REFERENCES

1. Fairchild, M.D., and Johnson, G.M. Image appearance modeling. SPIE/IS&T Electronic Imaging Conference. 5007 (2003):149-160.

2. Fairchild, M.D., and Johnson, G.M. Meet iCAM: A next-generation color appearance model, Proceedings of IS&T/SID 10th Color Imaging Conference, pp. 33-38. Scottsdale, 2002.

3. Johnson, G.M., and Fairchild, M.D. Rendering HDR images, Proceedings of IS&T/SID 11th Color Imaging Conference, pp. 36-41.Scottsdale, 2003.

4. Moroney, N., and Tastl, I. A Comparison of Retinex and iCAM for Scene Rendering. Journal of Electronic Imaging, 13 (2004):139-145.

5. Berns,R.S. Principles of color Technology. 3rd ed. New York: John Wiley & Sons, 2000; pp.13-17.

6. Wright, W. D. A re-determination of the trichromatic coefficients of the spectral colours, Transactions of the Optical Society , 30 (1928): 141-164.

7. Guild, J. The colorimetric properties of the spectrum, Philosophical Transactions of the Royal Society of London, 230 (1931): 149-187.

8. CIE 1931 color space [Online]. Available from: http://en.wikipedia.org/wiki/CIE_1931_color_space. [2006.February 27]

9. Berns, R.S. Principles of Color Technology. 3rd ed. New York: John Wiley & Sons, 2000; p.6.

10. Berns, R.S. Principles of Color Technology. 3rd ed. New York: John Wiley & Sons, 2000; p.7.

11. Judd, D.B., and Wyszecki, G. Color in business, Science and Industry. 3rd ed. New York: John Wiley & Sons, 1975; p.102.

12. Billmeyer,F.W., and Saltzman, M. Principles of color Technology. 2nd ed. New York: John Wiley & Sons, 1981; p.44.

13. Billmeyer,F.W., and Saltzman, M. Principles of color Technology. 2nd ed. New York: John Wiley & Sons, 1981; p.45.

14. Berns, R.S. Principles of Color Technology. 3rd ed. New York: John Wiley & Sons, 2000; p.165.

15. sRGB color space [Online]. Available from: http://en.wikipedia.org/wiki/SRGB_color_space. [2006.February 27]

16. Image enhancement [Online]. Available from: http://www.webopedia.com/TERM/I/imageenhancement.html. [2006.March 3]

17. Tutorials: Image Histograms Part1 [Online]. Available from: http://cambridgeincolour.com/tutorials/histograms1.htm. [2006. March 3]

18. Dynamic range [Online]. Available from: http://www.library.cornell.edu/presenvation/tutorial/intro/into-05.html. [2006. March 3]

19. Tutorials: Bit depth [Online]. Available from: http://cambridgeincolour.com/tutorials/bit-depth.htm. [2006.March 3]

20. Image filtering [Online]. Available from: http://www.schorsch.com/kbase/glossary/image_filtering.html. [2006.March 1]

21. Gaussian smoothing [Online]. Available from: http://www.cee.hw.ac.uk/hipr/html/gsmooth.html. [2006.March 15]

22. Convolution [Online]. Available from: http://www.cee.hw.ac.uk/hipr/html/convolue.html. [2006.March 15]

23. Convolution filters [Online]. Available from: http://www.mini.pw.edu.pl/~kotowski/Grafika/Images/Index.html. [2006.March15]

24. Fairchild, M.D., and Johnson, G.M. The iCAM framework for image appearance, image differences, and image quality [Online], Available from: http://www.cis.rit.edu/mcsl/iCAM/pub/JEIiCAM.pdf [2004].

25. Moroney, N., et al. The CIECAM02 color appearance model, IS&T/SID 10th Color Imaging Conference, pp. 23-27. Scottsdale, 2002.

26. High dynamic range [Online]. Available from: http://www.debevec.org/Research/HDR/. [2004]

27. Xiao, F., et al. High dynamic range imaging of natural scenes, IS&T/SID 10th Color Imaging Conference, pp. 337-342. Scottsdale, 2002.

