_i, y_i))^2 + (pix_b(x_i, y_i))^2}$

- Sort the color vectors f_i in increasing (or decreasing) sequence of color vectors into a 9-element array.
- Insert the color components $pix_r(x_{med}, y_{med})$, $pix_g(x_{med}, y_{med})$, $pix_b(x_{med}, y_{med})$ of the median into the target pixel of the color image.
- No new colors are created

12.b Color Image Processing, 12.10 Median Low Pass Filter

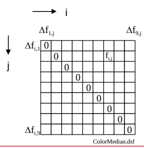
4th Option:

- Apply 3 x 3 mask on the color vectors.
- Calculate for all pairs of mask elements i,j the differences of the color vectors $f_{i,j}$.

i.e. $\Delta f_{i,j} = \left\| f_i - f_j \right\| = \sqrt{(r_i - r_j)^2 + (g_i - g_j)^2 + (b_i - b_j)^2}$

in which r_i : red component $r(x_i,y_i)$ etc.

It results in a 9 x 9 array of color distances:



12.b Color Image Processing, 12.10 Median Low Pass Filter

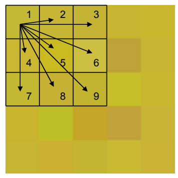
- The array contains the 9 distance vectors $d_i = (\Delta f_{i,1}, \Delta f_{i,2}, \ldots, \Delta f_{i,9})$, denoting the color distances of point i of the mask to all the other points within the mask, including to point i itself with color distance = 0.
- The value $D = \min\left\{ \sum_{j=1}^{9} \Delta f_{i,j} \right\}$

  denotes the color vector d_i of the pixel, the color of which is most similar to all the other pixels within the mask.
- The median of the elements of the vector d_i with value D: $\operatorname{med}\left\{ \Delta f_{i=\min,\, j} \right\}$ is assigned to the target pixel.

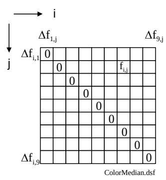
12.b Color Image Processing, 12.10 Median Low Pass Filter

- 5 x 5 synthetic image
- r = 200, g = 180, b = 50
- first/last line/column pixels with superimposed noise ± 3
- inner pixels with superimposed noise ± 20
- applying 3 x 3 mask changes only the inner pixels

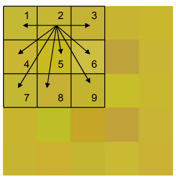
12.b Color Image Processing, 12.10 Median Low Pass Filter



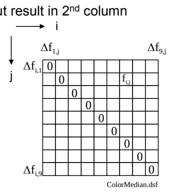
- Calculate differences $\Delta f_{i=1,j}$, j=1 .. 9, i=1 denoting the reference element 1, j denoting the mask elements 1 .. 9.
- Put result in 1st column



12.b Color Image Processing, 12.10 Median Low Pass Filter



- Continue calculating differences $\Delta f_{i=1,j}$, j=1 .. 9, i=2 denoting the reference element 2, j denoting the mask elements 1 .. 9.
- Put result in 2nd column



12.b Color Image Processing, 12.10 Median Low Pass Filter

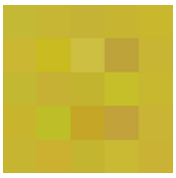
ColDist =

	0	1	2	3	4	5	6	7	8
0	0	33.481	36.674	23.452	17.972	10.05	14.177	22.204	36.742
1	33.481	0	33.226	19.723	23.791	24.739	29.155	37.656	31.796
2	36.674	33.226	0	17.117	20.347	33.764	32.772	20.833	5.196
3	23.452	19.723	17.117	0	7.81	18.466	20.712	18.628	16.31
4	17.972	23.791	20.347	7.81	0	13.491	13.638	16.912	21.095
5	10.05	24.739	33.764	18.466	13.491	0	9.055	25.259	33.985
6	14.177	29.155	32.772	20.712	13.638	9.055	0	26.038	34.366
7	22.204	37.656	20.833	18.628	16.912	25.259	26.038	0	20.616
8	36.742	31.796	5.196	16.31	21.095	33.985	34.366	20.616	0
9	194.753	233.568	199.93	142.219	135.056	168.809	179.914	188.145	200.106

