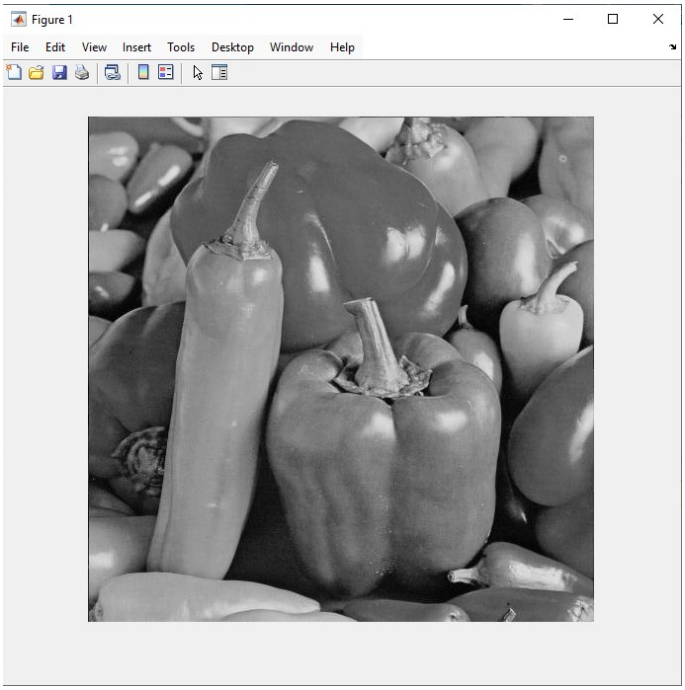
Practica 4

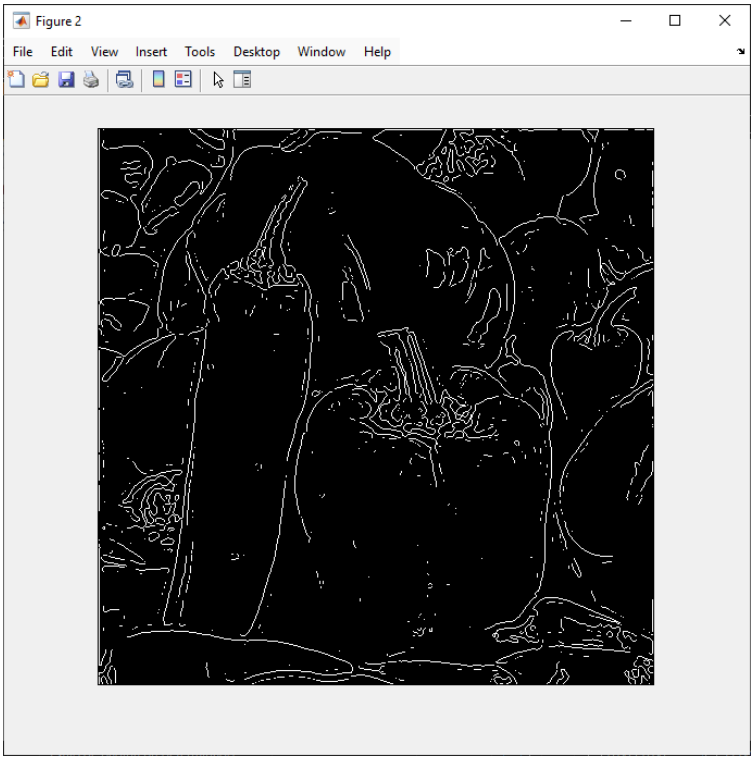
Ejercicio 8.A:

```
I=imread('../Imagenes/pimientos1.jpg');
figure
imshow(I)
%Devuelve una imagen binaria con los bordes. Tiene como argumentos la
%imagen, el metodo a usar, threshold y direccion.
B1=edge(I,'zerocross');
B2=edge(I,'Roberts');
B3=edge(I,'canny');
figure
imshow(B1)
figure
imshow(B2)
figure
imshow(B3)
```

