Question 1. Write an Auto Exposure Bracketing (AEB) function for Tegra (5 Points)

Answer:

Android Studio Codes

```
public void captureExposureStack(View v) {

    //TODO:hw4
    //TODO: Getting min and max exposures.
    Range<Long> exposureRange =
characteristics.get(CameraCharacteristics.SENSOR_INFO_EXPOSURE_TIME_RANGE);
    Long minimumExposure = exposureRange.getLower();
    Long maximumExposure = exposureRange.getUpper();
    Log.e(TAG, "minimumExposure: " + minimumExposure);
    Log.e(TAG, "maximumExposure: " + maximumExposure);
    Long prevExposure = minimumExposure;
    //check if 2* exposure >maximumExposure
    while (prevExposure + prevExposure < maximumExposure) {
        try {
            //sleep the system for 20ms between each capture.
            SystemClock.sleep(20);
            Log.e(TAG, "exposure: " + prevExposure);
            //TODO:update exposure time
            prevExposure = prevExposure+prevExposure;
            //create capture requester
            CaptureRequest.Builder requester =
mCameraDevice.createCaptureRequest(mCameraDevice.TEMPLATE_MANUAL);
            //TODO: set requester exposure time
            requester.set(CaptureRequest.SENSOR_EXPOSURE_TIME,prevExposure);
            //add surface
            requester.addTarget(mCaptureBuffer.getSurface());
            Log.e(TAG, "exposure: " + prevExposure);
            //check capture session and make capture request
            if (mCaptureSession != null)
                exposures.add(prevExposure);
                mCaptureSession.capture(requester.build(), null, null);
        } catch (CameraAccessException e) {
            Log.e(TAG, "Failed to build actual capture request", e);
        }
    }

}
```

According to the standard of pixels value, six images are chosen which are as follows.

