Basic Image Import, Processing, and Export

黃世強 (Sai-Keung Wong)

National Chiao Tung University, Taiwan

About an image

x

y

An image consists

of a set of pixels.

Each pixel has a coordinate (x,y)

x represents the column index.

y represents the row index.

The upper left corner is (1,1).

The pixels are organized in a rectangular shape.

Step 1: Read and Display an Image

```
I = imread('pout.tif');
imshow(I)
```

Step 2: Check How the Image Appears in the Workspace

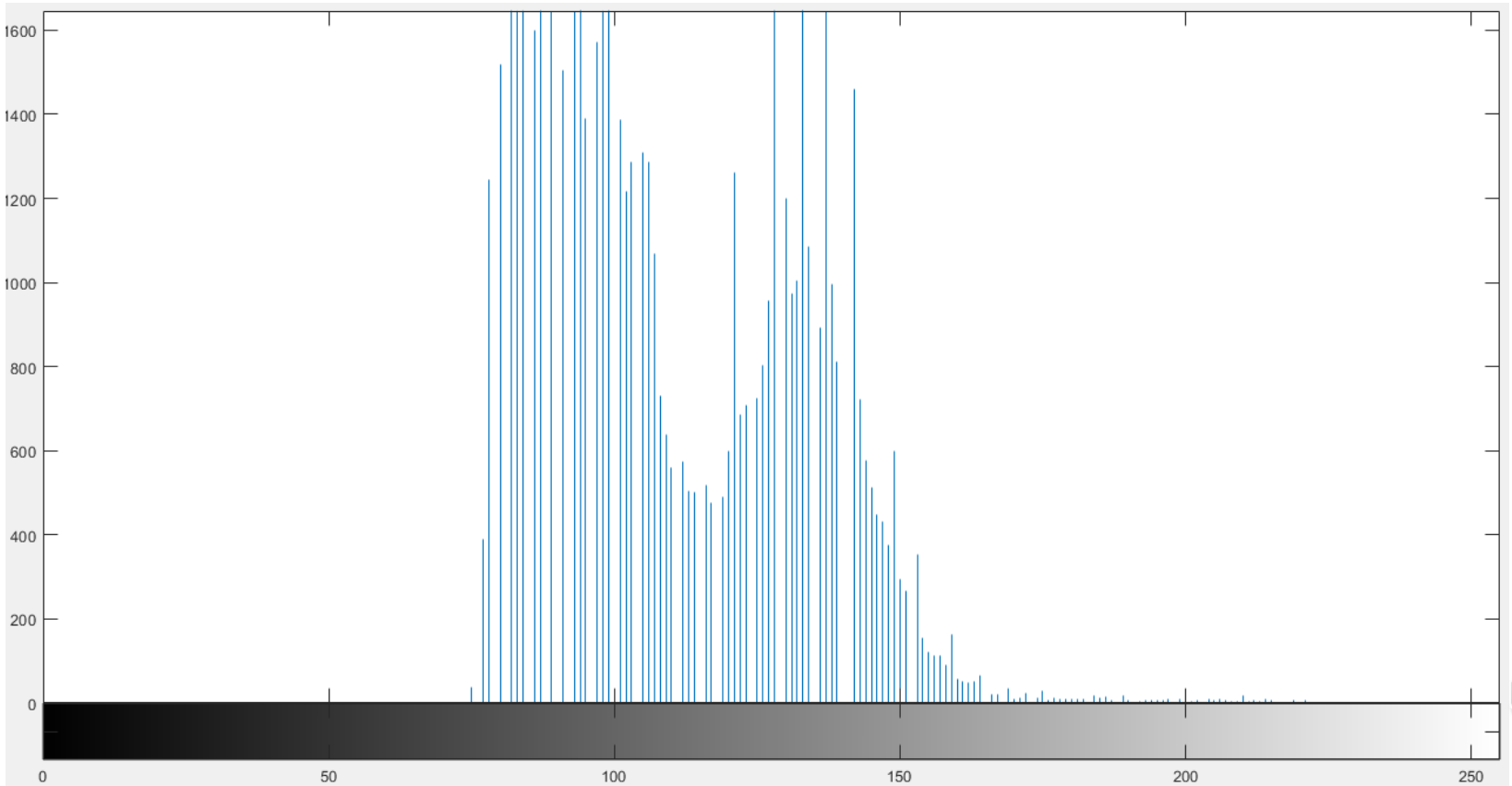
whos I

Name	Size	Bytes	Class	Attributes
I	291x240	69840	uint8	

Step 3: Improve Image Contrast

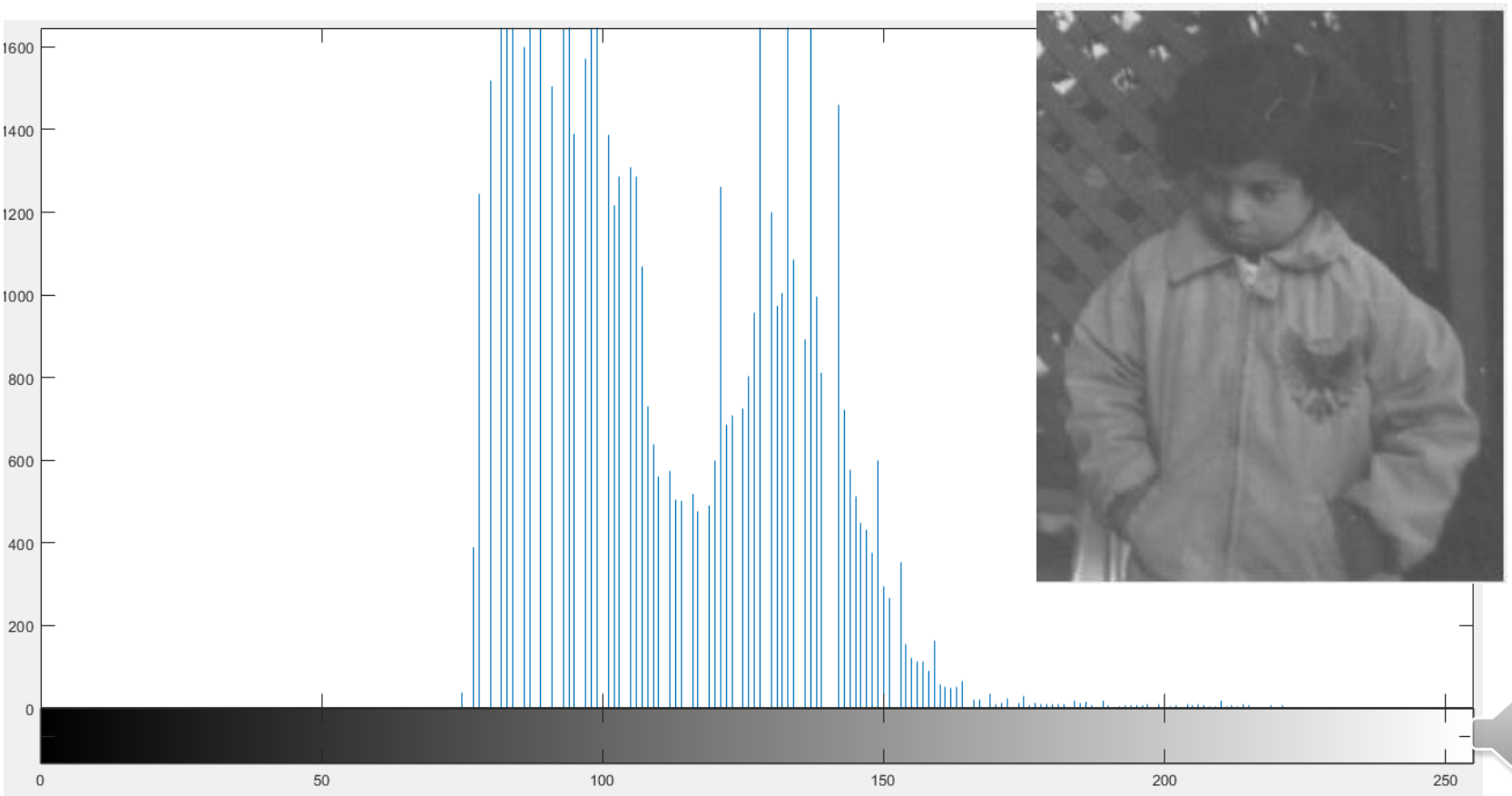
figure

imhist(I) %create a histogram



Step 3: Improve Image Contrast

Contrast is the difference in luminance or color that makes an image distinguishable.



Step 3: Improve Image Contrast

Contrast is the difference in luminance or color that makes an image distinguishable.



Step 3: Improve Image Contrast

```
I2 = histeq(I);
figure
imshow(I2)
```

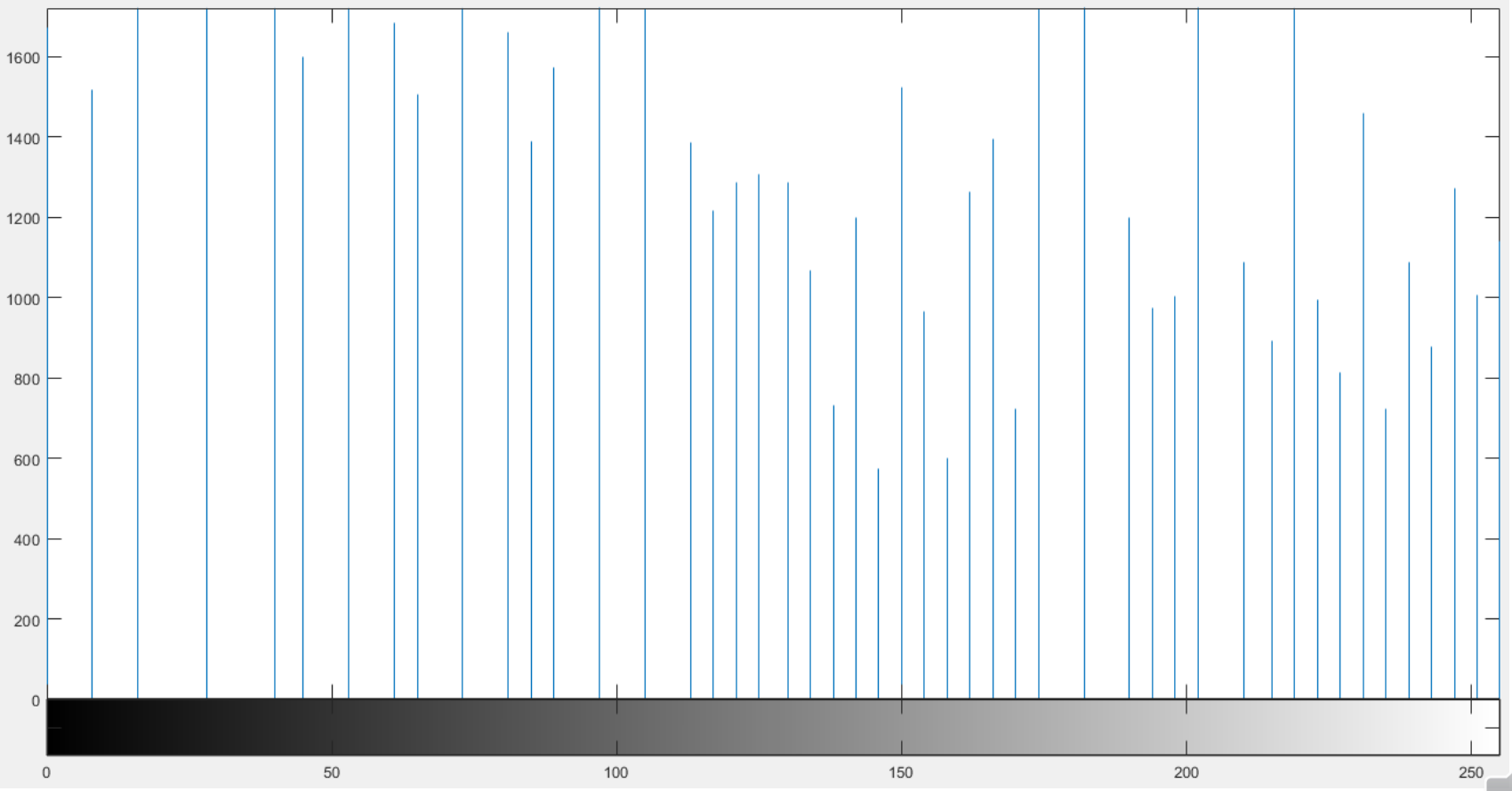
I

I2

histeq: Histogram equalization spreads the intensity values over the full range of the image.

