The goal of this project is building HDR image from multiple LDR images.

Step1: Recover Radiometric Response curve

To recover the response curve, the following equations should be used.

$$\mathcal{O} = \sum_{i=1}^{N} \sum_{j=1}^{P} \{w(Z_{ij}) [g(Z_{ij}) - \ln E_i - \ln \Delta t_j]\}^2 + \lambda \sum_{z=Z_{min}+1}^{Z_{max}-1} [w(z)g''(z)]^2 \quad \text{(Eq. 1)}$$

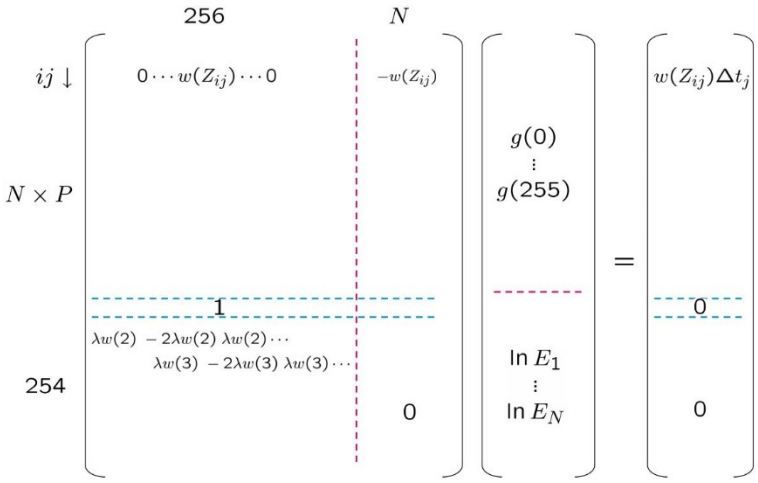
$$w(z) = \begin{cases} z - Z_{min} & \text{for } z \leq \frac{1}{2}(Z_{min} + Z_{max}) \\ Z_{max} - z & \text{for } z > \frac{1}{2}(Z_{min} + Z_{max}) \end{cases} \quad \text{(Eq. 2)}$$

Z is the pixel value;

E is the irradiance;

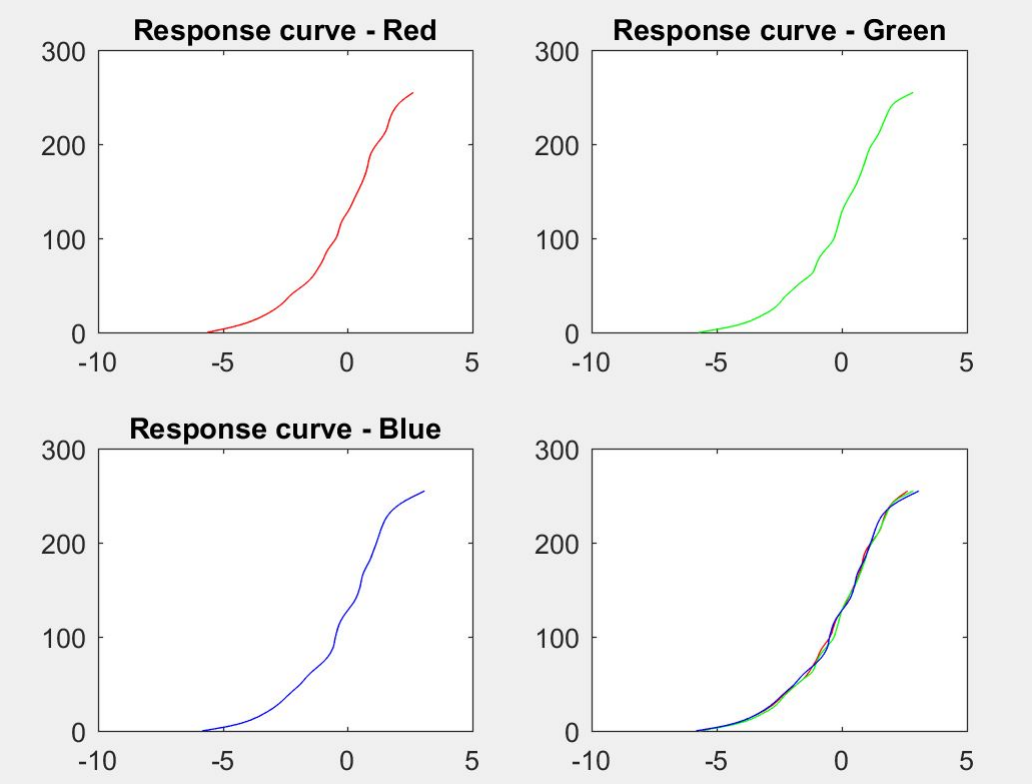
Delta t is the shutter speed;

W is the weighting function to emphasize the smoothness and fitting terms toward the middle of the curve.



To solve the over-determined system of linear equations (Eq. 1), least squares method can be used. Next, we should construct Ax = b to find x which is g and lnE.