28. Image quality assessment: From error visibility to structural similarity [Online]. Available from: http://www.cns.nyu.edu/~lcv/ssim/. [2006. February 17]

29. Sakamoto, K., and Urabe, H. Standard High Precision Pictures: SHIPP, The fifth Color Imaging Conference: Color Science, Systems and Applications. pp. 240-244.1997

APPENDICES

APPENDIX A

rgb2xyz

```
function [xyz]=rgb2xyz(infile);
% convert sRGB toXYZ
%rgb2xyz.m
%arguments
%infile-filename of input image
%25/8/2005

A=imread(infile);
img=im2double(A);
[m,n,c]=size(A);

%convert sRGB to RGB
for i=1:m
   for j=1:n
      for k=1:c
         if img(i,j,k)<=0.03928;
            xyz_img(i,j,k)=(img(i,j,k)./12.92);
         else
            xyz_img(i,j,k)=((img(i,j,k)+0.055)./1.055).^2.4;
         end
      end
   end
end

r=xyz_img(:,:,1);
g=xyz_img(:,:,2);
b=xyz_img(:,:,3);

%convert RGB to XYZ
x=(0.4124.*r)+(0.3576.*g)+(0.1805.*b);
```

```
y=(0.2126.*r)+(0.7152.*g)+(0.0722.*b);
z=(0.0193.*r)+(0.1192.*g)+(0.9505.*b);

xyz(:,:,1)=x.*100;
xyz(:,:,2)=y.*100;
xyz(:,:,3)=z.*100;
```

iCAM

```
function [xyz_inv] = iCAM(infile);
%This is a implementation of iCAM algorithm for image enhancement
% iCAM.m
%arguments:
%infile-filename of input image
%input 1. The size of the first and the second filter, utilized the size of filter
              equal to the size of image.
          2. The sigma ( σ ) value
              (1) The first filter use the sigma values about 50-60
              (2) The second filter use the sigma values about 200
%20/01/2006

ima=infile;
img= nozero(ima);                                                % load nozero.m
[m,n,c]=size(img);

% filter 1
xyz_wh = LP1(img,[m n],50);                                      % load LP1.m
Mcat=[0.7328,0.4296,-0.1624;-0.7036,1.6975,0.0061;0.0030,0.0136,0.9834];
Mcati = inv(Mcat);
RGB_wh = ccs(xyz_wh,Mcat);                                       % load ccs.m

% filter 2 & find FL and D factorr
ylow = LP2(img,[m n],200);                                       % load LP2.m
D= 1*(1-(1/3.6)*exp((-1*ylow-42)./92));
FL = (0.2.*((1./(5.*ylow+1)).^4).*(5.*ylow))+(0.1.*(1-
(1./(5.*ylow+1)).^4).*(5.*ylow).^(1/3));

%convert to RGB correspond
RGB_img = ccs(img,Mcat);
```

```
xyz_d65 = [95.5,100.0,108.88];
RGB_d65 = Mcat*xyz_d65';

Rc = (((D.*RGB_d65(1))./RGB_wh(:,:,1))+(1-D)).*RGB_img(:,:,1);
Gc = (((D.*RGB_d65(2))./RGB_wh(:,:,2))+(1-D)).*RGB_img(:,:,2);
Bc = (((D.*RGB_d65(3))./RGB_wh(:,:,3))+(1-D)).*RGB_img(:,:,3);

adapt_img = zeros(size(img));
adapt_img(:,:,1) = Rc;
adapt_img(:,:,2) = Gc;
adapt_img(:,:,3) = Bc;

xyz_ipt = ccs(adapt_img,Mcati);

%convert to IPT
xyz2lms = [0.4002,0.7077,-0.0807;-0.2280,1.1500,0.0612;0.0,0.0,0.9184];
lms2xyz = inv(xyz2lms);
lms = ccs(xyz_ipt,xyz2lms);

belowz = zeros(size(img));
belowz(:,:,1) = (lms(:,:,1)<= 0);
belowz(:,:,2) = (lms(:,:,2)<= 0);
belowz(:,:,3) = (lms(:,:,3)<= 0);

belowz = logical (belowz);

lms_nl(:,:,1) = abs(lms(:,:,1)).^(FL*.43);
lms_nl(:,:,2) = abs(lms(:,:,2)).^(FL*.43);
lms_nl(:,:,3) = abs(lms(:,:,3)).^(FL*.43);

lms_nl(belowz) = -1*lms_nl(belowz);

lms2ipt = [0.4000,0.4000,0.2000;4.4550,-4.8510,0.3960;0.8056,0.3572,-1.1628];
ipt2lms = inv(lms2ipt);
```

```
ipt_img = ccs(lms_nl,lms2ipt);
%InvertModel

%invert IPT to xyz
lms_inv = ccs(ipt_img,ipt2lms);
lmsc_inv(:,:,1) = abs(lms_inv(:,:,1)).^(1./(FL.*0.43));
lmsc_inv(:,:,2) = abs(lms_inv(:,:,2)).^(1./(FL.*0.43));
lmsc_inv(:,:,3) = abs(lms_inv(:,:,3)).^(1./(FL.*0.43));

xyz_ipt_inv = ccs(lmsc_inv,lms2xyz);

%invert to RGB adaptation
M = [0.8562,0.3372,-0.1934;-0.8360,1.8327,0.0033;0.0357,-0.0469,1.0112];
Mi = inv(M);
RGB_adapt_inv = ccs(xyz_ipt_inv,M);

xyz_whitepoint = [100.0,100.0,100.0];
RGB_whitepoint = M*xyz_whitepoint';
RGB_d65n=M*xyz_d65';

Rc_inv = (((D.*RGB_whitepoint(1))./RGB_d65n(1))+(1-D)).*RGB_adapt_inv(:,:,1);
Gc_inv = (((D.*RGB_whitepoint(2))./RGB_d65n(2))+(1-D)).*RGB_adapt_inv(:,:,2);
Bc_inv = (((D.*RGB_whitepoint(3))./RGB_d65n(3))+(1-D)).*RGB_adapt_inv(:,:,3);

RGB_inv = zeros(size(img));
RGB_inv(:,:,1) = Rc_inv;
RGB_inv(:,:,2) = Gc_inv;
RGB_inv(:,:,3) = Bc_inv;

xyz_inv = ccs(RGB_inv,Mi);
```

