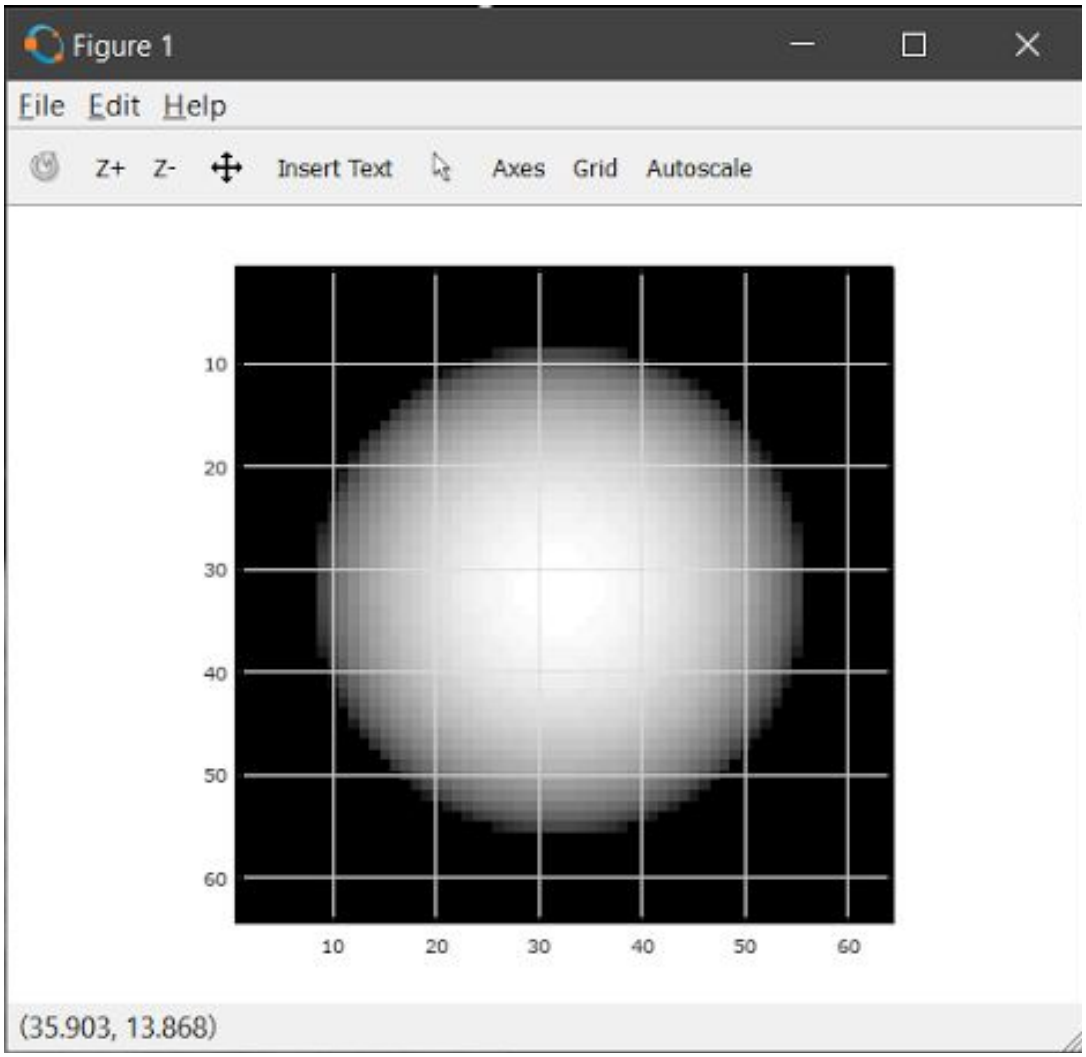
IT-524 COMPUTER VISION -ASSIGNMENT 1

Kunal Suthar
201401131

1. Consider a hemispherical surface z(x, y) = sqrt(r^2 - x^2 - y^2) + z0 . Select an image size 64X64. With the origin as the centre, and r=24 (say), find the surface normal representing the gradients p(x,y) and q(x,y) analytically. Assume the surface to be Lambertian. The object is illuminated by a point light source with (ps,qs) = (0,0) . Generate the corresponding image.

```
r = 24;
gridsize = 64;
E = zeros(gridsize, gridsize);
mid = gridsize/2 ;

    for x = -31:32
        for y = -31:32
            if ((x)^2 + (y)^2) <= r^2
                numerator = 1.0*( sqrt(r^2 - (x)^2 - (y)^2));
                denominator =  1.0*r;

                answer = numerator/denominator;
                if answer<=0
                    answer = 0;
                end
                E((mid+x),(mid+y)) = answer;
            end
        end
    end
imshow(E);
disp(E);
```

Generated Image:



Observations: This is 64*64 grid, and the source light direction of (ps,qs)=(0,0) is parallel to the optical axis passing through the center of the sphere, which is equivalent to (32,32) in the grid. That is why the center point (32,32) has the maximum intensity.

Image for source position
(0.8389, 0.7193, 1)
(0.5773, 0.6363, 1)
(0.3638, 0.5865, 1)
(0.1763, 0.5596, 1)
(−0.1763, −0.5596, 1)
(−0.3638, −0.5865, 1)
(−0.5773, −0.6363, 1)
(−0.8389, −0.7193, 1)

2. With these source positions, generate 8 different images considering the surface as sphere (see Assignment file). Consider these images as observations E_1 to E_8. For every (x,y) find p, q and rho using least squares method. Display these as images.Also find the MSE between estimated p, q maps and true p, q maps. You will get two MSE values. What is the conclusion?

```
ps_list=[0.8389;0.5773;0.3638;0.1763;-0.1763;-0.3638;-0.5773;-0.8389];
qs_list=[0.7193;0.6363;0.5865;0.5596;-0.5596;-0.5865;-0.6363;-0.7193];
for i = 1:8
    ps_list(i)=ps_list(i)/sqrt(ps_list(i)^2+qs_list(i)^2+1);
    qs_list(i)=qs_list(i)/sqrt(ps_list(i)^2+qs_list(i)^2+1);
end
%ps=ps/sqrt(ps^2+qs^2+1);
%qs=qs/sqrt(ps^2+qs^2+1);
r = 24;
gridsize = 64;
mat= zeros(gridsize, gridsize);
mid = gridsize/2 ;

E=zeros(gridsize,gridsize,8);

for i=1:8

    for x = -31:32
        for y = -31:32
            if ((x)^2 + (y)^2) <= r^2
                numerator = (((x)*ps_list(z)) + ((y) * qs_list(z)) + sqrt(r^2 - (x)^2 -
(y)^2));
                denominator =  1.0 * (r * (1 + ps_list(i)^2 + qs_list(i)^2));
```

```
                answer = numerator/denominator;
                if answer<=0
                    answer = 0;
                end
                mat((mid+x),(mid+y)) = answer;
            end
        end
    end
    figure(i), imshow(E)

end
```

Generated Images:

Image for source-1:

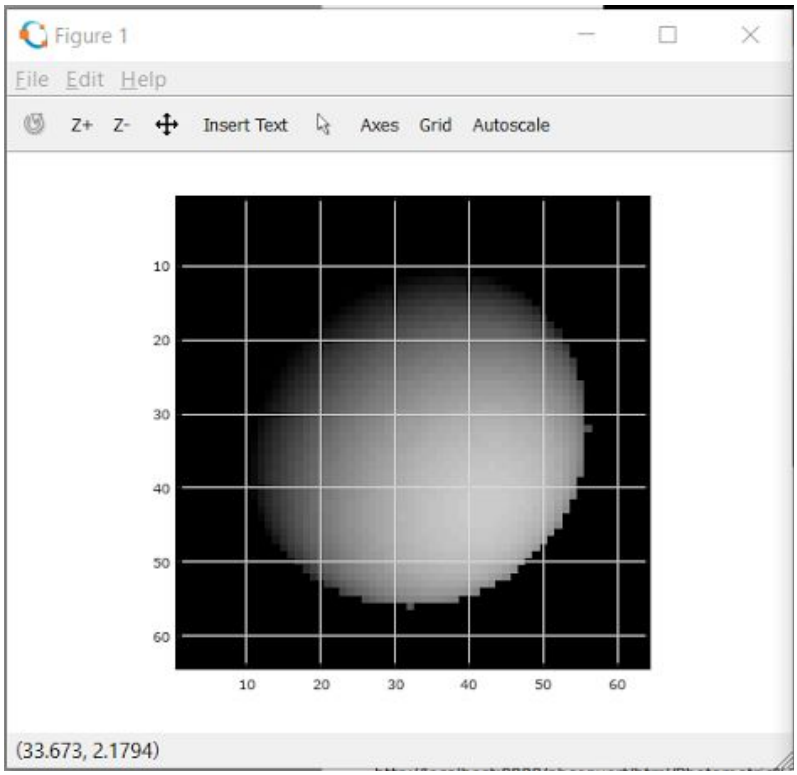


Image for source-2

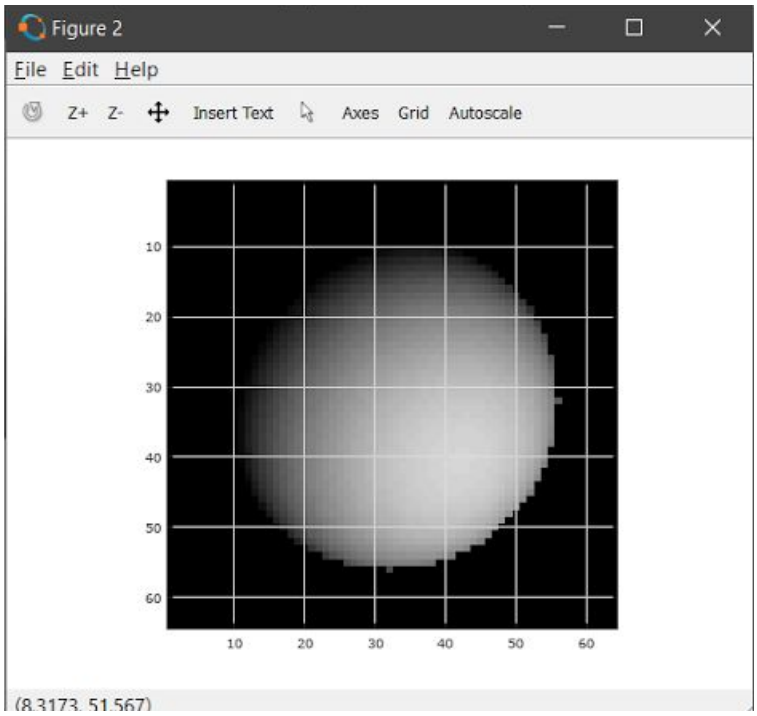


Image for source-3

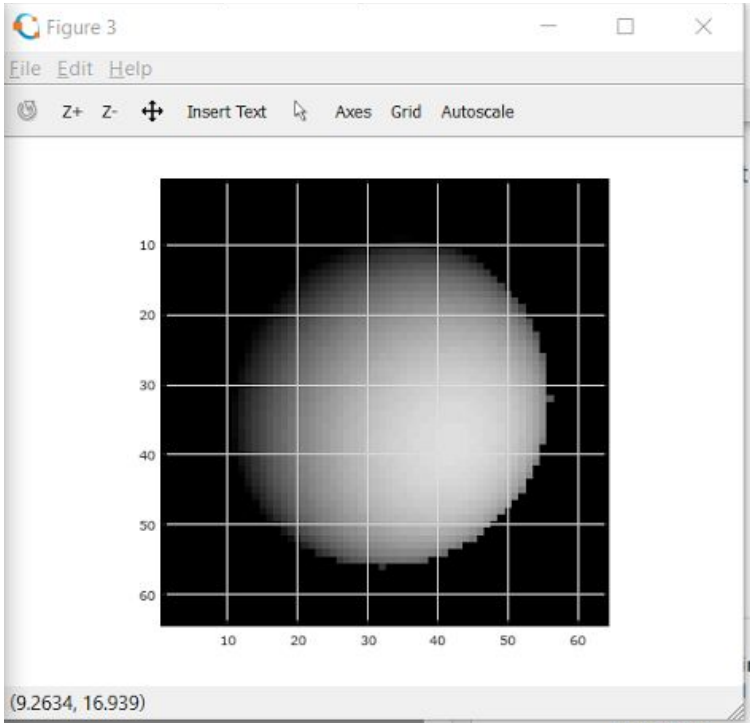


Image for source-4

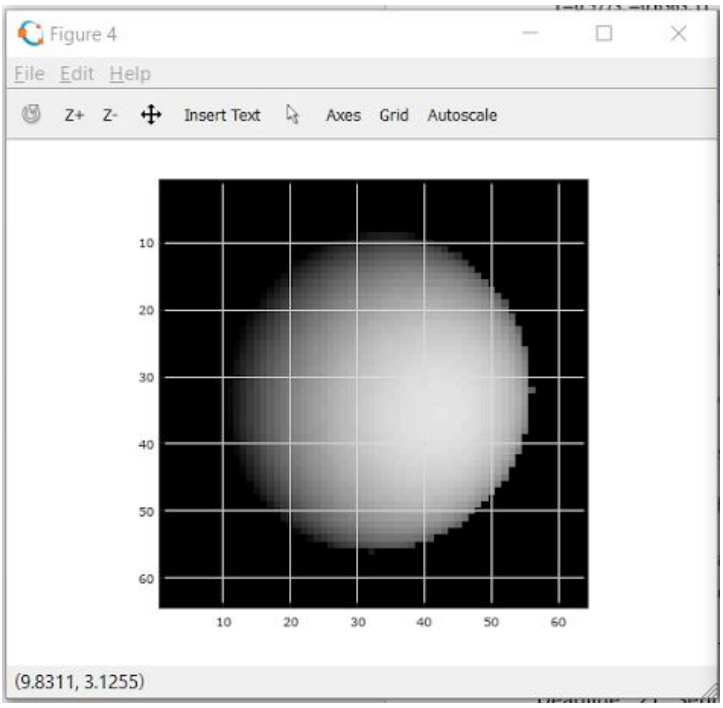


Image for source-5