The response curves are shown below.



Step2: Recover HDR radiance map

The radiance map can be recovered using $\ln(E[i]) = \frac{1}{P}\sum_{j=1}^{P}(g(Z[i,j]) - \ln(B[j]))$

A false color radiance map is shown below.

Step3: Tone mapping

Method 1: Gamma correction

First, normalize irradiance E using $E_{norm}[i] = \frac{E[i]-E_{min}}{E_{max}-E_{min}}$

Then, apply gamma correction to the image with $E_{gamma}[i] = E_{norm}^{\gamma}[i]$

The output is shown below.

Method2: global tone mapping from Reinhard

First, we need to convert radiance image from color to grayscale using $L[i] = rgb2gray(E_{norm}[i])$

Second, calculate the log average exposure using $L_{avg} = \exp\left(\frac{1}{N}\sum_i \ln(L[i])\right)$

Next, apply Reinhard tone-mapping with

$$T[i] = {}^{a}/_{L_{avg}}\, L[i] \qquad L_{tone}[i] = \frac{T[i]\left(1+{}^{T[i]}/_{T_{max}^2}\right)}{1+T[i]}$$

Then, define the scaling operator $M[i] = {}^{L_{tone}[i]}/_{L[i]}$

Last, multiply M to each color channel of radiance image.

The output is shown below.

Reinhard

Appendix

- **Code – use gsolve function, get response curve**

```
clc
clear all;
image6 = imread('C:\Users\dingx\Vision_HW4\code_test\1_2.jpg');
image5 = imread('C:\Users\dingx\Vision_HW4\code_test\1_4.jpg');
image4 = imread('C:\Users\dingx\Vision_HW4\code_test\1_8.jpg');
image3 = imread('C:\Users\dingx\Vision_HW4\code_test\1_15.jpg');
image2 = imread('C:\Users\dingx\Vision_HW4\code_test\1_30.jpg');
image1 = imread('C:\Users\dingx\Vision_HW4\code_test\1_60.jpg');

im1 = double(image1);
im2 = double(image2);
im3 = double(image3);
im4 = double(image4);
im5 = double(image5);
im6 = double(image6);

s = 480*640;
R =
[reshape(im1(:,:,1),[s,1]),reshape(im2(:,:,1),[s,1]),reshape(im3(:,:,1
),[s,1]),reshape(im4(:,:,1),[s,1]),reshape(im5(:,:,1),[s,1]),reshape(i
m6(:,:,1),[s,1])];
G =
[reshape(im1(:,:,2),[s,1]),reshape(im2(:,:,2),[s,1]),reshape(im3(:,:,2
),[s,1]),reshape(im4(:,:,2),[s,1]),reshape(im5(:,:,2),[s,1]),reshape(i
m6(:,:,2),[s,1])];
B =
[reshape(im1(:,:,3),[s,1]),reshape(im2(:,:,3),[s,1]),reshape(im3(:,:,3
),[s,1]),reshape(im4(:,:,3),[s,1]),reshape(im5(:,:,3),[s,1]),reshape(i
m6(:,:,3),[s,1])];

%lambda in [1,5]
l = 100;

%take 1000 random samples from R,G,B
% x = round(1000*rand);
% x = 1000;
% sample = randperm(size(R,1),x);

%take 1000 evenly distributed pixels
sample_even = (1:300:size(R,1));
x = size(sample_even,2);
R_sub = zeros(x,6);
G_sub = zeros(x,6);
B_sub = zeros(x,6);

% for j=1:x
%     R_sub(j,:) = R(sample(j),:);
%     G_sub(j,:) = G(sample(j),:);
```

```
%     B_sub(j,:) = B(sample(j),:);
% end

for j=1:x
    R_sub(j,:) = R(sample_even(j),:);
    G_sub(j,:) = G(sample_even(j),:);
    B_sub(j,:) = B(sample_even(j),:);
end


W = zeros(256,1);
for i=1:256
    W(i) = weight(i);
end

%Ax=B, get the B
b  = [1/60;1/30;1/15;1/8;1/4;1/2];
b = log(b);
b1 = zeros(x*6,6);
for j=1:size(b,1)
    b1(:,j) = b(j);
end

[g_R,lE_R] = gsolve(R_sub,b1,l,W);
[g_G,lE_G] = gsolve(G_sub,b1,l,W);
[g_B,lE_B] = gsolve(B_sub,b1,l,W);

X_R = zeros(x,6);
for k=1:6
X_R(:,k) = lE_R + b1(1:x,k);
end

subplot(2,2,1)
plot(g_R,(0:255),'r')
title('Response curve - Red')
hold on

subplot(2,2,2)
plot(g_G,(0:255),'g')
title('Response curve - Green')
hold on

subplot(2,2,3)
plot(g_B,(0:255),'b')
title('Response curve - Blue')
hold on

subplot(2,2,4)
plot(g_R,(0:255),'r')
hold on
plot(g_G,(0:255),'g')
hold on
```

```
plot(g_B,(0:255),'b')

% plot(X_R(:,1),R_sub(:,1),'.')
% hold on
% plot(X_R(:,2),R_sub(:,2),'.')
% hold off
```