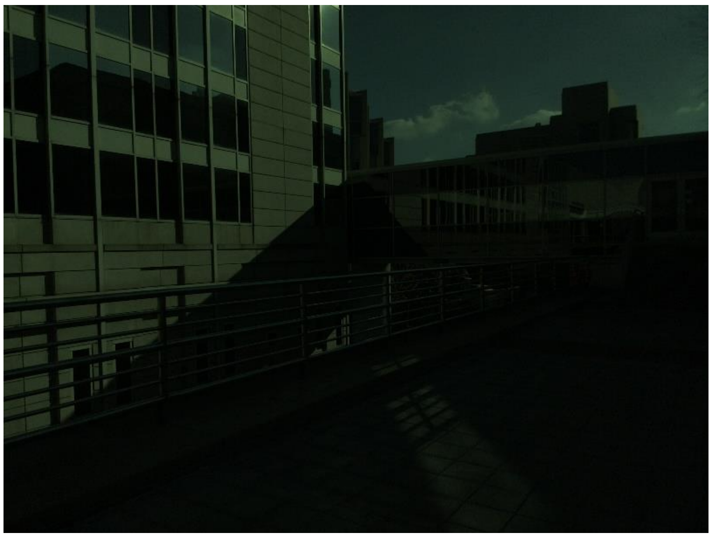
Figure 1. exposure time= 67200e-09 s

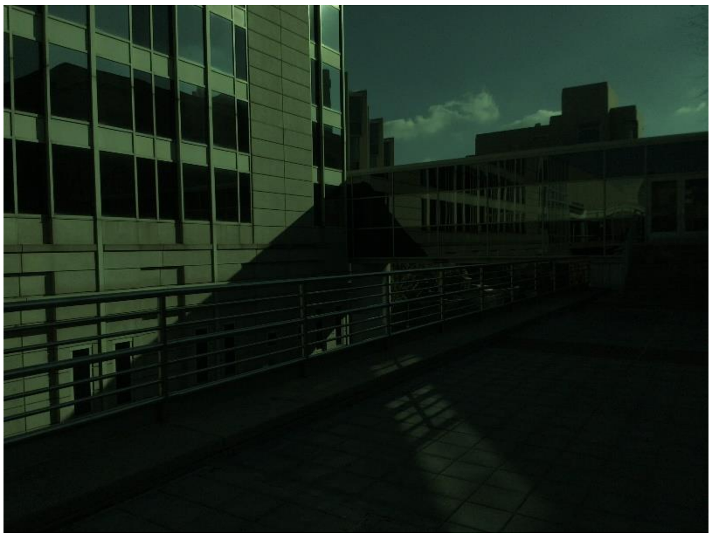
Figure 2. exposure time= 134400e-09 s

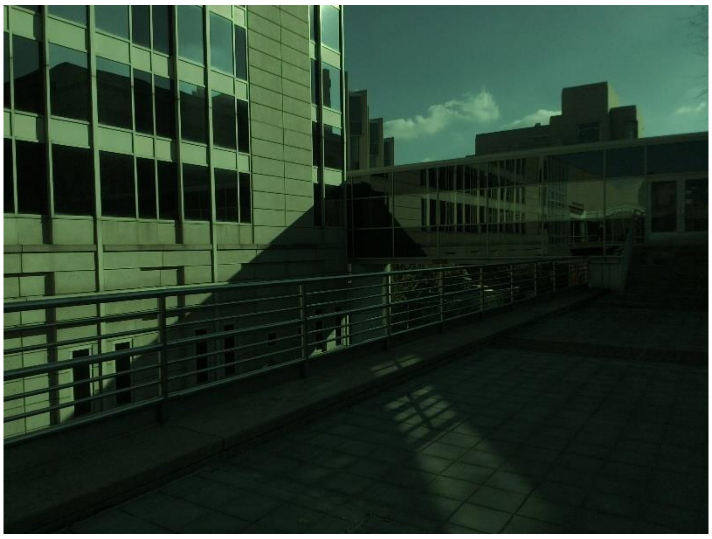
Figure 3. exposure time= 268800e-09 s

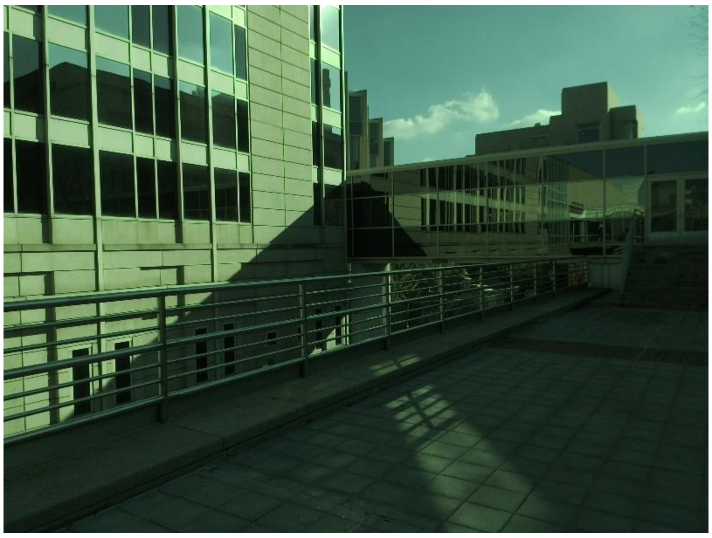
Figure 4. exposure time= 537600e-09 s

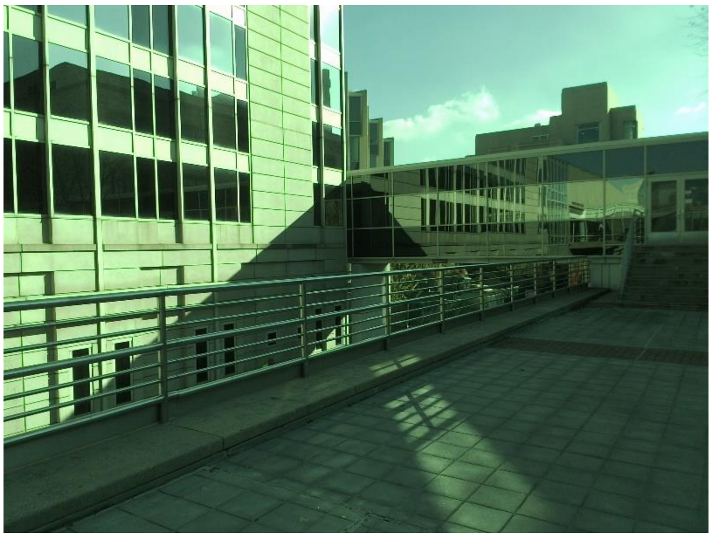
Figure 5. exposure time= 1075200e-09 s

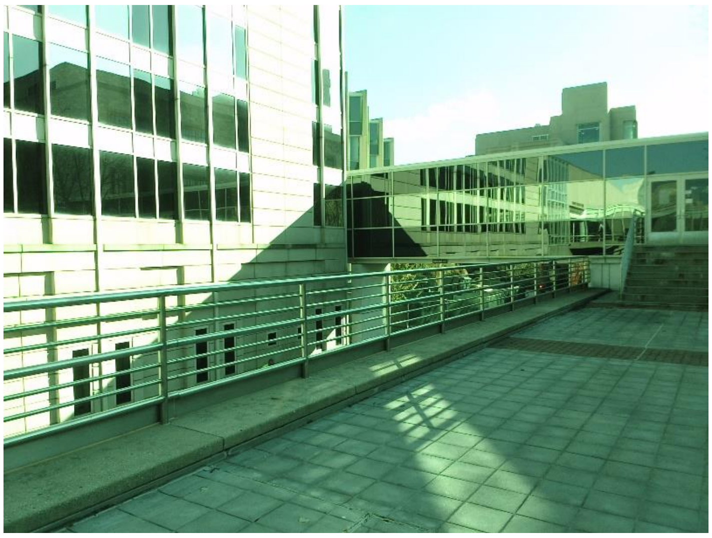
Figure 6. exposure time= 2150400e-09 s

Question 2. Write a program to find the camera response curves for the shield tablet (2 Points)

Answer:

When increasing the value of l, the fitted data becomes more compact and the response cure gets smoother. Results of l=0.1,3,5 of red channel and l=5 of red, green and blue channel are as follows. The rest of processing is based on l=5.

