

Proyecto: Imágenes HDR (High Dynamic Range)

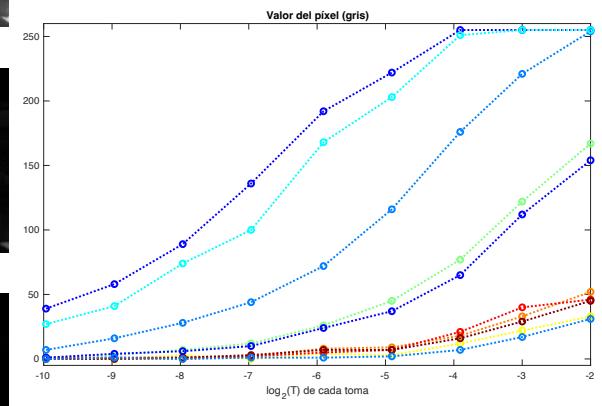
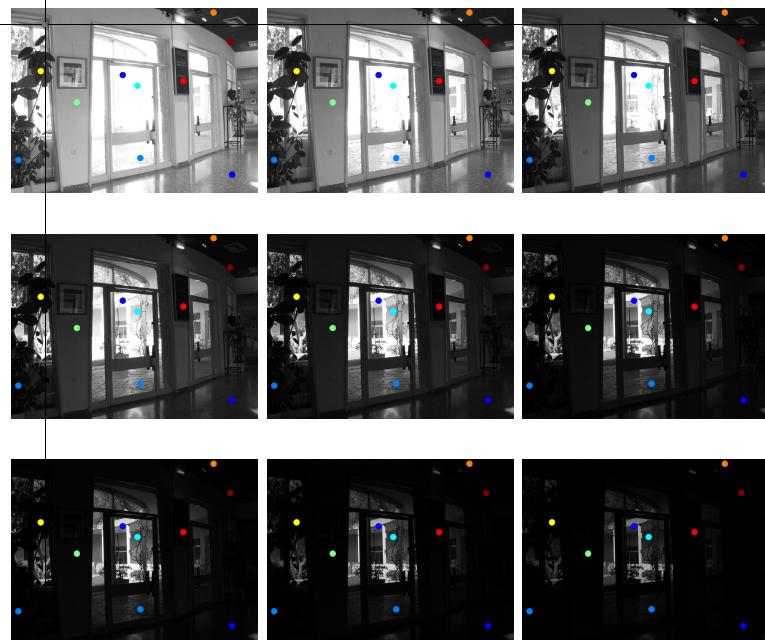
Apellidos, Nombre: Fernández Cabrita, Alejandro

Apellidos, Nombre: Evangelista Sarabia, Carlos

Apellidos, Nombre: De Almeida Pissarra, Inês

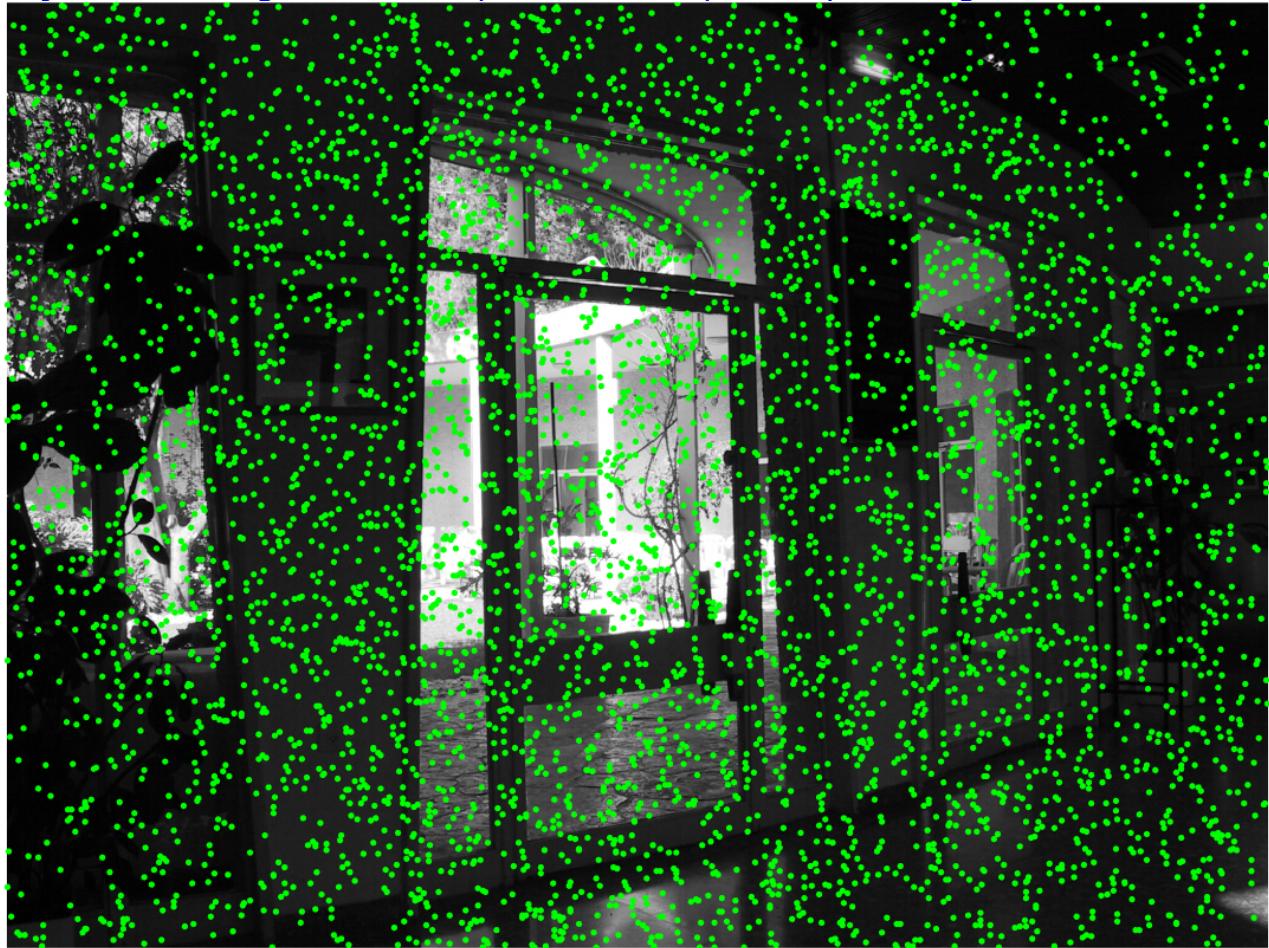
1) Lectura de imágenes y extracción de datos a usar

