MEMORIA DE LAS PRÁCTICAS

Teledetección

Sergio Pérez Martín || Miguel Velasco Romero
UNIVERSIDAD DE SEVILLA

Tabla de contenido

Práctica 1

Código 1.6:

```
function y = qsort (x)
    longitud=length(x);
    if longitud <= 1
        y = x;
    else
        x1=[];
        x2=[];
        for i=1:longitud-1

            if x(i) < x(longitud)
                x1=[x1 x(i)];
            else
                x2=[x2 x(i)];
            end
        end
    y=[qsort(x1) x(longitud) qsort(x2)];
    end
end
```

Práctica 2

Código 2.3 + Imagen propia tratada (antes/después):

```
function c = corte(img, porcentaje)
    histograma=histo(img);
    total=sum(sum(img));
    percentil = total * (porcentaje /100);
    corte = ceil(percentil);
    suma=0;
    for h = 1:length(histograma)
        suma=double(suma)+double(histograma(h));
        if suma >= corte
            m=h;
            break
        end
    end
    suma=0;
    for h = length(histograma):-1:1
        suma=double(suma)+double(histograma(h));
        if suma>= corte
            M=h;
            break
        end
    end
    imshow(img)
    figure(2)
    c=expan(img,m,M);
end
```

Imagen antes de ser tratada:

Ilustración 1 - Imagen original no tratada

Imagen después de ser tratada:

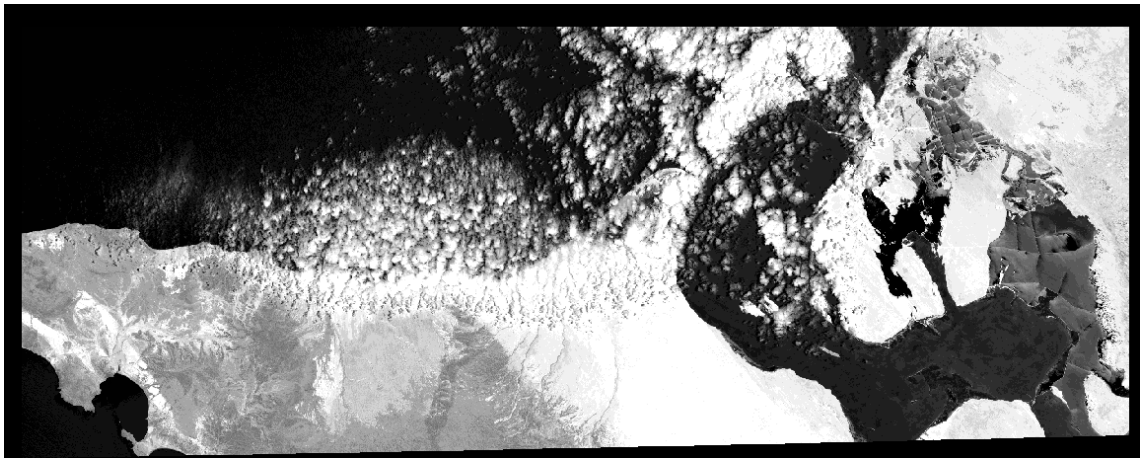

Ilustración 2 -Imagen tratada por un corte de colas al 1%

Código 2.4 + Imagen propia tratada (antes/después):

```
function img2 = ecual(img)
    histograma=histo(img);
    hitogramaAcu = zeros(1,255);
    ac = 0;
    for h = 1:255
        ac = ac + histograma(h);
        histogramaAcu(h) = ac;
    end
    [F,C] = size(img);
    pValidos = 0;
    for f = 1:F
        for c = 1:C
            nd = img(f,c);
            if nd>0
                pValidos = pValidos + 1;
            end
        end
    end
    PE = 255 / pValidos;
    im2=uint8(zeros(F,C));
    for f = 1:F
        for c = 1:C
            nd = img(f,c);
            if nd == 0
                nuevoNd = 0;
            else
                nuevoNd = histogramaAcu(nd) * PE;
            end
            im2(f,c) = nuevoNd;
        end
    end
    imshow(img)
    figure(2)
    imshow(im2)
```

Imagen antes de ser tratada:

Ilustración 3 - Imagen original no tratada

Imagen después de ser tratada:

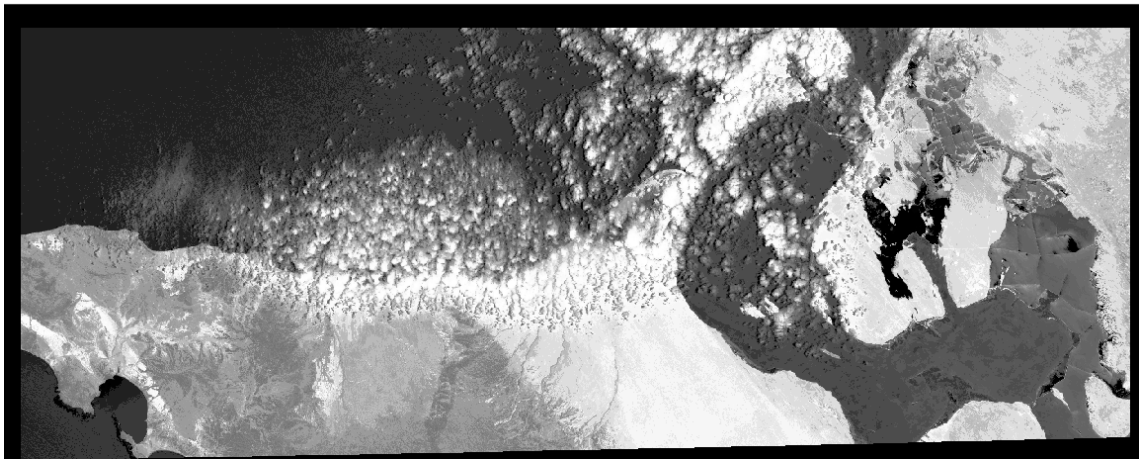

Ilustración 4 - Imagen tratada por el método de ecualización

Práctica 3

Código 3.3 + Imagen propia tratada (antes/después):

```
function im2=filtro(im, cf)
    [Fila,Columna] = size(cf);
    sumatorio = sum(sum(cf));
    cfN = cf;
    for fila = 1:Fila
        for columna = 1:Columna
            cfN(fila,columna) = double(cf(fila,columna)/sumatorio);
        end
    end
    [F,C] = size(im);
    im2= uint8(zeros(F,C));
    for f = 2:F-1
        for c = 2:C-1
            en = double(im(f-1:f+1, c-1:c+1));
            ndp = sum(sum(en .* cfN));
            im2(f,c) = ndp;
        end
    end
    imshow(im2)
end
```

