IT-524 COMPUTER VISION -ASSIGNMENT 2

Kunal Suthar
201401131

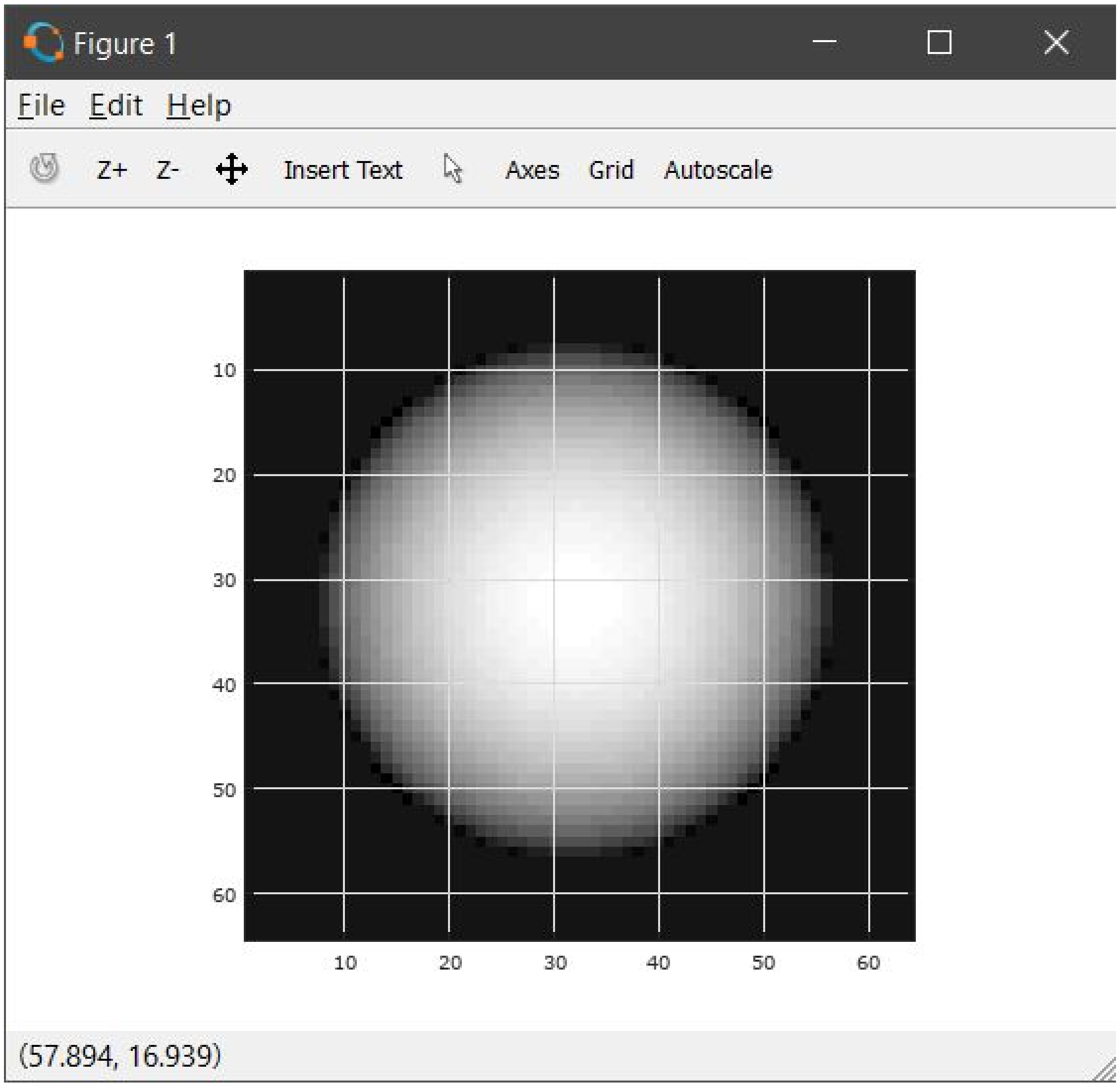
Use regularization based approach finding p,q and z with source as (0,0), (0.8389,0.7193), (0.5773,0.6363) and without any noise. How does the recovered depth compare with the actual depth? Radius of sphere=24.