Adjuntad una captura de la figura tras pinchar en 10 o 15 puntos variados



Extracción de datos en N puntos de las P tomas

Adjuntad la imagen obtenida para N=5000 puntos y el código de vuestra función.



```

function Zdata=extraer_datos(hdr_data,N)
    [alto, ancho, P] = size(hdr_data);
    Zdata = zeros(P, N);
    P2 = zeros(P, 1);
    n = 1;
    figure();
    imshow(hdr_data(:,:,:,5)/255);
    while n~>N
        i=floor(rand*(alto))+1;j=floor(rand*(ancho))+1;
        for k=1:P
            P2(k) = hdr_data(i, j, k);
        end
        if all(diff(P2)>=0)
            Zdata(:, n) = P2;
            n = n + 1;
            hold on;
            plot(j, i,'g.');
            hold off;
        end
    end
end
    
```

2) Resolución de las ecuaciones para obtener la función g(z)

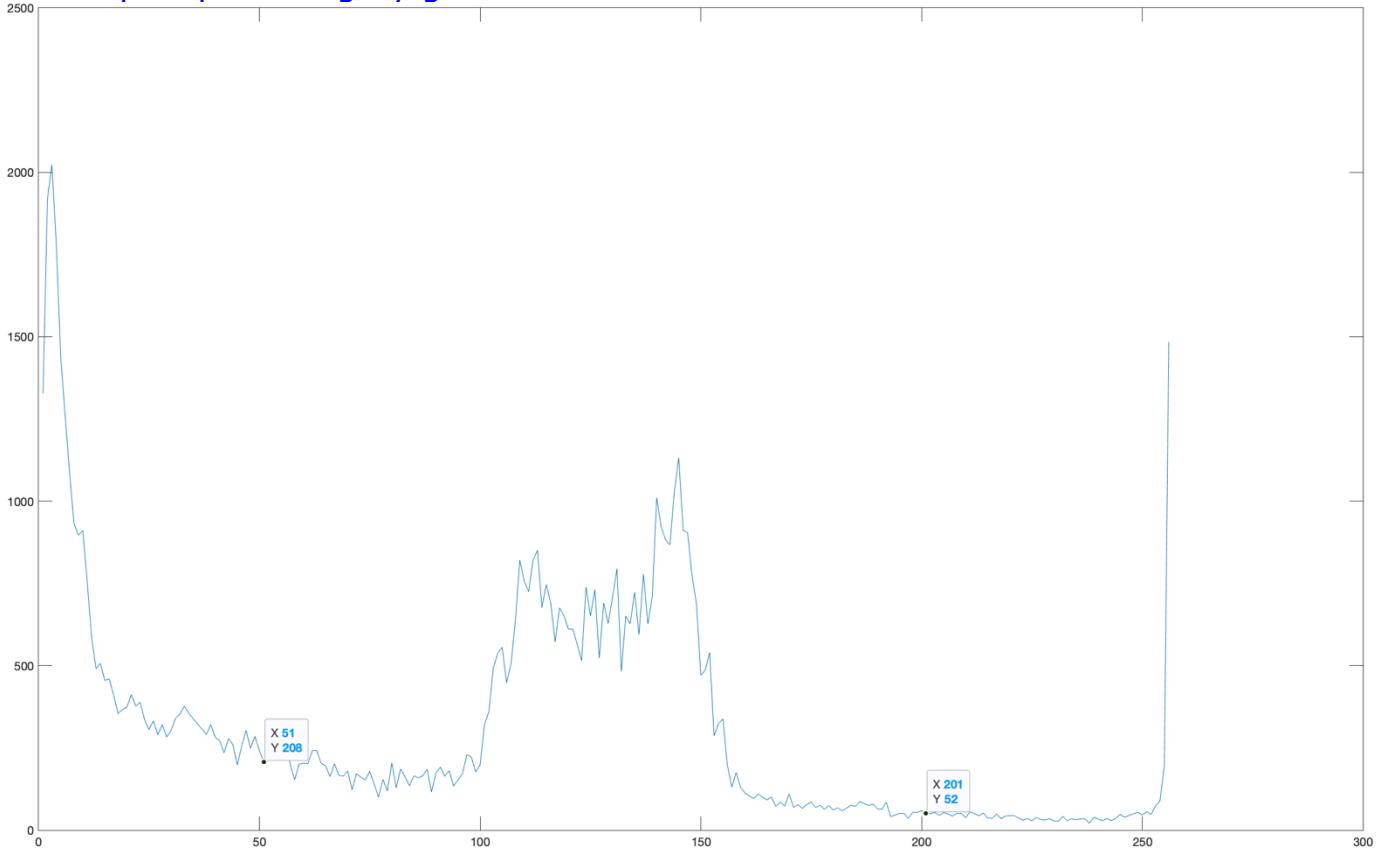
Para N=10000 puntos, determinar el número de entradas no nulas de la matriz H y qué tanto por ciento suponen.

