Color Histogram Normalization using Matlab and Applications in CBIR

László Csink, Szabolcs Sergyán
Budapest Tech

SSIP'05, Szeged

Outline

- Introduction
- Demonstration of the algorithm
- Mathematical background
- Computational background: Matlab
- Presentation of the algorithm
- Evaluation of the test
- Conclusion

Introduction (1)

- Retrieval in large image databases
  - Textual key based
    - Key generation is subjective and manual
    - Retrieval is errorless
  - Content Based Image Retrieval (CBIR)
    - Key generation is automatic (possibly time consuming)
    - Noise is unavoidable

Introduction (2)

- If a new key is introduced, the whole database has to be reindexed.
- In textual key based retrieval the reindexing requires a lot of human work, but in CBIR case it requires only a lot of computing.

Introduction (3)

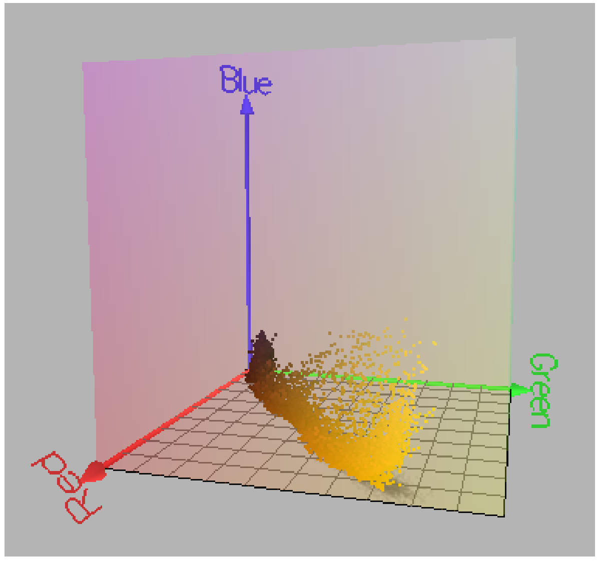
- CBIR
  - Low level
    - Color, shape, texture
  - High level
    - Image interpretation
    - Image understanding

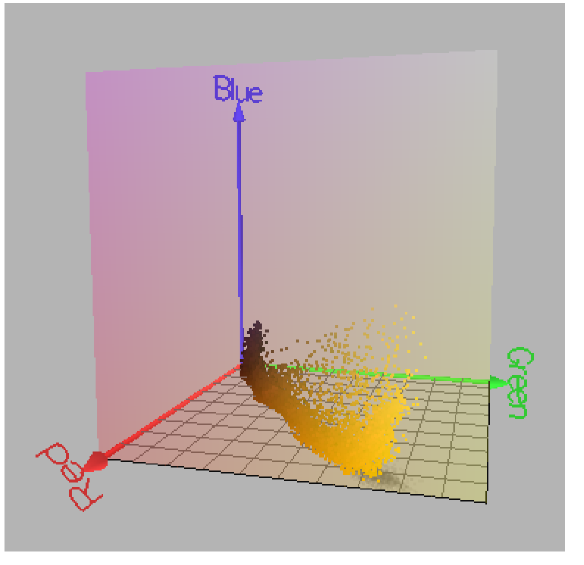
Introduction (4)

- Typical task of low level CBIR
  - Search for a given object using similarity distance based on content keys
- One way of defining similarity distance is to use color histograms – we concentrate on this approach in the present talk

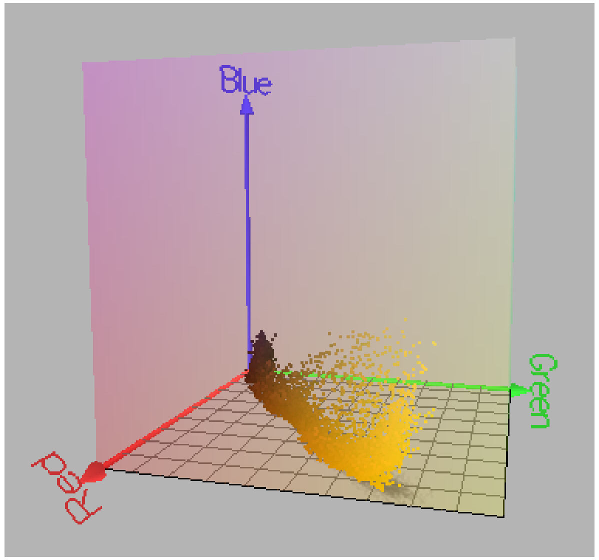
Demonstration of the algorithm

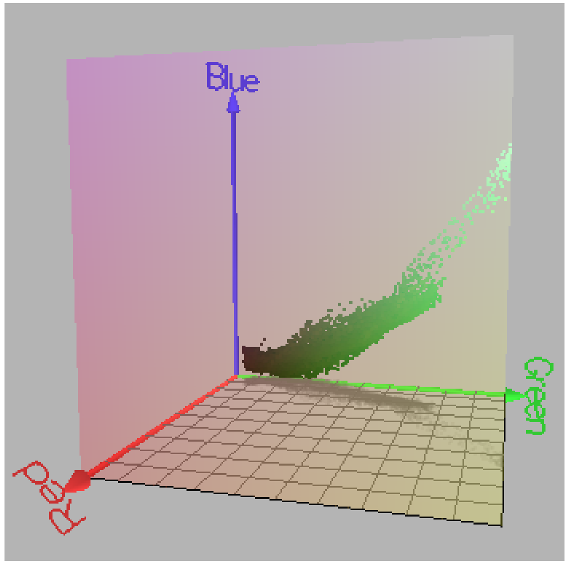
Image versions and their histograms





Two images and their histograms

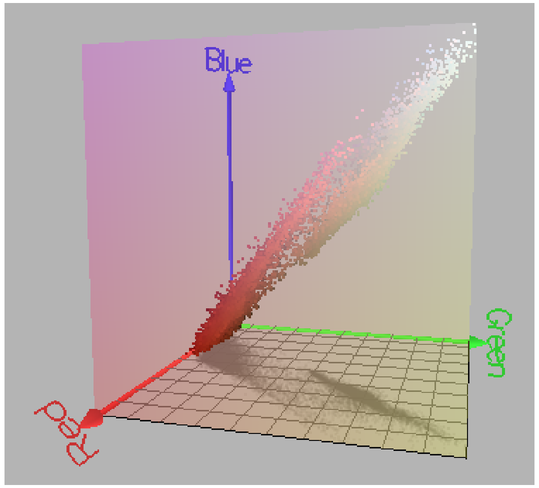


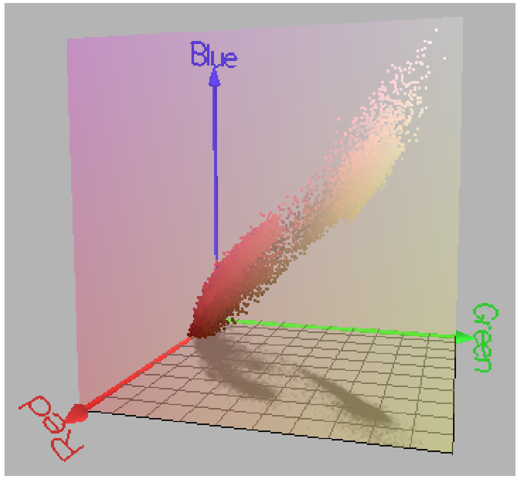


Similarity distance between two image histograms

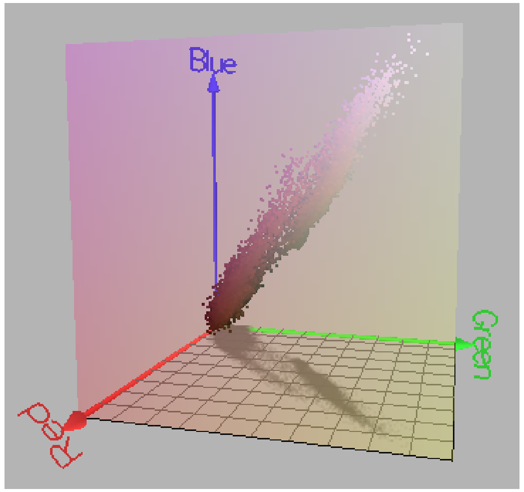
$$d_{\text{Euclid}}(hx, hy) = \sqrt{\sum_{r=1}^{4}\sum_{g=1}^{4}\sum_{b=1}^{4}\left(hx_{rgb} - hy_{rgb}\right)^2}$$

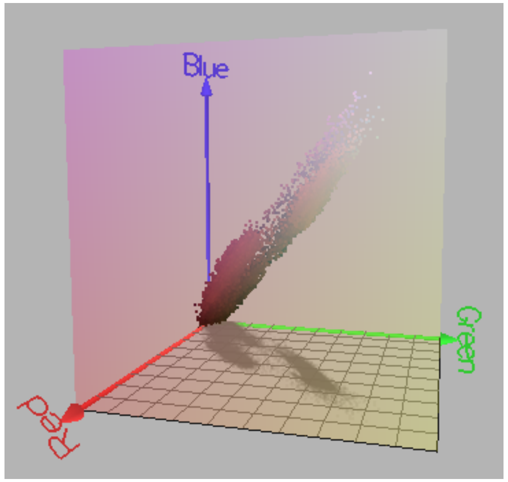
Different illuminations



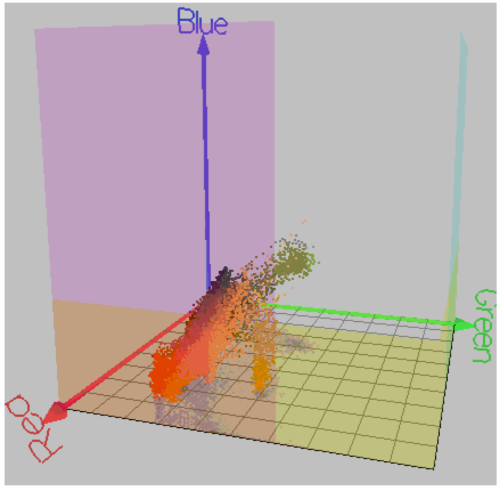


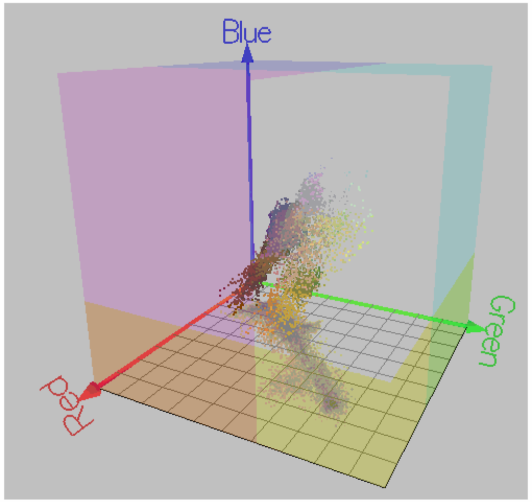
Normalized versions



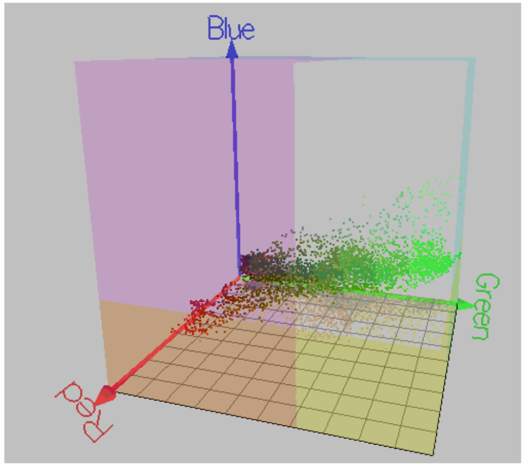


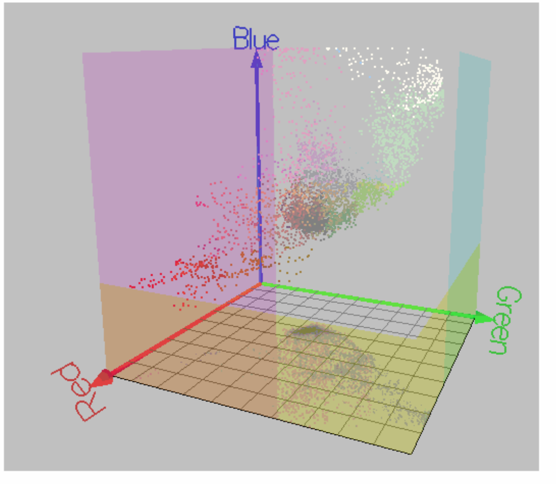
Normalization may change image outlook





Normalization may change image outlook





Mathematical background

Color cluster analysis

- $f = \lfloor f_{ij} \rfloor \quad i = 1, \ldots, M; \; j = 1, \ldots, N$

1. Compute the cluster center of all pixels *f* by *m*=*E*[*f*]. *m* is a vector which points to the center of gravity.
2. $C = E\left[(f - m)(f - m)^T\right]$

   The eigenvalues (λ_1, λ_2, λ_3) and eigenvectors of *C* are computed directly.
3. Denote the eigenvector belonging to the largest eigenvalue by $v=(a,b,c)^T$.

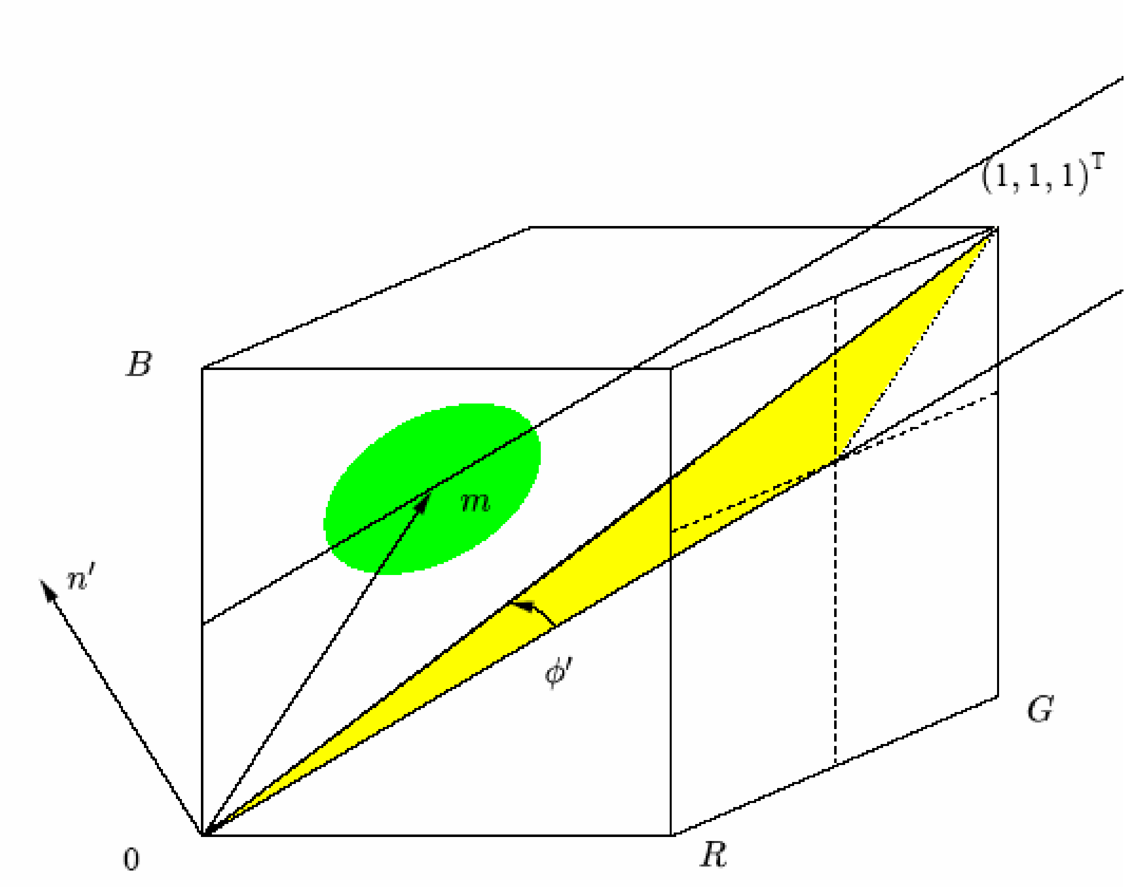
Rodrigues formula

Rotating *v* along *n* (*n* is a unit vector) by θ: *Rv*, *where*

$$R = I + U(n)\sin\theta + U(n)^2(1-\cos\theta)$$

$$U(n) = \begin{pmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{pmatrix}$$

Color rotation in RGB-space



Color rotation in RGB-space

$$n' = (a,b,c)^{T} \times \frac{1}{\sqrt{3}}(1,1,1)^{T}$$

$$\cos q' = (a,b,c)^{T} \cdot \frac{1}{\sqrt{3}}(1,1,1)^{T}$$

Color rotation in RGB-space

4. Use the Rodrigues formula in order to rotate with θ' around n'.
5. Shift the image along the main axis of the RGB-cube by $(128,128,128)^T$.
6. Clip the overflows above 255 and the underflows under 0.

Computational background

Fundamentals of MATLAB

Presentation of the algorithm

MATLAB code

```
function color_normalization(FILENAME, OUTPUT);

inp_image = imread(FILENAME); % read input image
[m,n,d]=size(inp_image);     % get size of input image
f=double(inp_image);     % double needed for computations
M=zeros(m*n,3);
z=1;
mv=mean(mean(f));     % a vector containing the mean r,g and b value
v1=[mv(1),mv(2),mv(3)];     % means in red, green and blue
```

MATLAB code

```
for i=1:m
   for j=1:n
      v=[f(i,j,1),f(i,j,2),f(i,j,3)];  % image pixel at i,j
      M(z,:) = v - v1;    % image normed to mean zero
      z = z + 1;
   end
end
C = cov(M);  % covariance computed using Matlab cov function
```

MATLAB code

```
%find eigenvalues and eigenvectors of C.
[V,D]=eig(C); % computes the eigenvectors(V) and eigenvalues
              (diagonal elements of D) of the color cluster C
%get the max. eigenvalue meig and the corresponding eigenvector ev0.
meig = max(max(D)); % computes the maximum eigenvalue of C.
                    Could also be norm(C)
if(meig==D(1,1)), ev0=V(:,1);, end
if(meig==D(2,2)), ev0=V(:,2);, end
if(meig==D(3,3)), ev0=V(:,3);, end
% selects the eigenvector belonging to the greatest eigenvalue
```

MATLAB code

```
Idmat  =eye(3);     % identity matrix of dimension 3
wbaxis=[1;1;1]/sqrt(3); % unit vector pointing from origin along the main
                          diagonal
nvec   = cross(ev0,wbaxis);      % rotation axis , cross(A,B)=A×B
cosphi = dot(ev0,wbaxis)      % dot product, i.e. sum((ev0.*wbaxis))
sinphi = norm(nvec); %  sinphi is the length of the cross product of two
                          unit vectors
%normalized rotation axis.
nvec   = nvec/sinphi;    % normalize  nvec
```

MATLAB code

```
if(cosphi>0.99)
   f=uint8(f);
   imwrite(f,OUTPUT);  %in this case we dont normalize, output is
                         input etc.
else                   % we normalize
   n3 = nvec(3); n2 = nvec(2); n1 = nvec(1);
         % remember: this is a unit vector along the rotation axis
    U = [[ 0  -n3  n2]; [ n3   0  -n1]; [ -n2 n1  0]];
   U2 = U*U;
   Rphi = Idmat + (U*sinphi) + (U2*(1-cosphi));
```

MATLAB code

```
n0   = [0 0 0]';
n255 = [255 255 255]';
for i=1:m
  for j=1:n
    s(1)= f(i,j,1)-mv(1); % compute vector s of normalized image at i,j
    s(2)= f(i,j,2)-mv(2);
    s(3)= f(i,j,3)-mv(3);
    t = Rphi*s' ; % s transposed, as s is row vector, then rotated
    tt = floor(t + [128 128 128]'); % shift to middle of cube and
                                      make it integer
```

MATLAB code

```
      tt = max(tt,n0);                 % handling underflow
      tt = min(tt,n255);               % handling overflow

      g(i,j,:)=tt;
    end
  end

  g=uint8(g);
  imwrite(g,OUTPUT);
end      % end of normalization
```

Evaluation of the test

Test databases

- 5 objects
- Alternative color cubes
- 3 illuminations
- 3 background

- 90 images

Some test results (without normalization)

Some test results (with normalization)

n_DSC01151.JPG

n_DSC01150.JPG

52.6808

n_DSC01092.JPG

78.7583

n_DSC01103.JPG

107.3023

n_DSC01102.JPG

114.1666

n_DSC01161.JPG

116.0299

Conclusions

References

- Paulus, D., Csink, L., and Niemann, H.: Color Cluster Rotation. In: Proc. of International Conference on Image Processing, 1998, pp. 161-165
- Sergyán, Sz.: Special Distances of Image Color Histograms. In: Proc. of 5th Joint Conference on Mathematics and Computer Science, Debrecen, Hungary, June 9-12, 2004, pp. 92

Thank you for your attention!

Csink.laszlo@nik.bmf.hu Sergyan.szabolcs@nik.bmf.hu