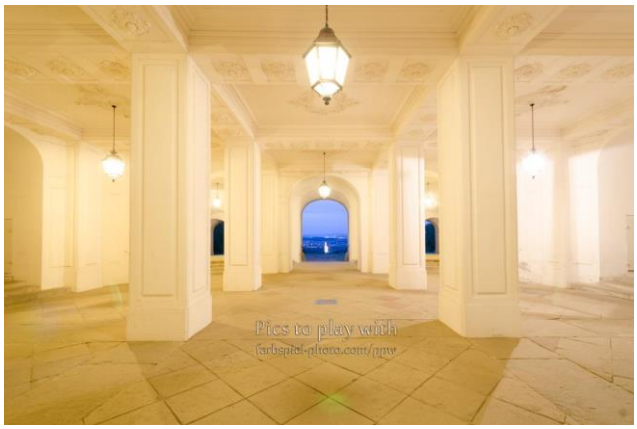
Programming Assignment 10

NikhilTank: 13110070

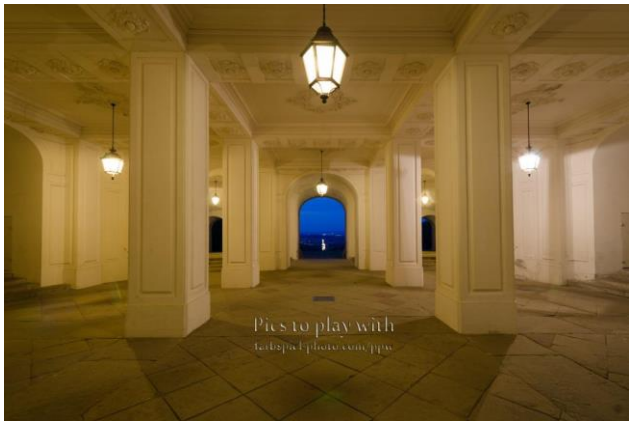
1. Given a set of multi-exposure images of a dynamic scene captured using a static camera, design an algorithm to generate the tone mapped HDR image of the scene without any ghosts.

Input images:

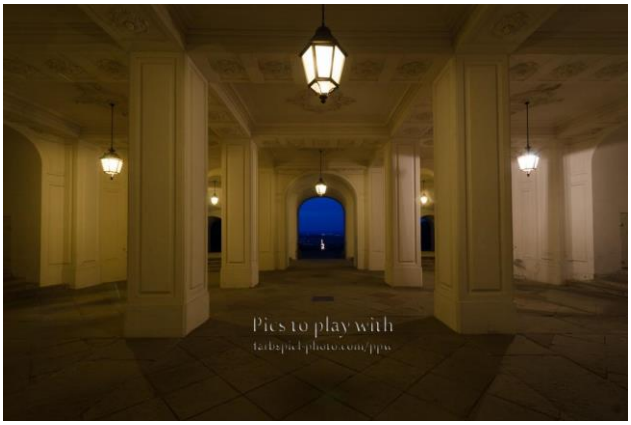


Into the Open (HDR) - ppw - 01.tif
> Shutter speed: 30s

Into the Open (HDR) - ppw - 02.tif
> Shutter speed: 15s (Image with unwanted object)



Into the Open (HDR) - ppw - 03.tif
> Shutter speed: 8s (Reference Image)



Into the Open (HDR) - ppw - 04.tif
> Shutter speed: 4s

Into the Open (HDR) - ppw - 05.tif
> Shutter speed: 2s

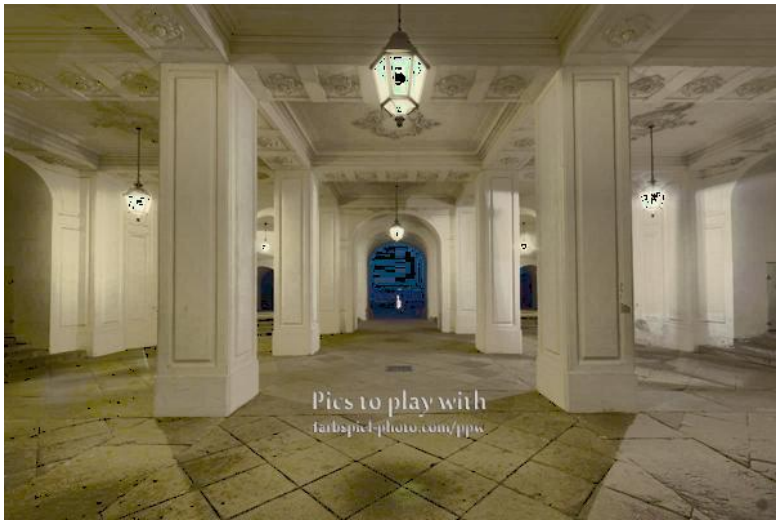
Into the Open (HDR) - ppw - 06.tif
> Shutter speed: 1s

After generating HDR from these imaging using algorithm for **Static scene (Tone mapped)**
We can see presence of Ghost

```
%% Matlab Code
%% Methodology to remove Ghost
% I have designed a DECISION matrix which will decide whether to keep a path
%from image or not, and make a list of it for each image.
clc
clear all
img1 = imread('ppw - 01.jpg');
img1 = imresize(img1, 0.5);
img2 = imread('ppw - 02.jpg');
img2 = imresize(img2, 0.5);
img3 = imread('ppw - 03.jpg');
img3 = imresize(img3, 0.5);
img4 = imread('ppw - 04.jpg');
img4 = imresize(img4, 0.5);
img5 = imread('ppw - 05.jpg');
img5 = imresize(img5, 0.5);
img6 = imread('ppw - 06.jpg');
img6 = imresize(img6, 0.5);

[x, y, c] = size(img1);
archive = {img1,img2,img3,img4,img5,img6};
B = [log(30); log(15); log(8); log(4); log(2); log(1)];
b = [30; 15; 8; 4; 2; 0];


%% generating weighting function
zmin = 0;
zmax = 255;
w = zeros(256,1);
for i = 0:255
   if i <= 1/2*(zmin+zmax)
      w(i+1) = i - zmin;
   elseif i > 1/2*(zmin+zmax)
      w(i+1) = zmax - i;
   end
end

%% generating response funtion
```

```
inew=zeros(256,6);
for z =1:c
   for p =1:6
      arc_temp = archive{p}(:,:,z);
      inew(:,p) = arc_temp(1:256);
   end
    g = gsolve(inew,B,5,w);
    G(:,z) = g;
end

%% finding what patch to keep
archive = {double(img1),double(img2),double(img3),double(img4),double(img5),double(img6)};
base = 40; %square path dimension
V = x/base; %patches in vertical
H = y/base; %patches in horizontal
d = {zeros(V, H, c),zeros(V, H, c),zeros(V, H, c),zeros(V, H, c),zeros(V, H, c),zeros(V, H, c)};% decision matrix
D = {zeros(V, H),zeros(V, H),zeros(V, H),zeros(V, H),zeros(V, H),zeros(V, H)};% decision matrix
for z = 1:c
   for p = 1:6
      for v = 1:V
         for h = 1:H
            out = 0;
            if (v-1)==0
               low_i = 1;
            else
               low_i = (v-1)*base;
            end
            high_i = v*base;

            if (h-1)==0
               low_j = 1;
            else
               low_j = (h-1)*base;
            end
            high_j = h*base;

            for i = low_i:high_i
               for j = low_j:high_j
                  if p+1 < 7
                     if abs( (log(b(p)*archive{p}(i,j,z))) - (log(b(3)*archive{3}(i,j,z))) ) / (log(b(3)*archive{3}(i,j,z))) > 0.15 % for p image and reference
image p=3
                        out = out + 1;
                     end
                  end
               end
            end

            if out > (x*y*c)*0.001 % decision for patch by 0.1% of outlines
               d{p}(v,h,z) = 0; % patch can't be kept inside, since does contain ghost
            else
               d{p}(v,h,z) = 1; % patch can be kept inside, since doesn't contain ghost
            end
         end
      end
   end
end

for p = 1:6
   for z = 1:c
      D{p}(:,:) = d{p}(:,:,1).*d{p}(:,:,2).*d{p}(:,:,3);
   end
end

%% getting HDR image
Num = zeros(x,y,c);
Den = zeros(x,y,c);
I = zeros(x,y,c);
for z = 1:c
```

```
  for v = 1:V
    for h = 1:H

      if (v-1)==0
        low_i = 1;
      else
        low_i = (v-1)*base;
      end
      high_i = v*base;

      if (h-1)==0
        low_j = 1;
      else
        low_j = (h-1)*base;
      end
      high_j = h*base;

      for i = low_i:high_i
        for j = low_j:high_j
          for p =1:6
            Num(i,j,z) = (D{p}(v,h))*w(archive{p}(i,j,z)+1)*(G(archive{p}(i,j,z)+1,z) - B(p))+ Num(i,j,z);
            Den(i,j,z)=  (D{p}(v,h))*w(archive{p}(i,j,z)+1) +  Den(i,j,z);
          end
          I(i,j,z) = Num(i,j,z)/Den(i,j,z);
        end
      end
    end
  end
end
I = exp(I);
tone = tonemap(I);
imwrite(tone, 'ImageHDR.jpg');
```



After **removing Ghost (Tone mapped)**
But artifacts are generated due to non-linear camera response function

```
%% Further code to do the semless bleding in the HDR image
%% smoothing
test = 0;
I = imresize(uint8(I), 0.5);
[x, y, c] = size(I);
Image = zeros(x,y,c);
for z = 1:c

  [Gmag, Gdir] = imgradient(I(:,:,z));
```

```
    divG = imdiv(Gmag);
    [x,y,c]=size(I);
    poiL = sparse(x*y,x*y);

    %% make matrix to poisson of an irradence
    for i =1:x*y %row variable %start & endpt included
       poiL(i,i) = (-4);
       if i-y >= 1
          poiL(i,i-y) = 1;
       end
       if i-1 >= 1
          if rem((i-1),y) ~= 0
             poiL(i,i-1) = 1;
          end
       end
       if i+y<=x*y
          poiL(i,i+y) = 1;
       end
       if i+1<=x*y
          if rem(i,y) ~= 0
             poiL(i,i+1) = 1;
          end
       end
    end

    %% make verticle source matrix
    G = sparse(x*y,1);
    r=0;
    for i = 1:x %row variable
       for j = 1:y
          G(i+j+r-1,1) = divG (i,j);
       end
       r = r + y - 1;
    end

    %% multiply the divG with inverse of poiL
    invpoiL(:,:) = inv(poiL(:,:));
    L_cap(:,:) = invpoiL(:,:)*G(:,:);

    %% transfer the verticle Image matrix to m*n matrix
    L_star = sparse(x,y);
    r=0;
    for i = 1:x %row variable
       for j = 1:y
          L_star (i,j) = L_cap(i+j+r-1,1) ;
       end
       r = r + y - 1;
    end
    test = test+1;
    Image(:,:,z) = L_star(:,:);

  end
  Image = exp(double(Image));
  Image = tonemap(Image);
  imwrite(Image,'HDR_smooth.jpg');
```

Reference:

[1] *ARTIFACT-FREE HIGH DYNAMIC RANGE IMAGING*
Orazio Gallo_, Natasha Gelfandz, Wei-Chao Chenz, Marius Tico z, and Kari Pulli.