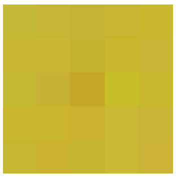
Example for source pixel (x=2, y=2)

- Sum up all differences $\Delta f_{i=const,j}$ for i = 0 … 8. Write sum in element i = 9.
- Search for D = min{ ∑Δfi,j } = 135.056, find i = 5. Sum of differences to pixel 5 (index 4) is the minimum.
- Search for median{$\Delta f_{3,j}$} = 16.912, find j = 7, which refers to element 8.
- Put color values r,g,b of element 8 to target pixel.
- Shift mask to next position.

12.b Color Image Processing, 12.10 Median Low Pass Filter



Original image



Corrected image

12.b Color Image Processing, 12.11 Edge Detection

Solution Ansatz ?

Intensity I= 0.299 r + 0.587 g + 0.114 b

12.b Color Image Processing, 12.11 Edge Detection

- High pass filtering of the intensity image may give wrong results.
- There are several options to construct color High Pass Filters for edge detection.

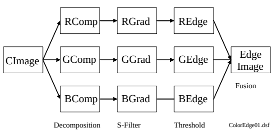
1. Edge filter applied separately on each color channel and superimposing all the edges.
2. Application of gradient filter (Sobel) on each color channel, calculation of the norm of a color gradient and generation of an edge image out of the color gradient
3. Edge filter using color vectors (Di Zenzo's algorithm)

12.b Color Image Processing, 12.11 Edge Detection

1st Option: Composition of binary edge images

- Decomposition of a color image into 3 color components (RGB).
- Application of gradient filter (Sobel) to calculate 3 binary edge images for RGB.
- Independent thresholds for each color channel.
  - Empirically defined
  - Global adaptive threshold (using fraction of global maximum gradient, see Ch. 3.2)
- Fusion of the 3 edge images to one edge image by OR-operation

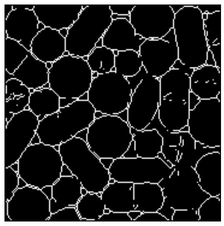
12.b Color Image Processing, 12.11 Edge Detection



- Fusion is done by OR-ing the edges in the R-, G- and B-channel

12.b Color Image Processing, 12.11 Edge Detection

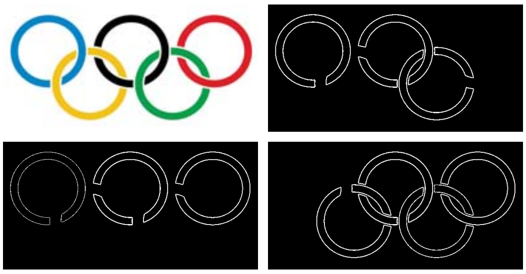
Colored Pills
© http://polpix.sueddeutsche.com/../medizin-wahnsinn.jpg



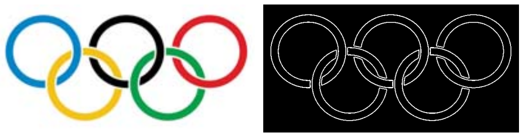
Fusion of Sobel filtered r,g,b components, edges skelettonized Image

12.b Color Image Processing, 12.11 Edge Detection

Olympic Rings: Sobelfilter red (upper right), green (lower left), blue (lower right). What happened?