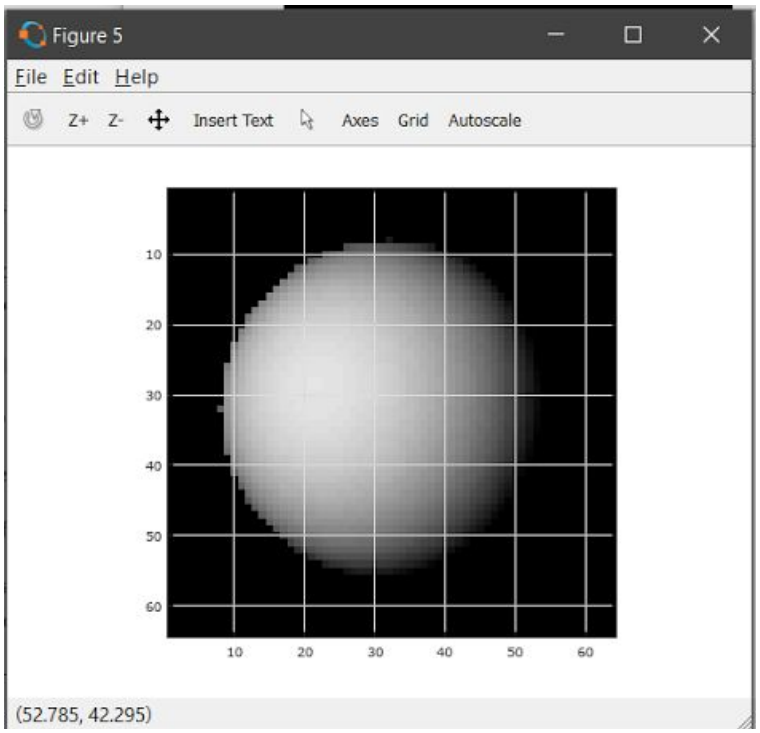


Image for source-6

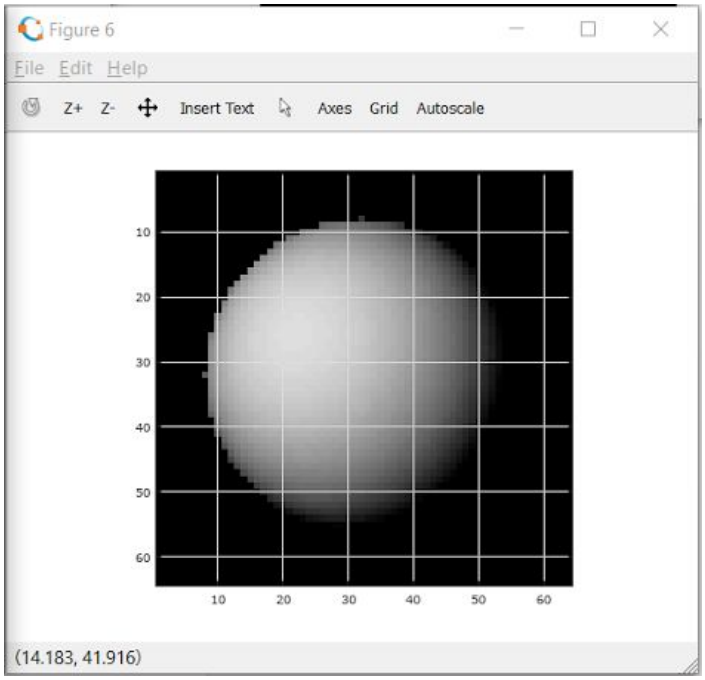


Image for source-7

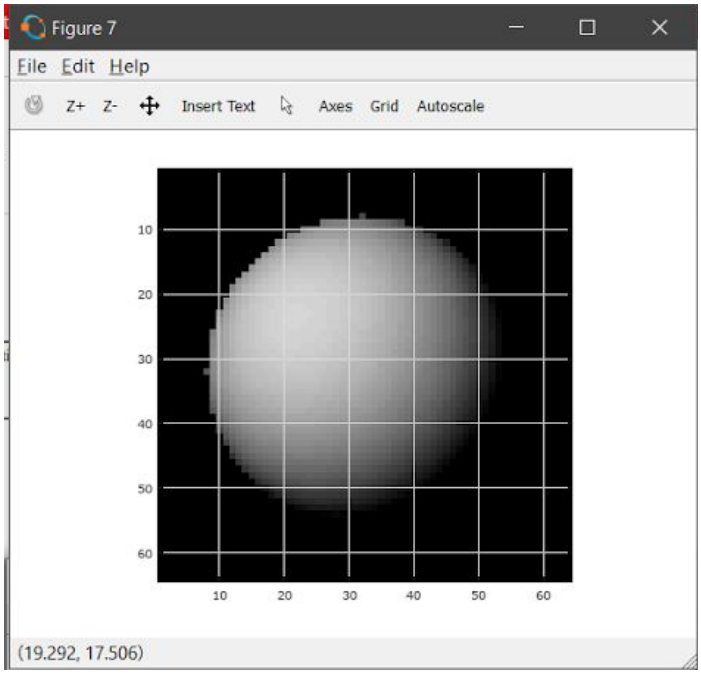
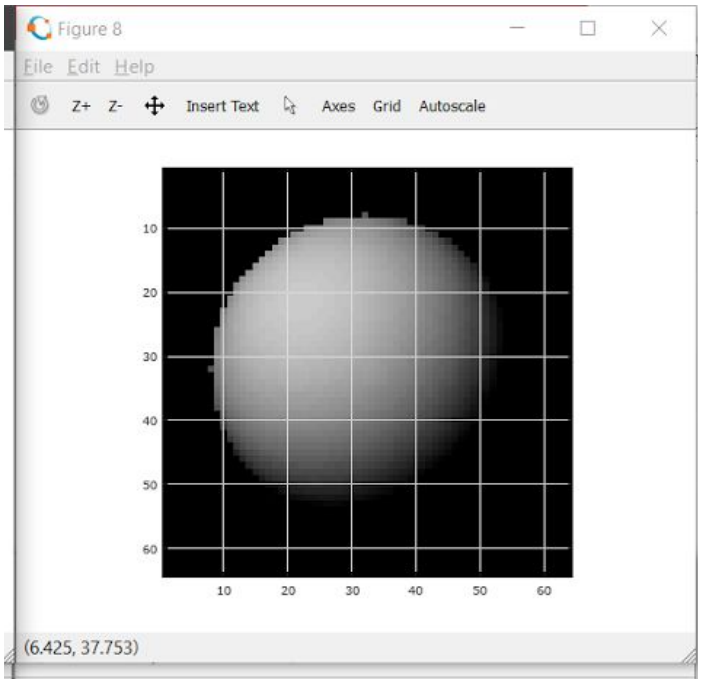


Image for source-8



Finding true values of p and q:

Code Snippet:

```
true_q = zeros(gridsize, gridsize);
    true_p = zeros(gridsize, gridsize);
    for x=-31:32
        for y= -31:32
            if(x^2+y^2<=r^2)
                z = sqrt(r^2 - x^2 - y^2);
            end
            if sqrt(r^2 - (x)^2 - (y)^2) >0
                true_p(center + x, center + y) = ( x / sqrt(r^2 - (x)^2 - (y)^2)) ;
                true_q(center + x, center + y) = ( y / sqrt(r^2 - (x)^2 - (y)^2)) ;
            end
        end
    end
 %   disp(true_p)
 figure(4),imshow(true_p);
 figure(5),imshow(true_q);
```

True P:

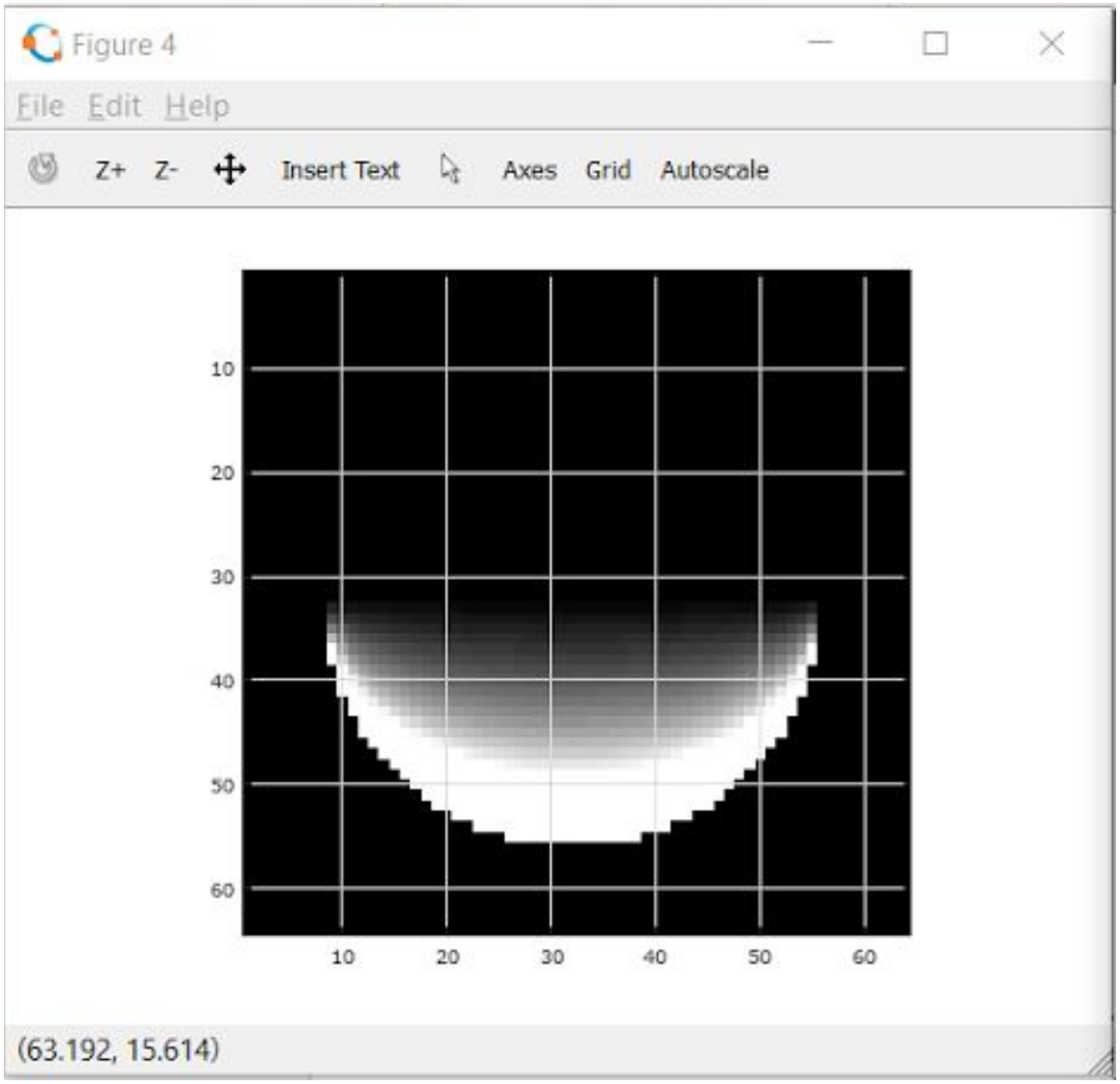


Figure 4
File Edit Help
Z+ Z- Insert Text Axes Grid Autoscale
10
20
30
40
50
60
10 20 30 40 50 60
(63.192, 15.614)

True Q:

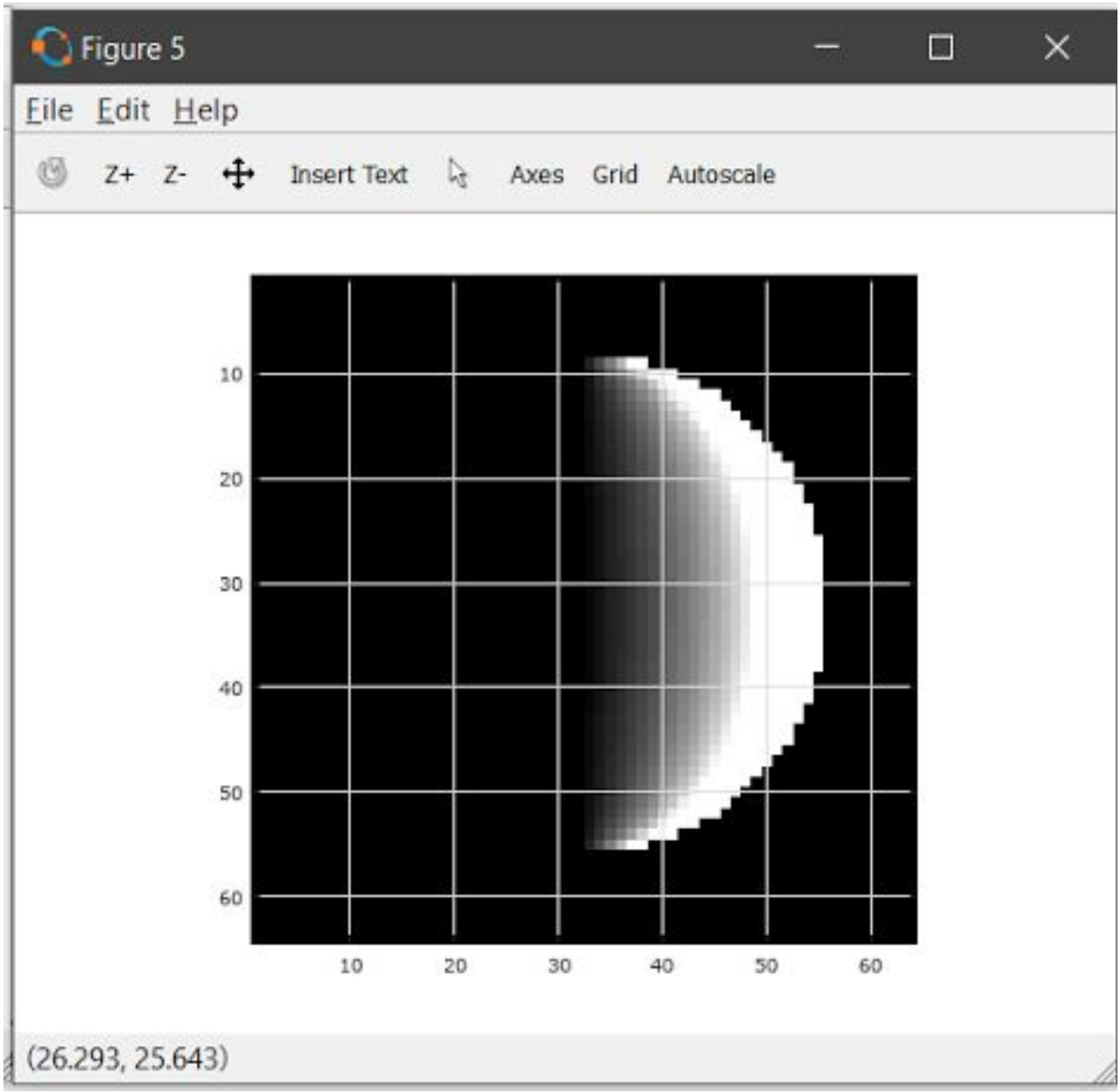


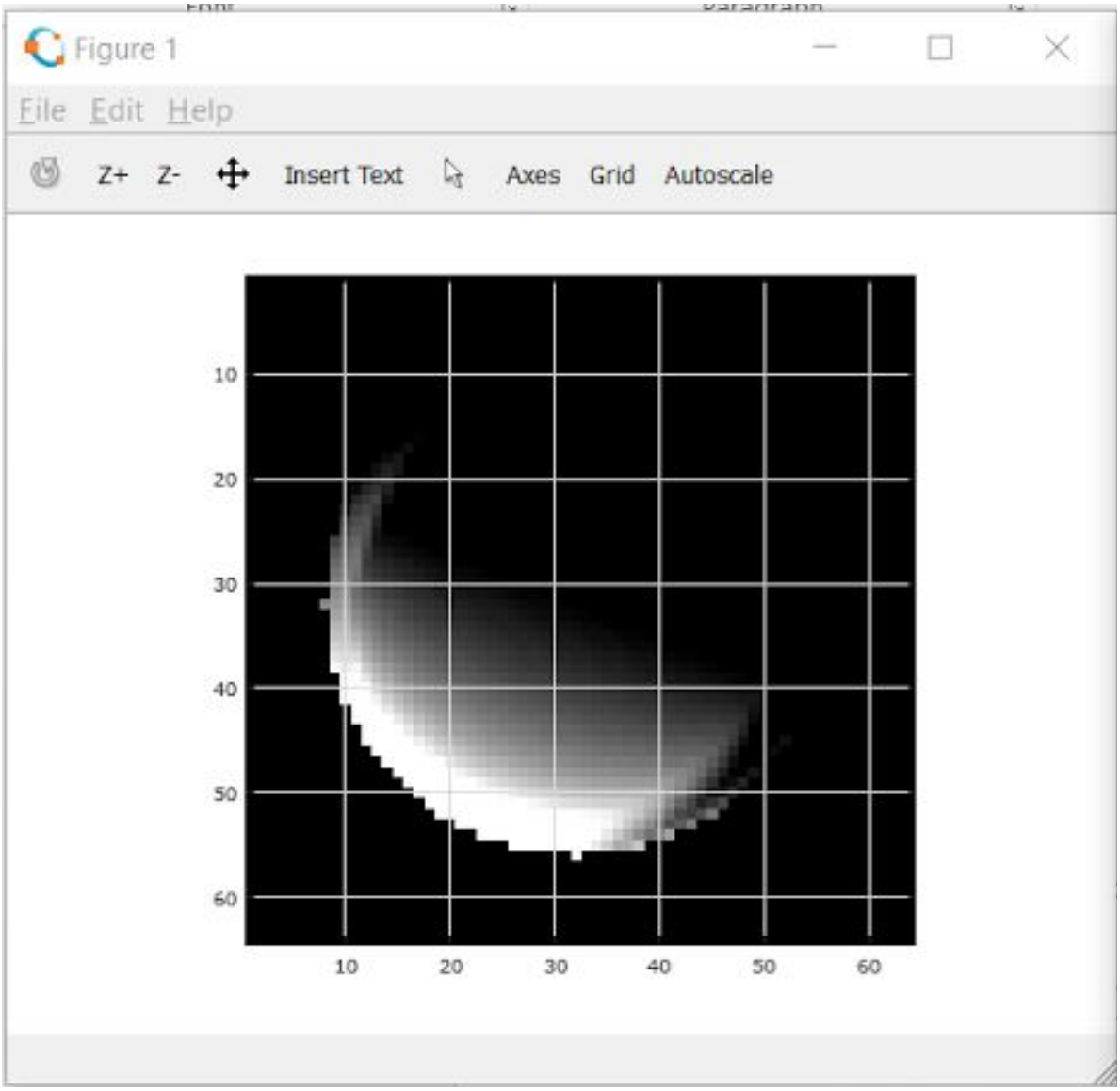
Figure 5
File Edit Help
Z+ Z- Insert Text Axes Grid Autoscale
10
20
30
40
50
60
10 20 30 40 50 60
(26.293, 25.643)

Finding P,Q, and albedo factor using Pseudo-Inverse Technique:

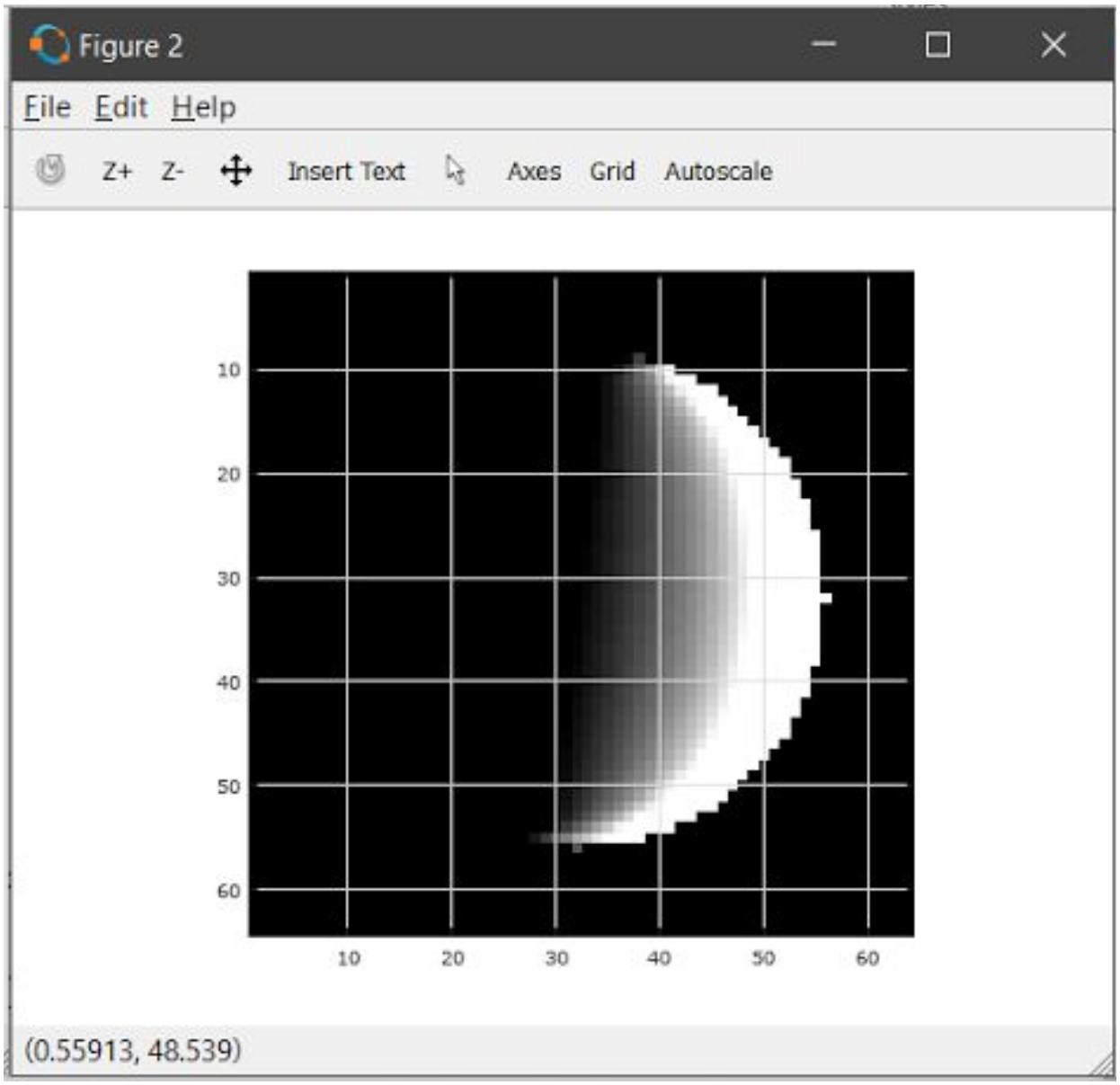
Code Snippet:

```
source_mat=[-1.*(ps),-1.*(qs),ones(8,1)];


for i = 1:8
    source_mat(i,3)=source_mat(i,3)/sqrt(source_mat(i,1)^2+source_mat(i,2)^2+1);
end


    for x=-31:32
        for y= -31:32
                    temp1 = reshape(E(center+x,center+y,:), [8,1]);
                    temp=pinv(source_mat)*(temp1);
                    estimated_p(center+x,center+y)=-1*temp(1)/temp(3);
                    estimated_q(center+x,center+y)=-1*temp(2)/temp(3);

estimated_albedo(center+x,center+y)=temp(3)/sqrt(estimated_p((center+x),(center+y))^2+estima
ted_q((center+x),(center+y))^2+1);
                     if isnan(estimated_p(center+x,center+y))
                        estimated_p(center+x,center+y)=0;
                    end
                    if isnan(estimated_q(center+x,center+y))
                        estimated_q(center+x,center+y)=0;
                    end
                    if isnan(estimated_albedo(center+x,center+y))
                        estimated_albedo(center+x,center+y)=0;
                    end
        end
    end



  disp(estimated_p)
  figure(1), imshow(estimated_p)
  figure(2), imshow(estimated_q)
  figure(3), imshow(estimated_albedo)
```

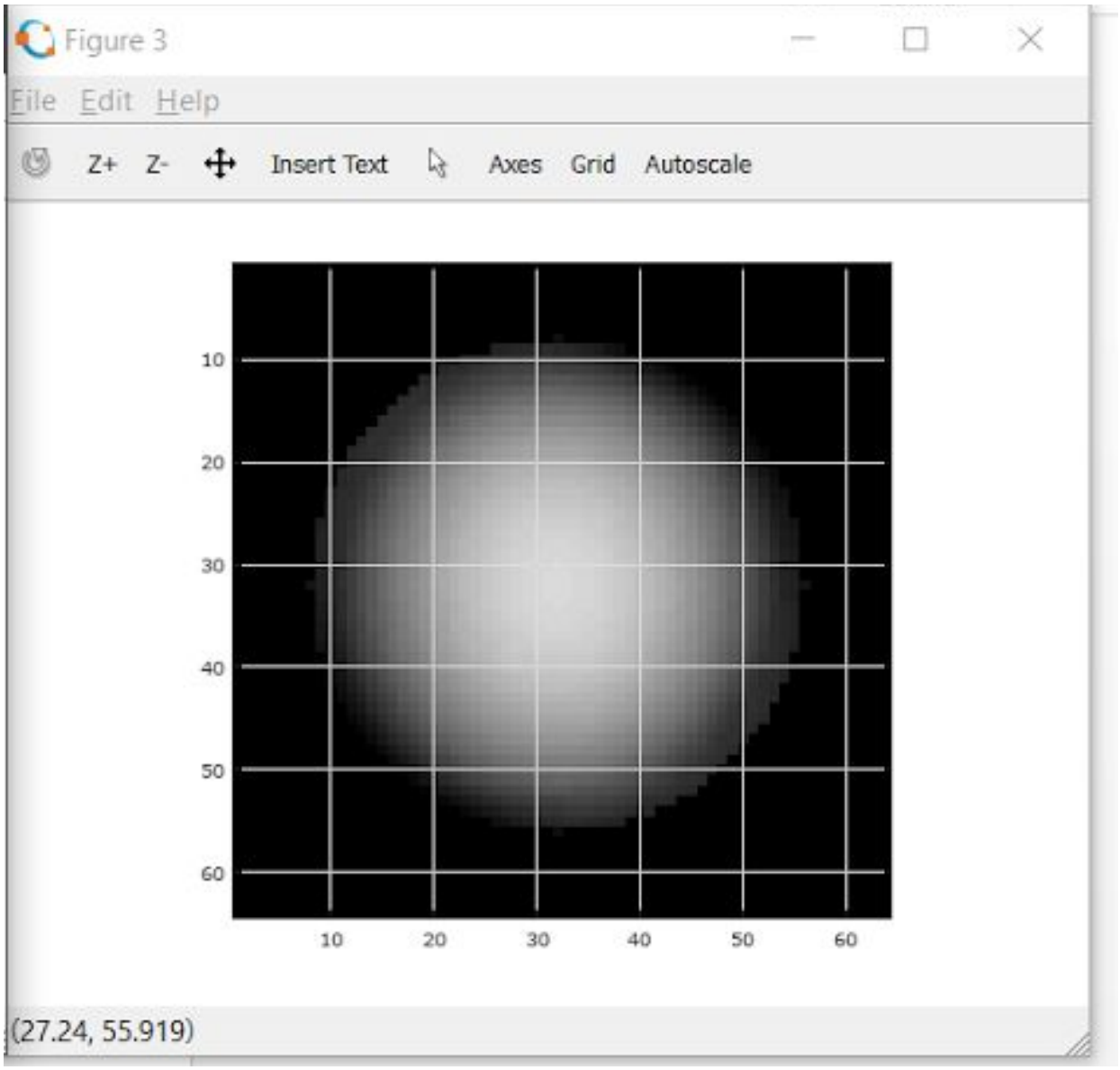
Constructed P using Pseudo-Inverse:



Constructed Q using Pseudo-Inverse:



Constructed Albedo-Factor using Pseudo-Inverse:

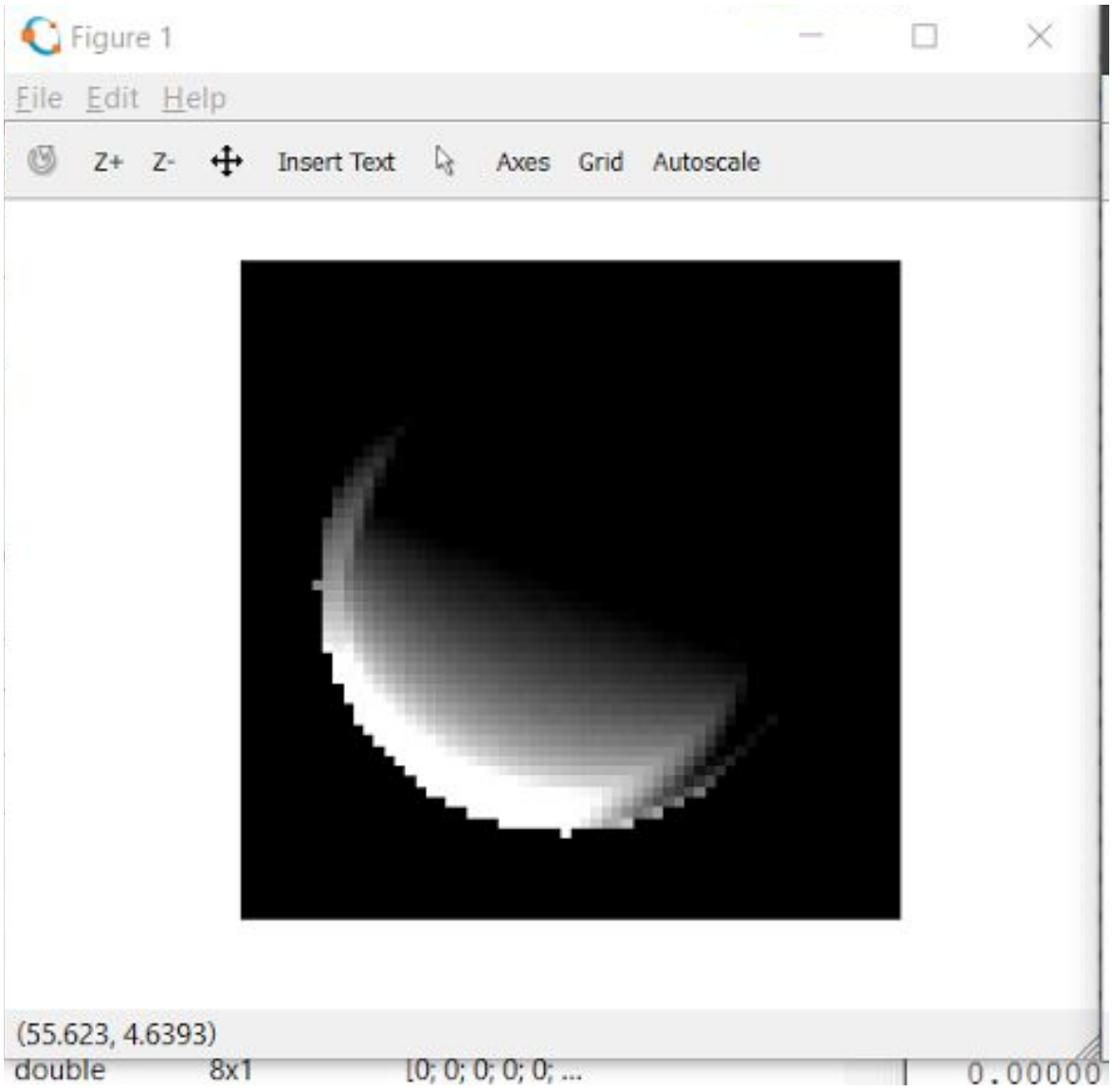


Finding P,Q, and albedo factor using SVD Technique:

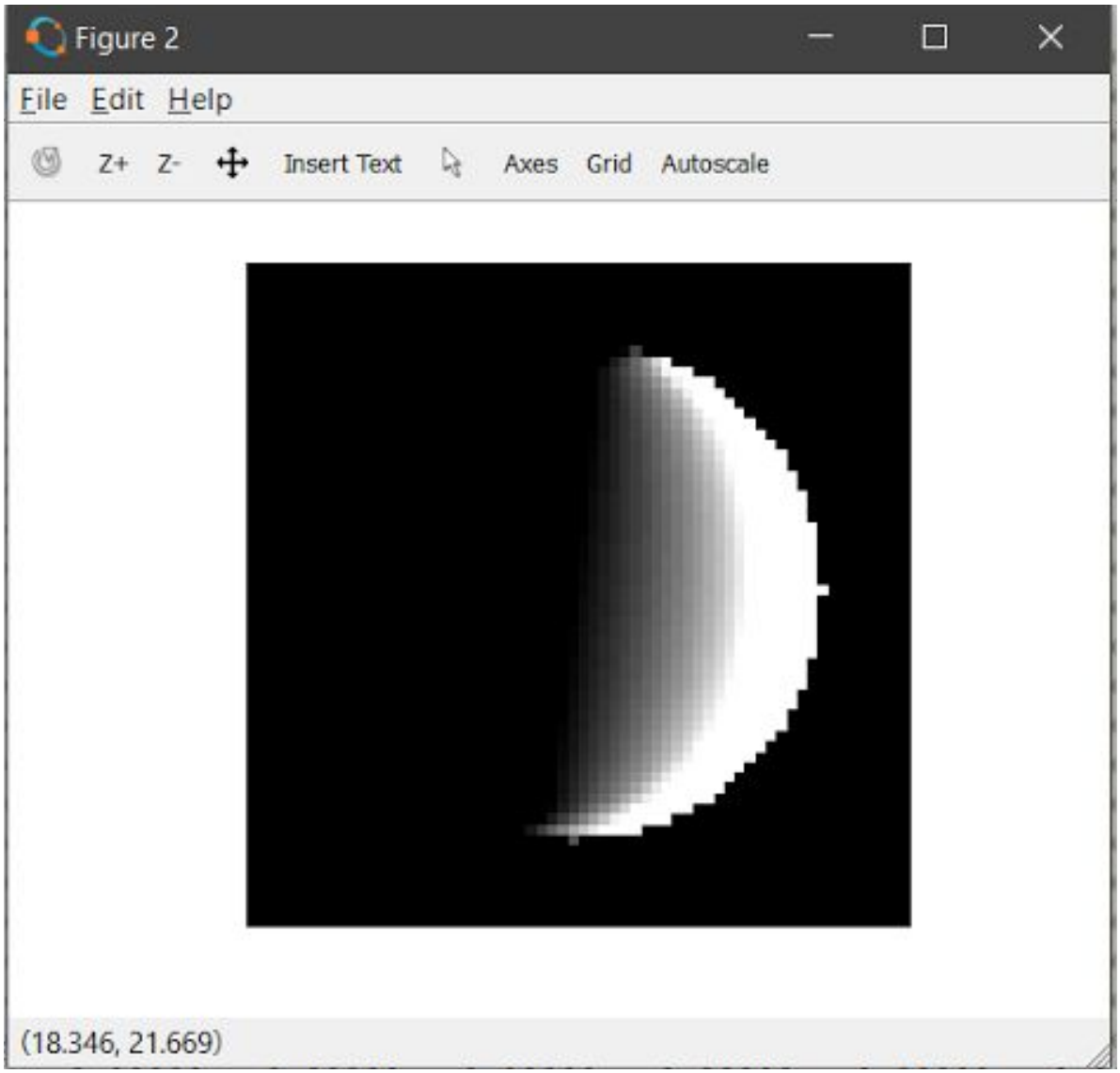
Code Snippet:

```
% SVD

    for x=-31:32
        for y= -31:32
                    temp1 = reshape(intensity_mat(center+x,center+y,:), [8,1]);
                     [U S V] = svd(source_mat, 0);
                     temp= V *inv(S)*(U'*(temp1));
                     estimated_p(center+x,center+y)=-1*temp(1)/temp(3);
                     estimated_q(center+x,center+y)=-1*temp(2)/temp(3);

estimated_albedo(center+x,center+y)=temp(3)/sqrt(estimated_p((center+x),(center+y))^2+estima
ted_q((center+x),(center+y))^2+1);
                     if isnan(estimated_p(center+x,center+y))
                         estimated_p(center+x,center+y)=0;
                     end
                     if isnan(estimated_q(center+x,center+y))
                         estimated_q(center+x,center+y)=0;
                     end
                     if isnan(estimated_albedo(center+x,center+y))
                         estimated_albedo(center+x,center+y)=0;
                     end

        end
    end

    figure(1), imshow(estimated_p)
    figure(2), imshow(estimated_q)
    figure(3), imshow(estimated_albedo)
```