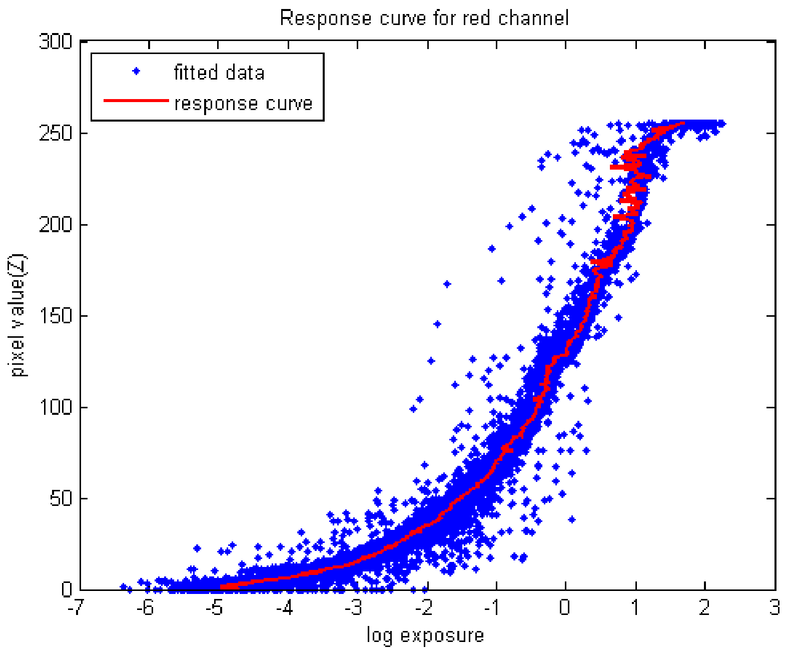

Figure 7. l=0.1

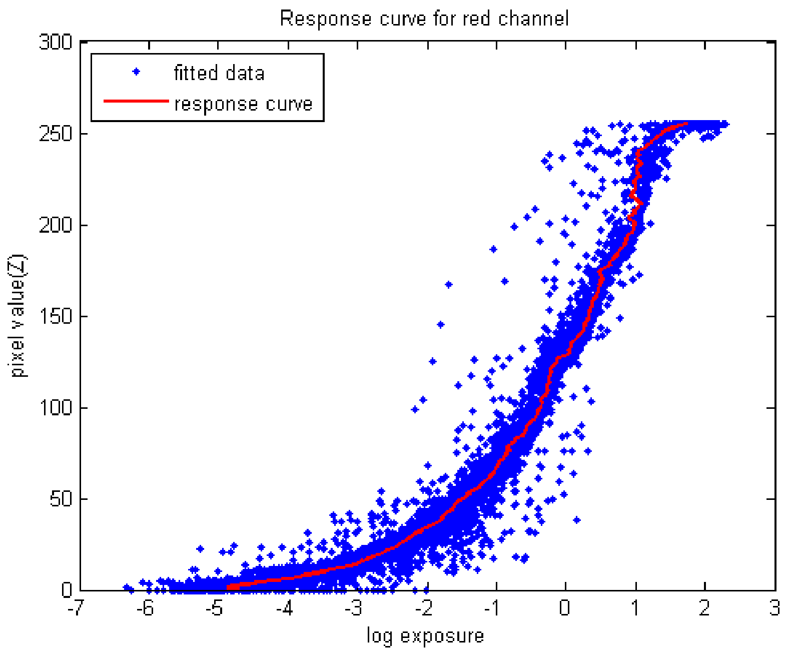

Figure 8. l=3

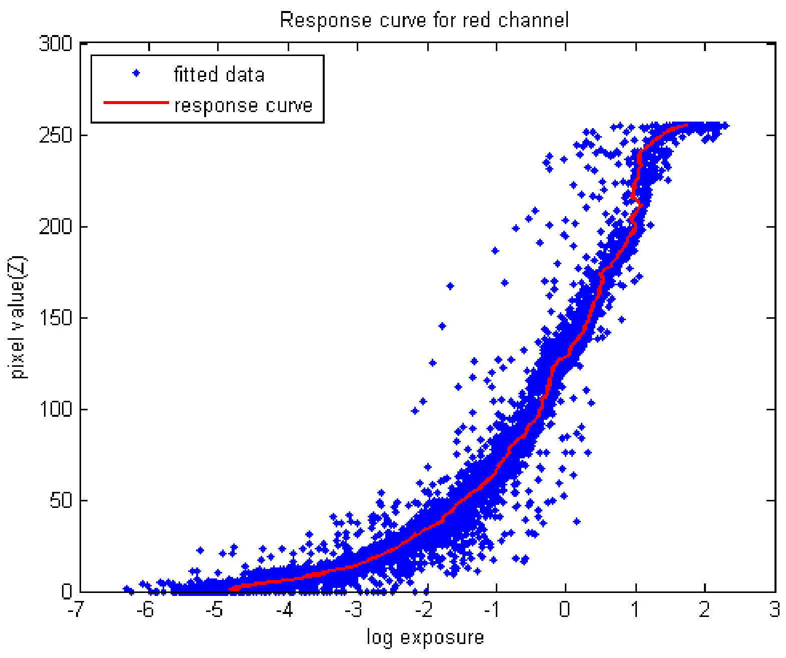

Figure 9. l=5

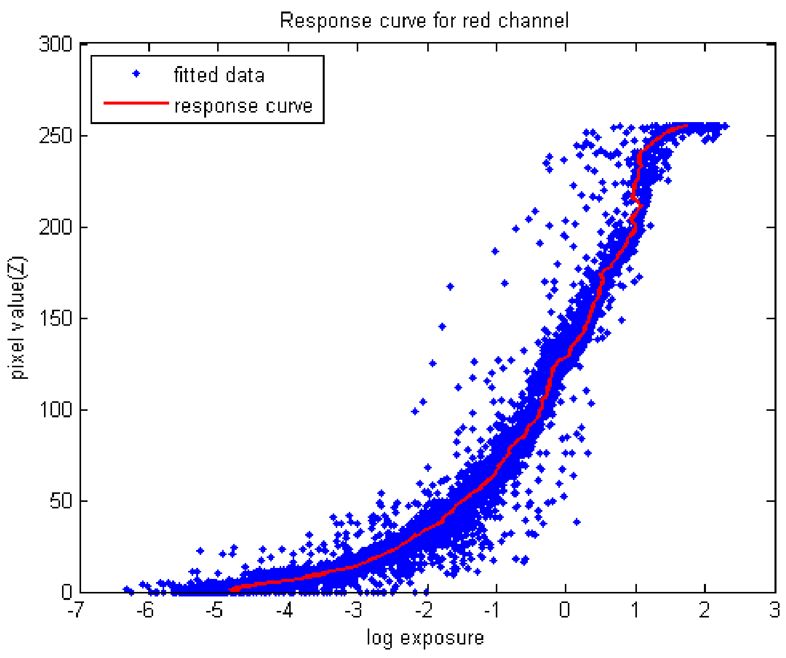

Figure 10. l=5 red channel

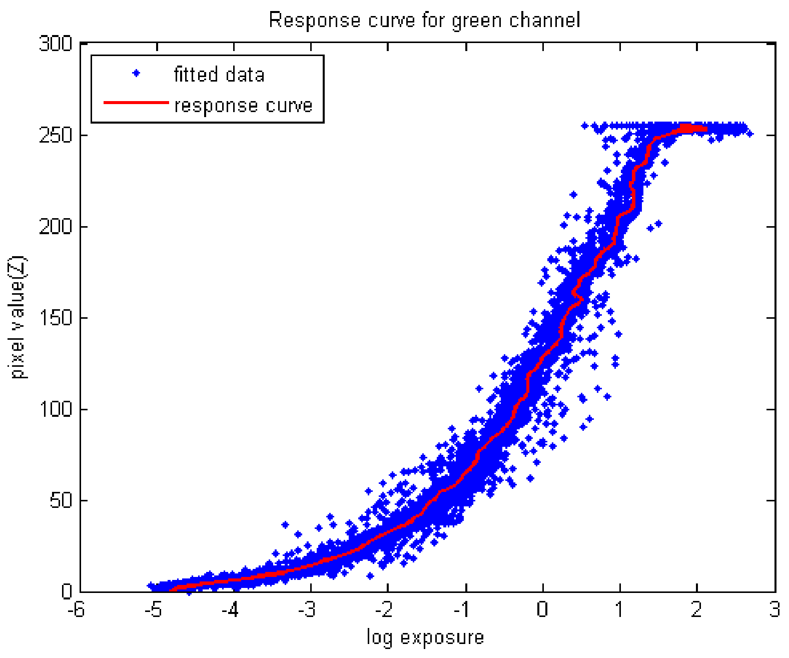

Figure 11. l=5 green channel

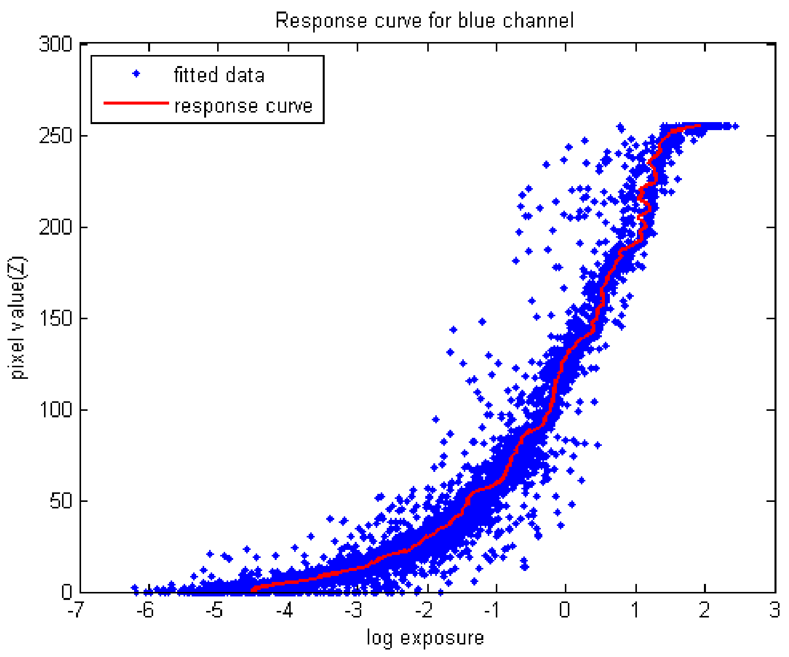

Figure 12. l=5 blue channel

Matlab Codes

```
clear;
clc;
B=[67200;
134400;
268800;
537600;
1075200;
2150400
        ];
B=B*10^-9;
k=size(B,1);
for i=1:k
        B(i)=log(B(i));
end
Io=cell(k,1);
Ior=cell(k,1);
Iog=cell(k,1);
Iob=cell(k,1);
Icr=cell(k,1);
Icg=cell(k,1);
Icb=cell(k,1);
p=randperm(1000);
for k=1:6
        imgname=strcat('D:\Courses Files\Introduction to Computational Photography\HW4\新建文件夹
\',num2str(k),'.jpg');
        Io{k}=imread(imgname);
        Ior{k}=Io{k}(:,:,1);
        Iog{k}=Io{k}(:,:,2);
        Iob{k}=Io{k}(:,:,3);
        Icr{k}=Ior{k}(339,:);
        Icg{k}=Iog{k}(339,:);
        Icb{k}=Iob{k}(339,:);
end
