CS170A -- HW#0 -- Solution form -- Octave

Your name: Xuemin He

Your UID: 204468663

Please upload only this notebook to CCLE by the deadline.

Policy for late submission of solutions: We will use Paul Eggert's Late Policy: N days late $\Leftrightarrow$ 2^N points deducted} The number of days late is $N = 0$ for the first 24 hrs, $N = 1$ for the next 24 hrs, etc., and if you submit an assignment H hours late, $2^{\lfloor H/24 \rfloor}$ points are deducted.

Problem 1: Images (30 points)

(a) color-to-grayscale transformation:
include both your function `grayscale(A)` and its result where `A` the RGB Mandrill image.

In [3]:

```
function X = grayscale(A)
  gray = (A(:,:,1)+A(:,:,2)+A(:,:,3))*(1/3);
  X(:,:,1) = uint8(256*gray);
  X(:,:,2) = uint8(256*gray);
  X(:,:,3) = uint8(256*gray);
end
```

(b) image saturation and oversaturation:
include both your function `saturate(A,t)` and its result where `A` the RGB Mandrill image, when t=0.25.

In [4]:

```
function X = saturate(A,t)
  GrayA = double(grayscale(A))/255;
  X = uint8(256*(t*A + (1-t)*GrayA));
end
```

(c) image brightening:
include both your function `brighten(A,t)` and its result where `A` the RGB Mandrill image, when t=0.25.

In [5]:

```
function X = brighten(A,t)
  X = uint8(256* A * t);
end
```

Problem 2: Color Models (30 points)

(a) RGB to YCbCr(R,G,B):

Prove that the result of `RGB to YCbCr(R,G,B)` are all in the range 0 t 255, provided R, G, and B are.

In [ ]:

```
Y = 0.29900*R + 0.58700*G + 0.11400*B
minY = uint8(0.29900*0 + 0.58700*0 + 0.11400*0) = 0
maxY = uint8(0.29900*255 + 0.58700*255 + 0.11400*255) = 255

C_b = -0.16874*R - 0.33126*G + 0.50000*B + 128
The first two terms are negative; the last two terms are positive:
minC_b = uint8(-0.16874*255 - 0.33126*255 + 0.50000*0 + 128) = 0
maxC_b = uint8(-0.16874*0 - 0.33126*0 + 0.50000*255 + 128) = 255

C_r = 0.50000*R - 0.41869*G - 0.08131*B + 128
The first and last terms are positive; the middle two terms are negative:
minC_r = uint8(0.50000*0 - 0.41869*255 - 0.08131*255 + 128) = 0
maxC_r = uint8(0.50000*255 - 0.41869*0 - 0.08131*0 + 128) = 255
```

(b) RGB to CMY(R,G,B):

Develop a similar kind of function `RGB_to_CMY(R,G,B)` for converting RGB to CMY values.

In [ ]:

```
function [C,M,Y] = RGB_to_CMY (R,G,B)
  M1 = ones(480,500);
  C = M1-(double(R)/255);
  M = M1-(double(G)/255);
  Y = M1-(double(B)/255);
end
```

(c) CMY Mandrill:

Show your result of `RGB_to_CMY()` for the Mandrill image by rendering it in RGB. (Please display the image in RGB -- with Cyan as Red, Magenta as Green, Yellow as Blue.)

In [ ]:

```
[R,G,B] = image2rgb(Mandrill);
R = uint8(256*R);
G = uint8(256*G);
B = uint8(256*B);
[C,M,Y] = RGB_to_CMY(R,G,B);
CMY_Mandrill = rgb2image(C,M,Y);
imshow(CMY_Mandrill)
```

Problem 3: Rotations (20 points)

The file `rotations_and_reflections.m` produces some 2x2 matrices, and shows how to define symbolic variables like θ. Using symbolic values, find a 3x3 matrix for $R_{123}(\psi, \theta, \phi)$. (Hint: blkdiag might help.)

In [1]:

```
syms phi theta psi
Rotation = @(t) [cos(t) -sin(t) ; sin(t) cos(t)];
R12_psi = blkdiag(Rotation(psi), 1);
R23_theta = blkdiag(1,Rotation(theta));
R12_phi = blkdiag(Rotation(phi),1);
Euler_angle = R12_psi * R23_theta * R12_phi

Euler_angle =

[ cos(phi)*cos(psi) - cos(theta)*sin(phi)*sin(psi), - cos(psi)*sin(phi) - cos(ph
i)*cos(theta)*sin(psi),  sin(psi)*sin(theta)]
[ cos(phi)*sin(psi) + cos(psi)*cos(theta)*sin(phi),   cos(phi)*cos(psi)*cos(thet
a) - sin(phi)*sin(psi), -cos(psi)*sin(theta)]
[                               sin(phi)*sin(theta),
cos(phi)*sin(theta),           cos(theta)]
```

```
  File "<ipython-input-1-3af6eb047735>", line 1
    syms phi theta psi
           ^
SyntaxError: invalid syntax
```

Problem 4: Slices (20 points)

(a) Global Average Temperature Anomaly:
plot the average (non-missing-value) temperature anomaly over the entire grid, for every year from 1916 to 2015.

In [1]:

```
GHCN_in_centigrade  = (GHCN(:,3:74) - 2500) / 100;

temperature_anomaly = reshape( GHCN_in_centigrade, [36, 12, 137, 72] );   % conv
ert to a 4D matrix, so we can use slices
size( temperature_anomaly )
number_of_all_GHCN_values = prod(size( temperature_anomaly ));


missing_values = (temperature_anomaly == -25);
number_of_missing_values = sum(sum(sum(sum( missing_values ))));

WORLD_latitude = 1:36
WORLD_longitude = 1:72

my_years = 1916:2015
temperature_anomaly = temperature_anomaly .* (~ missing_values);
my_slice = temperature_anomaly( WORLD_latitude, :, my_years - 1880 + 1, WORLD_lo
ngitude );
total_number_of_grid_squares = length(WORLD_latitude) * length(WORLD_longitude)
* 12;
N = total_number_of_grid_squares;

average_WORLD_anomaly_by_year = reshape( sum(sum(sum( my_slice, 4),2),1), [lengt
h(my_years) 1] ) / N


plot( my_years, average_WORLD_anomaly_by_year )
xlabel('year')
ylabel('temperature anomaly -- Celsius')
title('average global temperature anomaly by year')
```

```
  File "<ipython-input-1-f79353b9a9e6>", line 3
    GHCN_in_centigrade  = (GHCN(:,3:74) - 2500) / 100;
                                ^
SyntaxError: invalid syntax
```

(b) Global Warming:

Based on your plot, give your opinion on this question: is `global warming' real?

In [ ]:

```
The plot clearly shows an upward trend, indicating the real global warming.
```