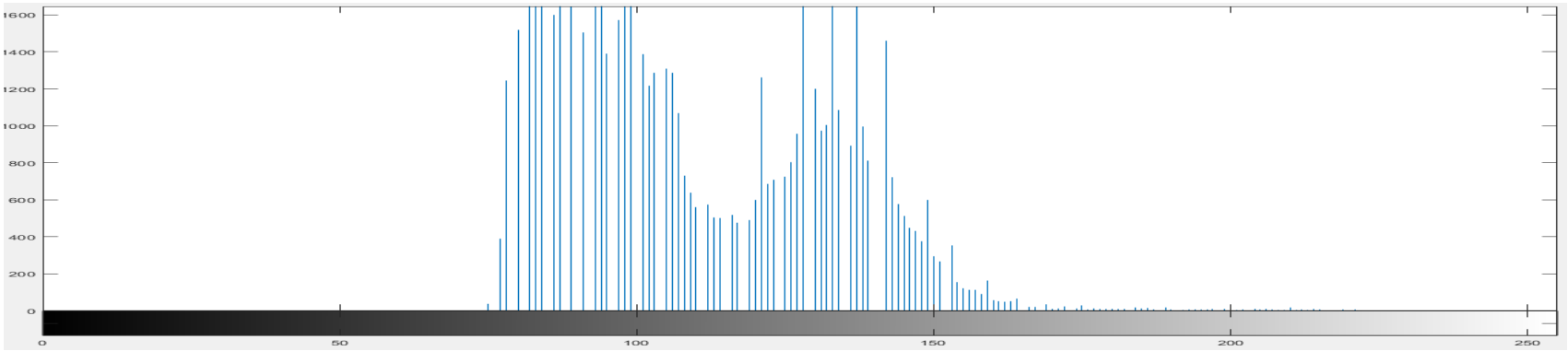
Step 3: Improve Image Contrast

imhist(I2) %create a histogram

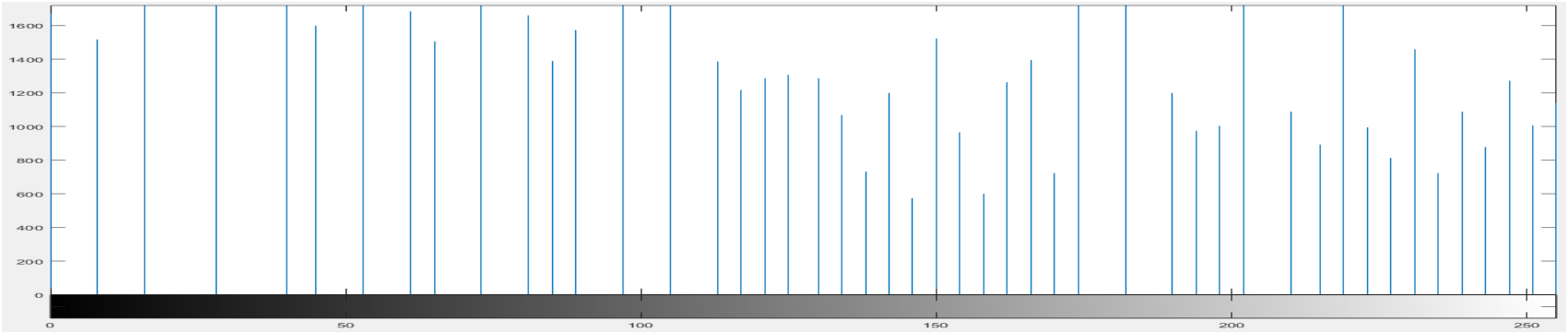


Step 3: Improve Image Contrast

imhist(I)



imhist(I2)



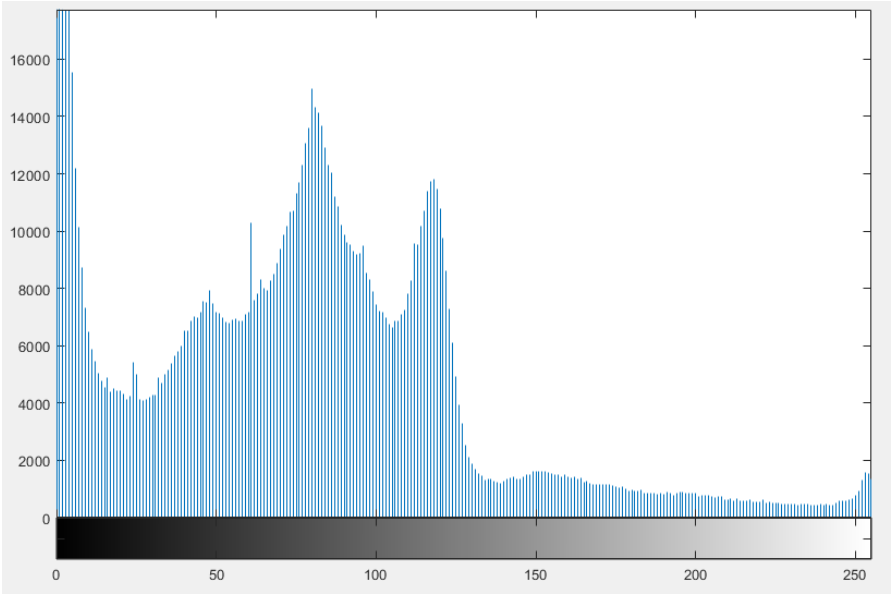
Step 4: Write the Adjusted Image to a Disk File

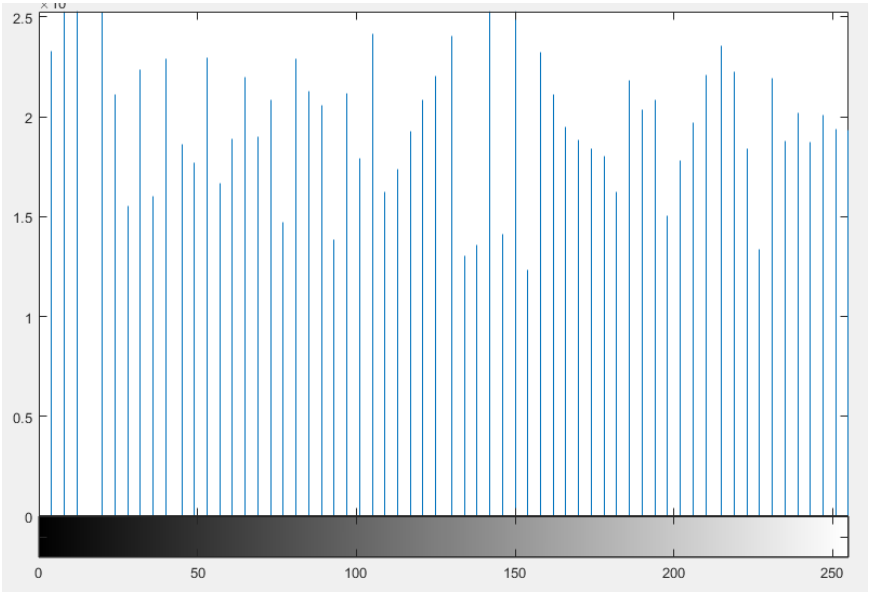
```
imwrite (I2, 'pout2.png');


// graphics formats, e.g., png, jpg, bmp, tif, …
```

Step 5: Check the Contents of the File

```
imfinfo('pout.tif')

ans =

struct with fields:

Filename: 'C:\Program Files\MATLAB\R2017b\toolbox\images\imdata\pout.tif'
FileModDate: '13-Apr-2015 01:23:12'
FileSize: 69296
Format: 'tif'
FormatVersion: []
Width: 240
Height: 291
BitDepth: 8
ColorType: 'grayscale'
```

Example


16000
14000
12000
10000
8000
6000
4000
2000
0
0
50
100
150
200
250


2.5
2
1.5
1
0.5
0
0
50
100
150
200
250

Functions

```
B = rgb2gray(A);  % grayscale image
imwrite (B, 'image2.png');  % write an image to a file
C = imresize(B,1.5); %resize an image
C = imresize(B,[300,150]); % specify the dimension
C = imrotate(B,35);  % rotate an image
imtool(B)      %use image viewer to view an image
D = imcrop(A,[160 140 110 180]);  % crop an image
A(2,15,:)      %get red, green, blue of a pixel (2,15)
```

Functions
Exercises

```
B = rgb2gray(A);  % grayscale image
```

Functions

size(I) % get the dimension of an image I

ans =

  525 791 3

  width x height x color_channels

rgb2gray
grayscale image

I = imread(…)

B = rgb2gray(I);

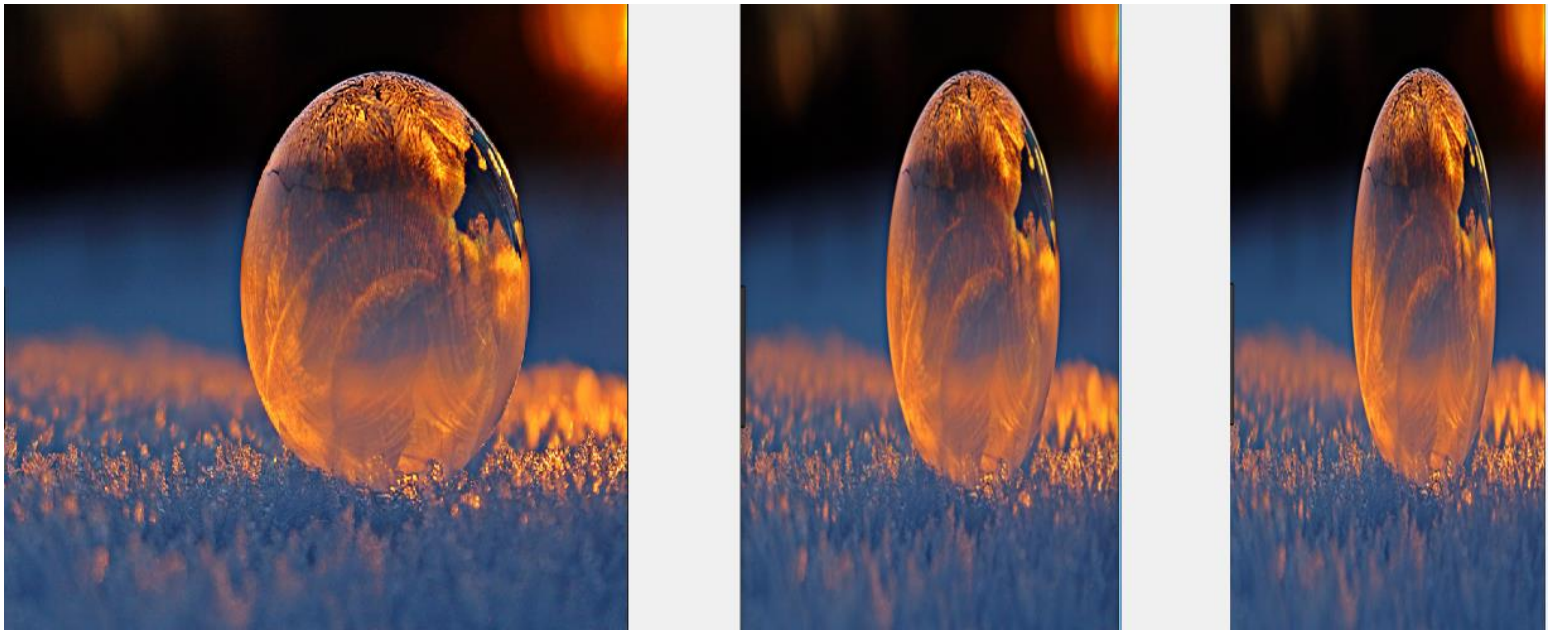
figure

imshow(I)

Imshow(B)

imresize: resize an image

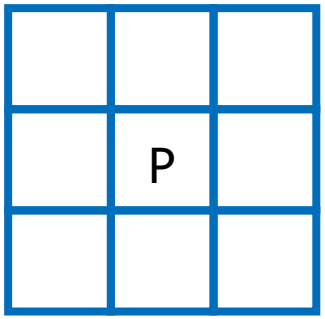
```
C = imresize(I, [500, 500], 'nearest');
D = imresize(I, [500, 300], 'bilinear');
E = imresize(I, [500, 250], 'bicubic');
figure, imshow(C), figure, imshow(D), figure, imshow(E)
```



imresize: resize an image

imresize(A, [NUMROWS NUMCOLS], METHOD)

METHOD can be a string naming a general interpolation method:



'nearest' - nearest-neighbor interpolation

'bilinear' - bilinear interpolation

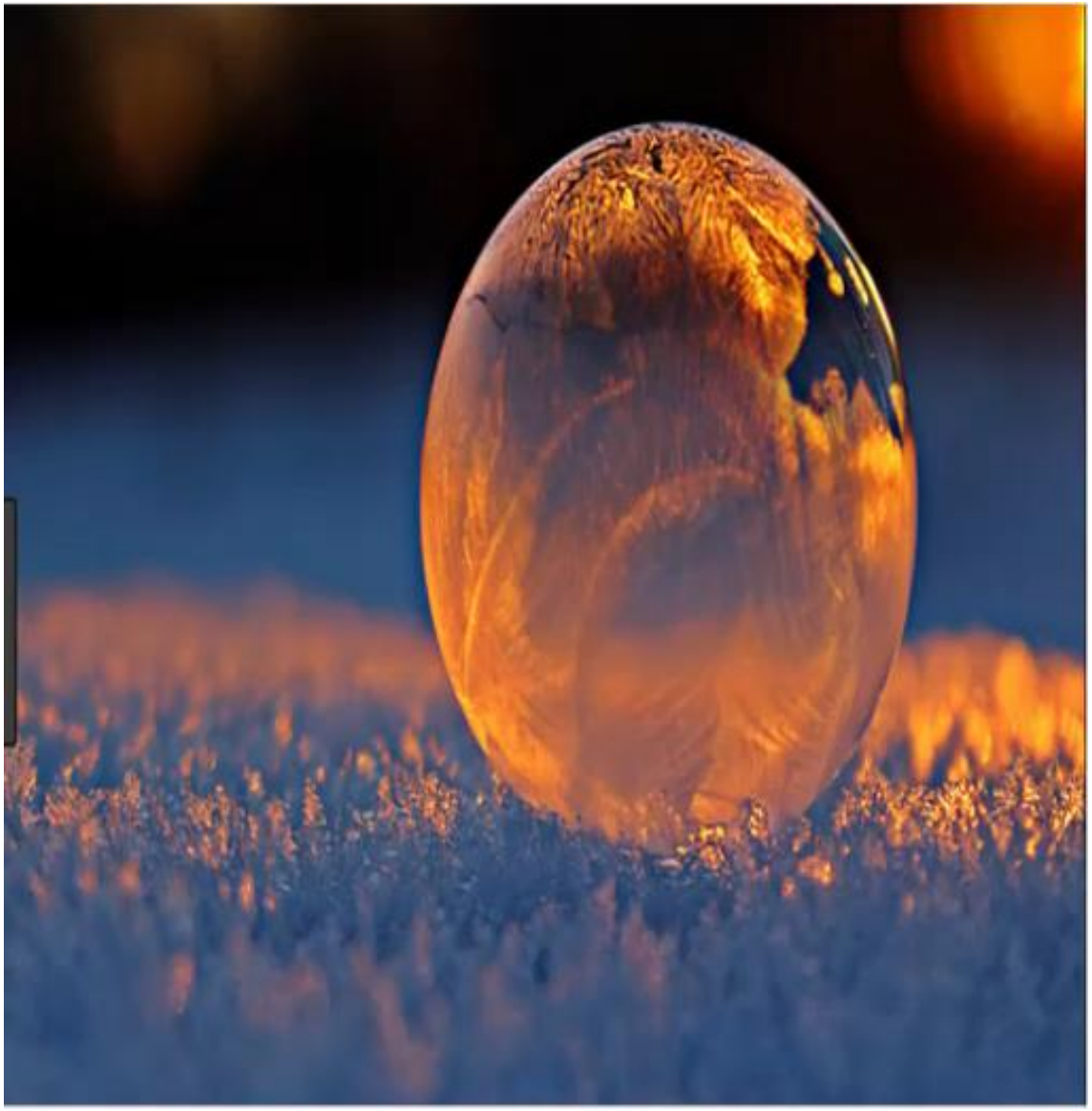
'bicubic' - cubic interpolation; the default method

imrotate: rotate an image

imrotate(B,n); rotate the image with n degrees

Write a program to rotate an image from 0 degree to 360 degrees, and show the animation.