[gr]=Responsecurve(Icr,B,1,5);
[gg]=Responsecurve(Icg,B,2,5);
[gb]=Responsecurve(Icb,B,3,5);

function [g]=Responsecurve(I,B,r,l)
RGB={' red';' green';' blue'};
Z=zeros(size(I{1},1)*size(I{1},2),size(I,1));
X=zeros(size(I{1},1)*size(I{1},2),size(I,1));
for i=1:size(I,1)
        Z(:,i)=reshape(I{i},size(I{1},1)*size(I{1},2),1);
end
[g,lE]=gsolve(Z,B,l);
for i=1:size(I{1},1)*size(I{1},2)
        for j=1:size(I,1)
                X(i,j)=lE(i)+B(j);
        end
end
```

```
ZX(:,1)=reshape(X,size(X,1)*size(X,2),1);
ZX(:,2)=reshape(Z,size(Z,1)*size(Z,2),1);
figure();
plot(ZX(:,1),ZX(:,2),'.','color','b');
hold on
y=0:1:255;
plot(g(y+1),y,'-','color','r','LineWidth',2);
legend('fitted data','response curve','location','northwest');
xlabel('log exposure');
ylabel('pixel value(Z)');
imgname=strcat('Response curve for',RGB(r),' channel');
title(imgname);
end
```

Question 3. Recover the HDR radiance map of the scene (3 Points)

Answer:

The HDR radiance maps of red, green and blue channel are as follows.