xyz2rgb

```
function [rgb]=xyz2rgb(infile);

%convert XYZ to RGB
%xyz2rgb.m
%arguments:
%infile-filename of input image
%25/8/2005

xyz=(infile);
[m,n,c]=size(xyz);
x=xyz(:,:,1)./100;
y=xyz(:,:,2)./100;
z=xyz(:,:,3)./100;

r=(3.2406.*x)+(-1.5374.*y)+(-0.4986.*z);
g=(-0.9692.*x)+(1.8760.*y)+(0.0416.*z);
b=(0.0556.*x)+(-0.2040.*y)+(1.0570.*z);

o(:,:,1)=(r);
o(:,:,2)=(g);
o(:,:,3)=(b);

%convert RGB to sRGB
for i=1:m
   for j=1:n
      for k=1:c
         if o(i,j,k)<=0.00304;
            rgb(i,j,k)=o(i,j,k).*12.92;
         else
            rgb(i,j,k)=(1.055.*(o(i,j,k).^(1/2.4)))-0.055;
         end
```

```
        end
    end
end

eliminate >1 and <0
for i=1:m
    for j=1:n
        for k=1:c
            if rgb(i,j,k)>1;
                rgb(i,j,k)=1;
            elseif rgb(i,j,k)<0;
                 rgb(i,j,k)=0;
            end
        end
    end
end
```

Change Color Space (ccs)

```
function out = ccs(a,b)

%change color space
%ccs.m
%The input image consists of three input image, (a)
%colormatrix,(b)
%The out image has the same format
%The 3x3 color matrix converts column vectors in the input image
%representation in to column vectors in the output representation.
%11/02/2005

insize= size (a);
a = reshape (a,insize(1)*insize(2),insize(3));
at = a';
out = b*at;
outt = out';
a = reshape (a,insize(1),insize(2),insize(3));
out = reshape (outt,insize(1),insize(2),insize(3));
```

Nozero

```
function   new = nozero(infile);

%nozero.m
%arguments:
%infile-filename of input image
%3/01/2006

in= (infile);
%in=im2double(A);
[m,n,c]=size(in);

for i=1:m
   for j=1:n
      for k=1:c
         if in(i,j,k)<=0;
            out(i,j,k)=0.0001;
         else
            out(i,j,k)=(in(i,j,k));
         end
      end
   end
end

new=zeros(size(in));
new(:,:,1)=out(:,:,1);
new(:,:,2)=out(:,:,2);
new(:,:,3)=out(:,:,3);
```

LP1

```
function out = LP1(a,h1,z)

%the first low-pass filter
%LP1.m
%arguments:
%a-input image file
%h1-the size filter
%z-the sigma value
%30/08/2005

filter = fspecial('gaussian',h1,z);

out = zeros(size(a));
out(:,:,1) = imfilter(a(:,:,1),filter);
out(:,:,2) = imfilter(a(:,:,2),filter);
out(:,:,3) = imfilter(a(:,:,3),filter);
```