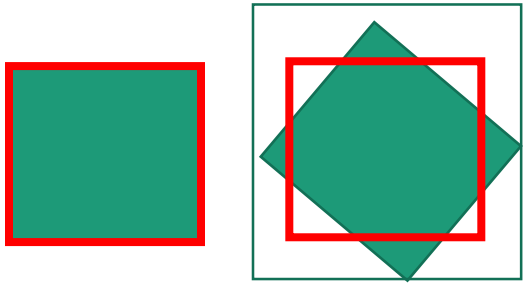
Imrotate: Animation Demo



imrotate: rotate an image

```
close all; h1 = figure(1)
set(h1,'Position',[10 10 500 500]);  // (10,10): position of upper left corner
I1 = imresize(I, [ 500 500]);
imshow(I1);
input('Press Enter to start...');
i = 0;
while i <= 360
   F = imrotate(I,i*1);
   I1 = imresize(F, [ 500 500]);  imshow(I1);
   i = i + 1; pause(0.0333)
end
```


imrotate: rotate an image



```
close all; h1 = figure(1)
set(h1,'Position',[10 10 500 500]);
I1 = imresize(I, [ 500 500]);
imshow(I1);
input('Press Enter to start...');
i = 0;
while i <= 360
   I1 = imrotate(I,i*1);
%   I1 = imresize(F, [ 500 500]);  % no resize
   imshow(I1);
   i = i + 1; pause(0.0333)
end
```

MATLAB resizes the figure window from time to time.

imrotate: rotate an image

```
close all; h1 = figure(1)
set(h1,'Position',[10 10 500 500]);
I1 = imresize(I, [ 500 500]);
imshow(I1);
input('Press Enter to start...');
i = 0;
while i <= 360
   F = imrotate(I,i*1);
   I1 = imresize(F, [ 500 500]);  imshow(I1);
  set(h1,'Position',[10 10 500 500]);          % heavy blinking
   i = i + 1; pause(0.0333)
end
```

Example
Modify an image

```
x = size(I);
I2 = I;
for i = [1:x(1)]
    f = i/x(1);
    I2(i,:,:) = I(i,:,:)*f;
end
imshow(I2);
```

Row index

column index

color channels

Color image. 3 channels (or components): red, green, blue

Example

```
x = size(I);
I2 = I;
for i = [1:x(1)]
    f = i/x(1);
    for j = [1:x(2)]
        I2(i,j,1) = I(i,j,1);
        I2(i,j,2) = I(i,j,2);
        I2(i,j,3) = I(i,j,3)*f;
    end
end
imshow(I2);
```

Example

```
x = size(I);
I2 = I;
for i = [1:x(1)]
    f = i/x(1);
    for j = [1:x(2)]
        v = sum( I2(i,j,:) )/3;
        I2(i,j,1) = v;
        I2(i,j,2) = v;
        I2(i,j,3) = v;
    end
end
imshow(I2);
```

Example

```
x = size(I);
I2 = I;
for i = [1:x(1)]
    f = i/x(1);
    for j = [1:x(2)]
        v = sum( I2(i,j,:) )/3;
        I2( i,j,: ) = v;
    end
end
imshow(I2);
```

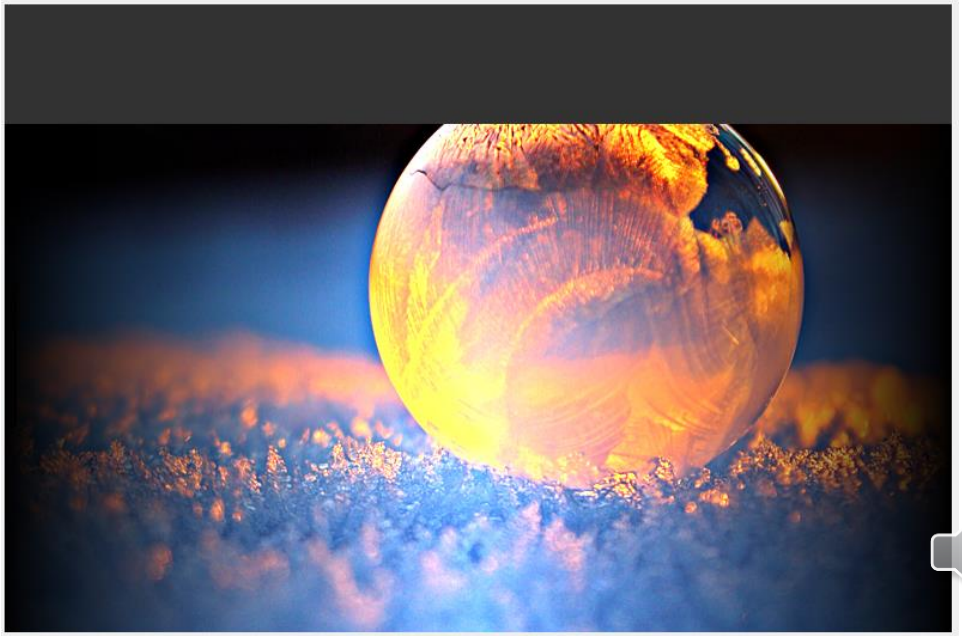
Example

```
x = size(I);
I2 = I;
for i = [1:x(2)]
    if (i < x(2)/2)
        f = i/x(2)*5;
    else
        f = 5-i/x(2)*5;
    end
    I2(:,i,:) = I(:,i,:)*f;
end
imshow(I2);
% 0->2.5->0
```

Example

I2(1:100, :, :) = 50;

imshow(I2);



Example

I2(:, 1:100, :) = 50;

imshow(I2);

colon operator

Image Modification

```
I = imread('tmp.png');
K = I;
K(:) = 0;                          %black
figure, imshow(I);
figure, imshow(K);
```

Image Modification

```
I = imread('tmp.png');
K = I;
K(:) = 128;                    %gray. Need to know data type
figure, imshow(I);
figure, imshow(K);
```

Image Modification

```
I = imread('tmp.png');
K = ones(size(I), 'double');
figure, imshow(I);
figure, imshow(K);
```

Image Modification

```
I = imread('tmp.png'); close all;
K = ones(size(I), 'double'); s = size(K); i = 1;
while i <= s(2)
    K(:,i,1:3) = i/s(2);     i = i + 1;
end
figure, imshow(K);
```

Image Modification

```
I = imread('tmp.png'); close all;
K = ones(size(I), 'double'); s = size(K); i = 1;
while i <= s(2)
    K(:,i,1:3) = i/s(2);     i = i + 1;
end
figure, imshow(K.*K);
```

Image Modification

```
I = imread('tmp.png'); close all;
K = ones(size(I), 'double'); s = size(K); i = 1;
while i <= s(2)
    K(:,i,1:3) = i/s(2);     i = i + 1;
end
figure, imshow(K.*K.*K);          // intensity value in [0,1]
```

Image Modification

```
close all; K = ones(size(I), 'double'); s = size(K);
Id = im2double(I); i = 1;
while i <= s(2)                    % s(2): number of columns
    K(:,i,1:3) = i/s(2); i = i + 1;
end
figure, imshow(Id.*K); % K is a mask
```

Image Modification

```
close all; s = size(K); K = ones(size(I), 'double');
Id = im2double(I); i = 1;
while i <= s(2)
    K(:,i,1) = 1.5*i/s(2); K(:,i,2:3) = i/s(2); i = i + 1;
end
figure, imshow(Id.*K); // element-wise multiplication
```

Image Modification

```
close all; s = size(K); K = ones(size(I), 'double');
Id = im2double(I); i = 1;
while i <= s(2)
    K(:,i,1) = 2*i/s(2); K(:,i,2:3) = i/s(2); i = i + 1;
end
figure, imshow(Id.*K);
```

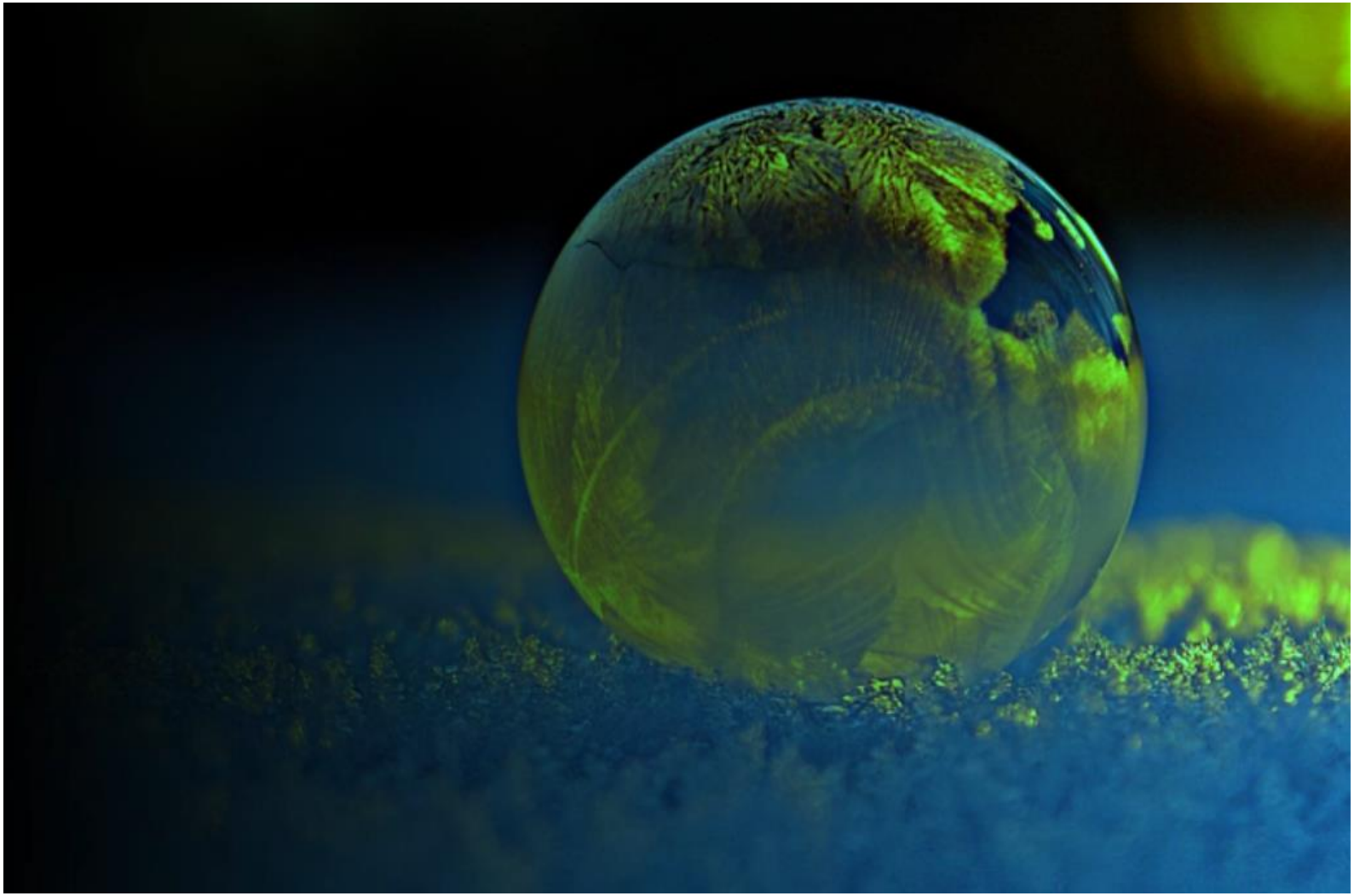
Image Modification

```
close all; s = size(K); K = ones(size(I), 'double');
Id = im2double(I); i = 1;
while i <= s(2)
    K(:,i,1) = 2*i/s(2); K(:,i,2:3) = 0.5*i/s(2); i = i + 1;
end
figure, imshow(Id.*K);
```

Image Modification

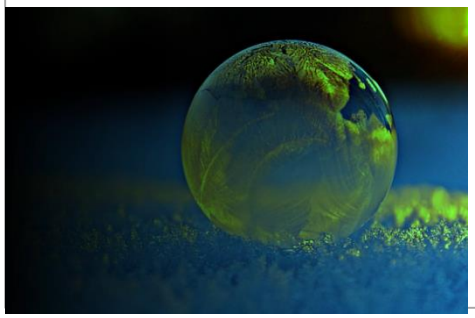
```
close all; s = size(K); K = ones(size(I), 'double');
Id = im2double(I); i = 1;
while i <= s(2)
    K(:,i,1) = 0.5*i/s(2); K(:,i,2:3) = 2*i/s(2); i = i + 1;
end
figure, imshow(Id.*K);
```