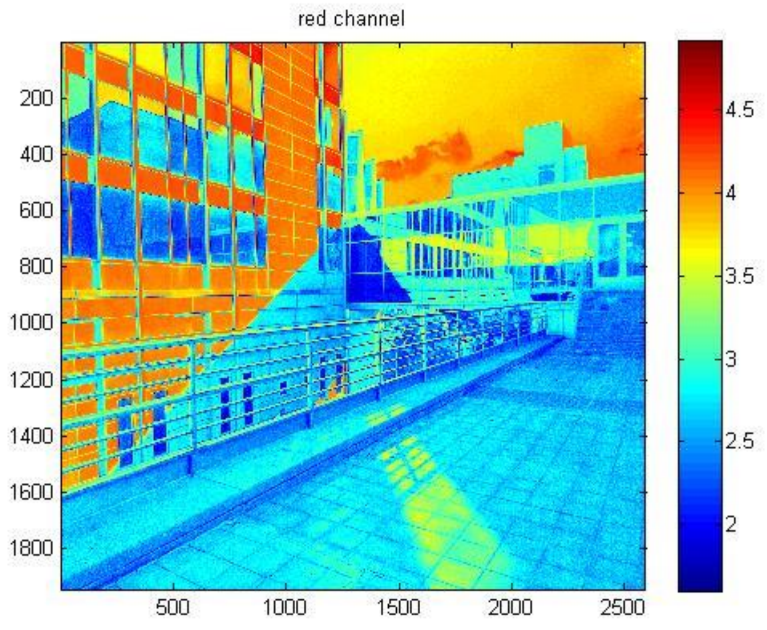

Figure 13. red channel

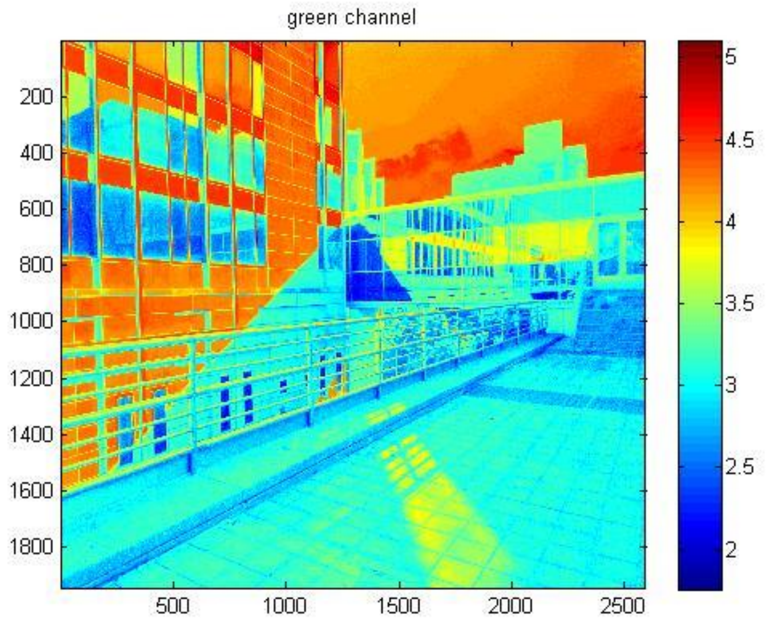

Figure 14. green channel

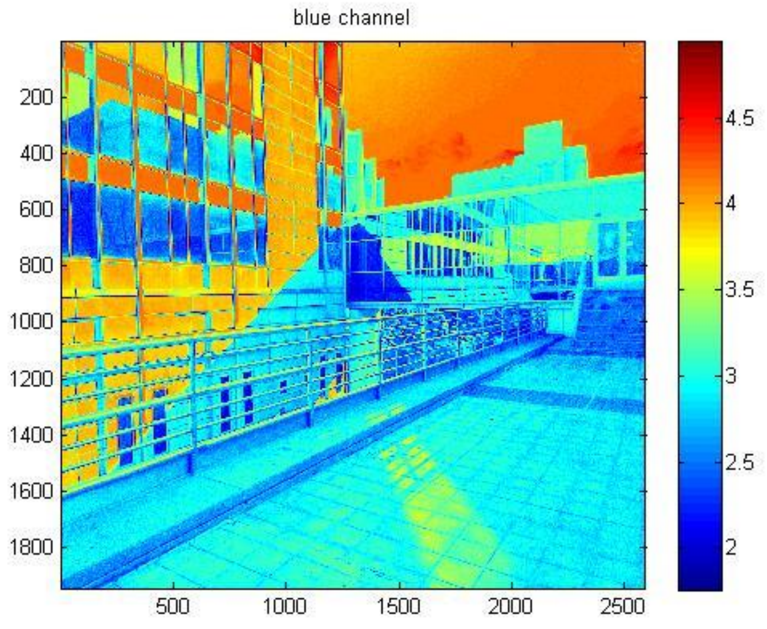

Figure 15. blue channel

The dynamic ranges of my scene are 10^4 for red channel, 10^4 for green channel and 10^3 for blue channel.

Matlab Codes

```
%%%%%% HDR radiance map %%%%%%
[Er,Er10,Ero]=map(Ior,B,gr);
[Eg,Eg10,Ego]=map(Iog,B,gg);
```

```
[Eb,Eb10,Ebo]=map(Iob,B,gb);
figure(),imagesc(Er10);colorbar;title('red channel');
figure(),imagesc(Eg10);colorbar;title('green channel');
figure(),imagesc(Eb10);colorbar;title('blue channel');
function [E,E10,Eo]=map(I,B,g)
m=size(I{1},1);
n=size(I{1},2);
E=zeros(m,n);
for i=1:m
    for j=1:n
        for k=1:size(B,1)
            gz=g(I{k}(i,j)+1);
            E(i,j)=E(i,j)+gz-B(k);
        end
        E(i,j)=E(i,j)/5;
    end
end
E10=log10(exp(E));
Eo=exp(E);
end
```

Question 4. Implement a tone mapping algorithm to display your HDR image (5 Points)

Answer:

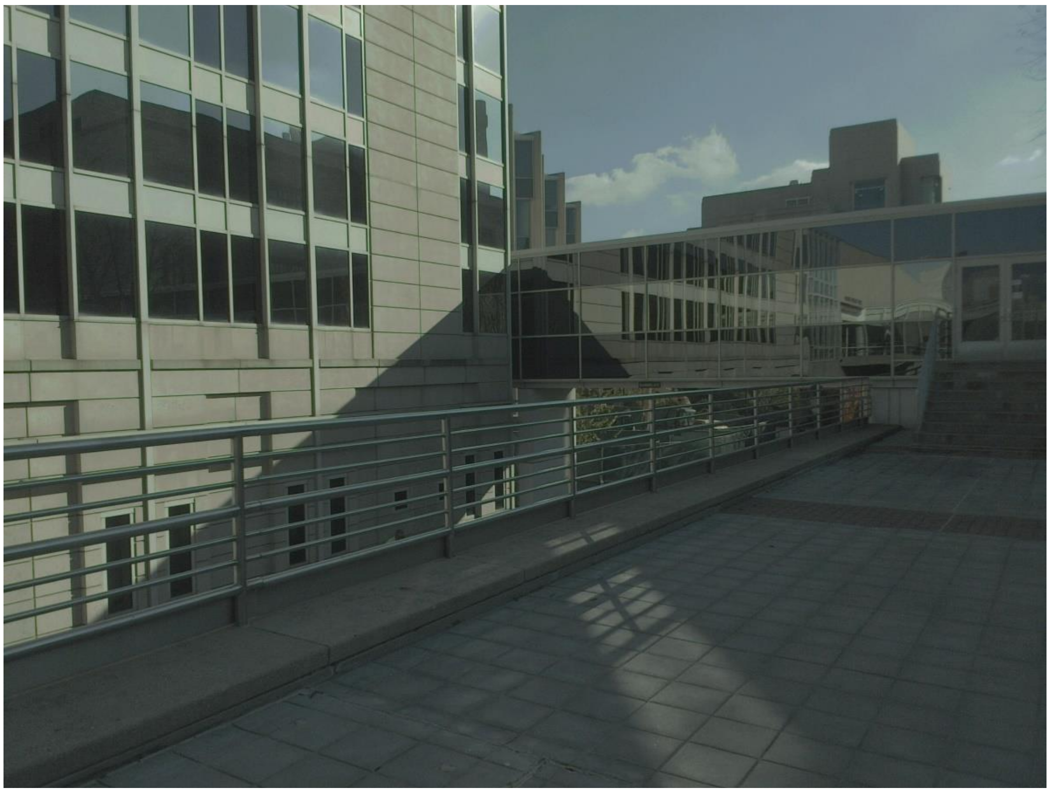

Figure 16. gamma correction γ=0.3

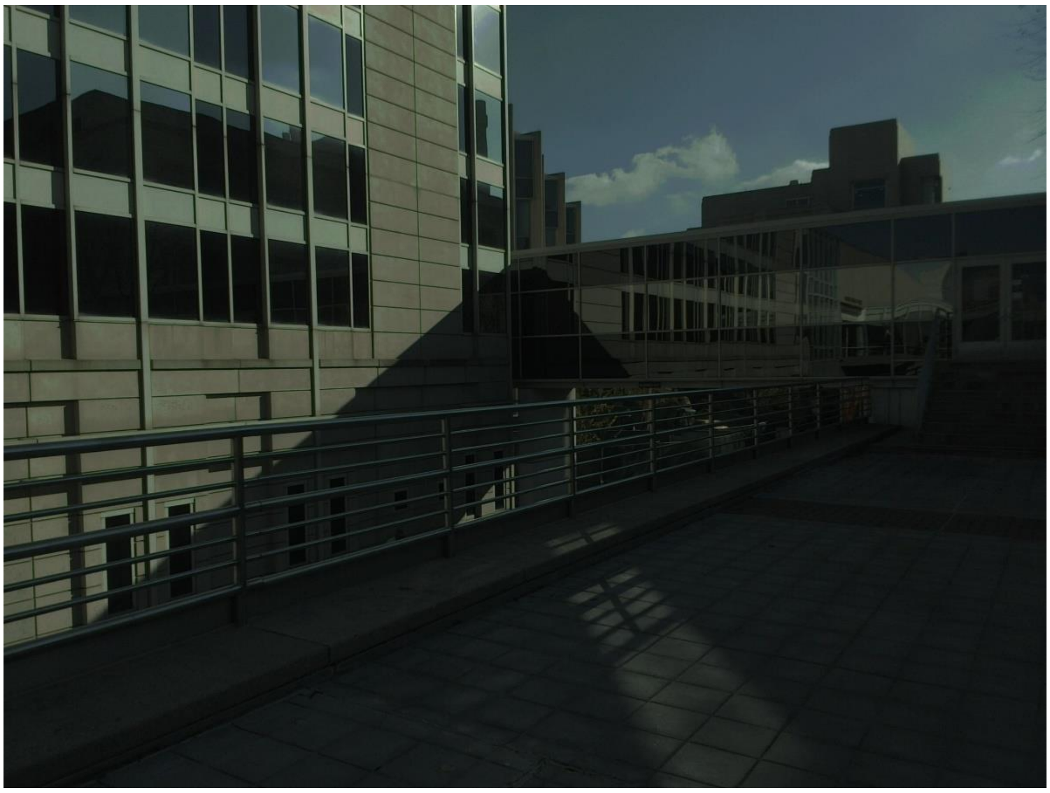

Figure 17. gamma correction γ=0.5

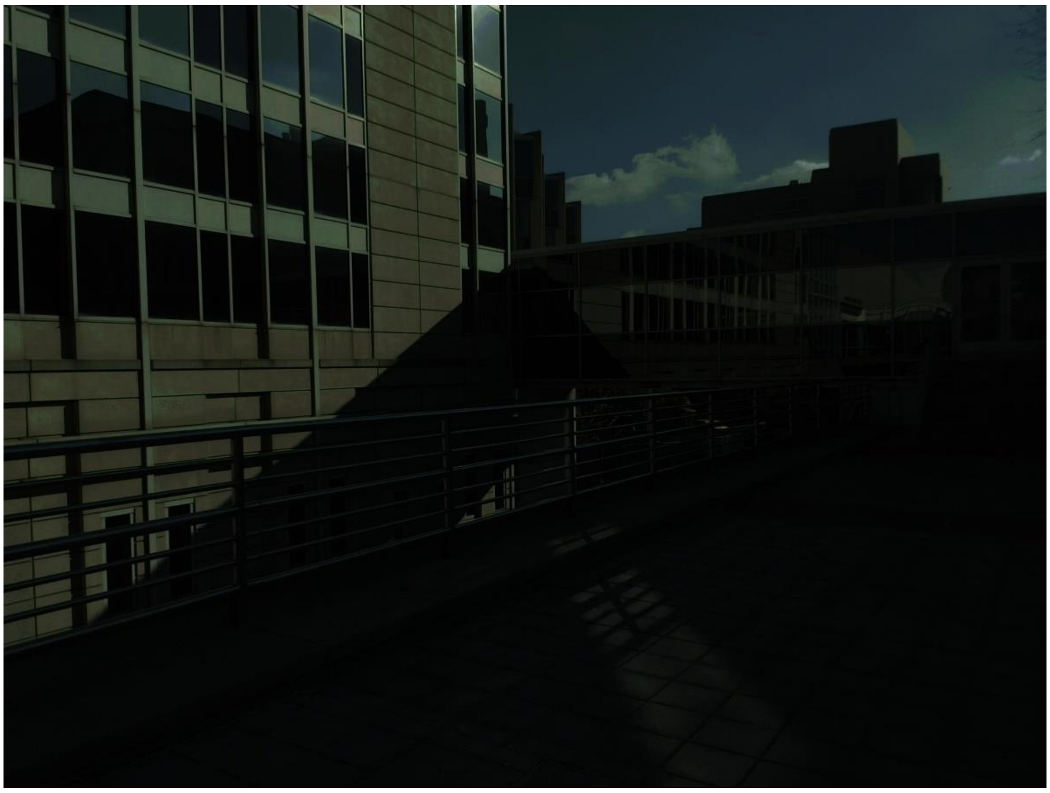

Figure 18. gamma correction γ=0.8

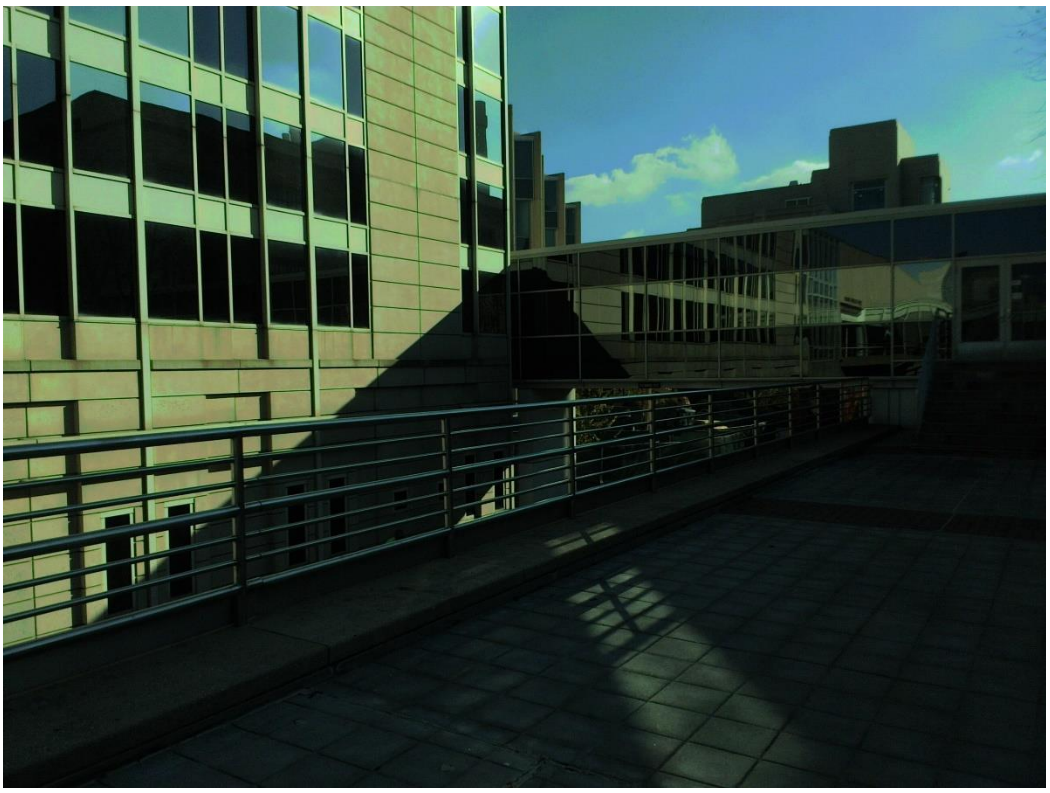
Figure 19. tone mapping algorithm a=0.18

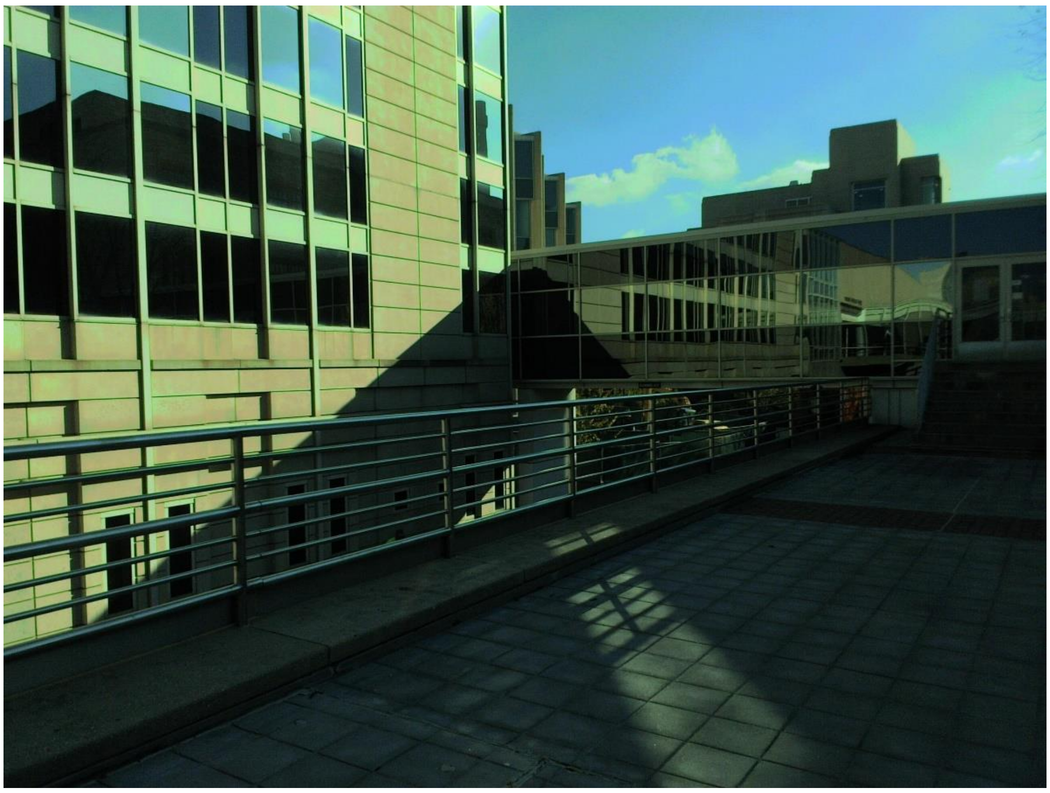
Figure 20. tone mapping algorithm a=0.28

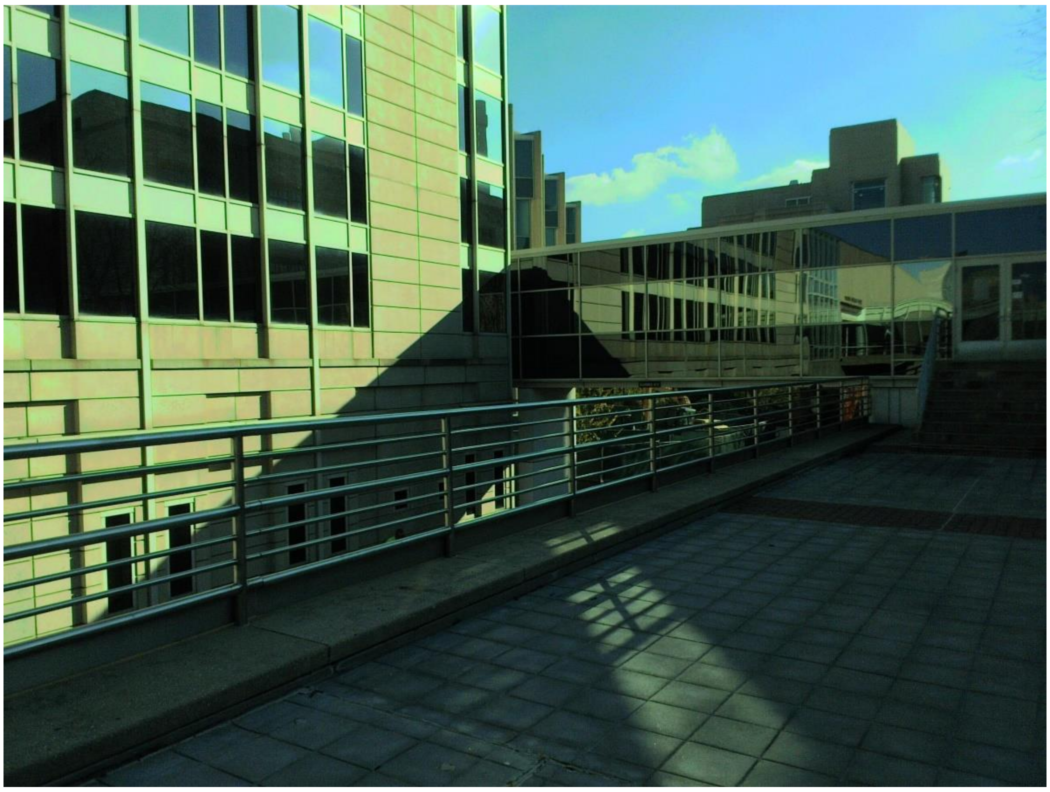
Figure 21. tone mapping algorithm a=0.38

From the results above, we can see that when a=0.38 and γ=0.5, we can obtain relative high quality images.

Matlab Codes

```
%%%%%% HDR image%%%%%%
[Ergamma,Ernorm]=HDRimage(Ero,0.3);
figure(),imagesc(Ergamma);colorbar;
[Eggamma,Egnorm]=HDRimage(Ego,0.3);
figure(),imagesc(Eggamma);colorbar;
[Ebgamma,Ebnorm]=HDRimage(Ebo,0.3);
figure(),imagesc(Ebgamma);colorbar;
Enormrgb(:,:,1)=Ernorm;
Enormrgb(:,:,2)=Egnorm;
Enormrgb(:,:,3)=Ebnorm;
Egamma(:,:,1)=Ergamma;
Egamma(:,:,2)=Eggamma;
Egamma(:,:,3)=Ebgamma;
L=rgb2gray(Enormrgb);
Lavg=0;
for i=1:size(L,1)
    for j=1:size(L,2)
        Lavg=Lavg+log(L(i,j));
    end
end
```

```
Lavg=exp(Lavg/(size(L,1)*size(L,2)));
T=0.28/Lavg.*L;
Ltone=T.*(1+T./(max(max(T)))^2)./(1+T);
M=Ltone./L;
Rnew=M.*Ernorm;
Gnew=M.*Egnorm;
Bnew=M.*Ebnorm;
New(:,:,1)=Rnew;
New(:,:,2)=Gnew;
New(:,:,3)=Bnew;
figure(),imshow(New);
title('New');
figure(),imshow(Egamma);
title('Gamma');
figure(),imshow(Enormrgb);
title('Enormrgb');
imwrite(New,'New.jpg','jpeg');
imwrite(Egamma,'Gamma.jpg','jpeg');
imwrite(Enormrgb,'Enormrgb.jpg','jpeg');

function [Egamma,Enorm]=HDRimage(E,r)
Enorm=(E-min(min(E)))./(max(max(E))-min(min(E)));
m=size(E,1);
n=size(E,2);
Egamma=zeros(m,n);
for i=1:m
    for j=1:n
        Egamma(i,j)=Enorm(i,j)^r;
    end
end
end
```

Problems encountered:

1. The pictures captured by the tablet are somehow strange. We all don`t know why they appear to be kind of green.
2. The dynamic of the radiance maps is lower than 10^6. Maybe it is because the features of the scene doesn`t own a wide range radiance.