LP2

```
 function out = LP2(a,h,z)

%The second low-pass filter
%LP2.m
%arguments:
%a-input image file
%h-the size of filter
%z-the sigma value
%30/08/2005

filter = fspecial('gaussian',h,z);
out = imfilter(a(:,:,2),filter);
```

imgQ

```
function [ssim] = imgQ(img1, img2);

%This is function comparison between the reference image (high quality image)
% and the image enhancement
%imgQ.m
%arguments:
%img1-the reference image
%img2-the image enhancement
%18/02/2006

xyz1 = rgb2xyz(img1);
xyz2 = rgb2xyz(img2);

img1 = xyz1(:,:,2);
img2 = xyz2(:,:,2);

% default setting by ssim_index%
  window = fspecial('gaussian', 11, 1.5);
  K(1) = 0.01;
  K(2) = 0.03;
%   L = 255; % default setting
  L = 100;   % my setting

[ssim, map] = ssim_index(img1, img2, K, window, L);          % load ssim_index.m
```

ssim_index

```
function [mssim, ssim_map] = ssim_index(img1, img2, K, window, L)

%download from http://www.cns.nyu.edu/~lcv/ssim/.
%ssim_index.m
%18/02/2006
%========================================================================
%SSIM Index, Version 1.0
%Copyright(c) 2003 Zhou Wang
%All Rights Reserved.
%
%The author is with Howard Hughes Medical Institute, and Laboratory
%for Computational Vision at Center for Neural Science and Courant
%Institute of Mathematical Sciences, New York University.
%
%----------------------------------------------------------------------
%Permission to use, copy, or modify this software and its documentation
%for educational and research purposes only and without fee is hereby
%granted, provided that this copyright notice and the original authors'
%names appear on all copies and supporting documentation. This program
%shall not be used, rewritten, or adapted as the basis of a commercial
%software or hardware product without first obtaining permission of the
%authors. The authors make no representations about the suitability of
%this software for any purpose. It is provided "as is" without express
%or implied warranty.
%----------------------------------------------------------------------
%
%This is an implementation of the algorithm for calculating the
%Structural SIMilarity (SSIM) index between two images. Please refer
%to the following paper:
%
%Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image
```

```
%quality assessment: From error visibility to structural similarity"
%IEEE Transactios on Image Processing, vol. 13, no. 4, pp.600-612,
%Apr. 2004.
%
%Kindly report any suggestions or corrections to zhouwang@ieee.org
%
%----------------------------------------------------------------------
%
%Input : (1) img1: the first image being compared
%       (2) img2: the second image being compared
%       (3) K: constants in the SSIM index formula (see the above
%          reference). defualt value: K = [0.01 0.03]
%       (4) window: local window for statistics (see the above
%          reference). default widnow is Gaussian given by
%          window = fspecial('gaussian', 11, 1.5);
%       (5) L: dynamic range of the images. default: L = 255
%
%Output: (1) mssim: the mean SSIM index value between 2 images.
%          If one of the images being compared is regarded as
%          perfect quality, then mssim can be considered as the
%          quality measure of the other image.
%          If img1 = img2, then mssim = 1.
%       (2) ssim_map: the SSIM index map of the test image. The map
%          has a smaller size than the input images. The actual size:
%          size(img1) - size(window) + 1.
%
%Default Usage:
%   Given 2 test images img1 and img2, whose dynamic range is 0-255
%
%   [mssim ssim_map] = ssim_index(img1, img2);
%
%Advanced Usage:
%   User defined parameters. For example
%
```

```
%  K = [0.05 0.05];
%  window = ones(8);
%  L = 100;
%  [mssim ssim_map] = ssim_index(img1, img2, K, window, L);
%
%See the results:
%
%  mssim                        %Gives the mssim value
%  imshow(max(0, ssim_map).^4)  %Shows the SSIM index map
%
%========================================================================

if (nargin < 2 | nargin > 5)
  ssim_index = -Inf;
  ssim_map = -Inf;
  return;
end

if (size(img1) ~= size(img2))
  ssim_index = -Inf;
  ssim_map = -Inf;
  return;
end

[M N] = size(img1);

if (nargin == 2)
  if ((M < 11) | (N < 11))
  ssim_index = -Inf;
  ssim_map = -Inf;
    return
  end
  window = fspecial('gaussian', 11, 1.5);%
  K(1) = 0.01;      % default settings
  K(2) = 0.03;    %
  L = 255;                          %
end

if (nargin == 3)
  if ((M < 11) | (N < 11))
  ssim_index = -Inf;
  ssim_map = -Inf;
    return
  end
  window = fspecial('gaussian', 11, 1.5);
  L = 255;
  if (length(K) == 2)
    if (K(1) < 0 | K(2) < 0)
  ssim_index = -Inf;
  ssim_map = -Inf;
  return;
    end
  else
  ssim_index = -Inf;
  ssim_map = -Inf;
  return;
  end
end

if (nargin == 4)
  [H W] = size(window);
  if ((H*W) < 4 | (H > M) | (W > N))
  ssim_index = -Inf;
  ssim_map = -Inf;
    return
  end
  L = 255;
  if (length(K) == 2)
```

```
    if (K(1) < 0 | K(2) < 0)
  ssim_index = -Inf;
  ssim_map = -Inf;
  return;
    end
  else
  ssim_index = -Inf;
  ssim_map = -Inf;
  return;
  end
end

if (nargin == 5)
  [H W] = size(window);
  if ((H*W) < 4 | (H > M) | (W > N))
  ssim_index = -Inf;
  ssim_map = -Inf;
    return
  end
  if (length(K) == 2)
    if (K(1) < 0 | K(2) < 0)
  ssim_index = -Inf;
  ssim_map = -Inf;
  return;
    end
  else
  ssim_index = -Inf;
  ssim_map = -Inf;
  return;
  end
end

C1 = (K(1)*L)^2;
C2 = (K(2)*L)^2;
```

```
window = window/sum(sum(window));


img1 = double(img1);
img2 = double(img2);


mu1   = filter2(window, img1, 'valid');
mu2   = filter2(window, img2, 'valid');
mu1_sq = mu1.*mu1;
mu2_sq = mu2.*mu2;
mu1_mu2 = mu1.*mu2;
sigma1_sq = filter2(window, img1.*img1, 'valid') - mu1_sq;
sigma2_sq = filter2(window, img2.*img2, 'valid') - mu2_sq;
sigma12 = filter2(window, img1.*img2, 'valid') - mu1_mu2;


if (C1 > 0 & C2 > 0)
  ssim_map = ((2*mu1_mu2 + C1).*(2*sigma12 + C2))./((mu1_sq + mu2_sq +
C1).*(sigma1_sq + sigma2_sq + C2));
else
  numerator1 = 2*mu1_mu2 + C1;
  numerator2 = 2*sigma12 + C2;
denominator1 = mu1_sq + mu2_sq + C1;
  denominator2 = sigma1_sq + sigma2_sq + C2;
  ssim_map = ones(size(mu1));
  index = (denominator1.*denominator2 > 0);
  ssim_map(index) =
(numerator1(index).*numerator2(indcx))./(denominator1(index).*denominator2(index
));
  index = (denominator1 ~= 0) & (denominator2 == 0);
  ssim_map(index) = numerator1(index)./denominator1(index);
end


mssim = mean2(ssim_map);


return
```