Demo
Periodic Effect



Periodic effect

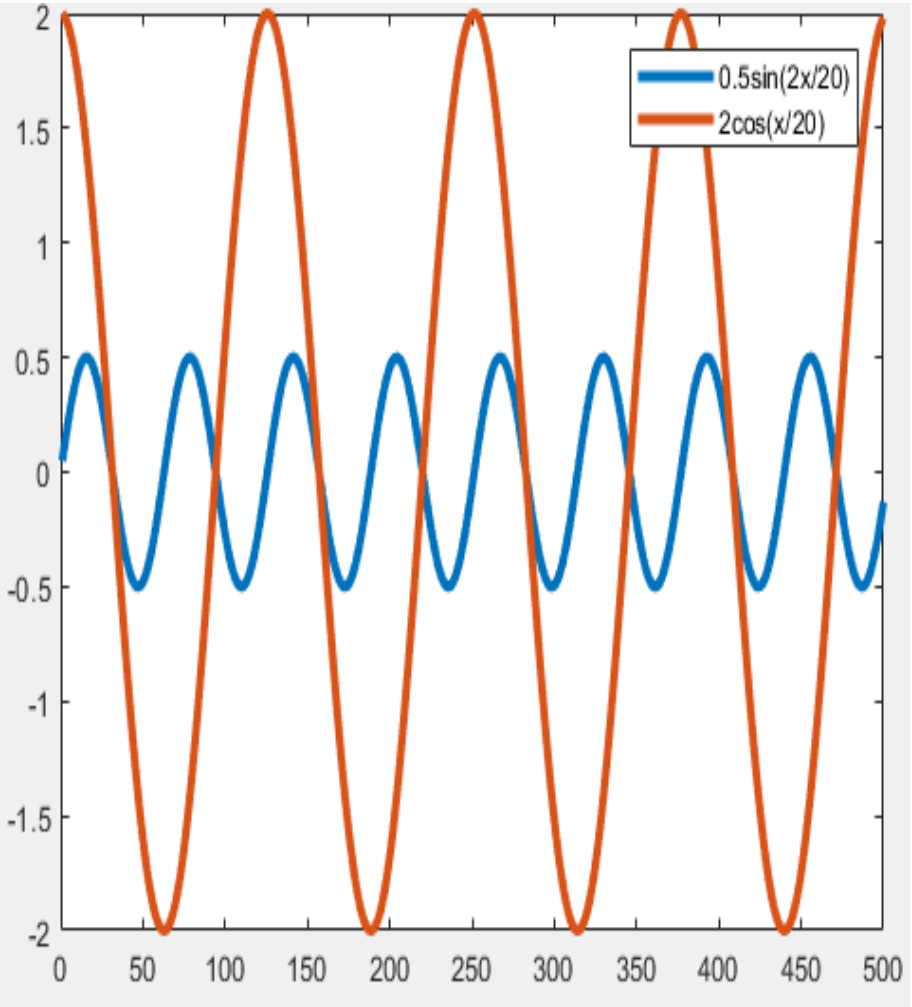
```
%increase blue and green intensity
close all;  I = imread('tmp.png');
s = size(I); K = ones(size(I), 'double');
figure, imshow(I);
input('Press ENTER to start...');
numFrames = 500
```



```
for f = [1:numFrames]
    Id = im2double(I);
    i = 1;
    rsin = sin(2*f/20);
    rcos = cos(f/20);
    rred = 0.5*rsin;
    rbluegreen = 2.0*rcos;
    while i < s(2)
        K(:,i,1) = rred*i/s(2);
        K(:,i,2:3) = rbluegreen*i/s(2);
        i = i + 1;
    end
    imshow(Id.*K);
    pause(0.033);
end
```

Periodic effect

```
for f = [1:numFrames]
      Id = im2double(I);
      i = 1;
      rsin = sin(2*f/20);
      rcos = cos(f/20);
      rred = 0.5*rsin;
      rbluegreen = 2.0*rcos;
      while i < s(2)
          K(:,i,1) = rred*i/s(2);
          K(:,i,2:3) = rbluegreen*i/s(2);
          i = i + 1;
      end
      imshow(Id.*K);
      pause(0.033);
End

//amplitude
```



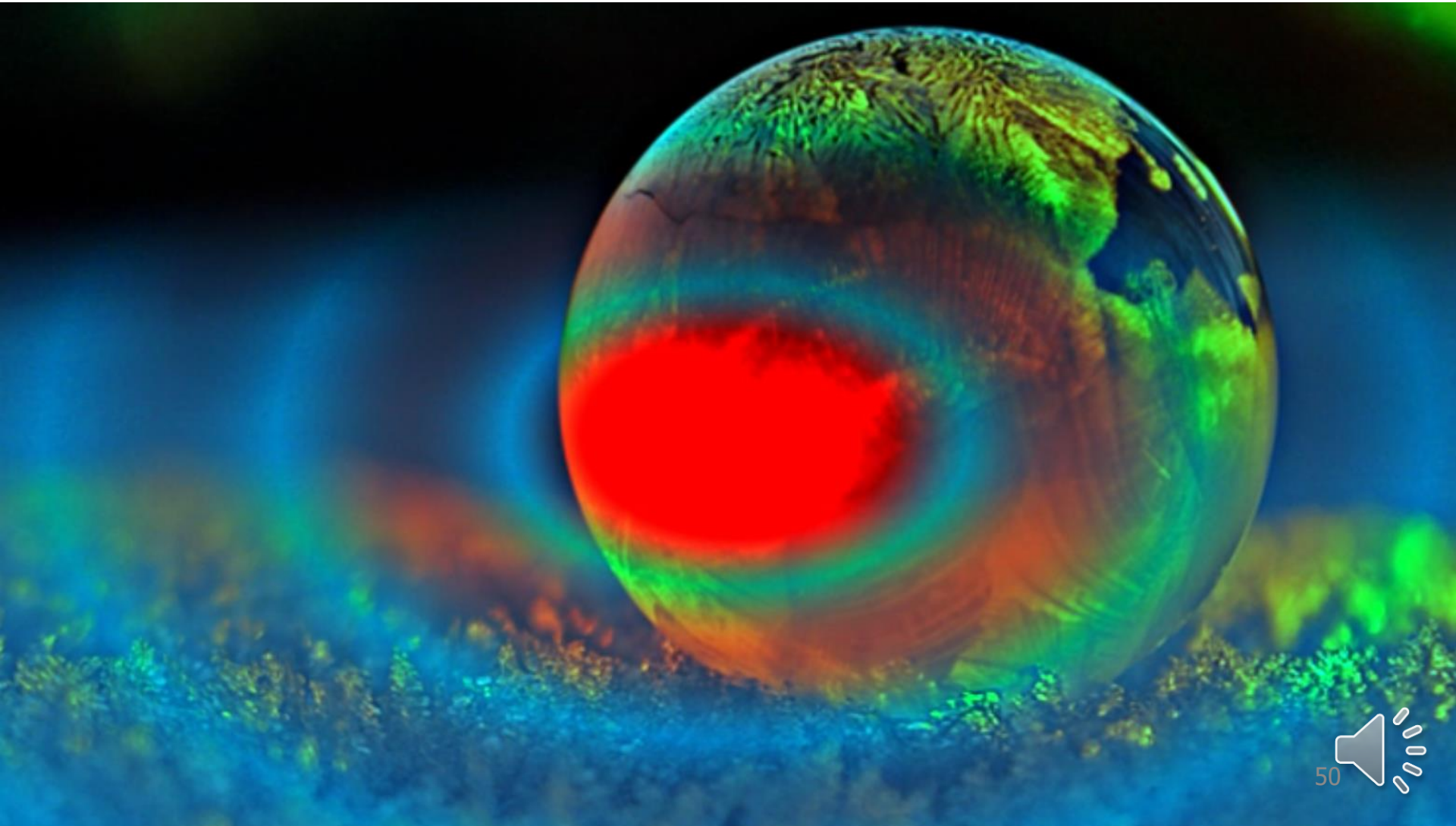
Periodic Effect
Circles

Periodic effect

```
close all;
I = imread('tmp.png');
I = imresize(I, [640, 640]);
s = size(I);
Id = im2double(I);
K = ones(size(I), 'double');
figure, imshow(I);
input('Press ENTER to start...');
numFrames = 500;
```

```
[x y] = meshgrid([1:s(2)]);
x = (x - s(1)/2) ./ 30;
y = (y - s(2)/2) ./ 30;
r = sqrt(x.^2 + y.^2) + eps;
z = abs(sin(r) ./r); % Maxican hat
zmax = max(max(z))
z = 5.*z ./ zmax;
K = Id;
for i = 1:s(1)
    for j = 1:s(2)
      K(i,j,:) = z(i,j); % construct a mask
    end
end
for f = [1:numFrames]
    k0 = 0.5 + 0.5*(sin(f/20));
    imshow(k0.*Id.*K);
    pause(0.033);
end
```

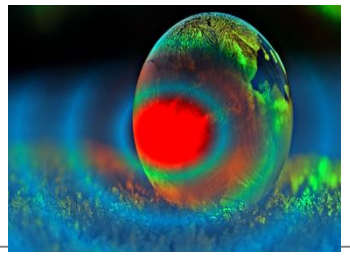
Periodic Effect Channels



Periodic effect: Channel Controls

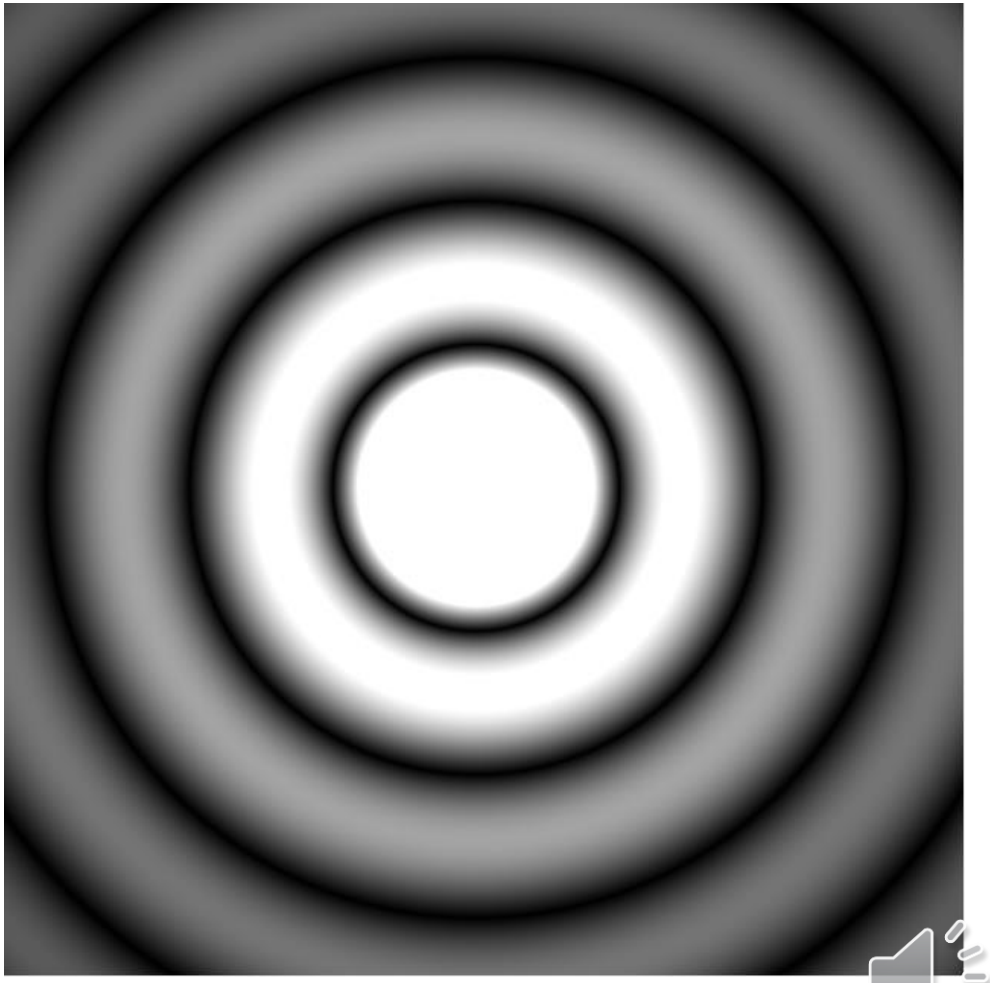
```
close all;
I = imread('tmp.png');
I1 = imresize(I, [640, 640]);
s0 = size(I);
Id = im2double(I1);
s1 = size(Id);
K = ones(size(Id), 'double');
figure, imshow(I);
input('Press ENTER to start...');
numFrames = 500;
[x y] = meshgrid([1:s1(2)]);
x = (x-s1(1)/2) ./ 30;
y = (y-s1(2)/2) ./ 30;
r = sqrt(x.^2 + y.^2) + eps;
```

```
z = abs(sin(r) ./r);
zmax = max(max(z))
z = 5.*z ./ zmax;
K = zeros(size(Id)); //initialize the mask
K(:,:,1) = z(:,:); // construct a mask
K(:,:,2) = z(:,:);
K(:,:,3) = z(:,:);
for f = [1:numFrames]
    k0 = 0.5 + 0.5*(sin(f/20));
    K1 = Id;
    K1(:,:,1) = k0.*Id(:,:,1).*K(:,:,1);
    K1(:,:,2) = k0.*Id(:,:,2).*(2-K(:,:,1));
    K1(:,:,3) = k0.*Id(:,:,3).*(2-K(:,:,1));
    K1 = imresize(K1, [s0(1) s0(2)]);
    imshow(K1);
    pause(0.033);
end
```



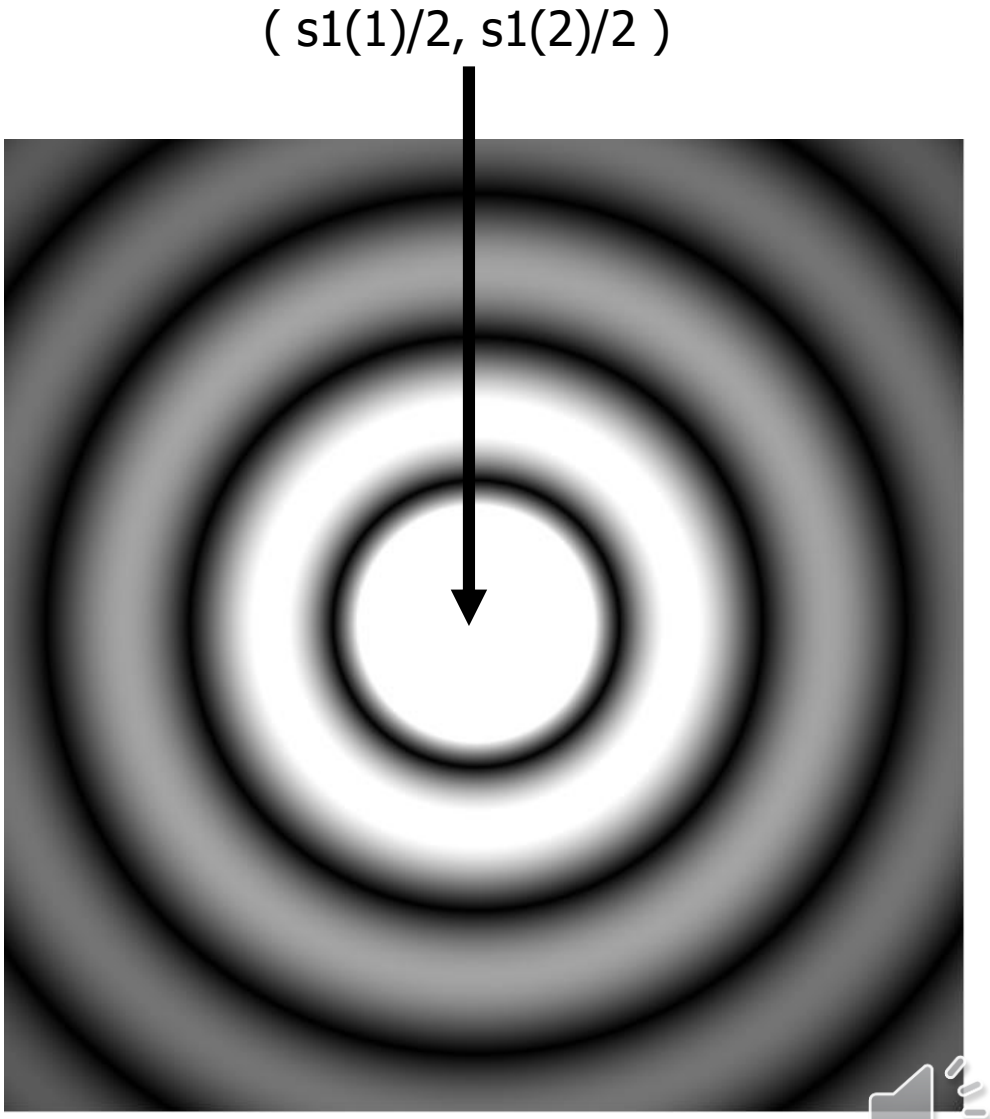
Map K (or Mask)

```
numFrames = 500;
[x y] = meshgrid([1:s1(2)]);
x = (x-s1(1)/2) ./ 30;
y = (y-s1(2)/2) ./ 30;
r = sqrt(x.^2 + y.^2) + eps;
z = abs(sin(r) ./r);
zmax = max(max(z))
z = 5.*z ./ zmax;
K = zeros(size(Id));
K(:,:,1) = z(:,:);
K(:,:,2) = z(:,:);
K(:,:,3) = z(:,:);
```



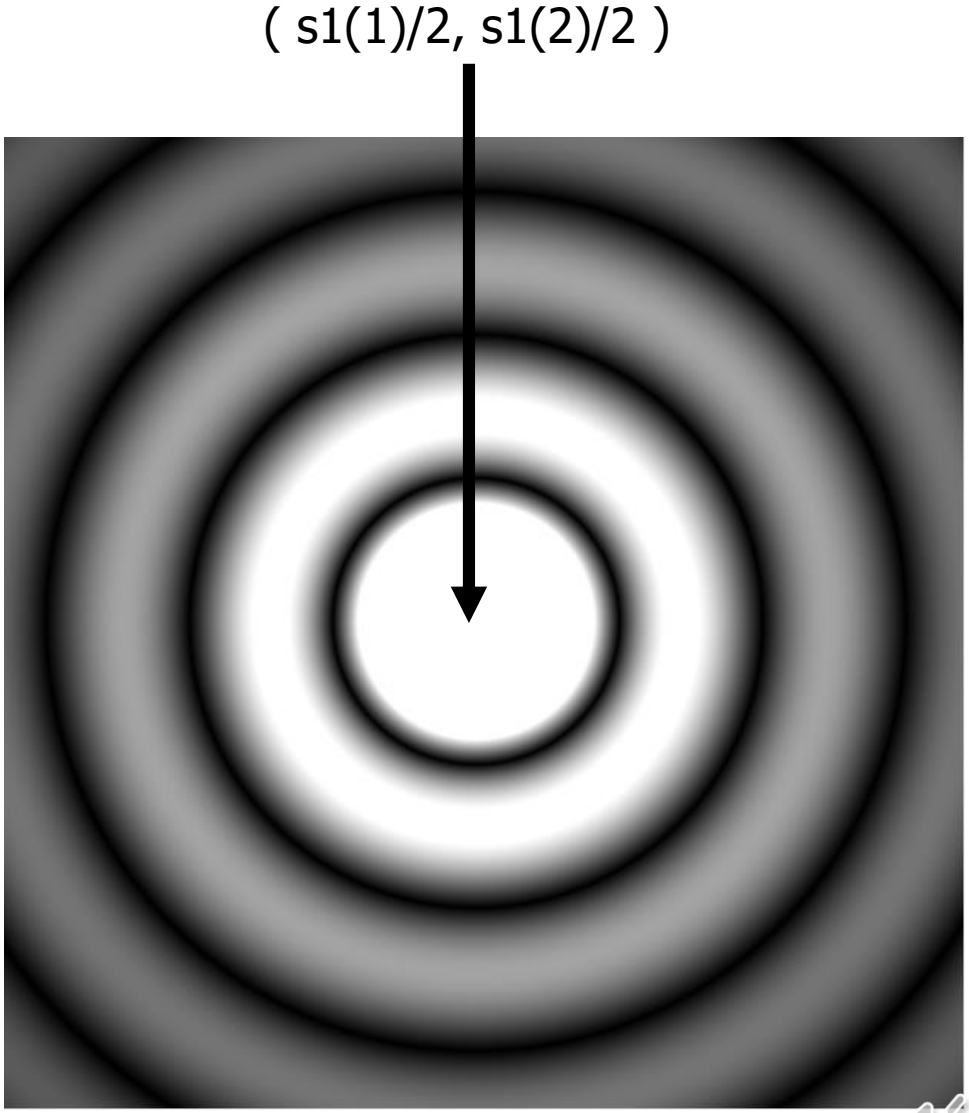
Map K

```
numFrames = 500;
[x y] = meshgrid([1:s1(2)]);
x = (x-s1(1)/2) ./ 30;
y = (y-s1(2)/2) ./ 30;
r = sqrt(x.^2 + y.^2) + eps;
z = abs(sin(r) ./r);
zmax = max(max(z))
z = 5.*z ./ zmax;
K = zeros(size(Id));
K(:,:,1) = z(:,:);
K(:,:,2) = z(:,:);
K(:,:,3) = z(:,:);
```

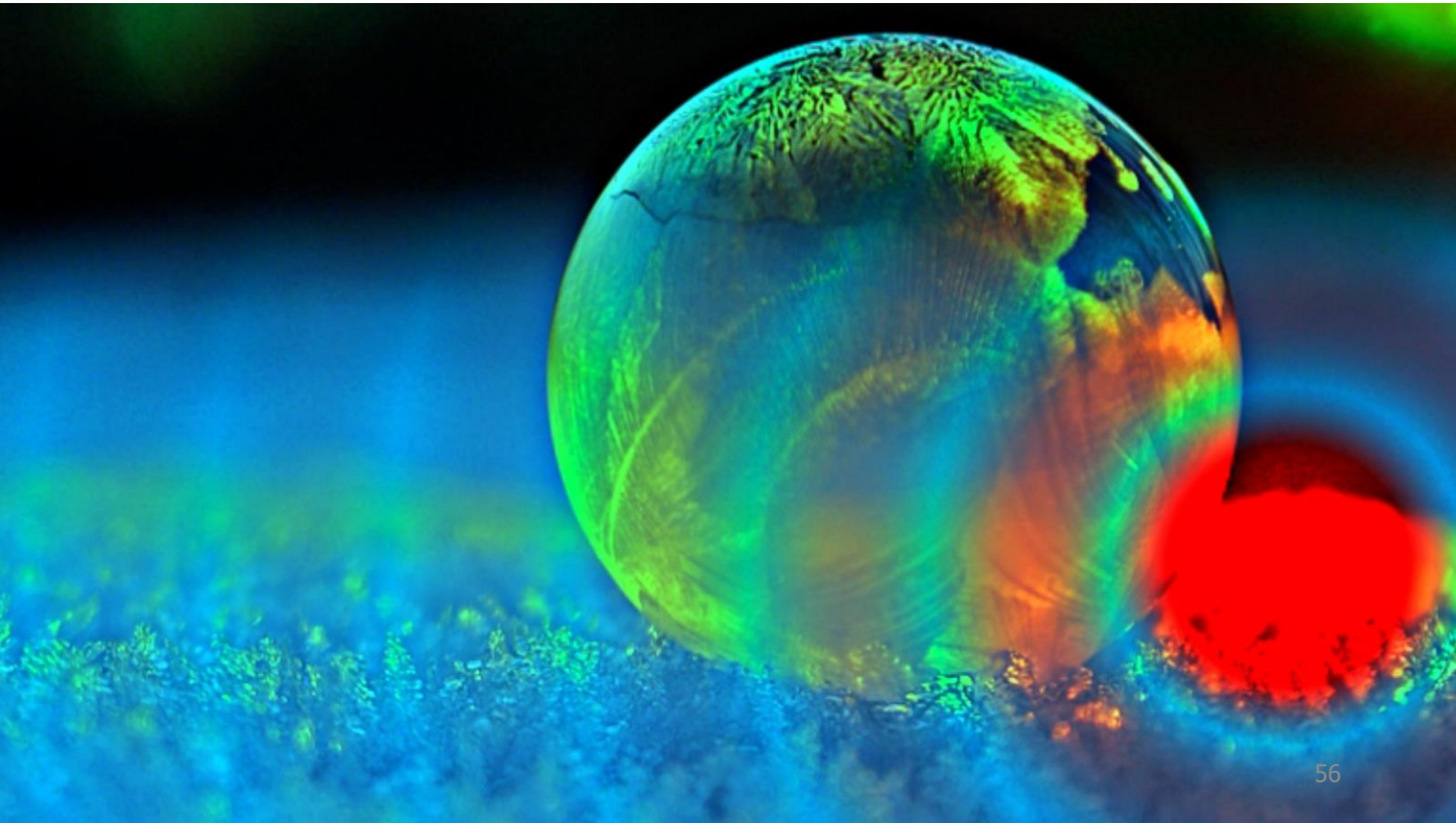


Map K

```
numFrames = 500;
[x y] = meshgrid([1:s1(2)]);
x = (x-s1(1)/2) ./ 30;
y = (y-s1(2)/2) ./ 30;
r = sqrt(x.^2 + y.^2) + eps;
z = abs(sin(r) ./r);
zmax = max(max(z))
z = 5.*z ./ zmax;

%K = zeros(size(Id));

K = cat(3, z, z, z);
%Concatenate
%K(:,:,1) = z(:,:);
%K(:,:,2) = z(:,:);
%K(:,:,3) = z(:,:);
```

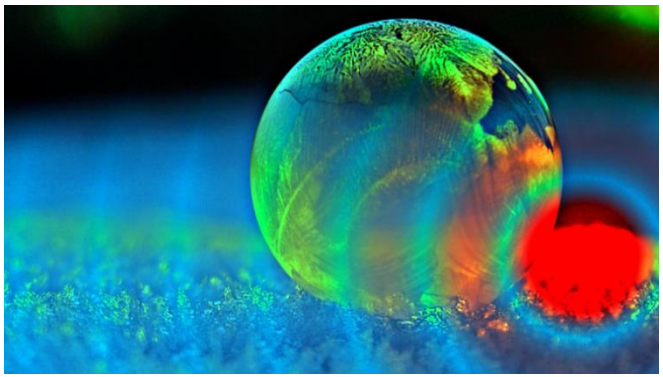


Periodic Effect
Elliptic Orbit



Periodic effect: Orbit

```
close all;
I = imread('tmp.png');
s0 = size(I);
Id = im2double(I);
figure, imshow(I);
input('Press ENTER to start...');

numFrames = 500
```



```
for f = [1:numFrames]
    f0 = f/numFrames*pi*2;
    xx = 0.8*s0(2)*cos(f0)/2;
    yy = 0.8*s0(1)*sin(f0)/2;
    [x y] = meshgrid([1:s0(2)], [1:s0(1)]);
    x = (x-s0(2)/2+xx) ./30;
    y = (y-s0(1)/2+yy) ./ 30;
    r = sqrt(x.^2 + y.^2) + eps;
    z = abs(sin(r) ./r);
    zmax = max(max(z))
    z = 5.*z ./ zmax;
    K = cat(3, z, z, z);
    k0 = 0.5 + abs(0.5*(sin(f/20)));
    K1 = cat(3, k0.*Id(:,:,1).*K(:,:,1),...
           k0.*Id(:,:,2).*(2-K(:,:,1)),...
           k0.*Id(:,:,3).*(2-K(:,:,1)) );
    imshow(K1); pause(0.033);
end
```

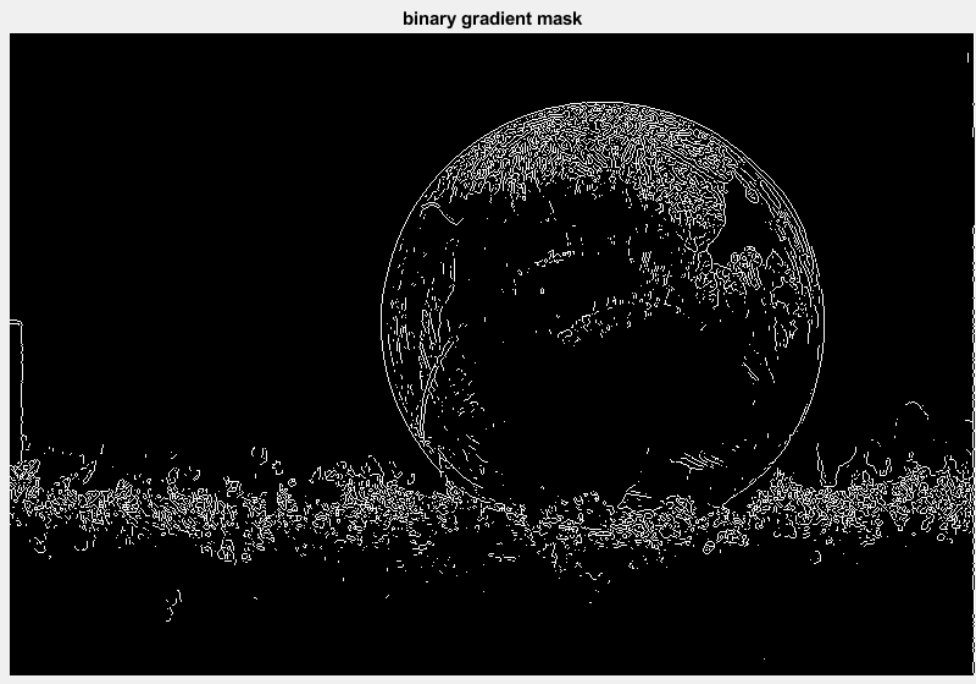
Variables

Name	Size	Bytes	Class	Attributes
I	525x791x3	1245825	uint8	
Id	525x791x3	9966600	double	
K	525x791x3	9966600	double	
K1	525x791x3	9966600	double	

```
K = cat(3, z, z, z);
k0 = 0.5 + abs(0.5*(sin(f/20)));
K1 = cat(3, k0.*Id(:,:,1).*K(:,:,1),...
         k0.*Id(:,:,2).*(2-K(:,:,1)),...
         k0.*Id(:,:,3).*(2-K(:,:,1)) );
```

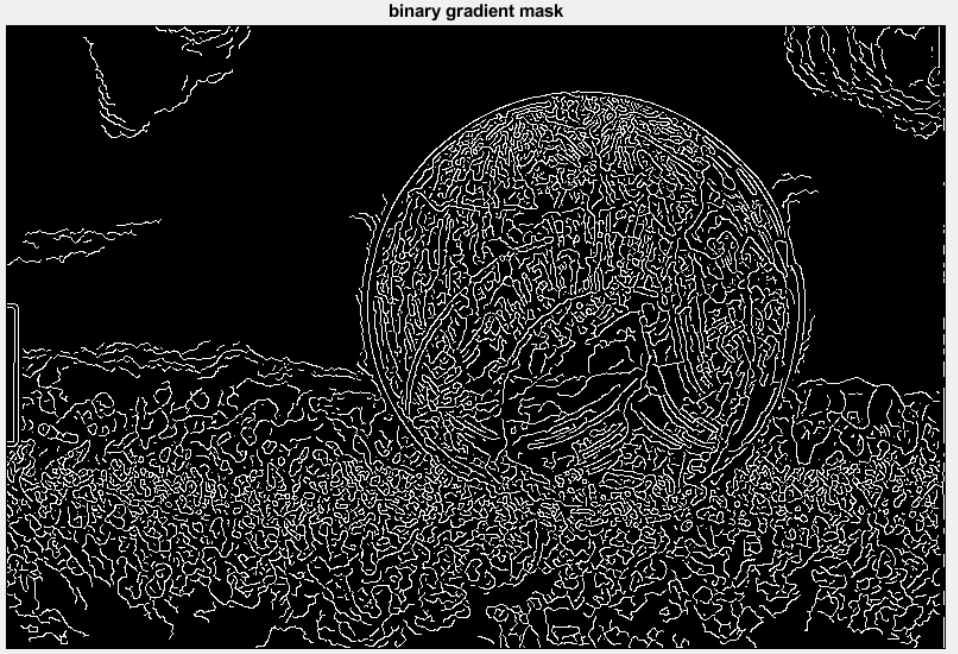
Edge Detection

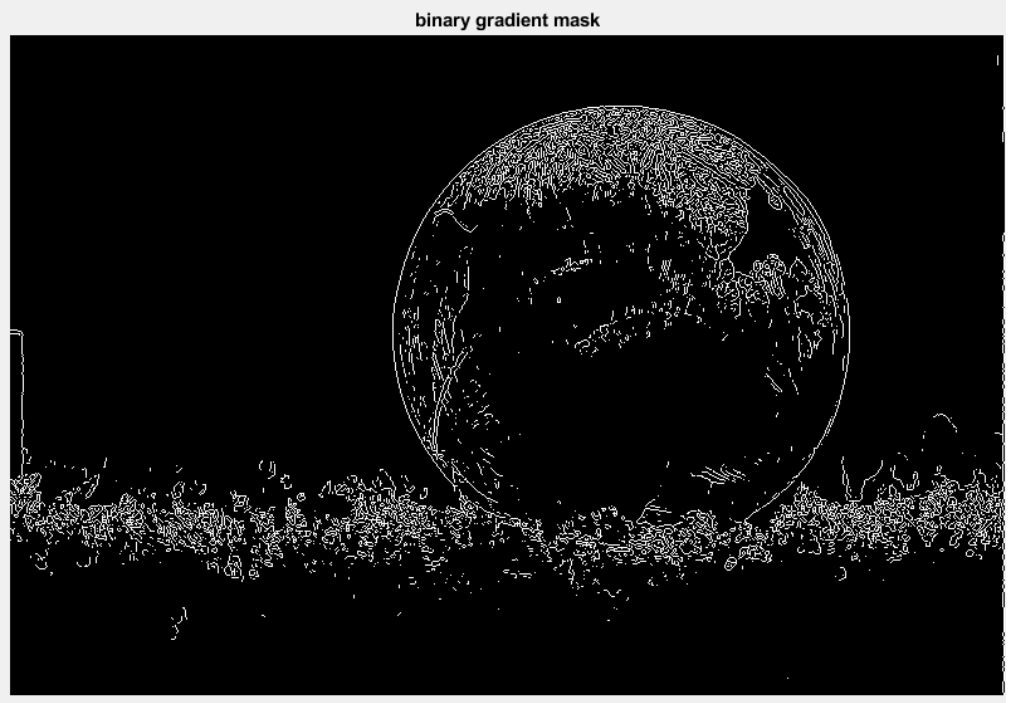
```
[~, threshold] = edge(G, 'sobel');
fudgeFactor = .5;
BWs = edge(G,'sobel', threshold * fudgeFactor);
figure, imshow(BWs), title('binary gradient mask');
```



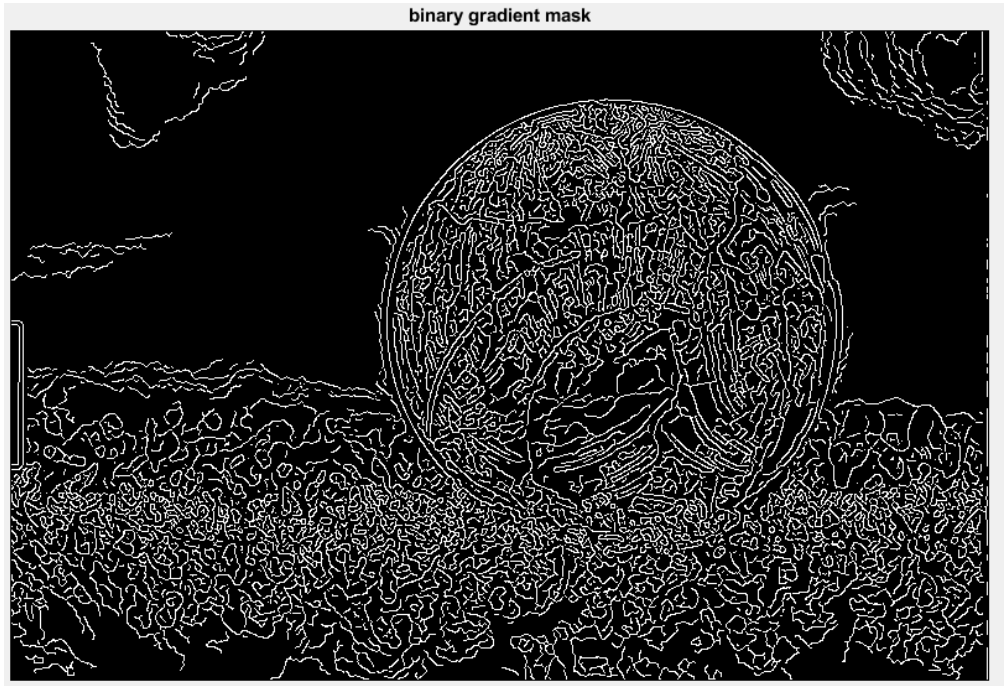
Edge Detection

```
[~, threshold] = edge(G, 'canny');
fudgeFactor = .5;
BWs = edge(G,'canny', threshold * fudgeFactor);
figure, imshow(BWs), title('binary gradient mask');
```



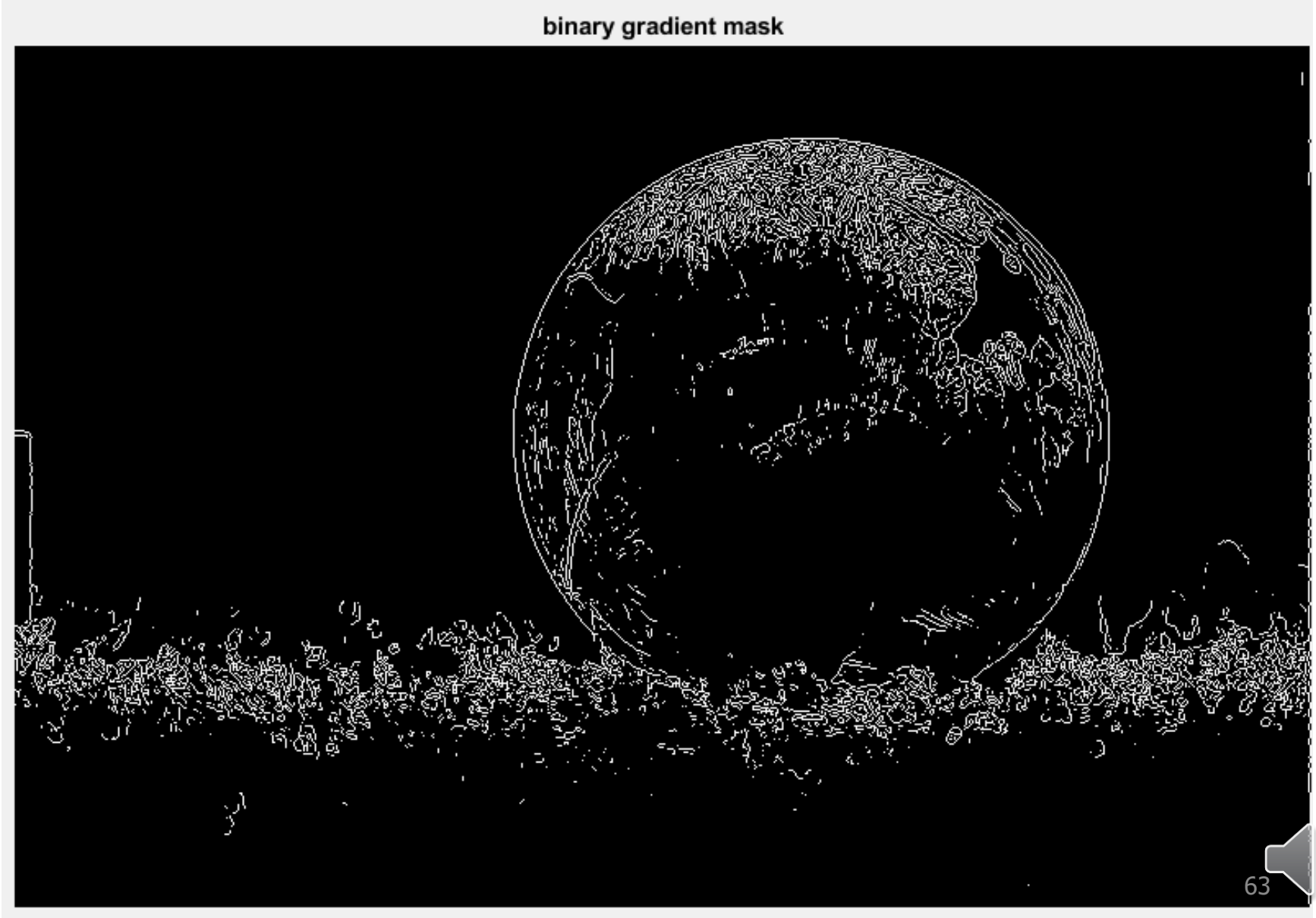


Sobel

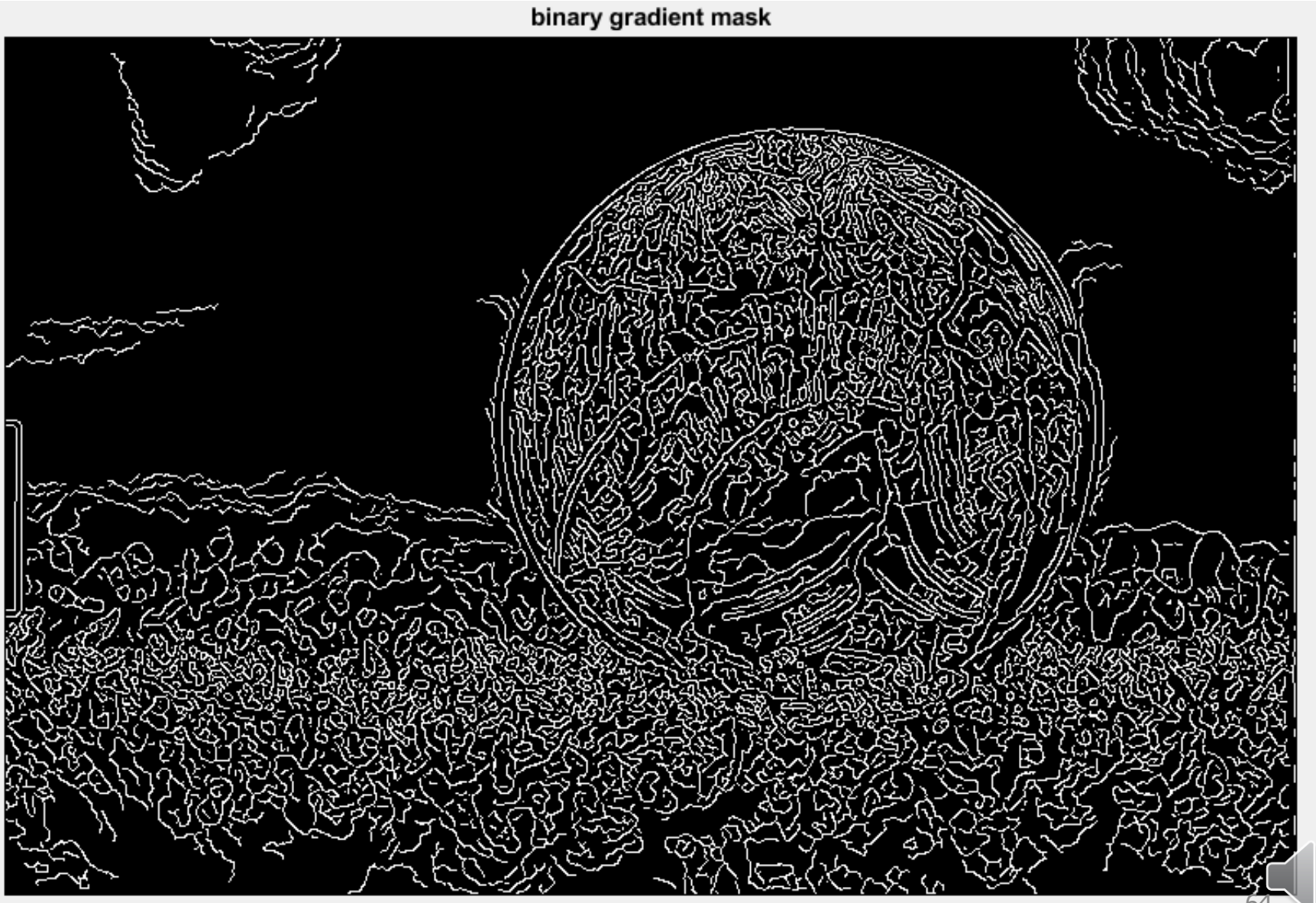


Canny

Sobel edge detection

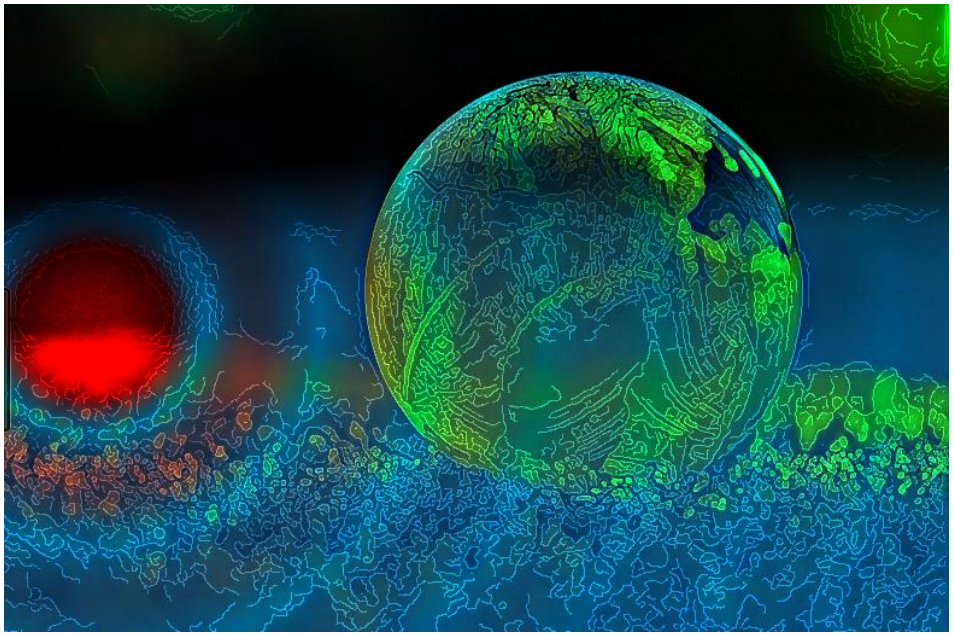


Canny edge detection

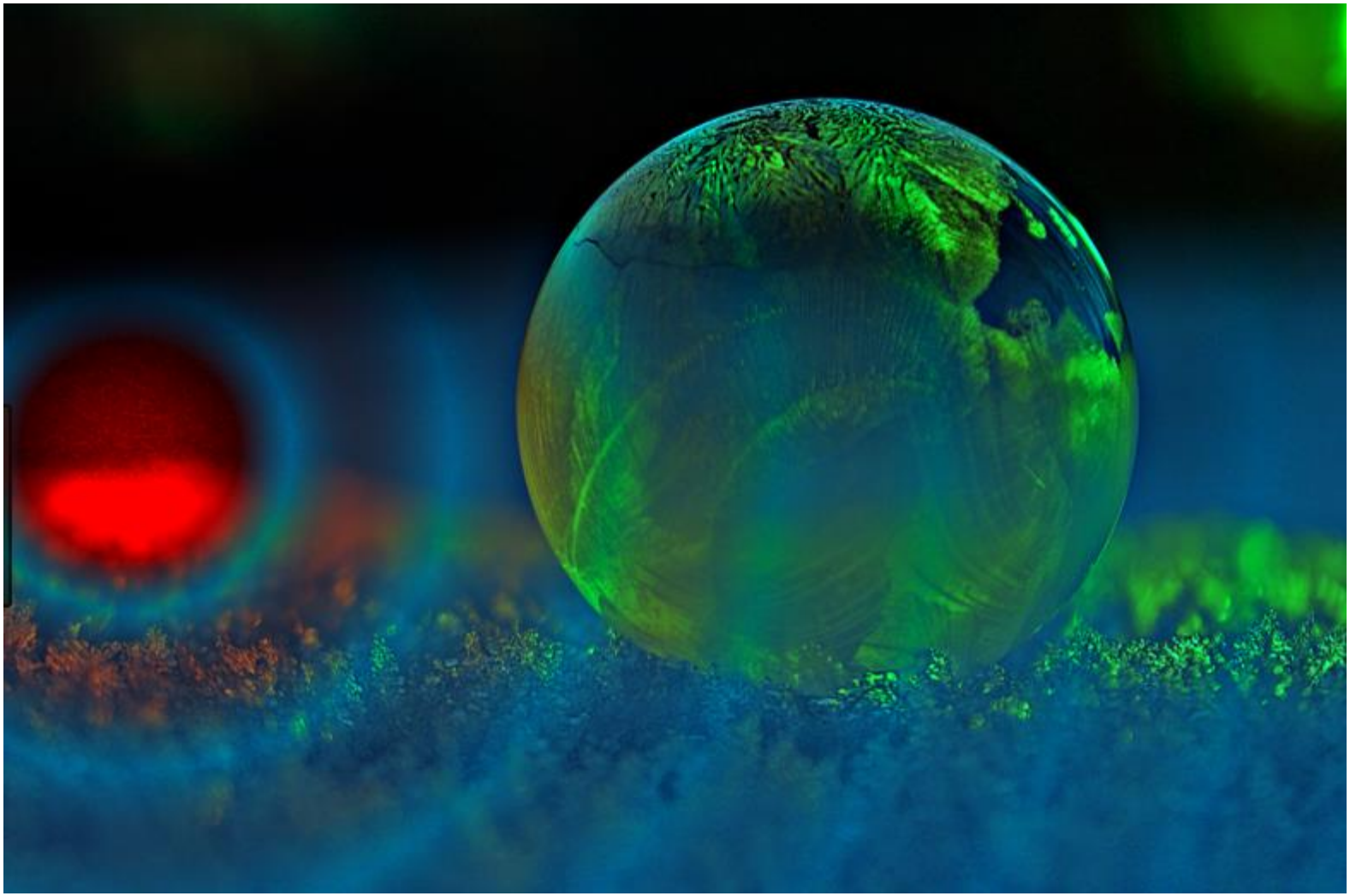


Edge + Image

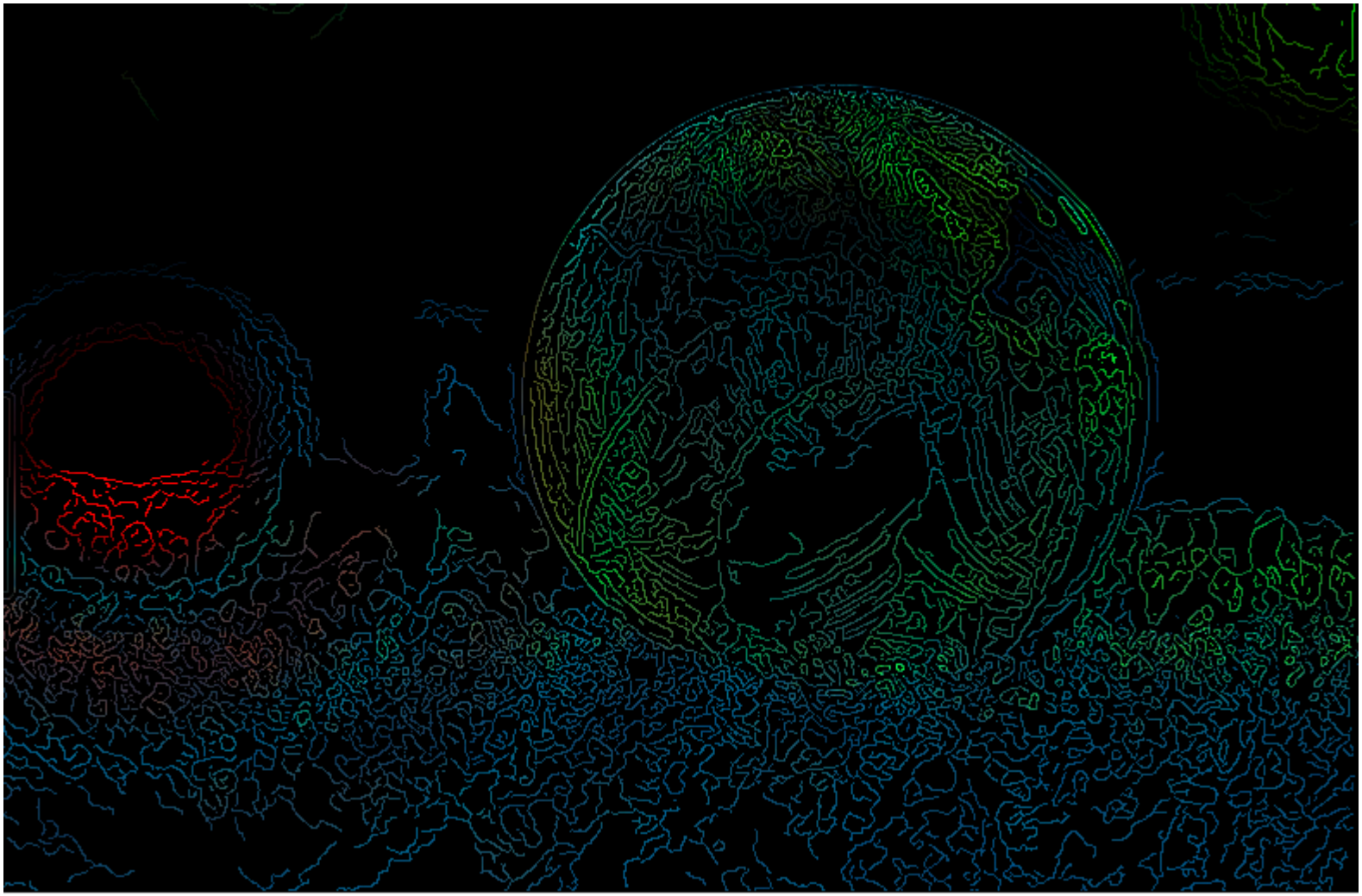
```
G = rgb2gray(K1);
[~, threshold] = edge(G, 'canny');
fudgeFactor = .5;
BWs = edge(G,'canny', …
threshold * fudgeFactor);
K2 = K1.*BWs;
imshow(K2 + K1)
```