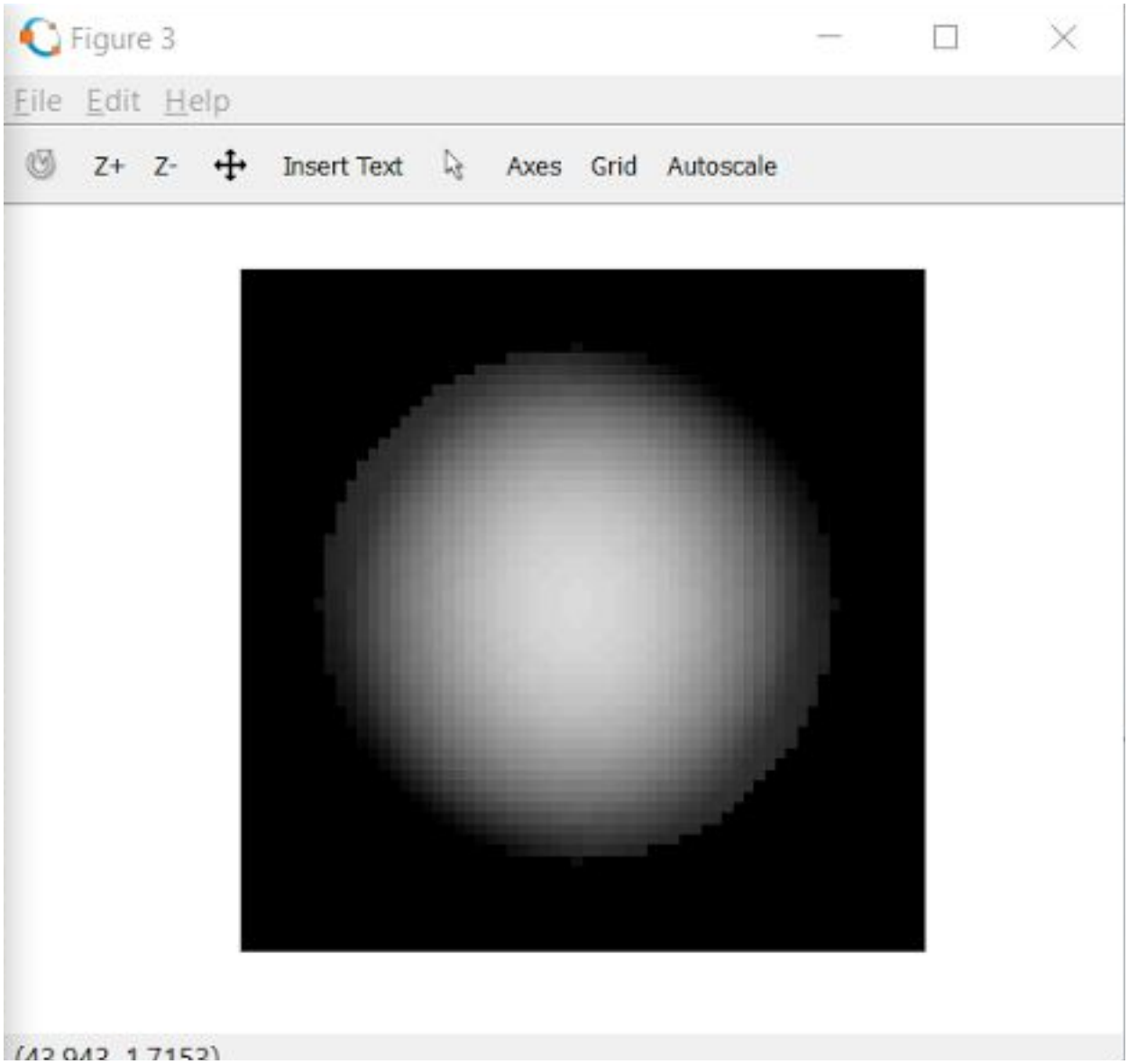
Constructed P using SVD:



Constructed Q using SVD:



Constructed Albedo-Factor using SVD:



- **ERRORS FROM BOTH THE TECHNIQUES:**

Code Snippet for both techniques:

```
P_Error = abs(true_p-estimated_p).^2;
   MSE_p = sum(P_Error(:))/numel(true_p);
   disp(MSE_p)

   Q_Error = abs(true_q-estimated_q).^2;
   MSE_q = sum(Q_Error(:))/numel(true_q);
   disp(MSE_q)
```

1. **Mean squared error for P using Pseudo-Inverse: 2.9737**
2. **Mean squared error for Q using Pseudo-Inverse: 2.7740**
3. **Mean squared error for P using SVD: 2.9737**
4. **Mean squared error for Q using SVD: 2.7740**