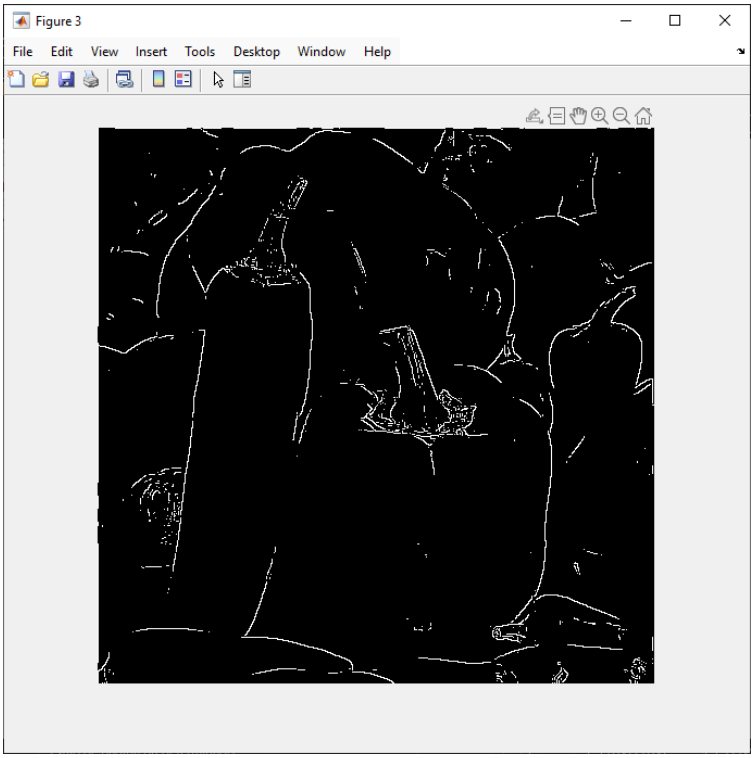
Tenemos como imagen de base:



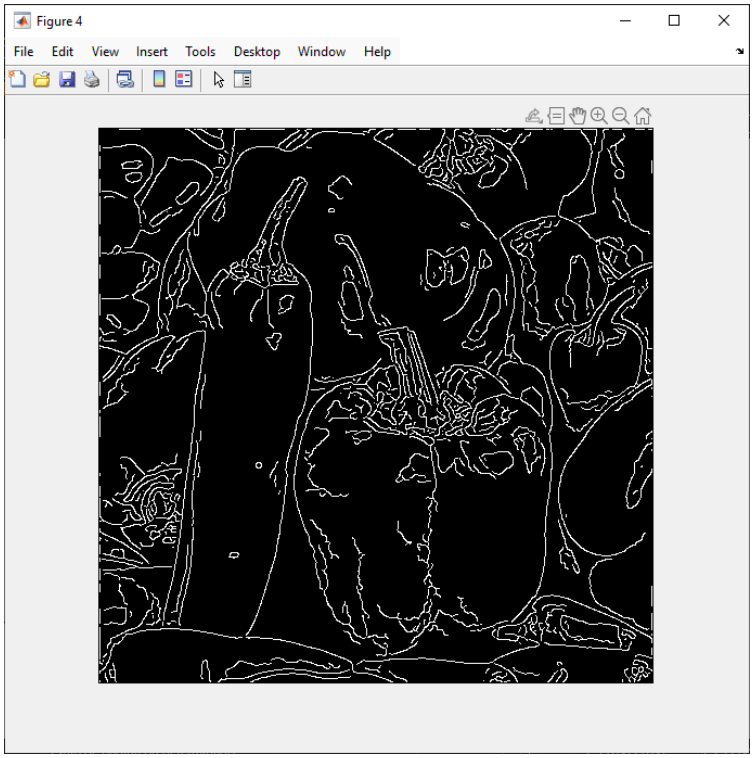
Si le aplicamos el algoritmo zerocross, obtenemos los siguientes bordes:



Para el algoritmo Roberts obtenemos:



Por ultimo para canny obtenemos:

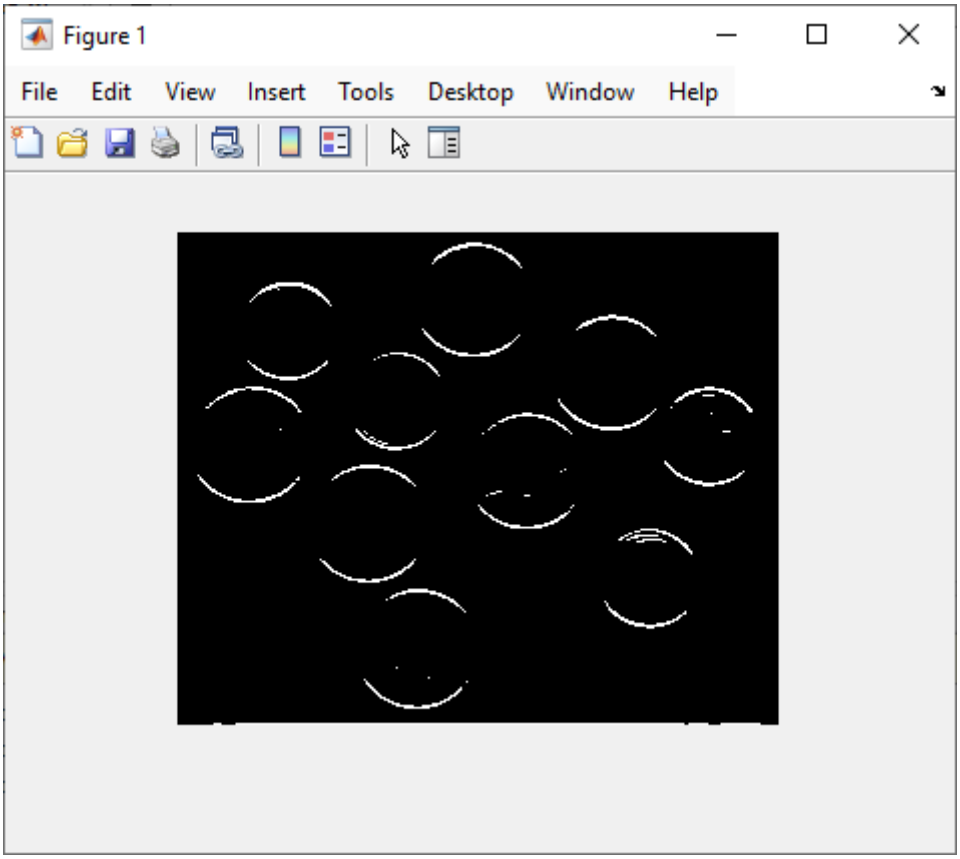


Ejercicio 8.B:

```
I=imread('../Imagenes/monedas.png');
%Una matriz kernel para detectar bordes horizontales
G=[1 1 1; 0 0 0; -1 -1 -1];
%Filtramos usando la matriz
J=filter2(g,I);
%Nos quedamos con los numeros enteros
J=abs(J);
%Le aplicamos un threshold
B=J>0.45*(max(J(:))-min(J(:)));
imshow(B)
```

Para detectar los bordes vamos a aplicar una matriz kernel a la imagen como filtro. La matriz la definimos de manera que detecte los cambios de intensidad de arriba hacia abajo detectando los bordes horizontales. Tras esto se lo aplicamos como filtro, nos quedamos con los valores absolutos,quitando decimales. Lo ultimo que necesitamos hacer es binarizar la imagen aplicándole un valor umbral donde decidimos donde es 1 o 0.