$$N = 10000 \quad P = 9$$

$$2*N*(P-1)+254*3 = 160762$$

$$160762/\text{Neq}*\text{Nincog} = 160762/(N*(P-1) + 255)*256 \quad 160762/20545280 = 0.7825\%$$

Cread la matriz H usando N=5000 puntos y adjuntad un plot de sum(H~=0). ¿Qué representan los valores de esta gráfica? A partir de la gráfica, indicad el número de veces que aparecen g_{50} y g_{200} en vuestras ecuaciones.



Representan el número de veces que aparecen g_{x-1} (si queremos g_{50} tenemos que buscar en esta gráfica X = 51). $g_{50} = 208$; $g_{200} = 52$

Uso de pesos: Indicad el peso que usaríais para la ecuación $g_{50} - g_{120} = \dots$

El peso es $W = \text{sqrt}((w(50+1)*w(120+1))) = 0.6341$

Adjuntad el código para crear H y b incorporando pesos.

```

function g = solve_G(Zdata,T)
[P, N] = size(Zdata);
Neq = N*(P-1) + 255;
b = zeros(Neq,1);
i = zeros(2*N*(P-1)+254*3, 1);
j = zeros(2*N*(P-1)+254*3, 1);
v = zeros(2*N*(P-1)+254*3, 1);

Z = zeros(P, 1);
contador = 1; indx = 1;
M = 256; t=(1:M)'/(M+1); w=(t.*(1-t)).^2; w=w/max(w);
for n=1:N
    Z = Zdata(:, n);
    dif = Z-128;
    [~, ref] = min(abs(dif));
    Zref = Z(ref);
    for k=[1:(ref-1) (ref+1):P]
        W = sqrt((w(Z(k)+1)*w(Zref+1)));
        i(indx) = contador; j(indx) = Z(k)+1; v(indx) = 1 * W;
        indx = indx+1;
        i(indx) = contador; j(indx) = Zref+1; v(indx) = -1 * W;
        indx = indx+1;
        b(contador) = log2(T(k)/T(ref)) * W;
        contador = contador + 1;
    end
end

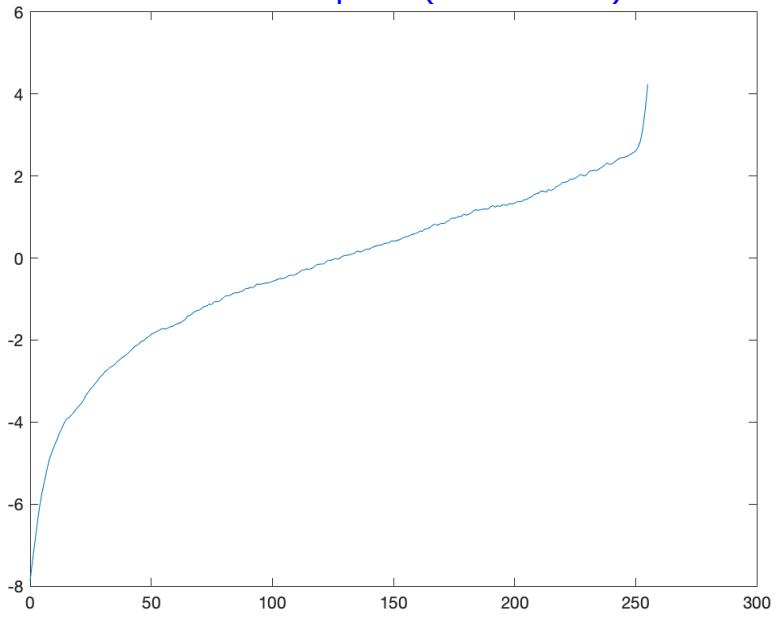
lambda = 1;
for a=1:254
    i(indx) = contador; j(indx) = a; v(indx) = -1 * lambda;
    indx = indx+1;
    i(indx) = contador; j(indx) = a+1; v(indx) = 2 * lambda;
    indx = indx+1;
    i(indx) = contador; j(indx) = a+2; v(indx) = -1 * lambda;
    indx = indx+1;
    contador = contador + 1;
end
i(indx) = contador; j(indx) = 129; v(indx) = 1 * lambda;

H = sparse(i,j,v,Neq,256);
end

```

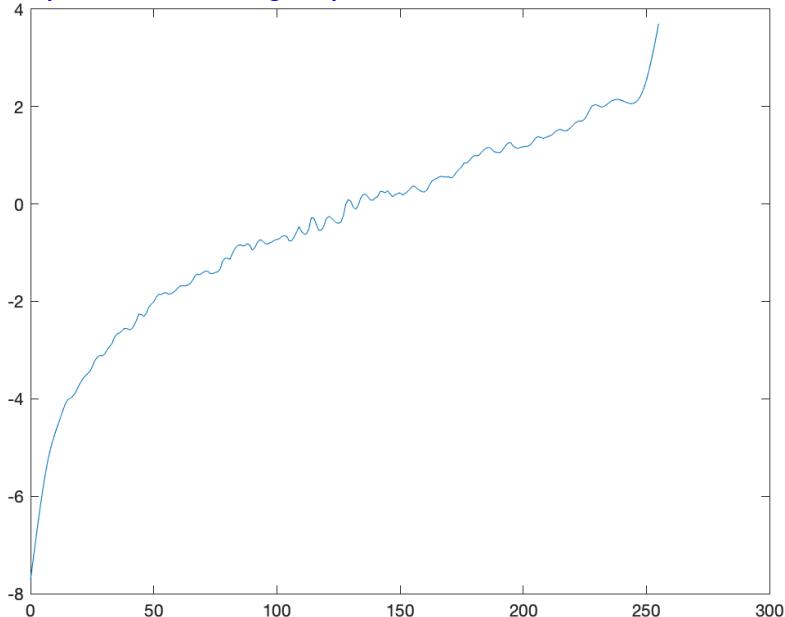
Resolución del sistema para obtener $g(z)$:

Usando $N=10000$ y $\lambda=1$, resolver y adjuntad una gráfica de la función $g(z)$ en función del valor de píxel (de 0 a 255). ¿Es la función $g(Z)$ estrictamente creciente?



No es estrictamente creciente.

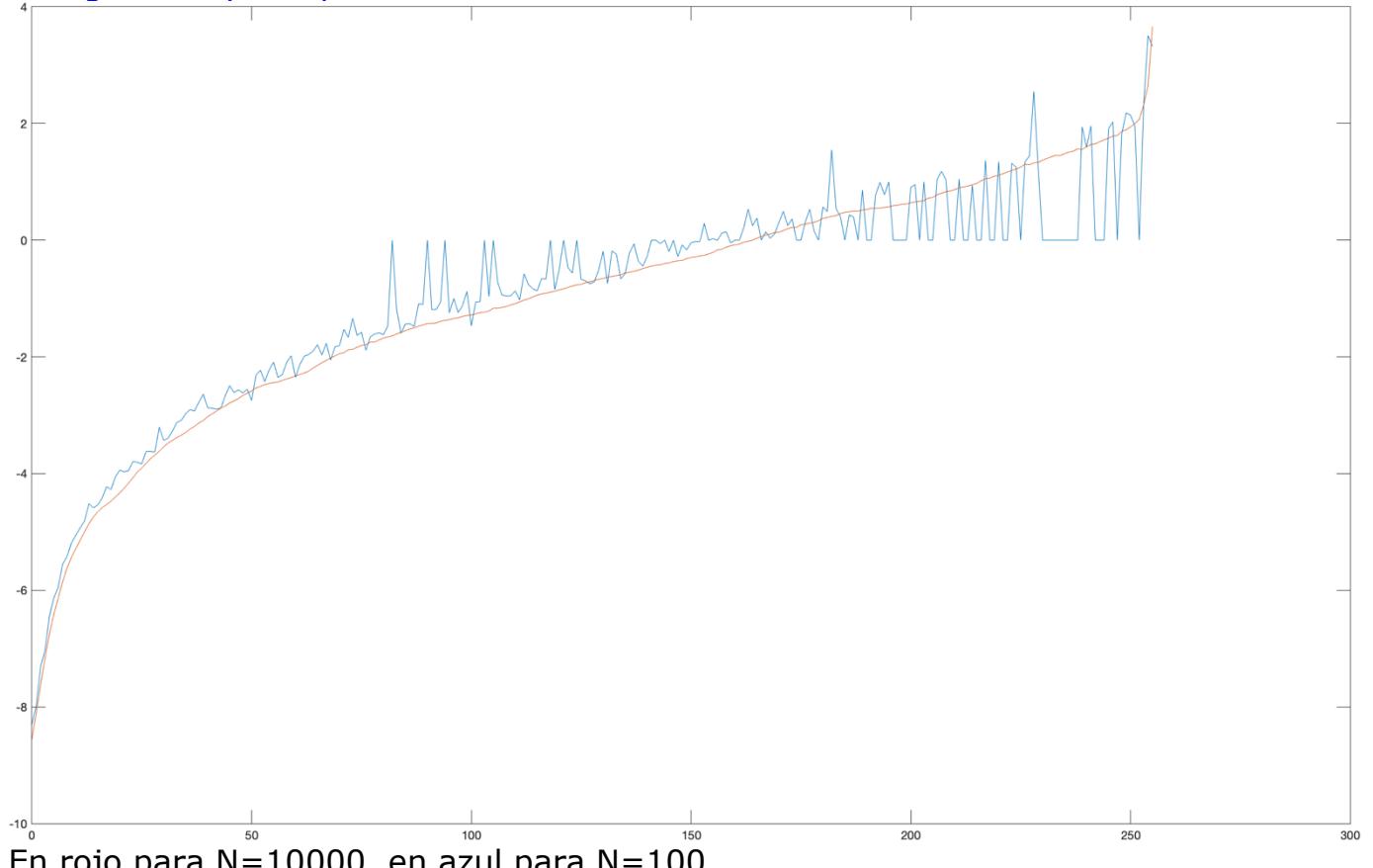
Resolver para $N=100$ puntos y adjuntad la nueva curva $g(Z)$. ¿Es ahora más visible el problema? ¿Qué podríamos hacer en este último caso para intentar arreglarlo?



Ahora el problema es más visible.

Si aumentamos λ , la función tiende a ser estrictamente creciente. Sin embargo, también convierte a g en una función un poco más incorrecta. Por ello, la mejor solución en este último caso es aumentar el número de puntos (N), ya que con $N=100$ no se garantiza que tengamos distribución de puntos en zonas oscuras, medias, claras.

Superponer en una gráfica las curvas $g(Z)$ obtenidas para $N=100$ y 100000 puntos con $\lambda=0$. Adjuntad la gráfica y explicar por qué para $N=100$, ciertos valores de $g(Z)$ son igual a 0 y completamente erróneos.



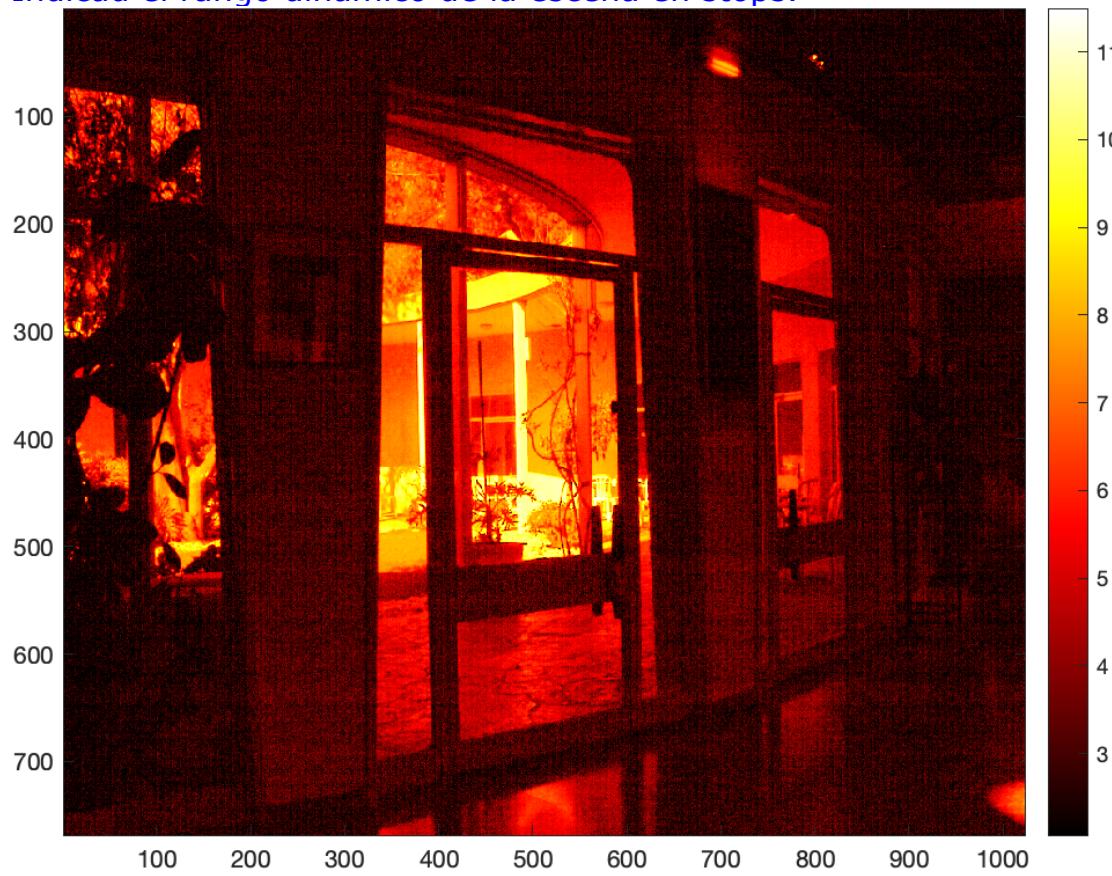
En rojo para $N=10000$, en azul para $N=100$.

Dado que lambda es 0, si no tenemos un píxel con valor x en ninguno de los puntos, $g(x)$ no aparecerá en las ecuaciones. Para $N = 100$ habrá muchos valores que no aparecerán, y por lo tanto en la gráfica de $g(Z)$ tendrán valor 0. (De la gráfica de $\text{sum}(H \sim 0)$ en un ejercicio anterior, que tenía $N=5000$, podemos ver que los puntos que menos aparecen son los que ahora em esta gráfica más aparecen com el valor 0).

3) Estimar la radiancia R de la escena

Adjuntad el código usado (puede hacerse con una sola línea de MATLAB).
 $\log2R = g(\text{hdr_data}(:,:,1)+1)-\log2(T(1));$

Adjuntad la imagen resultante.
Indicad el rango dinámico de la escena en stops.



El rango es 11 stops

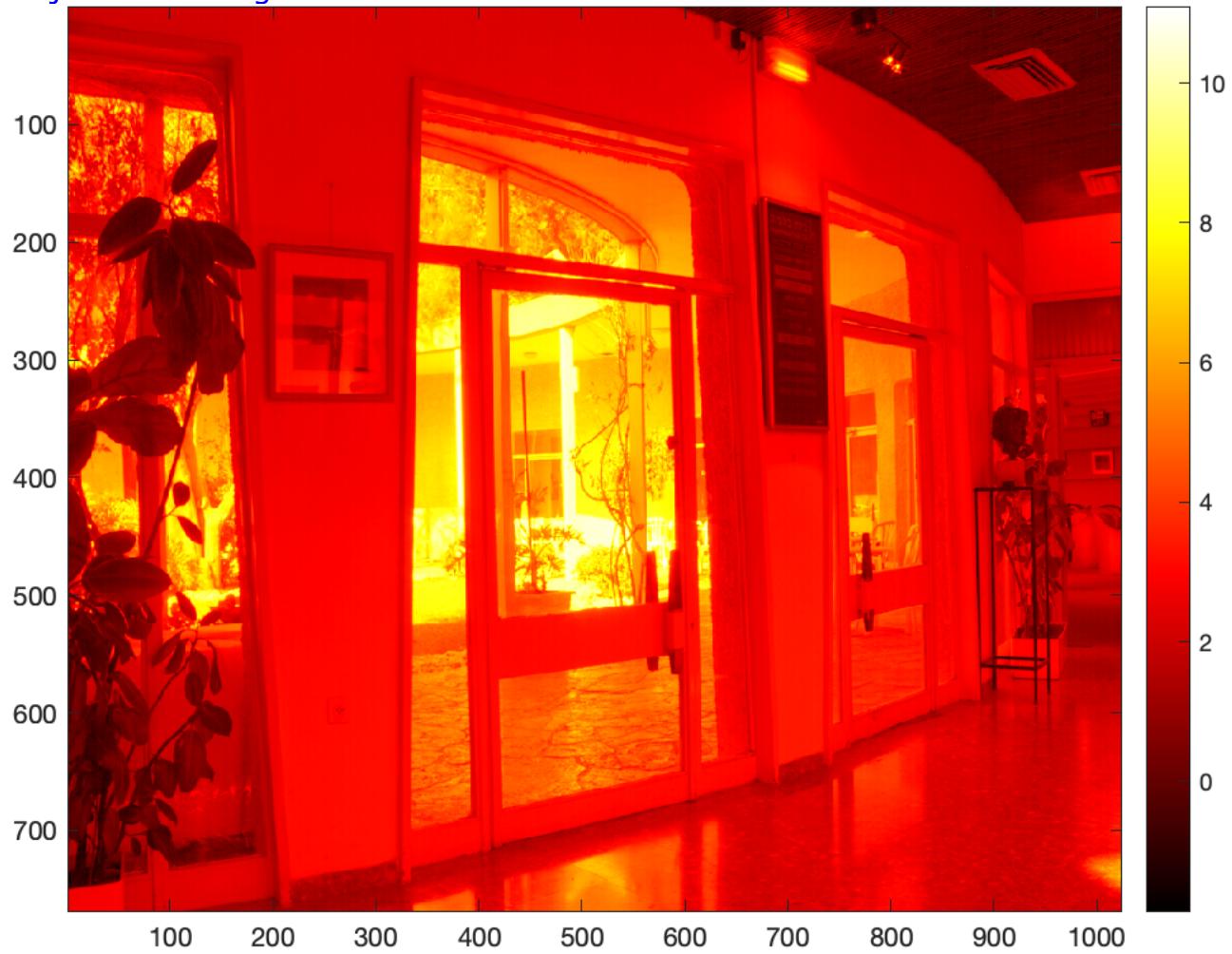
Adjuntad vuestro código de get_log2R().

```
function log2R = get_log2R(hdr_data,g,T)
    [alto, ancho, ~] = size(hdr_data);
    log2R = zeros(alto, ancho);
    M = 256; t=(1:M)'/(M+1); w=(t.*(1-t)).^2; w=w/max(w);
    for i=1:alto
        for j=1:ancho
            Z = hdr_data(i, j, :);
            V = g(Z(1,1,:)+1)-log2(T');

            dif = Z-128;
            [~, ref] = min(abs(dif));
            Zref = Z(ref);

            W = sqrt((w(Z+1)*w(Zref+1)));
            W = W/sum(W(:));
            log2R(i, j) = sum(V.*W);
        end
    end
end
```

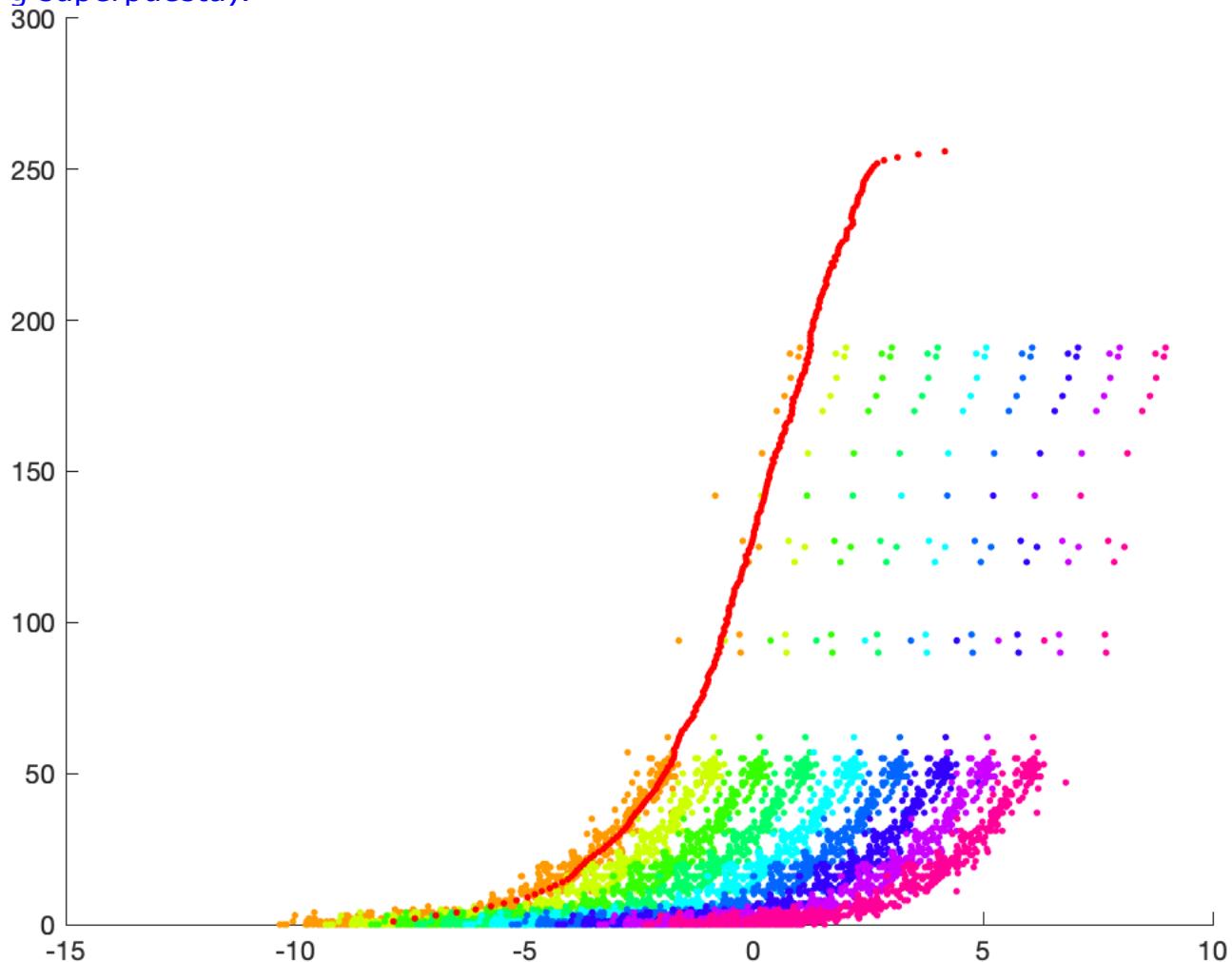
Adjuntad la imagen resultante.