Edge + Image
K1

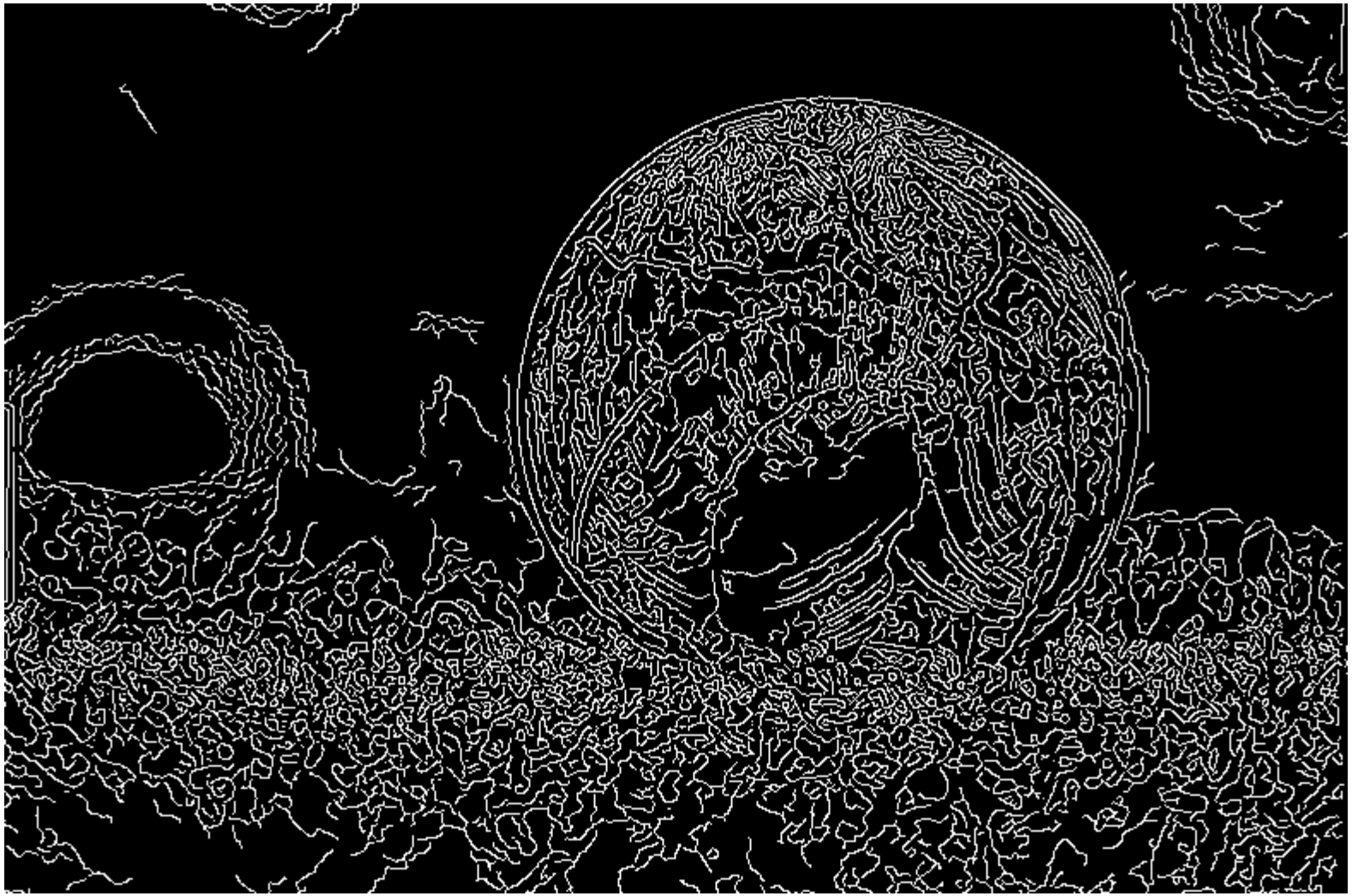


Edge + Image
K2

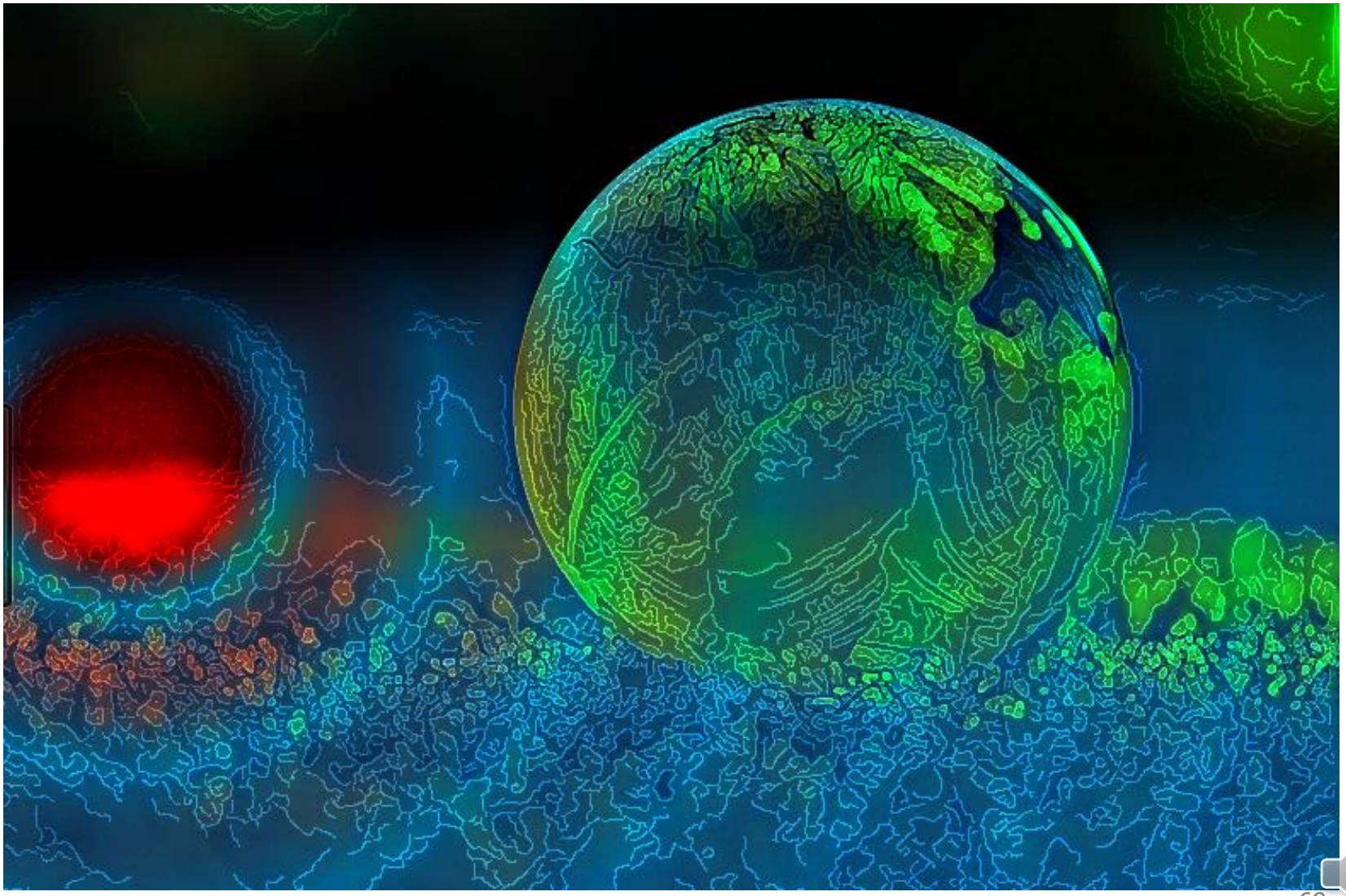


Edge + Image

Name	Size	Bytes	Class	Attributes
BWs	525x791	415275	logical	



Edge + Image
Edge Mask + Image



```
G = imnoise(I,'salt & pepper',0.02); % add noise
Imshow(G);
```

```
G = imnoise(I,'salt & pepper',0.1); % add noise
Imshow(G);
```

Files and Folders

- pwd displays the current working directory.
- cd Change current working directory.

dir	List folder contents
ls	List folder contents
pwd	Identify current folder
fileattrib	Set or get attributes of file or folder
exist	Check existence of variable, script, function, folder, or class
isfile	Determine if input is file
isfolder	Determine if input is folder
type	Display contents of file
visdiff	Compare two files or folders
what	List MATLAB files in folder
which	Locate functions and files

Detect and Measure Circular Objects in an Image

- Tutorials in matlab

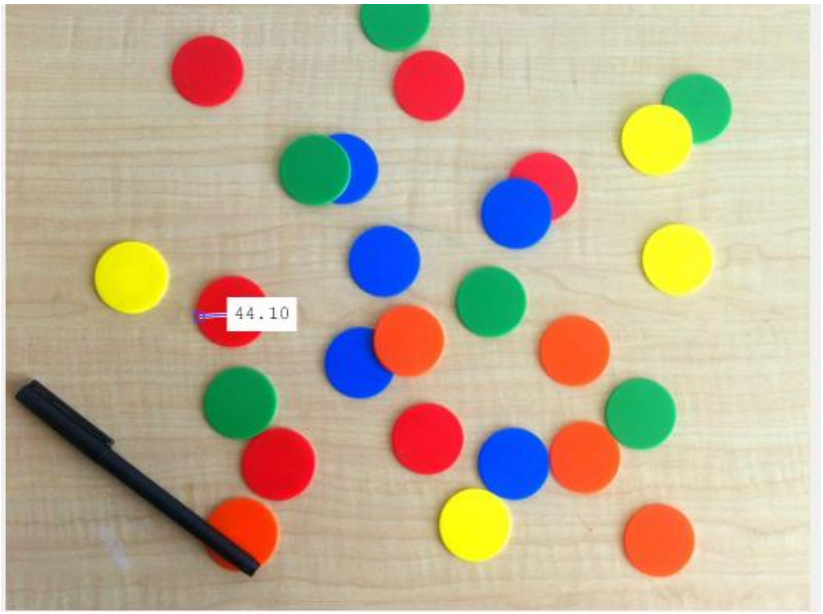
Step 1: Load Image

```
rgb = imread('coloredChips.png');
imshow(rgb)
```

Step 2: Determine Radius Range for Searching Circles

d = imdistline;

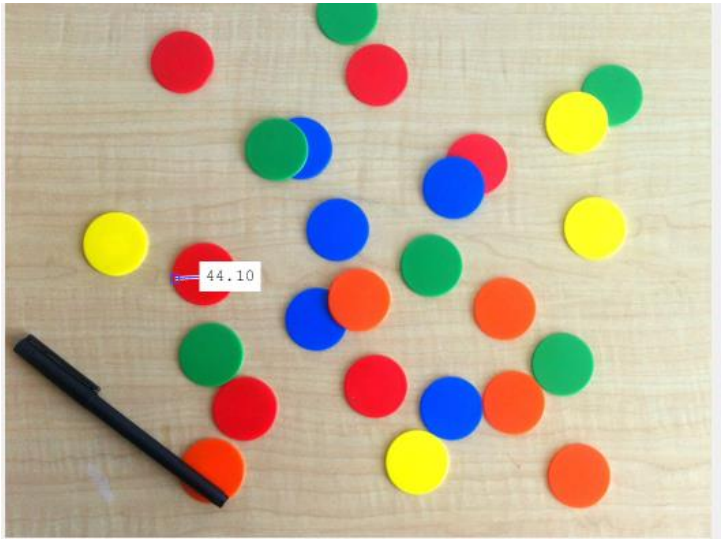
imdistline creates a draggable tool.



delete(d) %delete the tool

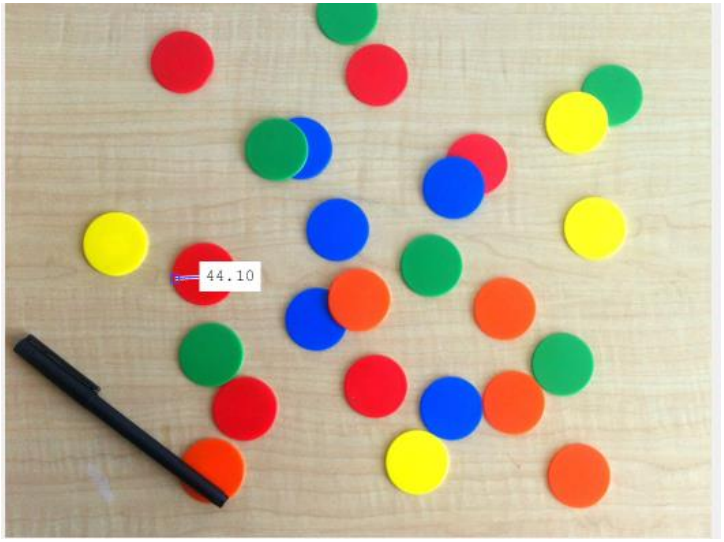
Step 3: Initial Attempt to Find Circles

```
gray_image = rgb2gray(rgb);
imshow(gray_image)

[centers,radii] = imfindcircles(rgb,[20 25],'ObjectPolarity','dark')
```



Step 4: Increase Detection Sensitivity

[centers,radii] = imfindcircles(rgb,[20 25],'ObjectPolarity','dark', ... 'Sensitivity',0.9)



Step 5: Draw the Circles on the Image

imshow(rgb)

h = viscircles(centers,radii);

Step 5: Draw the Circles on the Image

```
[centers,radii] = imfindcircles(rgb,[20
25],'ObjectPolarity','dark', 'Sensitivity',0.92); length(centers)


delete(h) % Delete previously drawn circles
h = viscircles(centers,radii);
```

Exercise: Draw an orange frame at the boundary.

Width = 5 pixels

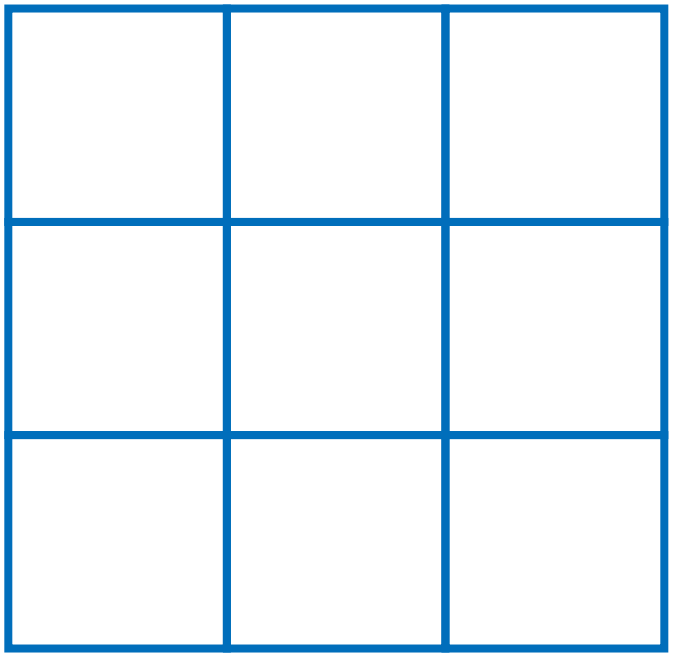
Exercise

Exercise: Draw a blue grid to the image.

Grid dimension = 3 x 3.

Grid frame width = 5 pixels.

Exercise



Summary

- imread — read an image
- imshow — show an image
- imhist — show histogram
- histeq — histogram equalization