12.b Color Image Processing, 12.11 Edge Detection



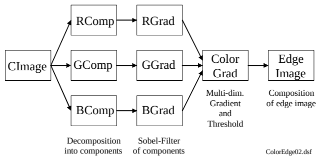
Fusion of Sobel filtered r, g, b components

12.b Color Image Processing, 12.11 Edge Detection

2nd Option: Analysis of the norm of a color gradient

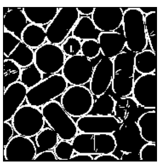
- Decomposition of color image into 3 color components (RGB).
- Application of gradient filter (Sobel) to calculate 3 color gradients.
- Calculation of the norm of a color gradient
- Generation of an edge image out of the color gradient (Lambert and Carron, 1999)

12.b Color Image Processing, 12.11 Edge Detection



- Calculation of the magnitude of the multi-dimensional gradient:
- ColorGrad = $\sqrt{(RGrad)^2 + (GGrad)^2 + (BGrad)^2}$
- Thresholding the value of ColorGrad

12.b Color Image Processing, 12.11 Edge Detection



Analysis of the norm of a color gradient (ColorGrad)

12.b Color Image Processing, 12.11 Edge Detection

3rd Option:
Calculation of the maximum change of the color vector c(x,y) in each pixel.

Silvano Di Zenzo: *A Note on the Gradient of a Multi-Image* in COMPUTER VISION, GRAPHICS AND IMAGE PROCESSING 33, p. 116-125 (1986)

Gonzalez, Woods: Digital Image Processing, Chapter 6.7.3, Eq. 6.7-3 to 6.7-9

Program:
www.mathworks.com/mathlabcentral/fileexchange/32623-gradient-image-of-multi-spectral-image/content/colgrad.m

12.b Color Image Processing, 12.11 Edge Detection

$$\vec{u}(x,y) = \frac{\partial}{\partial x} rImg(x,y)\,\vec{e}_r + \frac{\partial}{\partial x} gImg(x,y)\,\vec{e}_g + \frac{\partial}{\partial x} bImg(x,y)\,\vec{e}_b$$

$$\vec{v}(x,y) = \frac{\partial}{\partial y} rImg(x,y)\,\vec{e}_r + \frac{\partial}{\partial y} gImg(x,y)\,\vec{e}_g + \frac{\partial}{\partial y} bImg(x,y)\,\vec{e}_b$$

- Calculate the above derivatives e.g. by {- rImg(x-1,y) + rImg(x+1,y)} or by use of the Sobel masks.
- Calculate the following 2nd derivatives as scalar products.

$$g_{xx} = u^T * u = \left|\frac{\partial rImg(x,y)}{\partial x}\right|^2 + \left|\frac{\partial gImg(x,y)}{\partial x}\right|^2 + \left|\frac{\partial bImg(x,y)}{\partial x}\right|^2$$

$$g_{yy} = v^T * v = \left|\frac{\partial rImg(x,y)}{\partial y}\right|^2 + \left|\frac{\partial gImg(x,y)}{\partial y}\right|^2 + \left|\frac{\partial bImg(x,y)}{\partial y}\right|^2$$

$$g_{xy} = u^T * v = \frac{\partial rImg(x,y)}{\partial x} * \frac{\partial rImg(x,y)}{\partial y} + \frac{\partial gImg(x,y)}{\partial x} * \frac{\partial gImg(x,y)}{\partial y} + \frac{\partial bImg(x,y)}{\partial x} * \frac{\partial bImg(x,y)}{\partial y}$$

12.b Color Image Processing, 12.11 Edge Detection

Di Zenzo showed that the direction θ of maximum change of vector c(x,y) is given by

$$\theta = \frac{1}{2} * \arctan\left[\frac{2g_{xy}}{g_{xx} - g_{yy}}\right]$$

and that the magnitude of change $\Delta c(x,y) = c_2(x,y) - c_1(x,y)$ of two color Vectors in direction of θ is given by :

$$\Delta c(x,y) = +\sqrt{\frac{1}{2}\left[(g_{xx} + g_{yy}) + (g_{xx} - g_{yy}) * \cos(2\theta) + 2g_{xy} * \sin(2\theta)\right]}$$

12.b Color Image Processing, 12.11 Edge Detection

The solution θ_0 of $\theta = \frac{1}{2} * \arctan\left[\frac{2g_{xy}}{g_{xx} - g_{yy}}\right]$ is not unique.

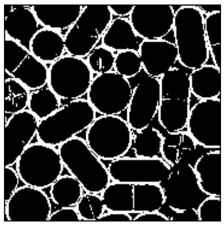
Because of $\tan(\theta) = \tan(\theta \pm \pi)$ and because of

$$\Delta c(x,y) = +\sqrt{\frac{1}{2}\left[(g_{xx} + g_{yy}) + (g_{xx} - g_{yy}) * \cos(2\theta) + 2g_{xy} * \sin(2\theta)\right]}$$

is dependent on $\cos(2\theta)$ and $\sin(2\theta)$, also $\theta_0 = \theta_0 \pm \pi/2$ are valid solutions.
Because of $\Delta c(x,y) = f(\theta) = f(\theta+\pi)$, $\Delta c(x,y)$ has to be computed for the interval $0 \le \theta < \pi$.
The two solutions θ_0 and $\theta_0 \pm \pi/2$ represent a pair of orthogonal directions of $\Delta c(x,y)$, one of which represents the maximum and the other the minimum of $\Delta c(x,y)$.

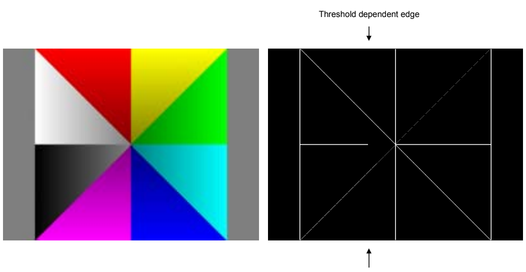
12.b Color Image Processing, 12.11 Edge Detection

Colored Pills



Edge detection using change of color vetors (Di Zenzo)

12.b Color Image Processing, 12.11 Edge Detection



12.b Color Image Processing, 12.11 Edge Detection

Di Zenzo's algorithm applied on natural image

12.b Color Image Processing, 12.12 Image Segmentation

- Image segmentation: Decomposition of an image into a number of K disjoint and non-empty regions R_k, k = 1 … K.
- Regions R_k are defined by some characteristics of the pixels (grey value, <u>color</u>, texture etc.).
- Completeness: $\sum_{k=1}^{K} R_k = N * M$ i.e. image size
- Non-Overlapping: $R_i \cap R_j = \emptyset \; \forall i \neq j$
- Connectivity: all pixels of a region R_k are connected by a defined connection, e.g. 4-neighborhood or 8-neighborhood.

12.b Color Image Processing, 12.12 Image Segmentation

- Uniformity criterion γ:
  - all pixels belonging to R_k have similar properties.

    $\gamma(R_k) = \text{TRUE}$
  - $\forall i \neq j$: the uniformity criterion is not satisfied for pixel $p_i \in R_i$ and $p_j \in R_j$.

    $\gamma(R_i, R_j) = \text{FALSE}$
- Boundaries of the regions should be smooth and not ragged.
- All regions R_k is assigned a unique label L_k.

12.b Color Image Processing, 12.12 Image Segmentation

- Segmentation strategies:
  - Bottom up: initially the whole image is treated a single region. Successively the region is divided into sub-regions by sharpening the uniformity criterion regarding the completeness, non-overlapping and connectivity up to a defined number of regions K.
  - Top down: initially each pixel of the image is treated to be its own region. Successively the regions are merged by widening the uniformity criterion regarding the completeness, non-overlapping and connectivity down to a defined number of regions K.