¿Cuál es ahora el rango dinámico de la escena en stops? ¿Es menor o mayor que antes? Justificar.

Es menor, la escena ahora tiene un rango dinámico de 10 stops. Esto es debido a la pérdida de detalle resultante de hacer la media ponderada para hallar la radiancia.

Adjuntad la gráfica resultante (datos de la fila alto/2 de todas las tomas + solución superpuesta).



Visualización de la imagen HDR:

Indicad máximo y mínimo de la radiancia obtenida y rango dinámico.

Max = 2330.14949854372

Min = 0.230641218817494

Rango Dinámico = 10102.9187692056

Adjuntad código usado para re-escalar e imagen resultante. ¿Cuál es el problema?

```
M = max(hdr(:));
```

```
m = min(hdr(:));
```

```
hdr = (hdr-m)/(M-m);
```

El problema es que no se ha conseguido asignar a cada valor de la radiancia con un alto rango dinámico, un valor de luminancia con un bajo rango dinámico (para su representación en las pantallas), de forma que sean visibles los detalles de las zonas oscuras y claras. Este es el problema que buscan solucionar los algoritmos de "tone mapping".

Probad con: `rgb=tonemap(hdr, 'AdjustSaturation', 3);` y adjuntad la imagen resultante usando `imshow()`.



Adjuntad dentro de esta entrega (en un .zip o similar) un script `crear_hdr.m` que genere la imagen HDR y la muestre. Incluid en vuestro script todas las funciones (`get_log2R`, `extraer_datos`, `solve_G`, ...) necesarias para que funcione (no hace falta incluir las imágenes de partida).

4) Fusión de exposiciones usando la pirámide laplaciana

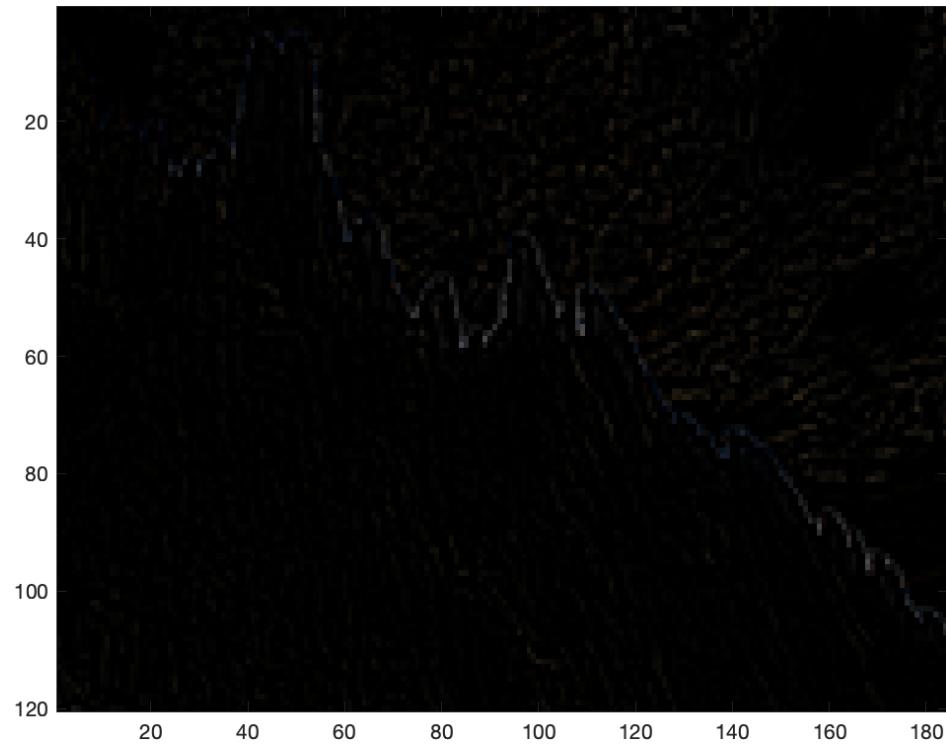
Pirámide Laplaciana:

Adjuntar código de vuestra función `lap.m`

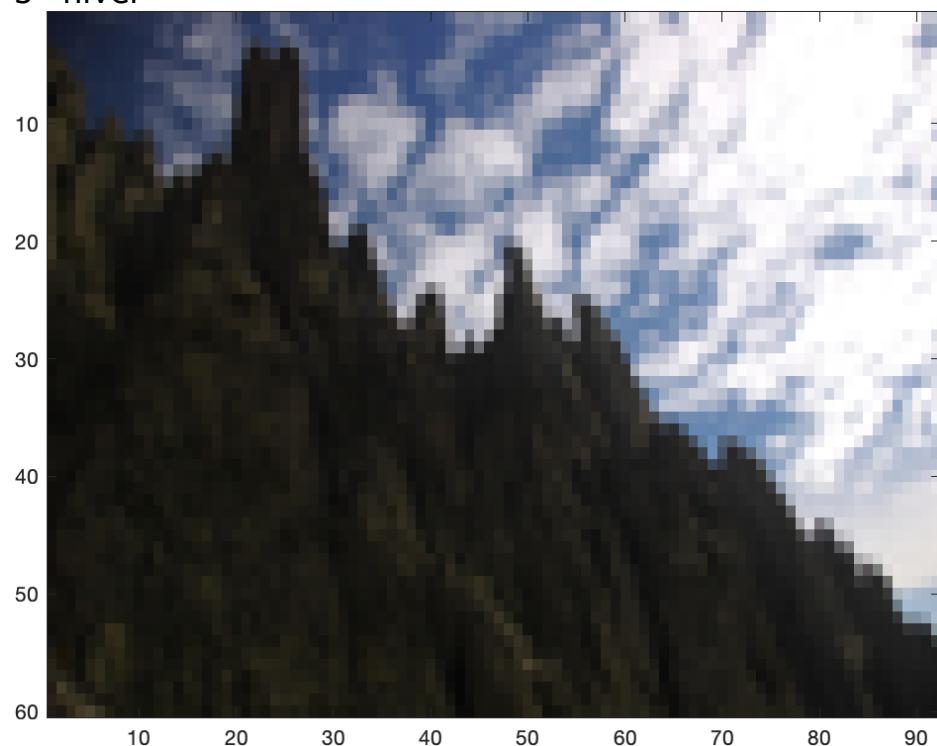
```
function p=lap(im,N)
p=cell(1,N);
for k=1:N-1
    im_red = imresize(im, 1/2);
    im2 = imresize(im_red, 2);
    p{k} = im-im2;
    im = im_red;
end
p{N} = im;
end
```

Visualizar los dos últimos niveles (4º y 5º) de la pirámide con `imagerc()` y adjuntad una captura de los resultados.

4º nivel



5º nivel



Combinación de las pirámides laplacianas:

Adjuntar código usado para combinar las 3 pirámides
 $N = 5;$

```

im1 = im2double(imread("exp_1.jpg"));
p1 = lap(im1, N);
im2 = im2double(imread("exp_2.jpg"));
p2 = lap(im2, N);
im3 = im2double(imread("exp_3.jpg"));
p3 = lap(im3, N);

pm = cell(1, N);
pm{5} = (p1{N} + p2{N} + p3{N})/3;

for l = 1:N-1
    [alto, ancho, ~] = size(p1{l});
    nivel_combinado = zeros(alto, ancho, 3);
    for i = 1:alto
        for j = 1:ancho
            detalle1 = p1{l}(i, j, :);
            detalle2 = p2{l}(i, j, :);
            detalle3 = p3{l}(i, j, :);

            norm1 = norm(detalle1(:));
            norm2 = norm(detalle2(:));
            norm3 = norm(detalle3(:));
            if norm1 >= norm2 && norm1 >= norm3
                nivel_combinado(i, j, :) = p1{l}(i, j, :);
            elseif norm2 >= norm1 && norm2 >= norm3
                nivel_combinado(i, j, :) = p2{l}(i, j, :);
            else
                nivel_combinado(i, j, :) = p3{l}(i, j, :);
            end
        end
    end
    pm{l} = nivel_combinado;
end

```

Inversión de la pirámide mezcla:

Adjuntad el valor máximo de la diferencia entre ambas, que debería ser muy bajo.
 $\text{Max}(\text{dif}(:)) = 3.3307\text{e-}16$

Adjuntar código usado para invertir la pirámide laplaciana.

```
HDR=pm{N};
```

```
for k=N-1:-1:1
    HDR = imresize(HDR, 2);
    HDR = HDR + pm{k};
end
```

Retoques finales:

Adjuntad el código usado para re-escalar la imagen, aplicar la ecualización adaptativa y el aumento de saturación.

```
v0 = min(HDR(:));
v1 = max(HDR(:));
HDR = (HDR-v0)/(v1-v0);
```

```
HSV = rgb2HSV(HDR);
```

```
HSV(:,:,:,3) = adapthisteq(HSV(:,:,:,3), 'ClipLimit', 0.01);
```

```
HSV(:,:,:,2) = HSV(:,:,:,2).^0.75;
```

Adjuntad imagen final resultante.



Adjuntad dentro de la entrega un 2º script fusion.m que implemente los pasos descritos en este algoritmo y muestre la imagen resultante. El script debe contener todas las funciones auxiliares para que funcione de forma autónoma (no hace falta incluir las imágenes de partida).

Volver a aplicad este algoritmo usando ahora como imágenes de entrada tres de las imágenes del apartado anterior, en particular: belg_3.jpg, belg_5.jpg y belg_7.jpg.
[Adjuntad la imagen resultante.](#)