Imagen antes de ser tratada:

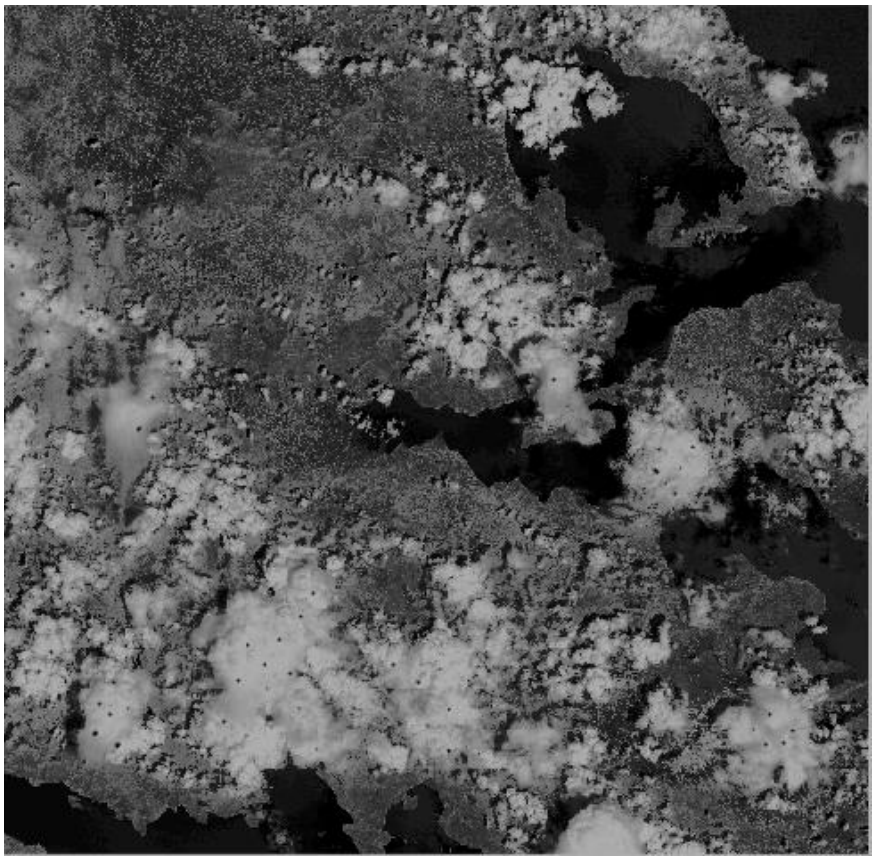

Ilustración 5 - Imagen original no tratada

Imagen después de ser tratada:

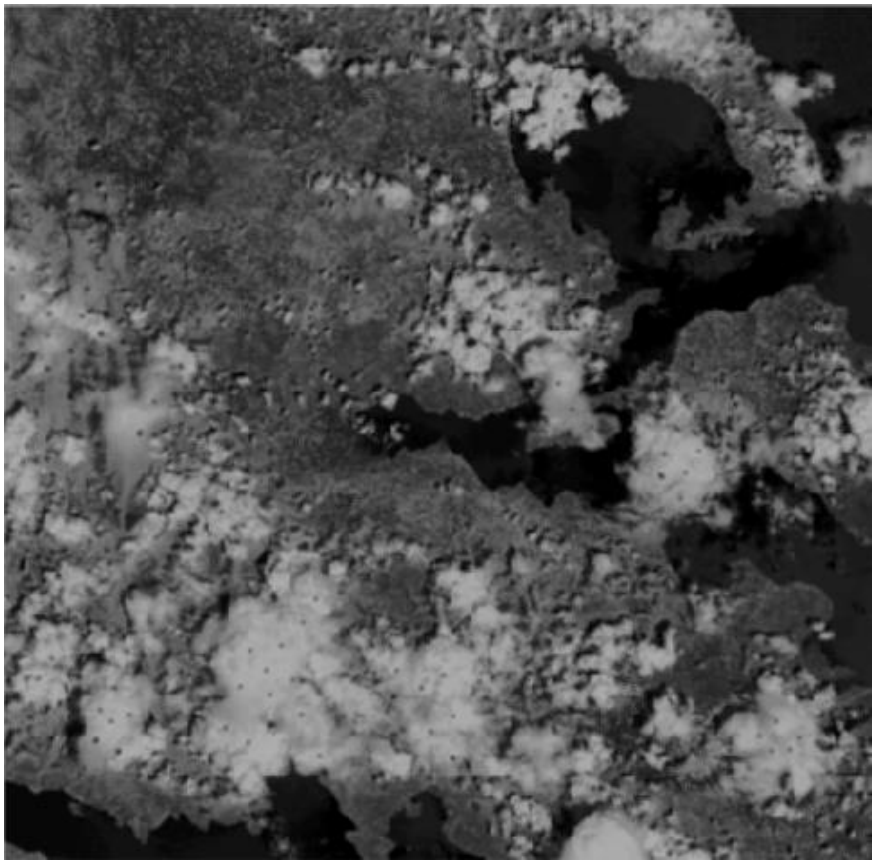

Ilustración 6 - Imagen tratada con el filtro de media

Código 3.4 + Imagen propia tratada (antes/después):

```
function im2 =filtroMedianaMejorado(im)
    [F,C] = size(im);
    im2= uint8(zeros(F,C));
    for f = 2:F-1
       for c = 2:C-1
            en = double(im(f-1:f+1, c-1:c+1));
            m = min(min(en));
            M = max(max(en));
            nd = median(reshape(en,1,9));
            if nd - m >= 40 || M - nd >= 40
                im2(f,c) = nd;
            else
                im2(f,c) = im(f,c);
            end
       end
    end
    imshow(im)
    figure(2)
    imshow(im2)
end
```

Imagen antes de ser tratada:

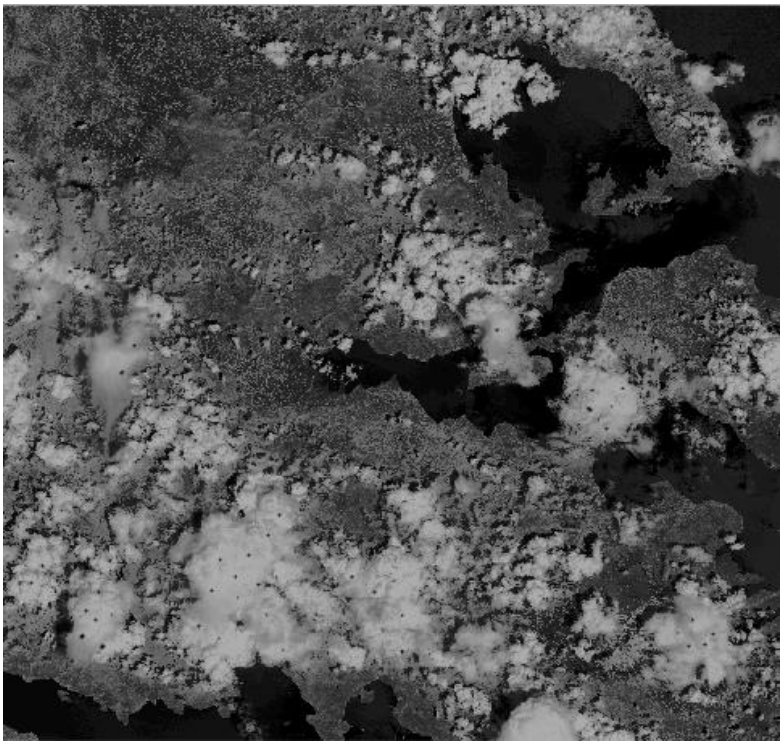

Ilustración 7 - Imagen original no tratada

Imagen después de ser tratada:

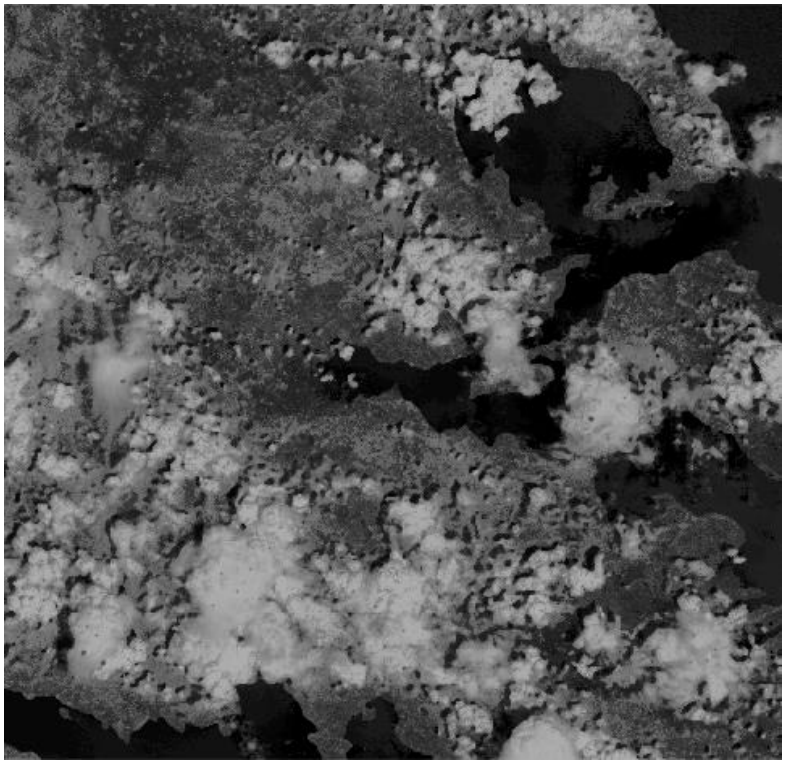

Ilustración 8 - Imagen tratada con filtro de mediana

Práctica 4

Código 4.2 + Imagen propia tratada (antes/después):

```
function img2 = amplia(img1)
    [F, C] = size(img1);
    F2 = F*2;
    C2 = C*2;
    img2 = zeros(F2, C2);
    img2(1:2:F2, 1:2:C2) = img1;
    for f = 1:2:F2
        for c = 2:2:(C2-1)
            nd1 = img2(f, c-1);
            nd2 = img2(f, c+1);
            ndp = (nd1+nd2)/2;
            img2(f,c) = ndp;
        end
    end
    for f = 2:2:(F2-1)
       f1 = img2(f-1, :);
       f2 = img2(f+1, :);
       fnd = (f1+f2)/2;
       img2(f, :) = fnd;
    end
    imshow(img1);
    figure(2)
    img2 = uint8(img2);
    imshow(img2);
end
```

Imagen antes de ser tratada:

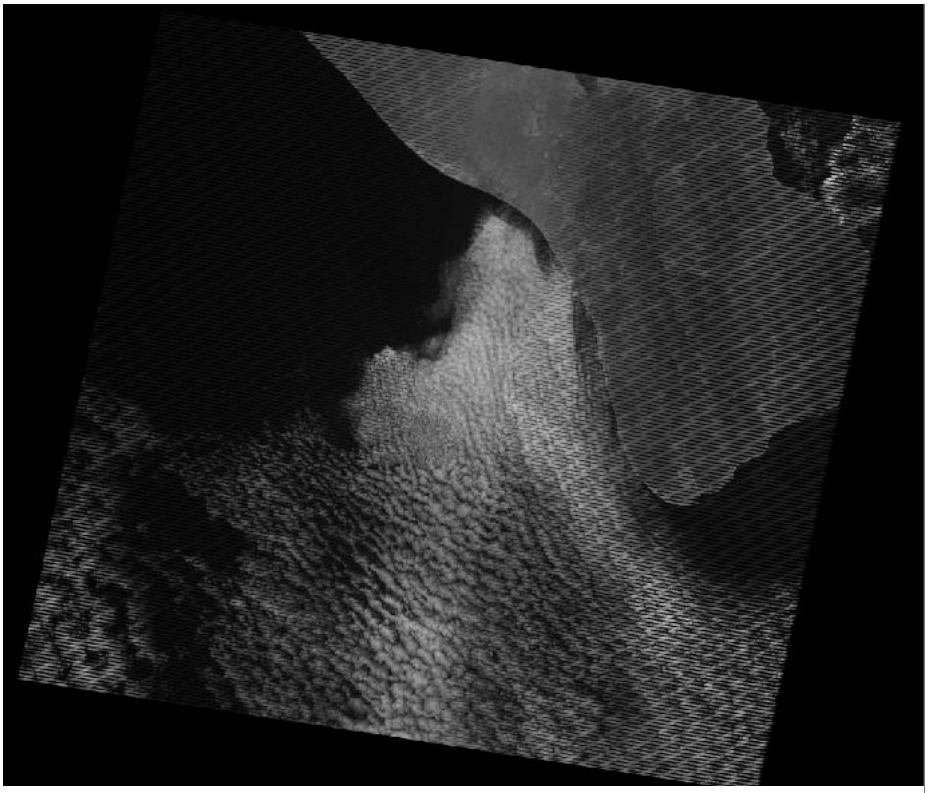

Ilustración 9 - Imagen original no tratada

Imagen después de ser tratada:

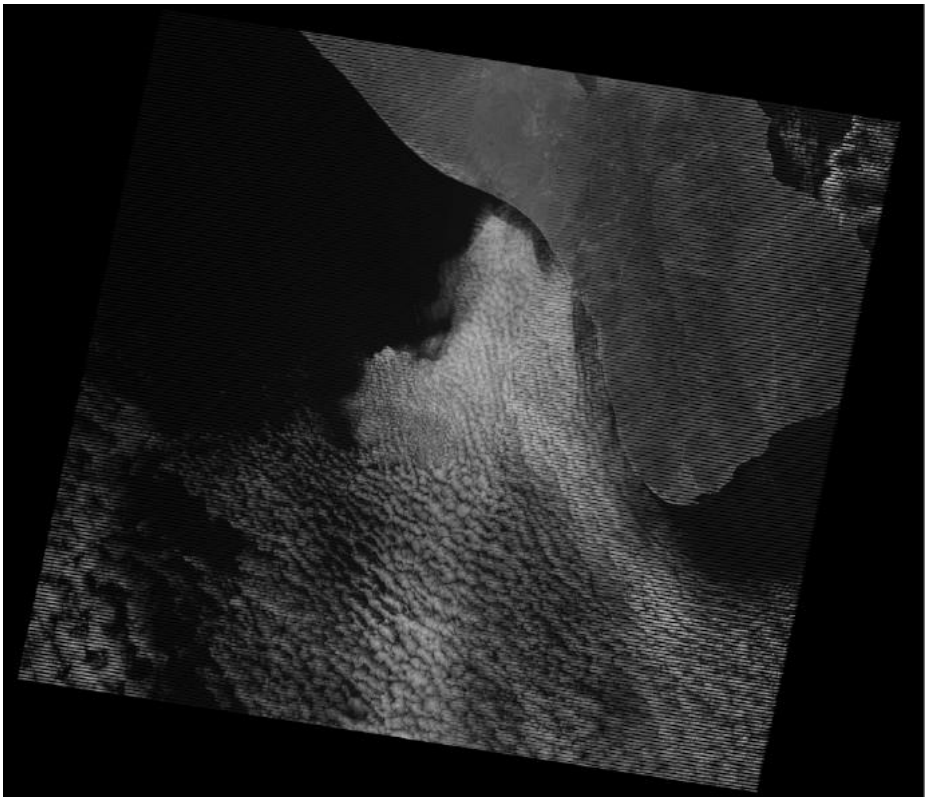

Ilustración 10 – Imagen ampliada por interpolación lineal

Código 4.4 + Imagen propia tratada (antes/después):

```
function img2 = ampliafft(img1, R)
    [F, C] = size(img1);
    FP = F * R;
    CP = C * R;
    IMG1 = fft2(img1);
    IMG2 = zeros(FP, CP);
    IMG2(1:(F/2), 1:(C/2)) = IMG1(1:(F/2), 1:(C/2));
    %Copiar img1 en los otros cuadrantes
    IMG2(FP-(F/2):FP, 1:(C/2)) = IMG1((F/2):F, 1:(C/2));
    IMG2(1:(F/2), CP-(C/2):CP) = IMG1(1:(F/2), (C/2):C);
    IMG2(FP-(F/2):FP, CP-(C/2):CP) = IMG1((F/2):F, (C/2):C);
    %Volver al dominio del espacio (ifft2) obteniendo ahora img2
    %Amplificar img2
    img2 = real(ifft2(IMG2))*((FP*CP)/(F*C));
    %Hacer cast uint8
    img2 = uint8(img2);
    %Pinta la imagen en pantalla
    imshow(img2);
end
```

Imagen antes de ser tratada:

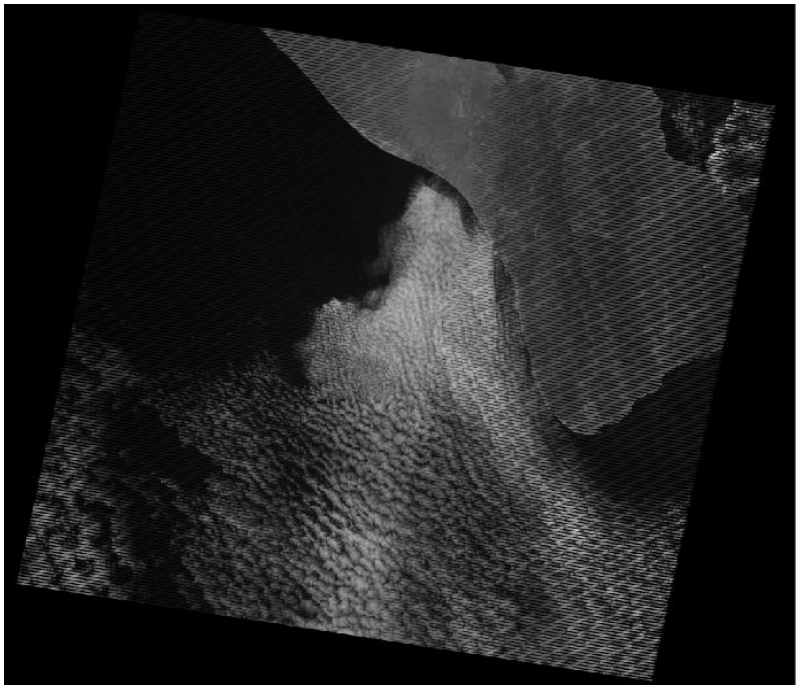

Ilustración 11 - Imagen original no tratada

Imagen después de ser tratada:

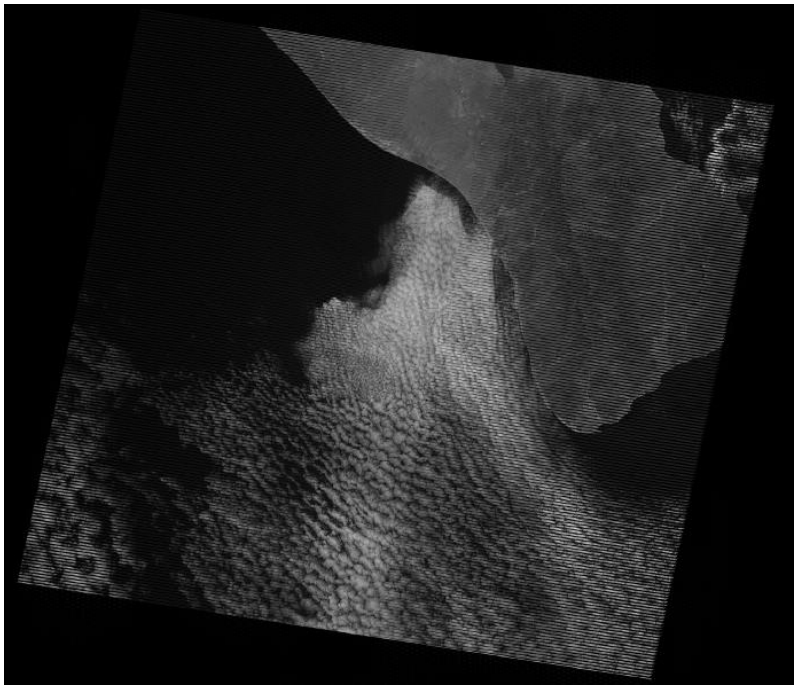

Ilustración 12 - Imagen ampliada mediante FFT (R=8)

Práctica 5

Código 5.3 + Imagen propia tratada (antes/después):

```
function img = ndvi(r, nir)
    r = im2double(r);
    nir = im2double(nir);
    [F, C] = size(r);
    img = zeros(F, C);
    for f = 1:F
        for c = 1:C
            nd = (nir(f, c)-r(f, c))/(nir(f, c)+r(f, c));
            nd = round(((nd+1)/2)*254+1);
            img(f, c) = nd;
        end
    end
    img = uint8(img);
    imshow(img);
end
```

Imagen antes de ser tratada:

Ilustración 13 - Imagen original no tratada

Imagen después de ser tratada:

Ilustración 14 - NDVI de la imagen original

Código 5.4 + Imagen propia tratada (antes/después):

```
function img2 = densi(img1, banda)
    img1 = img1(:,:,banda);
    GRIS = [128, 128, 128];
    AZUL = [0, 0, 255];
    ROJO = [255, 0, 0];
    VERDE = [0, 255, 0];
    AMARILLO = [255, 255, 0];
    [F, C] = size (img1);
    img2 = uint8(zeros(F, C, 3));
    for f = 1:F
        for c = 1:C
            nd = img1(f, c);
            if nd > 0
                if nd < 30
                    ndp = GRIS;
                else if nd < 87
                        ndp = AZUL;
                else if nd < 144
                        ndp = VERDE;
                else if nd < 231
                        ndp = AMARILLO;
                    else
                        ndp = ROJO;
                    end
                    end
                    end
                end
                img2(f, c, : ) = ndp;
            end
        end
    end
    imshow(img2);
end
```

Imagen antes de ser tratada:

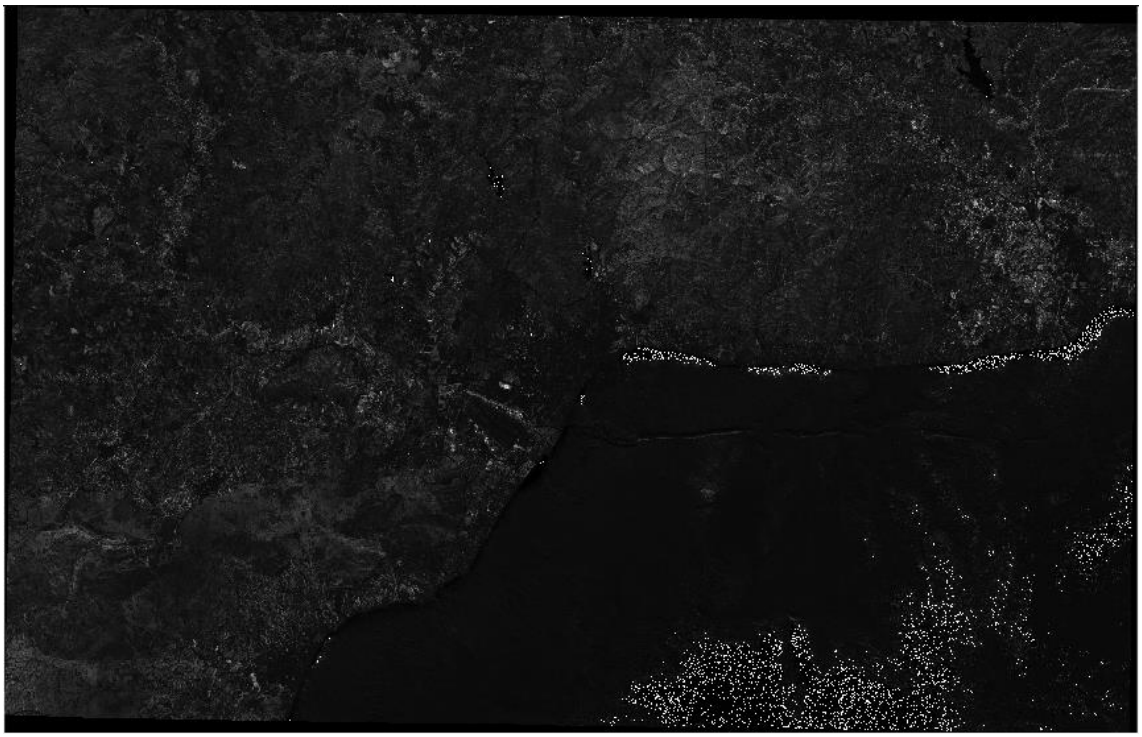

Ilustración 15 – NDVI de la imagen original

Imagen después de ser tratada:

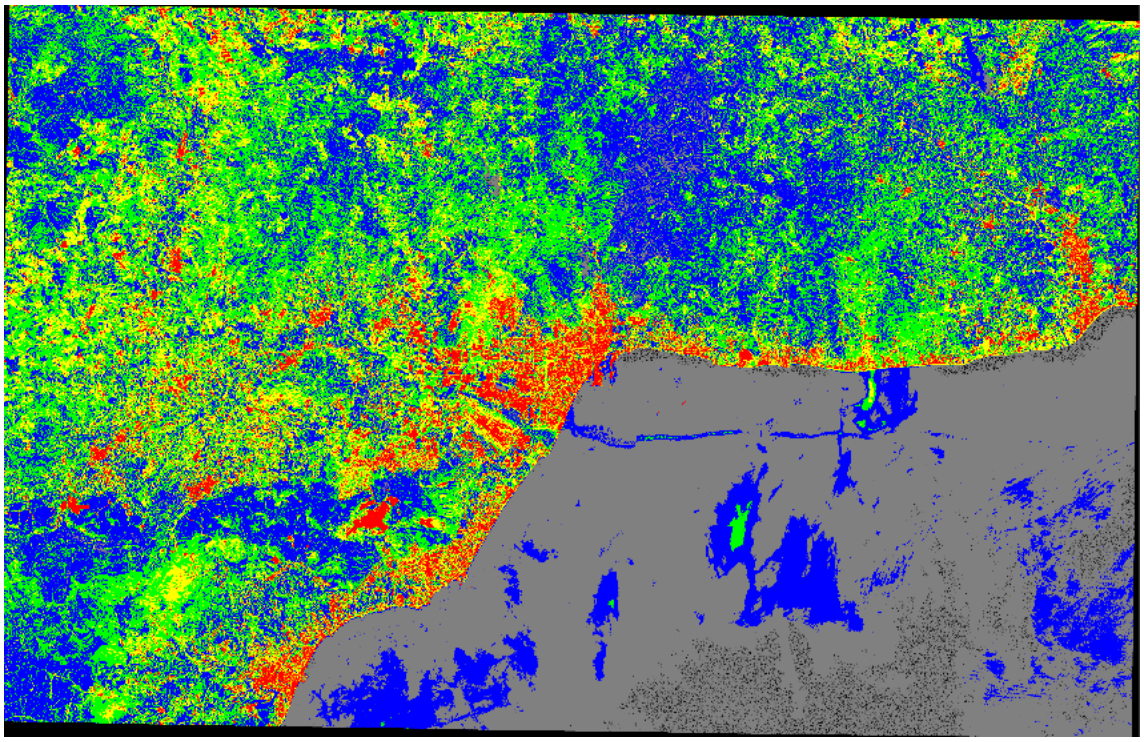

Ilustración 16 - Density Slicing sobre NDVI de la imagen original

Código 5.5 + Imagen propia tratada (antes/después):

```
function img2 = cortehsv(img1)
    img2 = rgb2hsv(img1);
    img2 = uint8(img2 .* 255);
    S = img2(:, :, 2);
    [F, C] = size(S);
    for f = 1:F
        for c = 1:C
            nd = S(f, c);
            if nd < 20
                img2(f, c, 2) = 1;
            else if nd > 30
                    img2(f, c, 2) = 255;
                end
            end
        end
    end
    img2 = double(img2) ./ 255;
    img2 = hsv2rgb(img2);
    img2 = uint8(img2 .* 255);
    imshow(img2);
end
```

Imagen antes de ser tratada:

Ilustración 17 - Imagen original no tratada

Imagen después de ser tratada:

Ilustración 18 - Corte de colas en HSV a la imagen original

Práctica 6

Código 6.1:

```
function img2 = isodata(img1, Cat)
    [F, C, B] = size(img1);
    img2 = uint8(zeros(F, C, B));
    colores = cell(1, Cat);
    for c = 1:Cat
        colores{c} = [randi(255), randi(255), randi(255)];
    end
    %Colocamos los centroides (puntos intermedios que determinan a que
    %categoria va cada pixel)
    b = 0;
    c = 0;
    centroides = cell(1, Cat);
    while b < Cat
        while c < Cat
            y = round(rand * F);
            x = round(rand * C);
            if y ~= 0 && x ~= 0
                c = c + 1;
            end
            for banda = 1:B
                centroides{c} = double(img1(y, x, banda));
            end
            if prod(centroides{c}) > 0
                b = b + 1;
            end
        end
    end
    bandas = zeros(1, B);
    distancias = zeros(1, Cat);
    grupos = cell(1, Cat);
    for n = 1:10
        for f = 1:F
            for c = 1:C
                for banda = 1:B
                    bandas(banda) = img1(f, c, banda);
                end
                nd = double(bandas);
                if prod(nd) > 0
                    for l = 1:length(centroides)
                        distancias(l) = norm(nd - centroides{l});
                    end
                    distanciaMenor = min(distancias);
                    categoria = find(distancias == distanciaMenor, 1,
'last');
                    img2(f, c, :slight_smile: = colores{categoria};
                    grupos{categoria} = {grupos{categoria} ; nd};
                end
            end
        end
        for l = 1:length(centroides)
            centroides{l} = mean(centroides{l});
        end
        imshow(img2)
    end
end
```

Imagen antes de ser tratada:

Ilustración 19 - Imagen original no tratada

Imagen clasificada en 5 categorias:

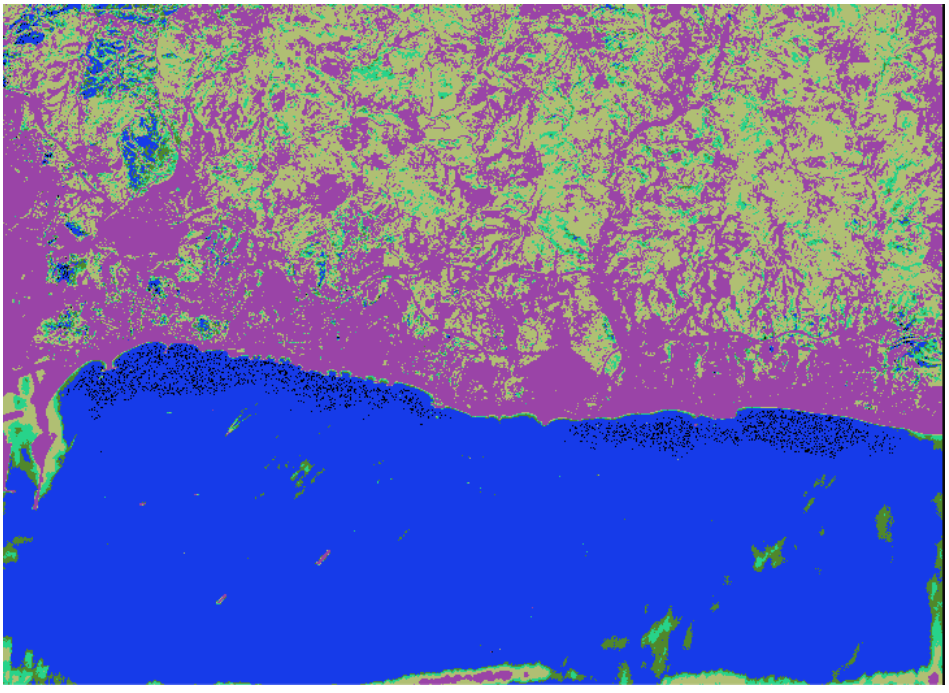

Ilustración 20 - Clasificación de la imagen original por ISODATA

Imagen clasificada en 3 categorias:

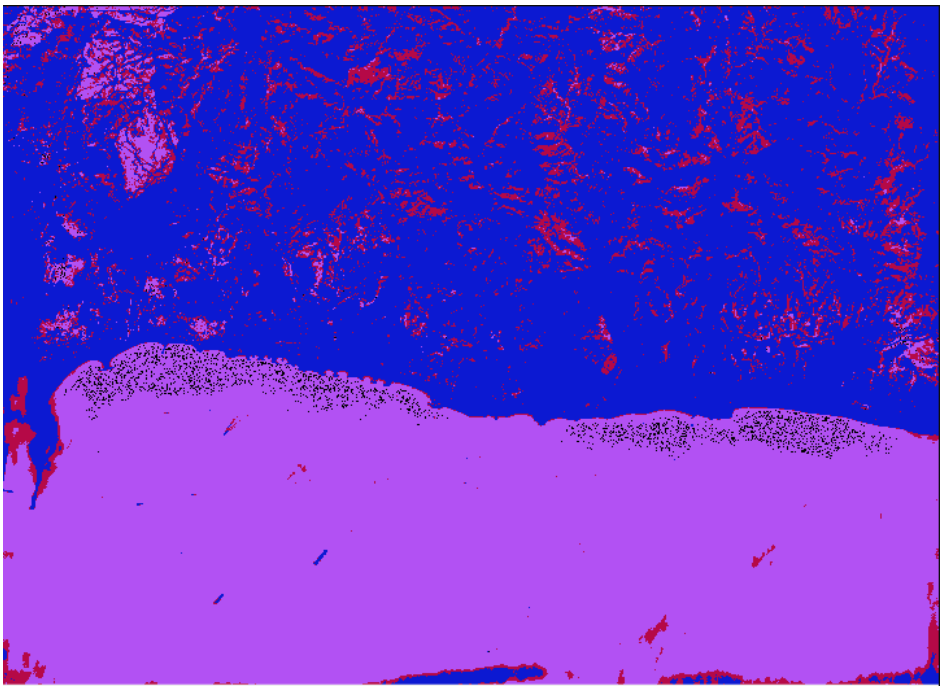

Ilustración 21 - Clasificación de la imagen original por ISODATA

Práctica 7

Código 7.1:

```
function img2 = geocorreccion(img1, banda)
    img1 = img1(:, :, banda);
    xGCP = [524625.4; 459306; 258512.5; 700009.6; 575802.7;
281661.5;...
        735857.1; 500650.5; 458620.8; 505291.2; 474595.9; 622546.4;...
        563010.9];
    yGCP = [4681912; 4364328.4; 4293178; 4163595; 4065420;
3991327.1;...
        4037990.8; 4094319.9; 4283915.4; 4451251.7; 4750412.2;
5100667;...
        5187212.2];
    cGCP = [1770; 1694; 1524; 1420; 1263; 893; 811; 581; 596; 691;
763;...
        1550; 1509];
    fGCP = [590; 869; 916; 1020; 1088; 1104; 1050; 945; 770; 635;
352;...
        166; 79];

    unos = ones(length(xGCP), 1);
    a = regress(fGCP, [unos, xGCP, yGCP]);
    b = regress(cGCP, [unos, xGCP, yGCP]);

    [F, C] = size(img1);
    img2 = uint8(zeros(F+500, C+500));
    x = -750000;
    y = 5400000;
    xTemp = x;
    for f = 1:F+500
        for c = 1:C+500
            fn = a(1) + a(2) * x + a(3) * y;
            cn = b(1) + b(2) * x + b(3) * y;
            x = x + 1000;
            if fn >= 1 && fn <= F && cn >= 1 && cn <= C
                img2(f, c) = img1(round(fn), round(cn));
            end
        end
        x = xTemp;
        y = y - 1000;
    end
    imshow(img2);
end
```

Imagen recortada previa a la corrección:

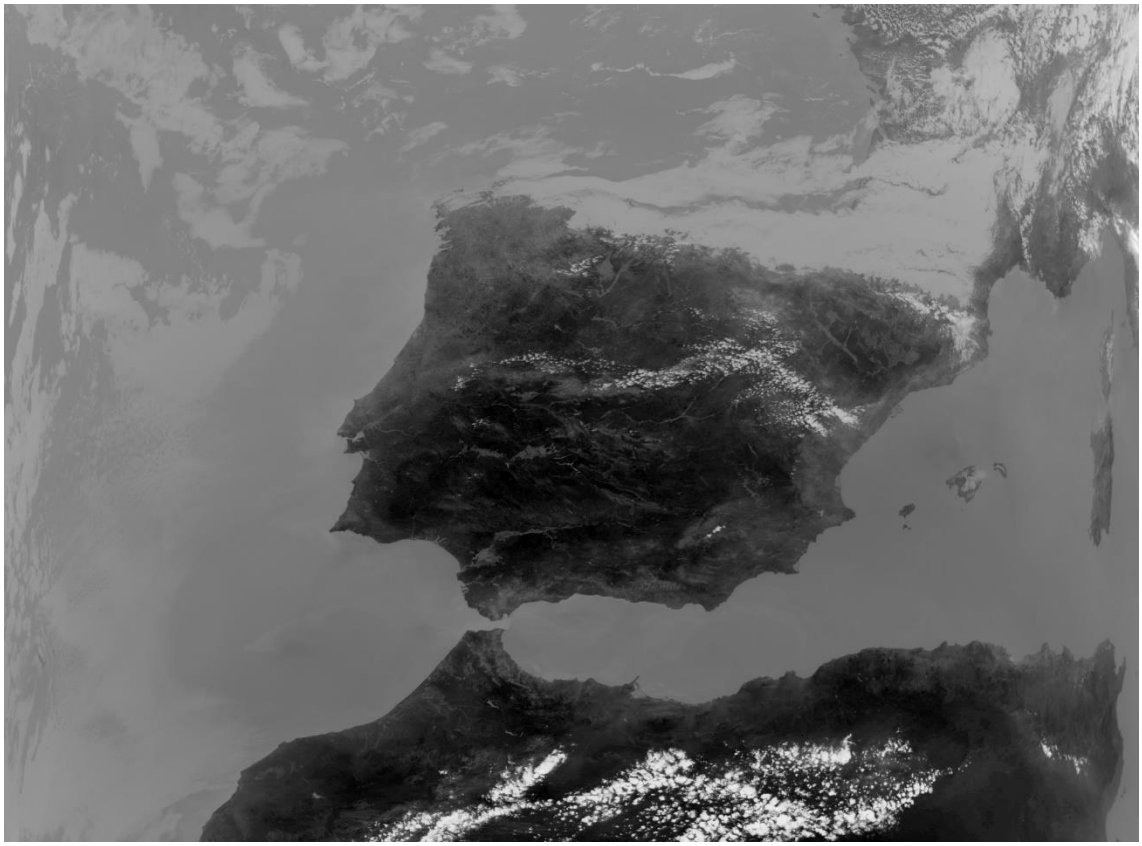

Ilustración 22 -- Imagen NOAA-ORBIT original recortada (banda 4)

Captura de Geoplanner con puntos de control:

Ilustración 23 -Selección de puntos de control en Geoplanner

Imagen corregida geométricamente:

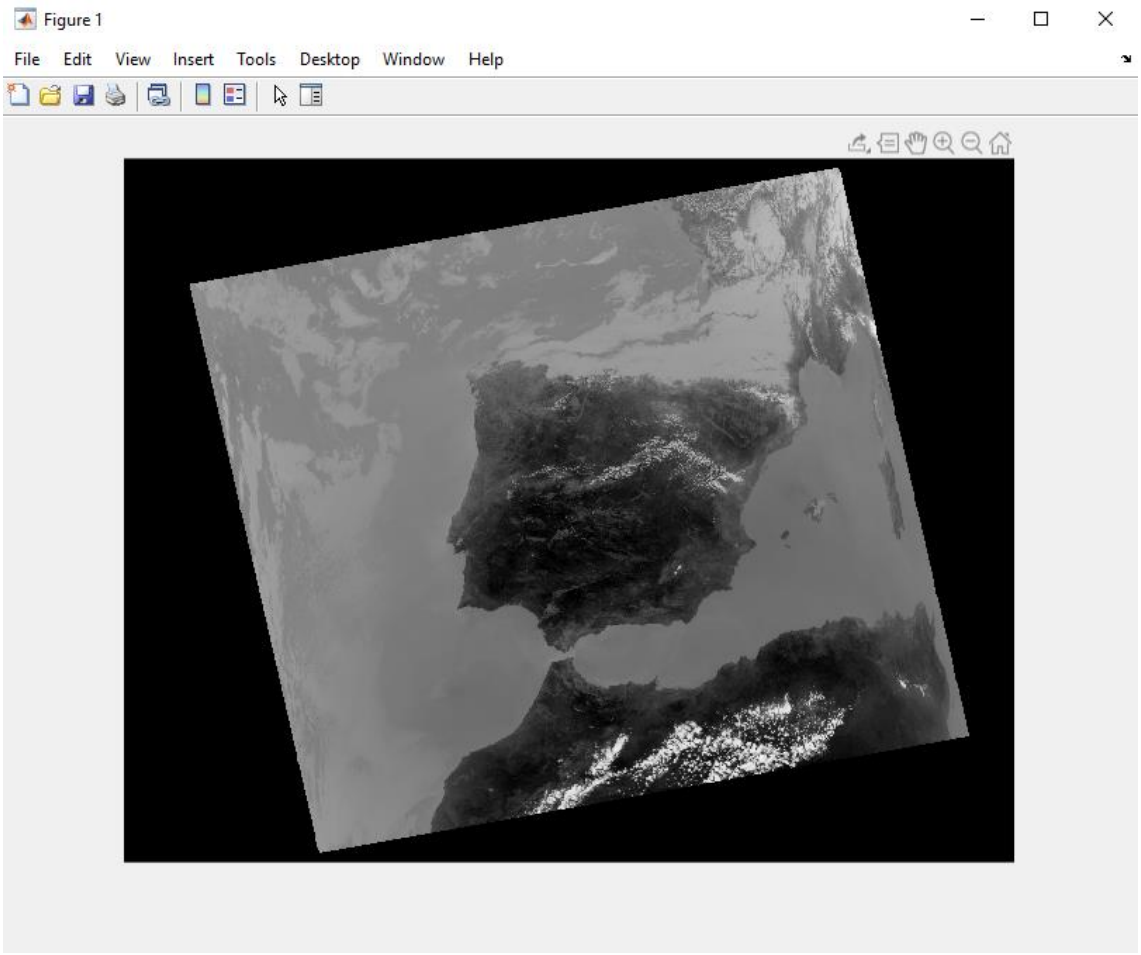

Ilustración 24 - Imagen corregid geométricamente