Code Snippet for generating E with (ps,qs)=0:

```
clear all;
close all;
% Image generator with source direction = s_orig
M=64;
N=64;
radius=24; % Radius of sphere
Depth = zeros(M,N);
p_init= zeros(M,N);
q_init= zeros(M,N);
E= zeros(M,N);
R = 0.2 * ones(M,N); % Background Light
s_orig=[0,0];  % p,q coordinate of source
for i=1:M,
    for j=1:N,
        current_radius = sqrt((i-M/2)^2 + (j-N/2)^2);
        if(current_radius < radius) % Assign Depth only for points that are part of sphere
in image
            Depth(i,j) = round(sqrt(radius^2 - (i-M/2)^2 - (j-N/2)^2)); % estimate depth for
each pixel of sphere
            p = (i-M/2)/Depth(i,j);
            q = (j-N/2)/Depth(i,j);
            % Calculate E from p,q & s
            temp = Rval(p, q, s_orig);
            p_init(i,j)=p;
            q_init(i,j)=q;
            % Ensuring nonegativity of E
            if(temp>0)
                E(i,j) = temp;
            else
                E(i,j) = 0;

            end
        end
    end
end
```

Image:



Code Snippet for finding p and q using iterative method:

```
M = size(E,1);
N = size(E,2);

En = E;

p_old = p_init;
```

```
q_old = q_init;
p_new = zeros(size(En));
q_new = zeros(size(En));

maxiter = 300; %Maximum number of iterations


isize= M;
iter = 0;
lambda =   50;

while (1)
    %For p
    if(1)
        for i = 2:isize-1
            for j = 2:isize-1

                    p_new(i,j) = Ravg(p_old, i, j)  + (1/lambda)*( En(i,j) -
Rval(p_old(i,j),q_old(i,j),s) )*Rp(p_old(i,j),q_old(i,j),s(1),s(2)) ;

            end
        end
    end
    %For q
    if(1)

         for i = 2:isize-1
            for j = 2:isize-1
                    q_new(i,j) = Ravg(q_old, i, j)  + (0.25/lambda)*( En(i,j) -
Rval(p_old(i,j),q_old(i,j),s))*Rq(p_old(i,j),q_old(i,j),s(1),s(2)) ;
                          end
         end
    end

    if  (iter == maxiter)
        break;

    end

    p_old = p_new;
    q_old = q_new;

    iter = iter + 1;
end

pn=p_new;
qn=q_new;
figure(1),imshow(mat2gray(pn));
figure(2),imshow(mat2gray(qn));
```

Code for Ravg, Rval, Rp and Rq respectively:

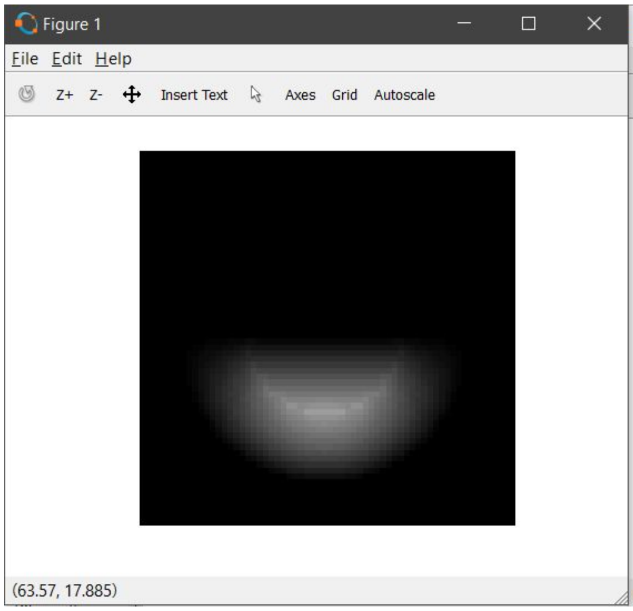
```
function val = Ravg(A,i,j)
    val = ( A(i,j-1) + A(i-1,j) + A(i,j+1) + A(i+1,j) )/4;% +(A(i-1,j-1)+
A(i-1,j+1)+A(i+1,j-1)+A(i+1,j+1))/12;
end
```

```
function val = Rval(p,q,s)
    val = (s(1)*p + s(2)*q + 1)/sqrt( (s(2)*s(2) + s(1)*s(1) +1) * (p*p + q*q +1));
end
```