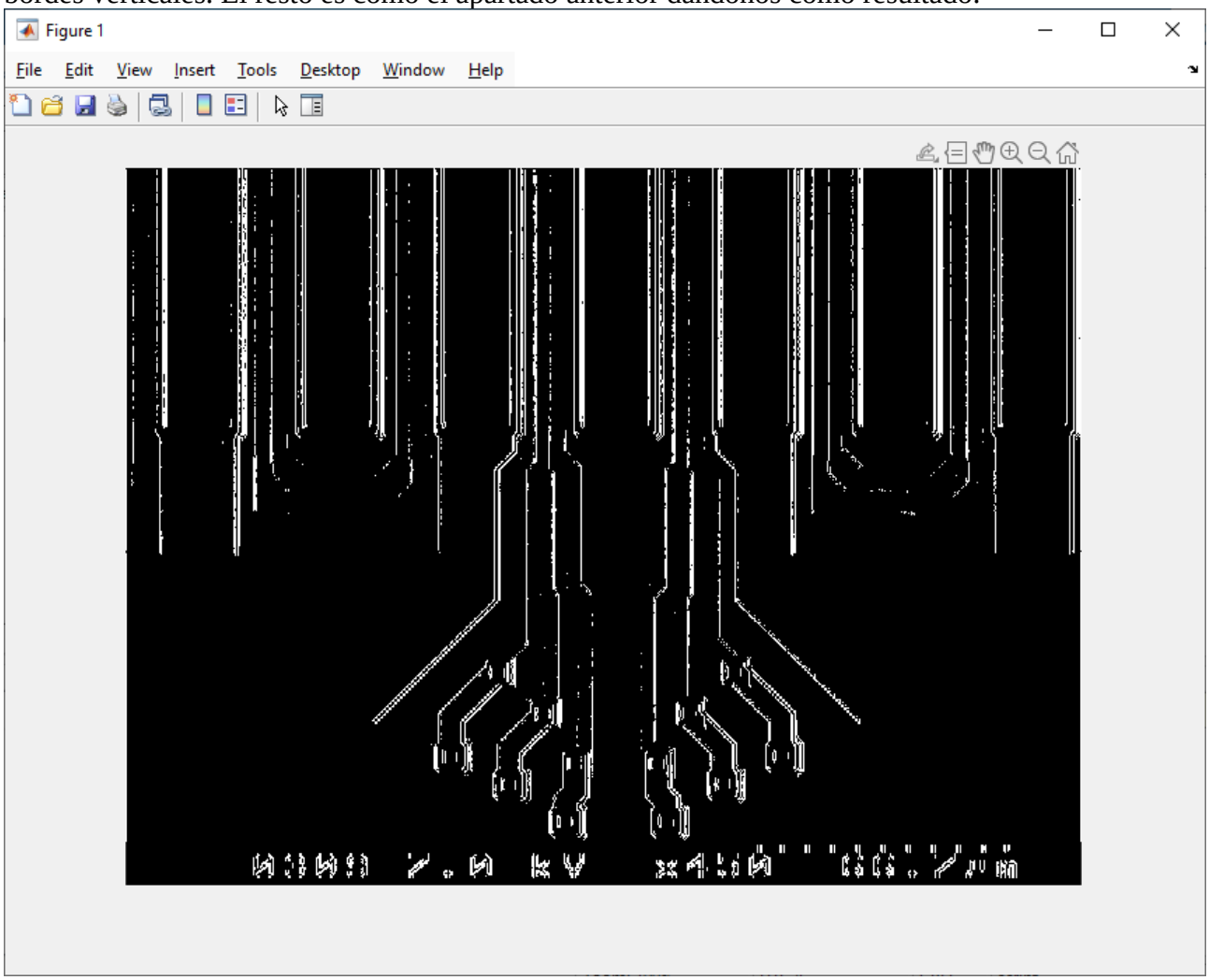
Estos nos da como resultado lo siguiente:



Ejercicio 8.C:

```
I=imread('../Imagenes/WAFER1.TIF');
%Una matriz kernel para detectar bordes verticales
g=[1 0 -1;
   1 0 -1;
   1 0 -1];
%Filtramos usando la matriz
J=filter2(g,I);
%Nos quedamos con los numeros enteros
J=abs(J);
%Le aplicamos un threshold
B=J>0.27*(max(J(:))-min(J(:)));
imshow(B)
```