APPENDIX B

Table B-1: The rank of Boat image in color filed.

	Rank				Total observer
	1	2	3	4	
Perfect image	5	5	4	6	20
Original image	6	7	6	1	20
iCAM	6	1	3	10	20
Imadjust	3	7	7	3	20

Table B-2: The rank of Boat image in detail filed.

	Rank				Total observer
	1	2	3	4	
Perfect image	11	8	0	1	20
Original image	4	4	2	10	20
iCAM	3	6	7	4	20
Imadjust	2	2	11	5	20

Table B-3: The rank of Japan image in color filed.

	Rank				Total observer
	1	2	3	4	
Perfect image	2	5	8	5	20
Original image	4	11	5	0	20
iCAM	0	1	4	15	20
Imadjust	14	3	3	0	20

Table B-4: The rank of Japan image in detail filed.

	Rank				Total observer
	1	2	3	4	
Perfect image	13	3	3	1	20
Original image	1	7	5	7	20
iCAM	2	9	2	7	20
Imadjust	4	1	10	5	20

Table B-5: The rank of Party image in color filed.

	Rank				
	1	2	3	4	Total observer
Perfect image	15	1	3	1	20
Original image	2	8	10	0	20
iCAM	0	0	1	19	20
Imadjust	3	9	8	0	20

Table B-6: The rank of Party image in detail filed.

	Rank				
	1	2	3	4	Total observer
Perfect image	13	5	2	0	20
Original image	4	4	10	2	20
iCAM	0	4	1	15	20
Imadjust	3	7	7	3	20

VITA

Miss Paradee Jetsamanpunt was born on August 26, 1981 in Bangkok, Thailand. She received a Bachelor's Degree of Science in Physics from the Faculty of Science, Srinakharinwirot University in 2002 and she has been a graduate student in Image Technology, Department of Photographic Science and Printing Technology Faculty of Science, Chulalongkorn University since 2003.