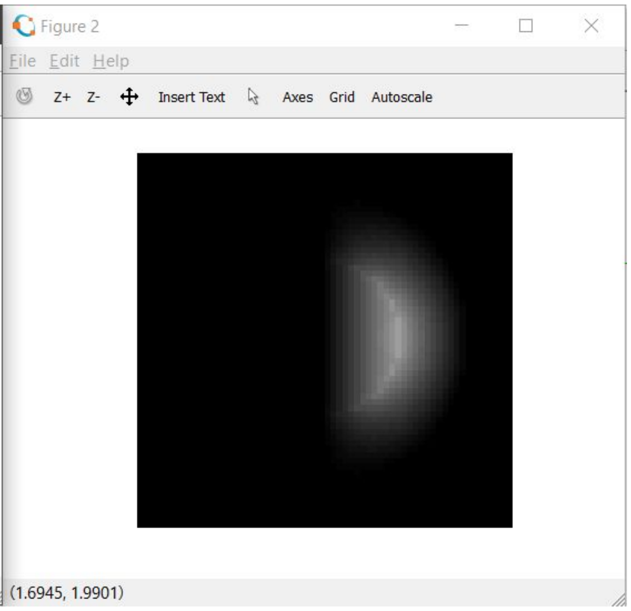
```
function val = Rp(p,q,ps,qs)
    val = (ps*(q*q +1) - p*(qs*q + 1))/(sqrt((qs*qs + ps*ps +1) * (p*p + q*q +1)) *
(p*p+q*q+1));
end
```

```
function val = Rq(p,q,ps,qs)
    val = (qs*(p^2 +1) - q*(ps*p + 1))/(sqrt((qs^2 + ps^2 +1) * (p^2 + q^2 +1)) *
(p^2+q^2+1));
end
```

Images of calculated p and q using iterative method:



P

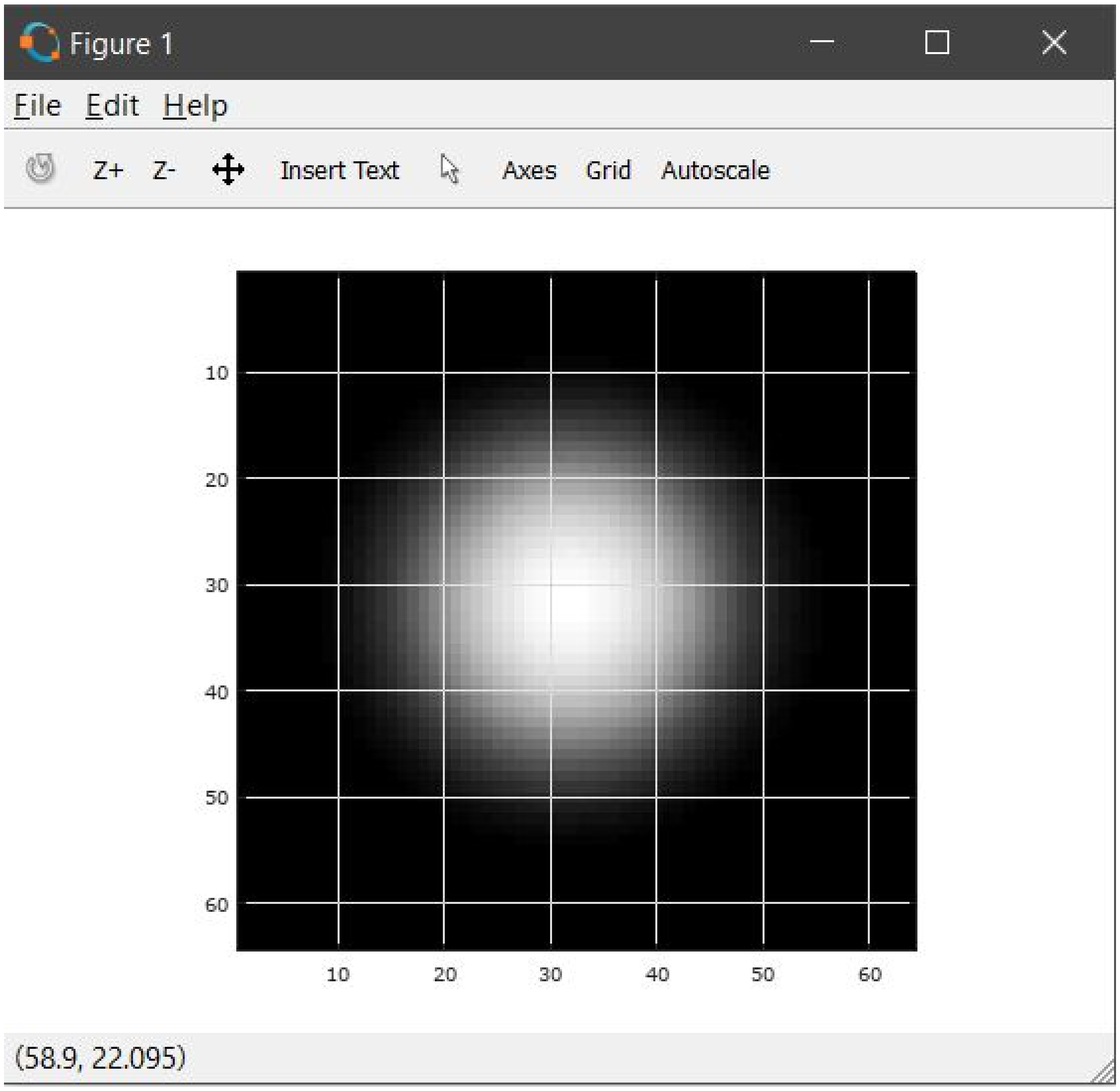


Q

Code Snippet for finding Z using iterative method:

```
M = size(E,1);
N = size(E,2);
zn = zeros(size(E));
z_old = zeros(size(E));
px = diff(pn,1,1);
qy = diff(qn,1,2);
% Apply the iterative method to estimate the value of depth at each point
iters = 10000
for kk = 1:iters,
    disp(kk)
    for i=2:(M-1),
        for j=2:(N-1),
             if mask(i,j)==1
                zn(i,j) = Ravg(z_old,i,j) + px(i,j) + qy(i,j);
             else
                 zn(i,j) = 0;
             end
        end
    end
    z_old = zn;
end
figure;
imshow(mat2gray(zn));
```

Image for Z using iterative method:



For (ps,qs)=(0.8389,0.7193)

Image for generated E:

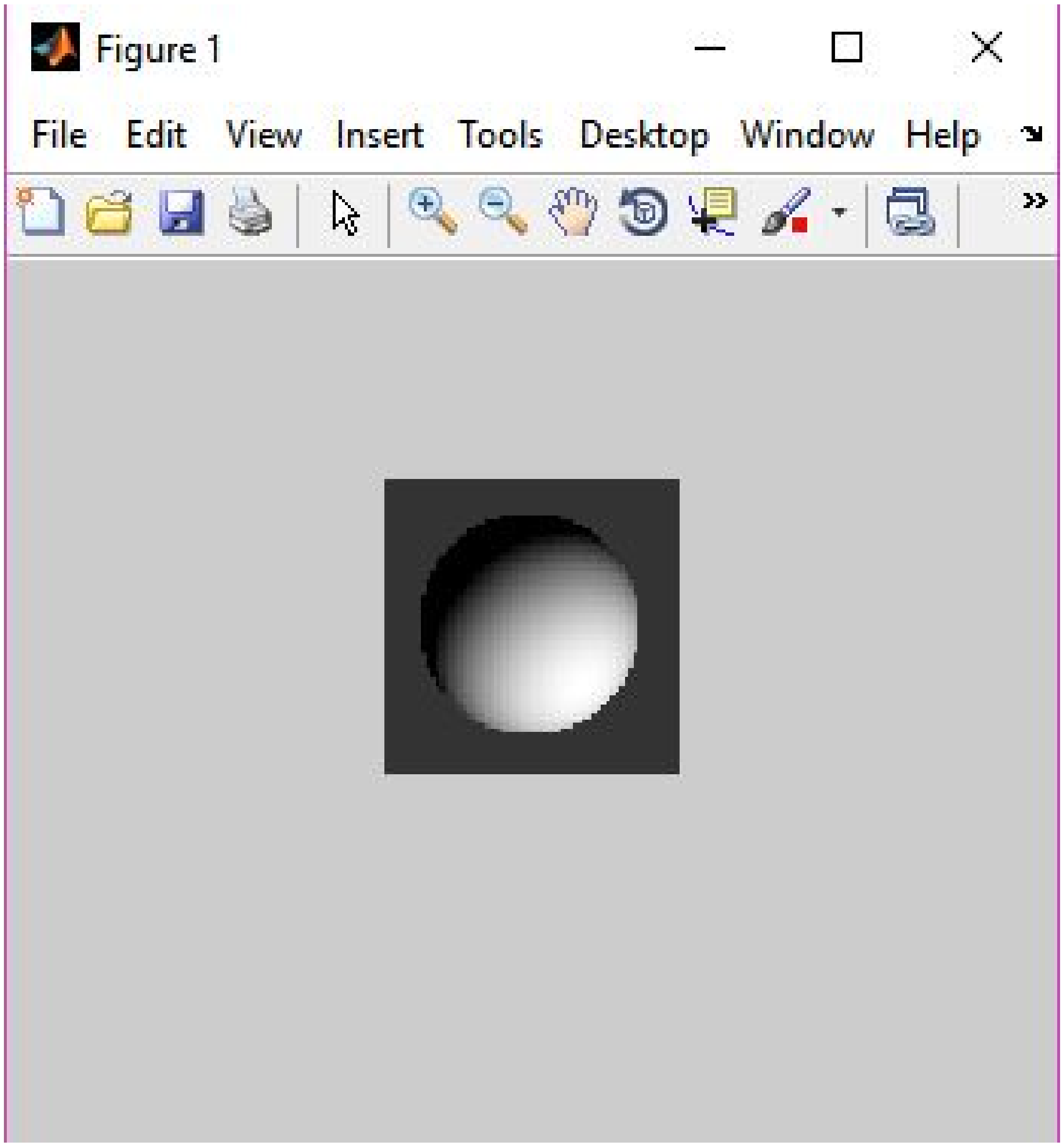
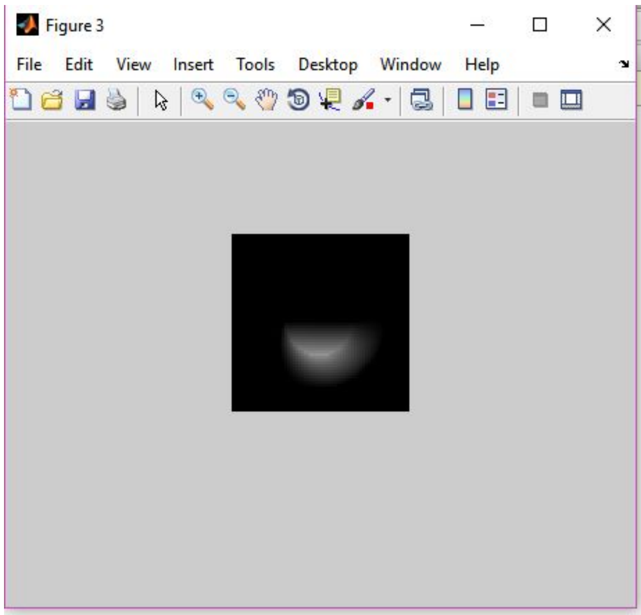
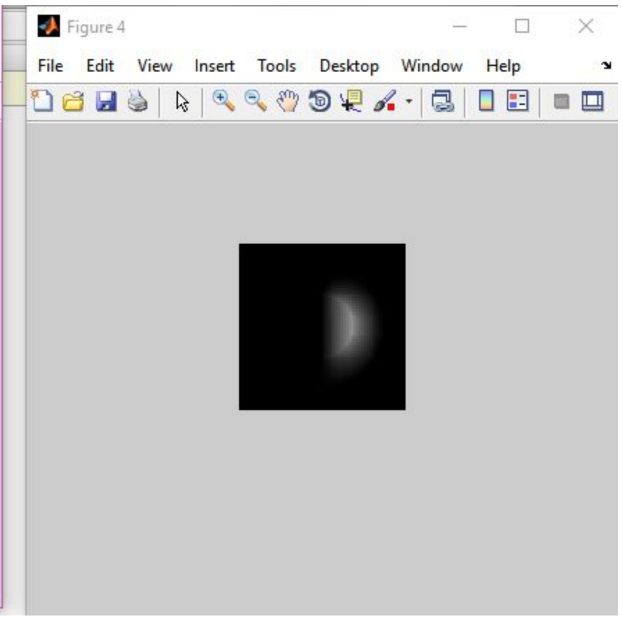