- **Code – get radiance map**

```
%%radiance map
sum_R = 0;
sum_G = 0;
sum_B = 0;

eR = zeros(s,6);
eG = zeros(s,6);
eB = zeros(s,6);

for m = 1:256
    for n = 1:6
        indices_R = find(R(:,n) == m-1);
        indices_G = find(G(:,n) == m-1);
        indices_B = find(B(:,n) == m-1);

        eR(indices_R,n) = g_R(m)-b(n);
        eG(indices_G,n) = g_G(m)-b(n);
        eB(indices_B,n) = g_B(m)-b(n);
    end
end

for n1 = 1:6
    sum_R = sum_R + eR(:,n1);
    sum_G = sum_G + eG(:,n1);
    sum_B = sum_B + eB(:,n1);
end

W_R = W(R+1);
W_G = W(G+1);
W_B = W(B+1);

radiance_mapR = sum_R/6;
radiance_mapG = sum_G/6;
radiance_mapB = sum_B/6;

radiance_mapR = reshape(radiance_mapR,[480,640]);
radiance_mapG = reshape(radiance_mapG,[480,640]);
radiance_mapB = reshape(radiance_mapB,[480,640]);

delta = 7;
% radiance = cat(3,radiance_mapR,radiance_mapG,radiance_mapB);
% h = heatmap(radiance,'x','y','symmetric','false');
```

```
L_w = 0.2126*radiance_mapR+0.7152*radiance_mapG+0.0722*radiance_mapB;
 L_w_bar = exp(mean(log(L_w(:) + delta))); %%% delta is a small number
 L = (0.045/L_w_bar)*L_w;
%%% 0.18 is the middle-grey key value;
%%% You may set the value to 0.09, 0.36, 0.54, 0.72.
HeatMap(flipud(L), 'colormap', 'jet', 'symmetric', 'false')
```

- **Code – normalization**

```
 EnormR = zeros(480,640);
 EnormG = zeros(480,640);
 EnormB = zeros(480,640);
 minE =
min([min(radiance_mapR(:)),min(radiance_mapG(:)),min(radiance_mapB(:))
]);
 maxE =
max([max(radiance_mapR(:)),max(radiance_mapG(:)),max(radiance_mapB(:))
]);
%normalization
for k = 1:480
   for l = 1:640
         EnormR(k,l) = (radiance_mapR(k,l)-minE)/(maxE-minE);
         EnormG(k,l) = (radiance_mapG(k,l)-minE)/(maxE-minE);
         EnormB(k,l) = (radiance_mapB(k,l)-minE)/(maxE-minE);
   end
end
```

- **Code – gamma correction**

```
%%gamma correction
gamma = 0.75;
A = 1;
EgammaR = A*EnormR.^gamma;
EgammaG = A*EnormG.^gamma;
EgammaB = A*EnormB.^gamma;

imageGamma = cat(3,EgammaR,EgammaG,EgammaB);
figure
imshow(imageGamma)
title('Gamma')
```

- **Code – Reinhard**

```
Enorm3 = cat(3,EnormR,EnormG,EnormB);
 M = tonemap(Enorm3);

Rnew = zeros(480,640);
Gnew = zeros(480,640);
Bnew = zeros(480,640);
```

```
for k3 = 1:480
    for k4 = 1:640
        Rnew(k3,k4) = M(k3,k4)*EnormR(k3,k4);
        Gnew(k3,k4) = M(k3,k4)*EnormG(k3,k4);
        Bnew(k3,k4) = M(k3,k4)*EnormB(k3,k4);
    end
end

newI = cat(3,Rnew,Gnew,Bnew);
figure
imshow (newI)% M = Ltone./L1;
title ('Reinhard')
```

- **Code – weighting function**

```
function w = weight(z)

Zmin = 1;
Zmax = 256;

if z<=0.5*(Zmin+Zmax)
    w = z - Zmin;
else
    w = Zmax - z;
end

end
```

- **Code – reinhard**

```
function M = tonemap(Enorm3)

L1 = rgb2gray(Enorm3);
L2 = log(L1);
sumL = sum(L2(:));
n = size(Enorm3,1)*size(Enorm3,2);
Lavg = exp((1/n)*sumL);
a = 0.18;
T = (a/Lavg)*L1;
Tmax2 = max(T(:))*max(T(:));
b = 1+ T/Tmax2;
% Ltone = T.*b/(1+T);
Ltone = zeros(480,640);
for p=1:480
    for h=1:640
        Ltone(p,h) = (T(p,h)*b(p,h))/(1+T(p,h));
    end
end
```

```
M = zeros(480,640);
for k1 = 1:480
    for k2 = 1:640
        M(k1,k2) = Ltone(k1,k2)/L1(k1,k2);
    end
end


end
```