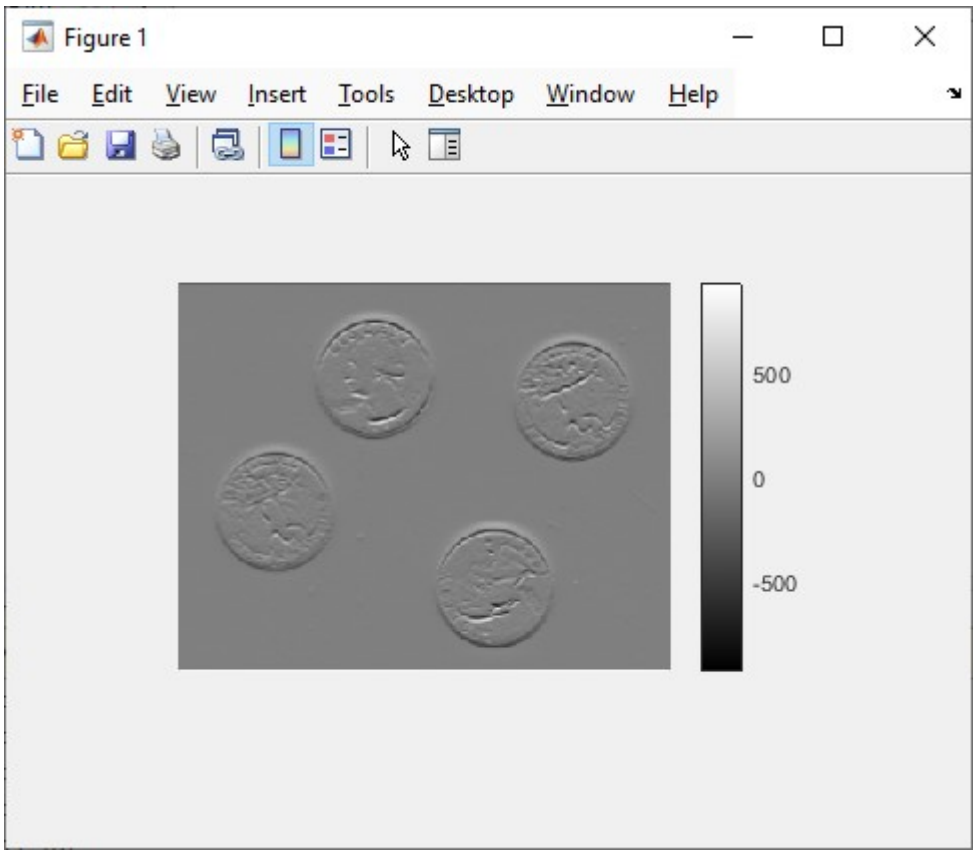
En este caso necesitamos conseguir los bordes verticales, por lo que la matriz kernel la hacemos para que detecte los cambios de intensidad de izquierda a derecha, de esta manera tenemos los bordes verticales. El resto es como el apartado anterior dándonos como resultado:



Ejercicio 8.D:

```
I=imread('../Imagenes/eight.tif');
%Creamos una matriz kernel de bordes horizontales centrandonos en el pixel
%horizontal
h=[1 2 1; 0 0 0; -1 -2 -1];
I1=filter2(h,I);
imshow(I1,[]), colorbar
```