12.b Color Image Processing, 12.12 Image Segmentation

- Solution of segmentation is not unique.
- Solution is dependent on
  - Connectivity
  - Scan direction
  - Uniformity criterion
  - Segmentation strategy
  - Used color space

Suggested papers:

Color Spaces and Image Segmentation by Laurent Busin, Nicolas Vandenbroucke, Ludovic Macaire, Univ. des Sciences et Technologies, Lille (F), 2007

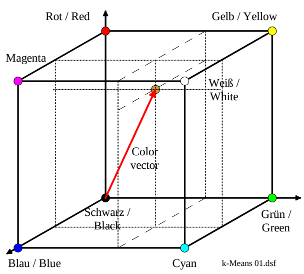
Color Image Edge Detection and Segmentation by S. B. Wesolkowski, Master-Thesis, Univ. of Waterloo, Ontario, Cda., Chapter 5

12.13 Segmentation by Unsupervised Learning, k-Means Clustering

- Application on color images
- RGB Color space wit corners (Red, Green, Blue, Yellow, Magenta, Cyan, White, Black
- Number of clusters : 8
- Position of clusters :

Red	255-0-0	Yellow	255-255-0
Green	0-255-0	Magenta	255-0-255
Blue	0-0-255	Cyan	0-255-255
White	255-255-255	Black	0-0-0

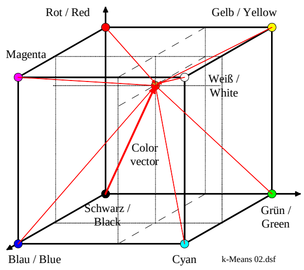
12.13 Segmentation by Unsupervised Learning, k-Means Clustering



The situation:

RGB color space defining eight colors : three primary colors R, G, B, three composed colores Y, M, C and two achromatic colors K (Key) and W. Color vector of a pixel p from origin (K) to individual color point Cp.

12.13 Segmentation by Unsupervised Learning, k-Means Clustering



Define the eight colors as the centers of eight clusters.
Calculate the differences of color vector Cp to the cluster centers.
Check for the minimum difference.
Calculate the average of all color vectors with minimum to the same center.
Update the average as the new cluster center.
Iterate the process until no change occurs.

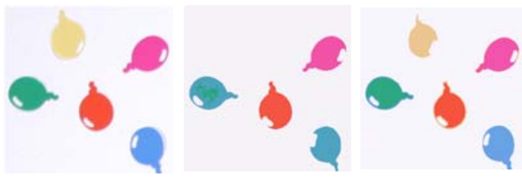
12.13 Segmentation by Unsupervised Learning, k-Means Clustering

R = 255, 0, 0
G = 0, 255,0
B = 0, 0, 255
Y = 255, 255,0
M = 255, 0, 255
C = 0, 255, 255
W = 255, 255, 255
K = 0, 0, 0

$$CM = \begin{array}{cccccccc} r & g & b & y & c & m & w & k \end{array}$$
$$\begin{pmatrix} 179 & 89 & 79 & 206 & 140 & 105 & 169 & 52 \\ 68 & 150 & 48 & 172 & 96 & 153 & 189 & 56 \\ 33 & 49 & 154 & 26 & 182 & 144 & 169 & 55 \end{pmatrix}$$

12.13 Segmentation by Unsupervised Learning, k-Means Clustering

12.13 Segmentation by Unsupervised Learning, k-Means Clustering



Original image   1st step   2nd step

Automatic color segmentation using corners of RGB color space.
After 1st step there still exist some color artifacts.
After 2nd step the artifacts disappear.

Review of Chapter 12B

- Low Pass Filter
  - Mean (and similar) filter on color channels
  - Median filter on color channels
  - Median filter on color vectors
  - Median filter on color differences
- High Pass Filter - Edge detection in color images
  - Edge filter on color channels
  - Edge filter using color gradients
  - Edge filter using color vectors (Di Zenzo's algorithm)
- Image segmentation