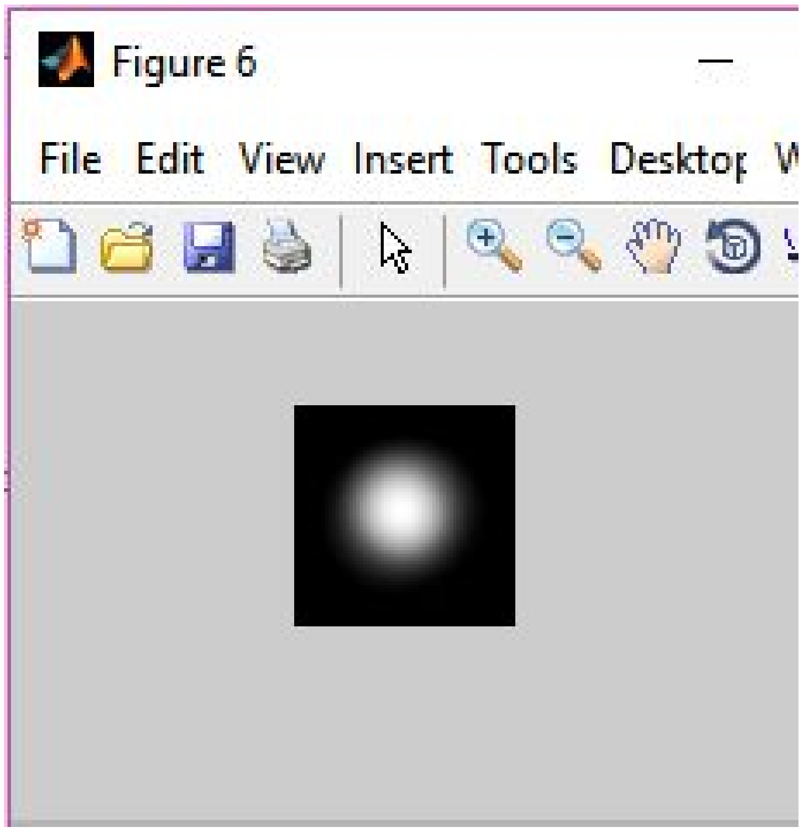
Image for generated p and q:





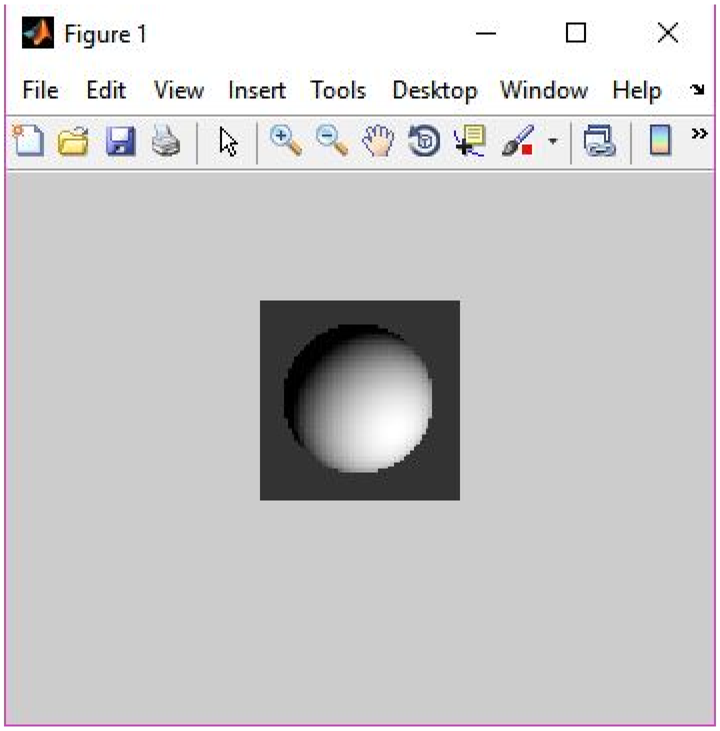
P **Q**

Image for generated Z:

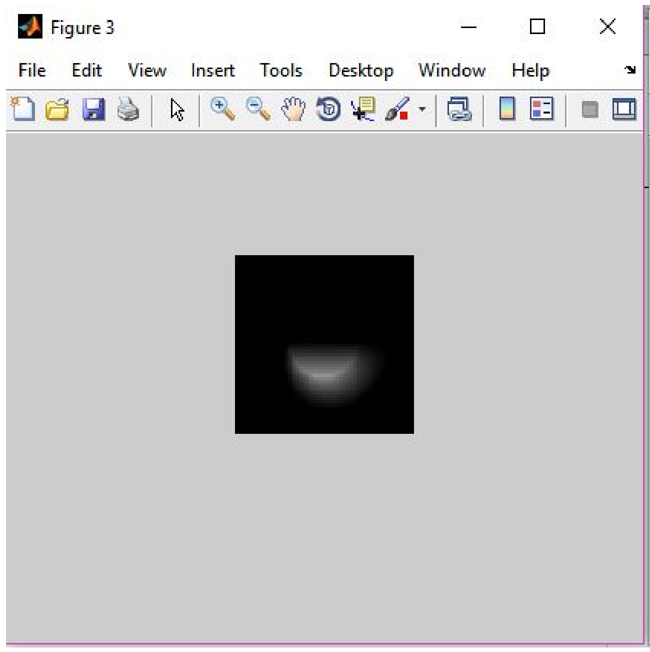


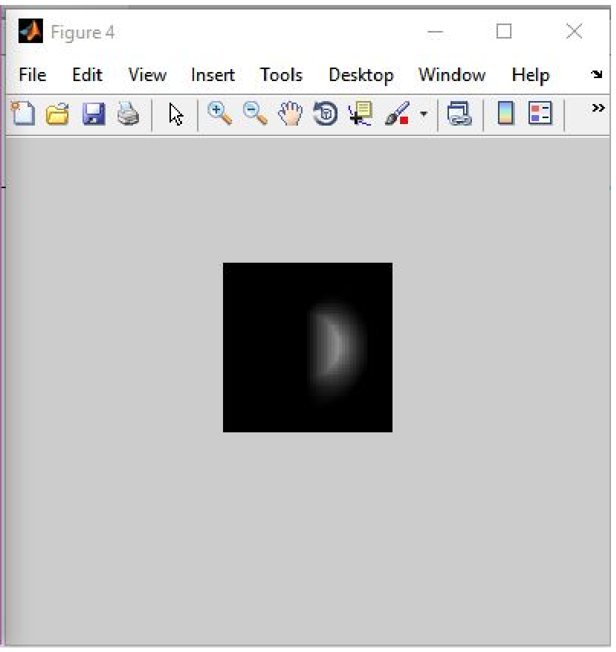
For (ps,qs)=(0.5773,0.6363):

Image for generated E:



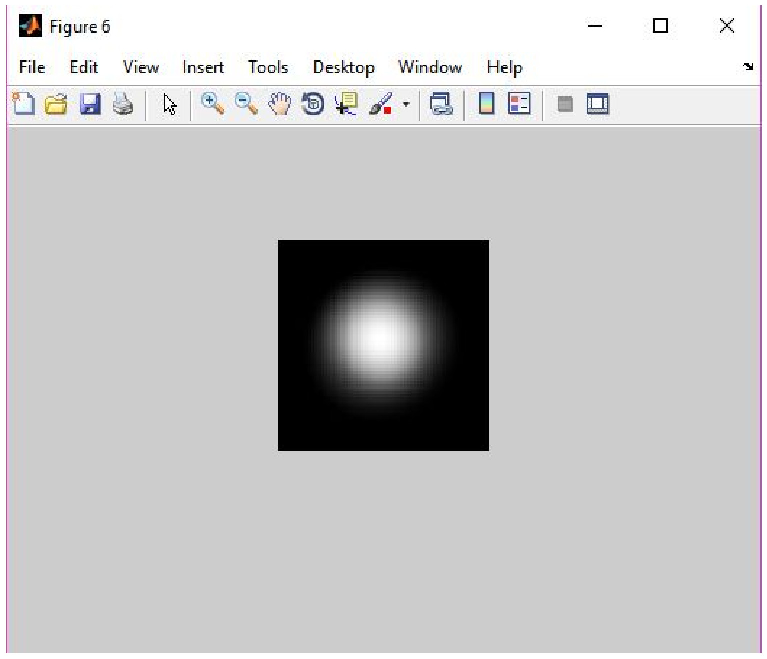
Images for generated p and q:





P **Q**

Image for generated Z:



Mean Squared Errors in Z:

Code Snippet:

```
Error = abs(Depth-zn).^2;
MSEz = sum(Error(:))/numel(zn);
```

Mean Square Error with (ps,qs)=(0,0): 35.3961
Mean Square Error with (ps,qs)=(0.8389,0.7193): 31.6883
Mean Square Error with (ps,qs)=(0.5773,0.6363):32.7267