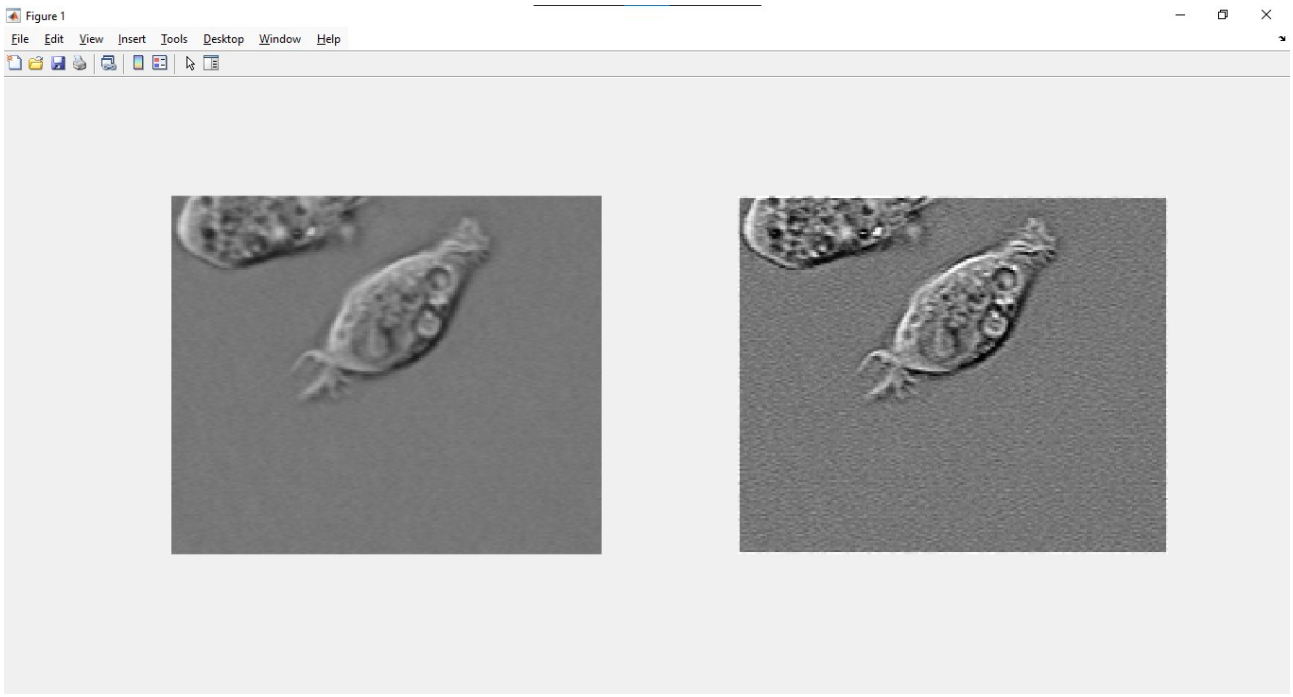
Le aplicamos a la imagen un filtro de bordes horizontales dándole bastante importancia los pixeles cercanos al centro. Esto nos da como resultado:



Ejercicio 9.A:

```
I=imread('../Imagenes/cell.tif');
%Crea un filtro del tipo que le digamos
h=fspecial('unsharp');
%Usamos el filtro y nos quedamos con los valores absolutos del resultado
J=abs(filter2(h,I));
%Convertimos los valores a un rango [0,1]
J1=J/255; % /max(J(:));
figure
subplot(1,2,1)
imshow(I)
subplot(1,2,2)
imshow(J1)
```

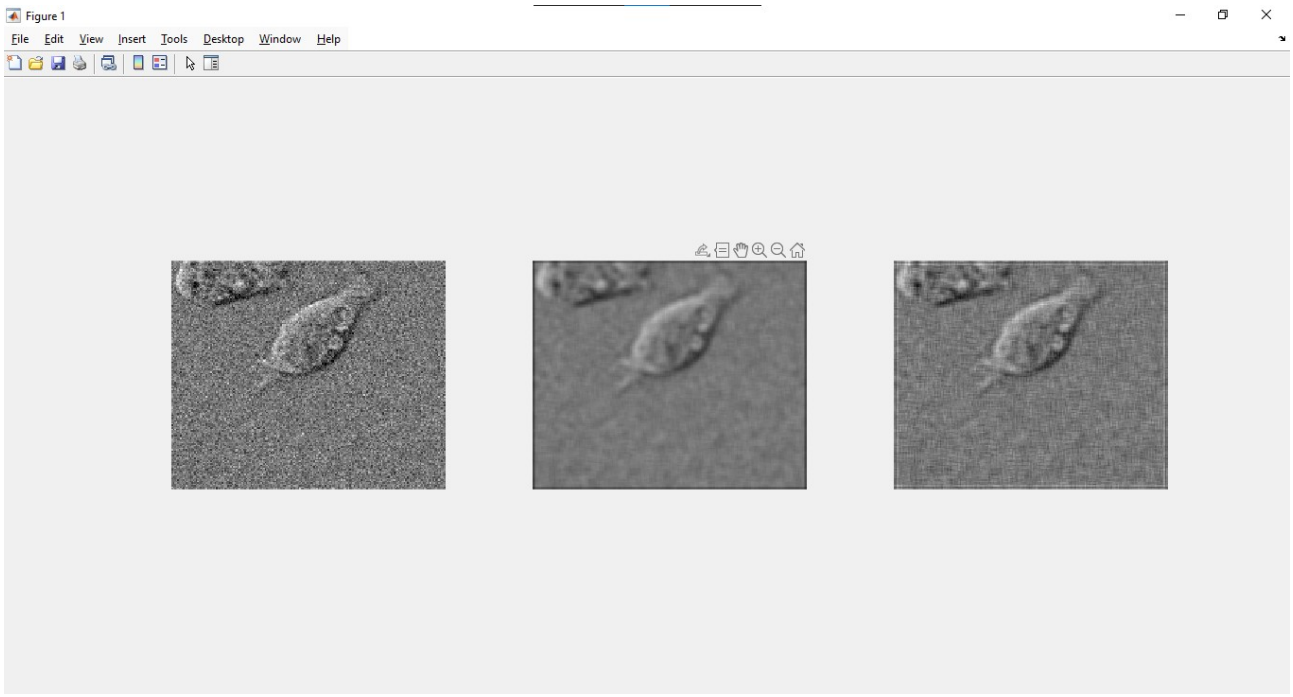
Creamos un filtro de realce de bordes con ‘fspecial(‘unsharp’)’ y se lo aplicamos a la imagen, dándonos como resultado:



Ejercicio 9.B:

```
I=imread ('../Imagenes/cell.tif');
%Le aplicamos un filtro para añadirle ruido a una imagen
%Le aplicamos el tipo que le digamos, la media y la varianza del ruido
J=imnoise(I, 'gaussian' ,0,0.01);
figure
subplot(1,3,1)
imshow(J)
%Creamos un filtro average de tamaño 5x5
g=fspecial('average',[5 5])
%Le aplicamos el filtro y convertimos los valores al rango [0,1]
M1=filter2(g,J)/255;
%Creamos otro filtro para realzar los bordes
h=fspecial('unsharp')
M2=abs(filter2(h,M1));
subplot(1,3,2)
imshow(M1); subplot(1,3,3), imshow(M2)
```

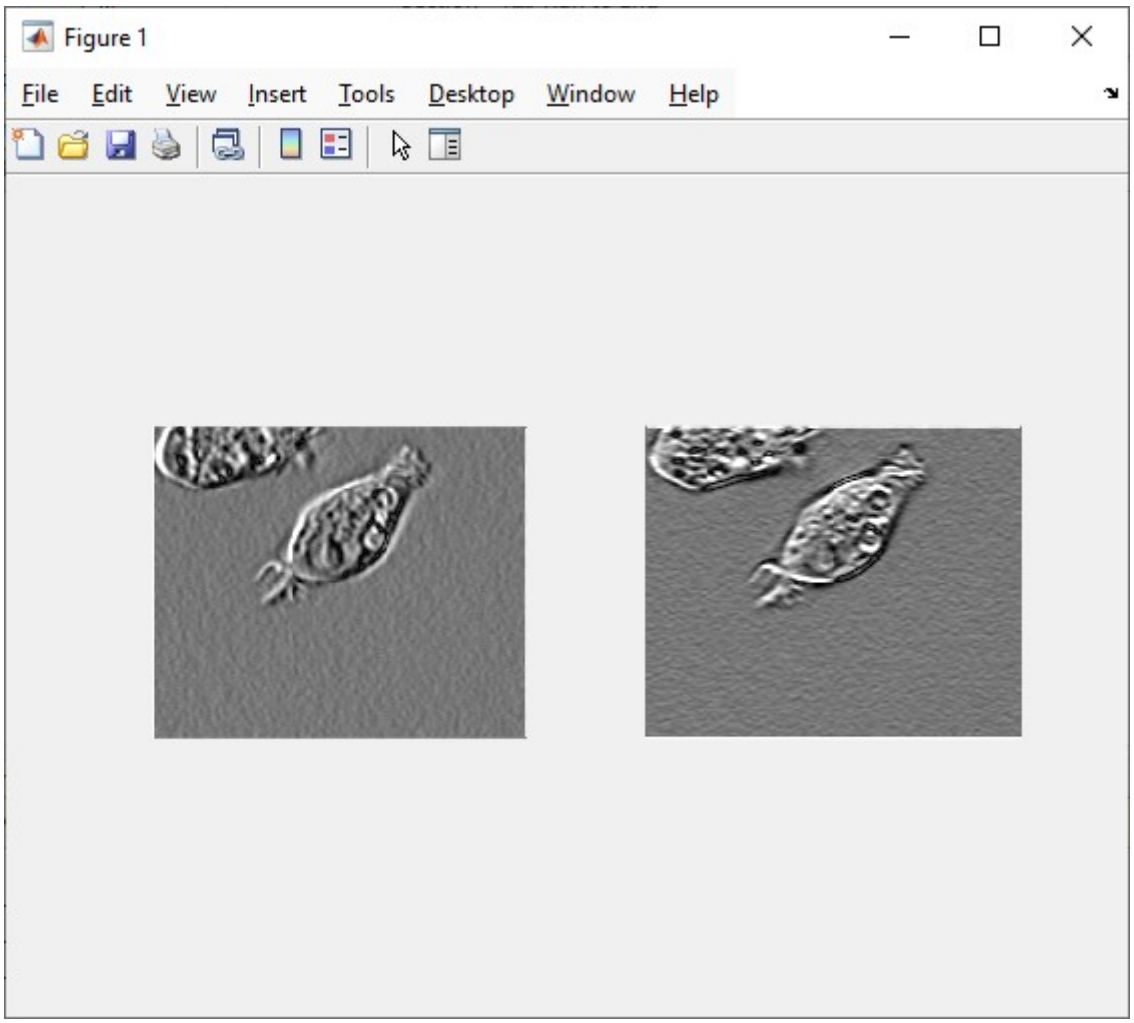
Para este caso, primero le añadimos ruido gaussiano a la imagen. Tras esto la tenemos que restaurar, así que probamos con dos filtros, el primero de ellos es el de media y el segundo el mismo usado en el apartado A. Esto nos da las siguientes imágenes:



Ejercicio 9.C:

```
I=imread('../Imagenes/cell.tif');
%Resalta verticalmente
gv=[-1 0 1;
    -1 1 1;
    -1 0 1];
%Resalta horizontalmente
gh=[ 1  1  1;
     0  1  0;
    -1 -1 -1];
Jv=abs(filter2(gv,I))/255;
Jh=abs(filter2(gh,I))/255;
figure, subplot(1,2,1), imshow(Jv);
subplot(1,2,2), imshow(Jh);
```

Creamos unos kernel parecidos a los usados en los bordes horizontales y verticales, solo que añadiendo un 1 extra para darle peso extra al pixel actual. Si aplicamos estos a la imagen obtenemos:



Ejercicio 9.D:

```
I=imread('../Imagenes/cell.tif');
%Ecualizamos el histograma
I=histeq(I);
figure, subplot(1,3,1)
imshow(I)
%Detectamos bordes con canny
B=edge(I,'canny');
subplot(1,3,2), imshow(B)
%Realzamos los bordes sumandole una pequeña cantidad de intesidad
R=double(I)/255+0.1*double(B);
subplot(1,3,3), imshow(R)
```

Para esta imagen realzamos los bordes sumándole una imagen binaria de sus bordes. Esto nos da como resultado:

Ejercicio 9.E:

```
I=imread('../Imagenes/BUG.TIF');
figure
subplot(1,2,1)
imshow(I)
I=histeq(I);
%Le aplicamos un filtro para resaltar los bordes
h=fspecial('unsharp');
J=abs(filter2(h,I));
%Le aplicamos una matriz para resaltar los bordes horizontales
gh=[ 1  1  1;
     0  0  0;
    -1 -1 -1];
B=abs(filter2(gh,J))/255;
subplot(1,2,2)
imshow(R)
```

Para realzar los bordes horizontales de la imagen, primero la ecualizo, tras esto le aplico un filtro para realzar los bordes en general y por ultimo le aplico otro para realzar solo los bordes horizontales. Esto nos da como resultado: