BIOENGENHARIA - LABORATORIO 02

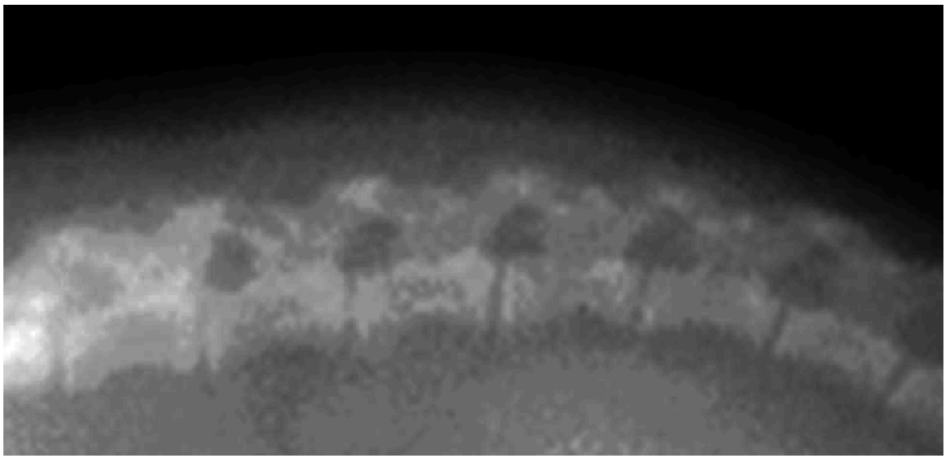
Table of Contents

Aluno: Pedro Henrique Garcia Macedo R.A.: 1829696

Ex 1.

```
clear

% a)
% Essa só consegui resolver na cagada kkkk.
% Quando r=0, s=0 ->  0 = a + b
% portanto a = -b

% Quando r = 0.85, s = 0.5 -> f(c) = 0.5;
% Divindo f(r=0.85)/f(r=1) obtemos, e isolando f(c) = 0
valor_inicial = 10;
[c, f] = fsolve(@(c) (2*exp(0.85*c)-exp(c)-1), valor_inicial)
% Obtemos o valor de 4.5510 para c;
% Quando r = 1, s = 1 -> isolamos b;
b = 1/(exp(c)-1);
a = -b;

% Obtendo os coeficientes, verificamos que a transformação dá
 exatamente
% os parâmetros de projeto
s = @(r) (a + b.*exp(c.*r));
s(0)
s(1)
s(0.85)

% b)
A = imread("p_3_1", "jpg");
% imread() retorna uint8 (num. inteiro sem sinal)
% é necessário converter para double() para poder fazer a divisão
A = double(A);
A_n = A./max(max(A));
A_s = s(A_n);
imshow(A_s)
```

```
Equation solved.

fsolve completed because the vector of function values is near zero
as measured by the default value of the function tolerance, and
the problem appears regular as measured by the gradient.



c =
    4.5510
f =
  -1.1346e-08
ans =
     0
ans =
     1
ans =
    0.5000
```



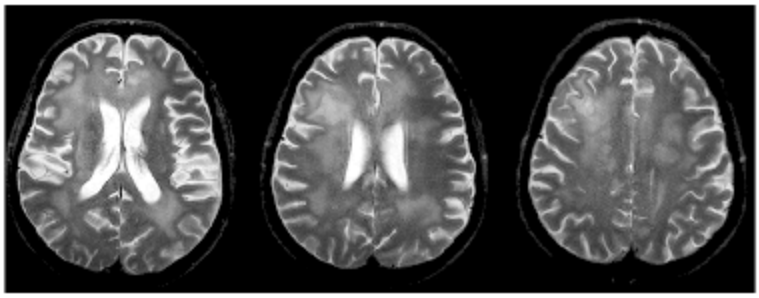
Ex 2.

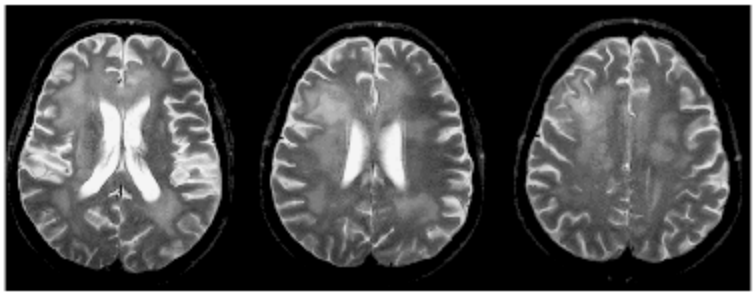
```
clc; clear;
A = imread("p_3_2", "jpg");
A = rgb2gray(A);
class(A)

ans =
    'uint8'
```

2. a)

O matlab não suporta inteiros com menos de 8 bits (ver doc Integers). Não podemos, portanto, economizar memória utilizando inteiros menores com menos que 8 bits. No entanto, podemos aplicar uma máscara de bits para zerar os MSB ou LSB.

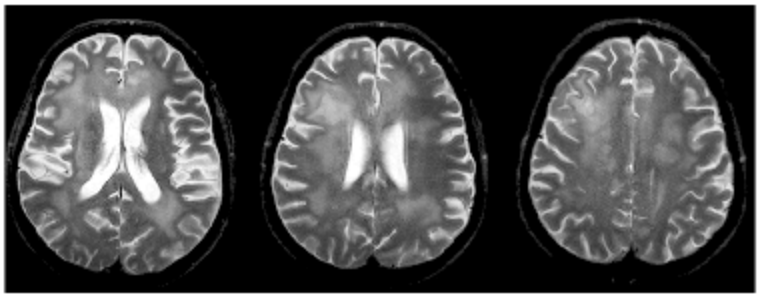
```
lsb3 = 248;
A_c = bitand(A, lsb3);

figure(1)
subplot(2,1,1)
imshow(A, [0 255]);

subplot(2,1,2)
imshow(A_c, [0 255]);
% Inspecionando, a diferença entre as imagens é pouco significativa.
```

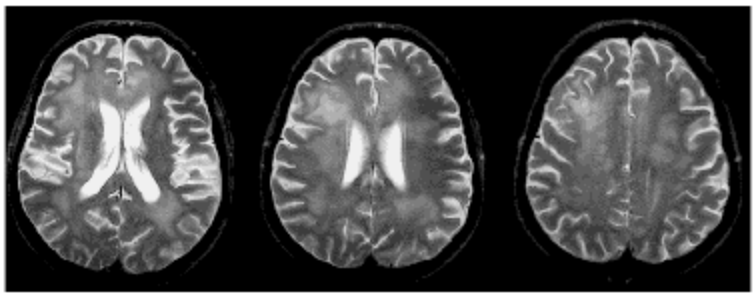




2. b)

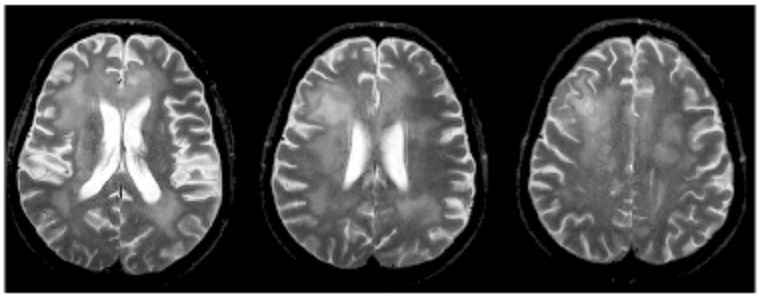
```
lsb4 = 240;
A_c = bitand(A, lsb4);

figure(2)
subplot(2,1,1)
imshow(A, [0 255]);

subplot(2,1,2)
imshow(A_c, [0 255]);

% Inspecionando, a diferença entre as imagens é perceptivel,
% mas pouco significativa.
```

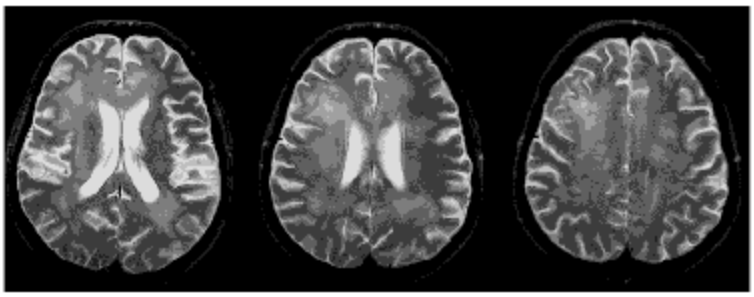




2. c)

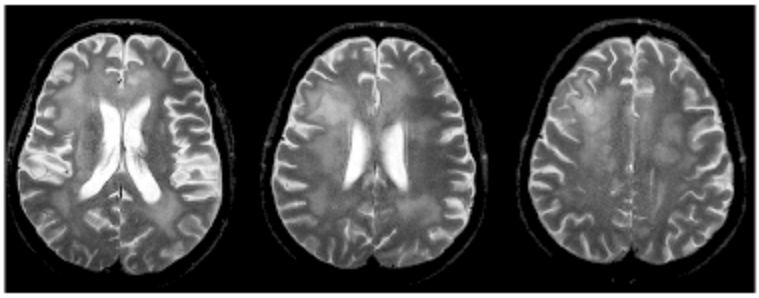
```
lsb5 = 224;
A_c = bitand(A, lsb5);

figure(3)
subplot(2,1,1)
imshow(A, [0 255]);

subplot(2,1,2)
imshow(A_c, [0 255]);
% Com menos que 3 bits MSB restantes, a percepcão é muito degradada.

% Olha que legal, dá pra inverter as cores usando A_c = bitxor(A,
 255);
```

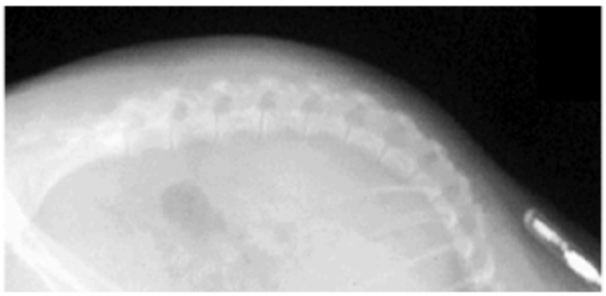


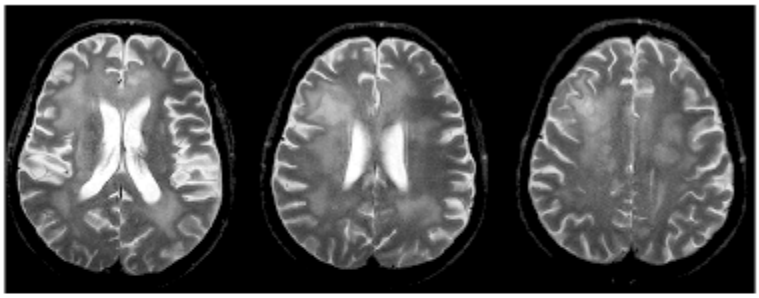


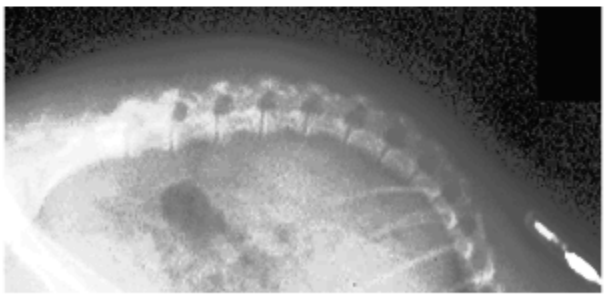
3.

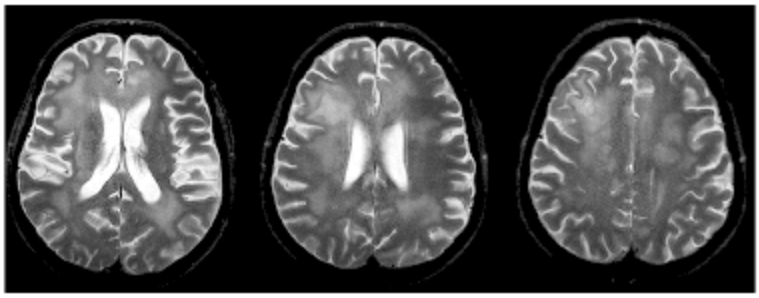
```
clc; clear;
I = imread("p_3_3", "jpg");
J = histeq(I);
figure(1)
imshow(I)
figure(2)
imshow(J)
figure(3)
imhist(I,255)
figure(4)
imhist(J, 255)

% Verificamos que características da imagem estão mais distintas
% mas há um nível perceptível de ruído também.
```

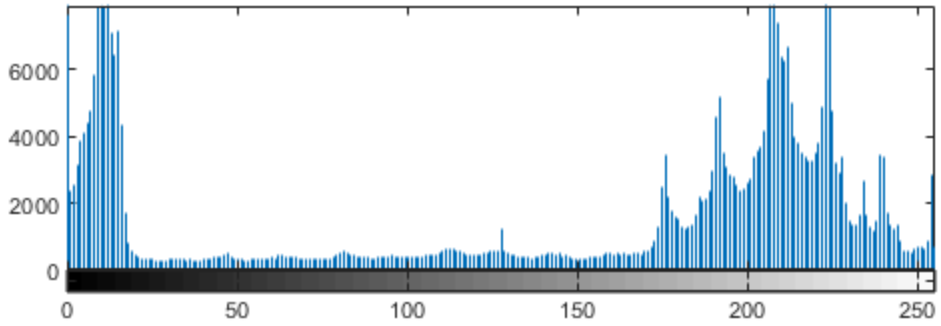


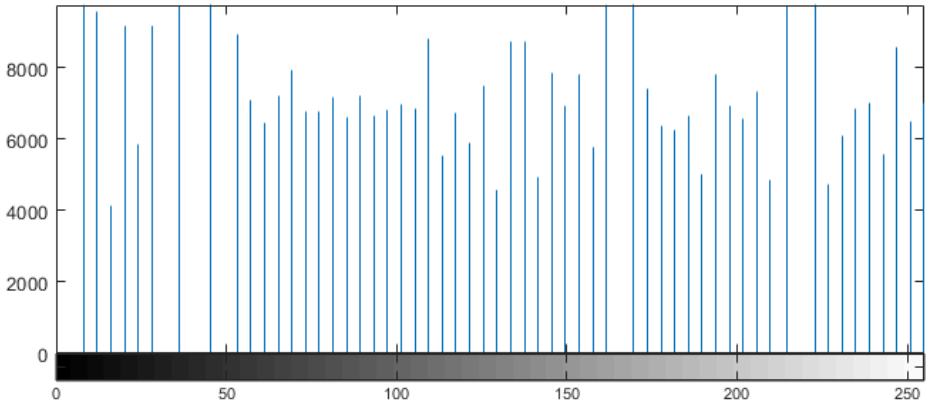








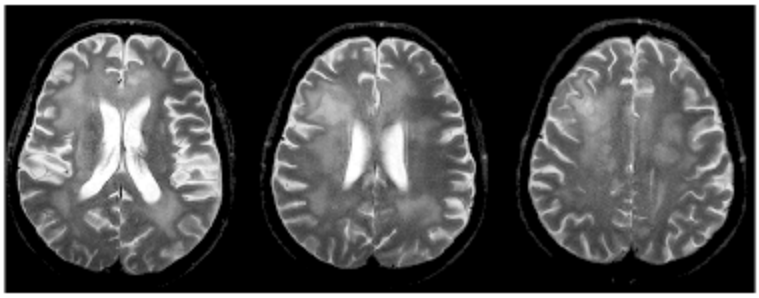


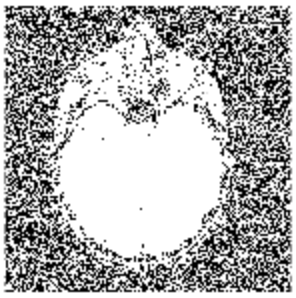


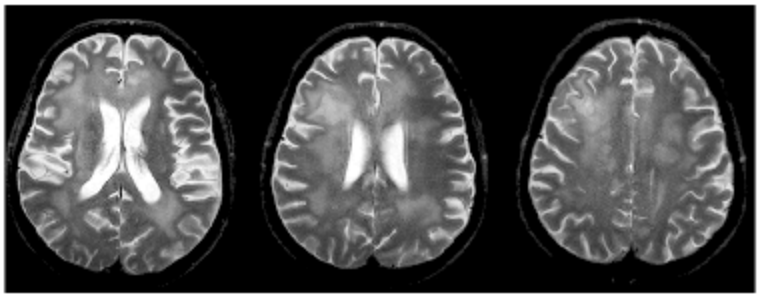
4. a)

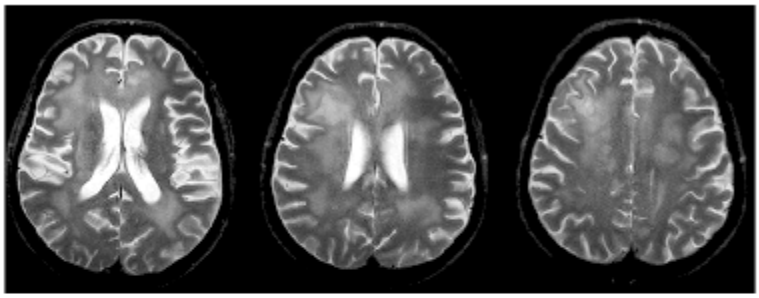
```
clear
load("p_3_4.mat")
I = noisy_MRI_image;
H = (1/9)*[1 1 1; 1 1 1; 1 1 1];
smooth_image = imfilter(I, H);


figure(1)
imshow(I)
```

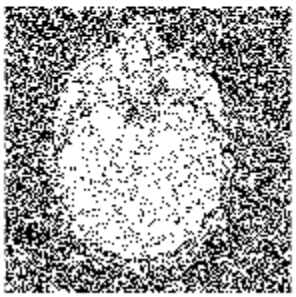
```
figure(2)
imshow(smooth_image)

% 4. b) verificamos que a imagem filtrada é mais 'lisa'

% 4. c)
H = (1/16)*[1 2 1; 2 4 2; 2 2 2];
smoother_image = imfilter(I, H);
figure(3)
imshow(smoother_image)

% 4. d)
J = imnoise(I, 'salt & pepper', .2);
k = medfilt2(J);
imshow(J);
figure(4)
imshow(k)

% vemos que o ruído é reduzido sem degradar as bordas.
```

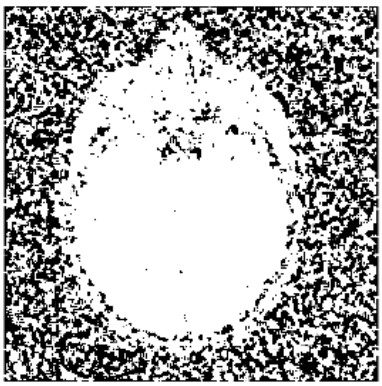








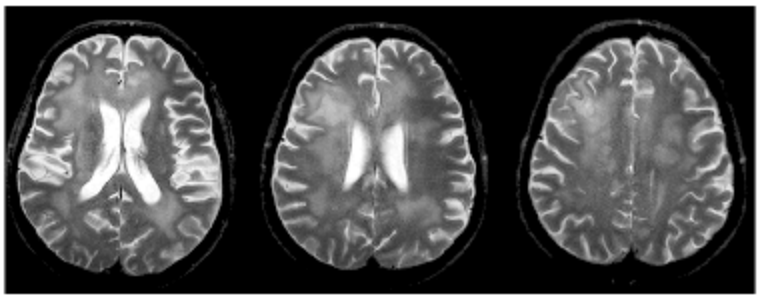


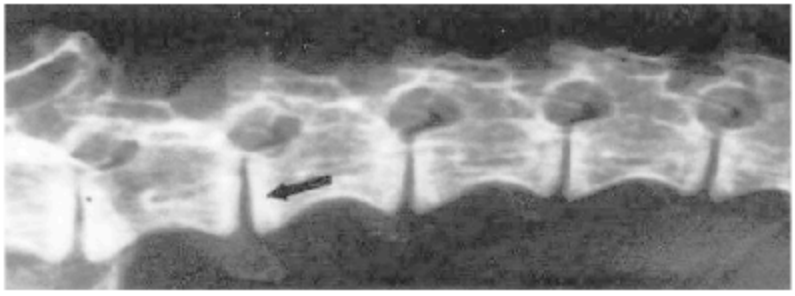


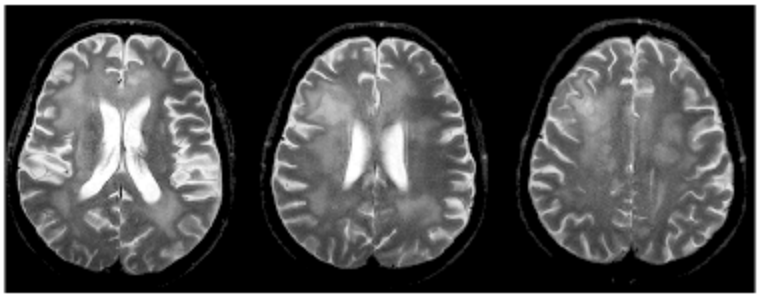
5.)

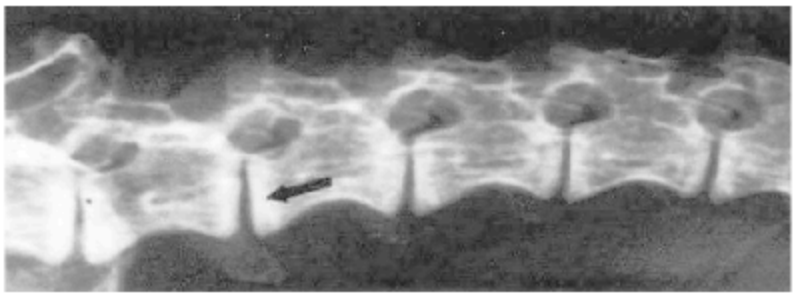
```
clear
load("p_3_5.mat")
I = vertebra;
% A = 1
A = 1;
H = (1/9)*[ 1 1 1; 1 -8 1; 1 1 1];
K = [0 0 0; 0 1 0; 0 0 0];
HB = ((A-1).*K) +H;
sharpened = imfilter(I, HB);
imshow(I);
figure(1)
imshow(sharpened)

% A = 1.05
A = 1.05;
H = (1/9)*[ 1 1 1; 1 -8 1; 1 1 1];
K = [0 0 0; 0 1 0; 0 0 0];
HB = ((A-1).*K) +H;
sharpened = imfilter(I, HB);
imshow(I);
figure(2)
imshow(sharpened)

% A = 1.15
A = 1.15;
H = (1/9)*[ 1 1 1; 1 -8 1; 1 1 1];
K = [0 0 0; 0 1 0; 0 0 0];
HB = ((A-1).*K) +H;
sharpened = imfilter(I, HB);
imshow(I);
figure(3)
imshow(sharpened)

% A = 1.2
A = 1.2;
```

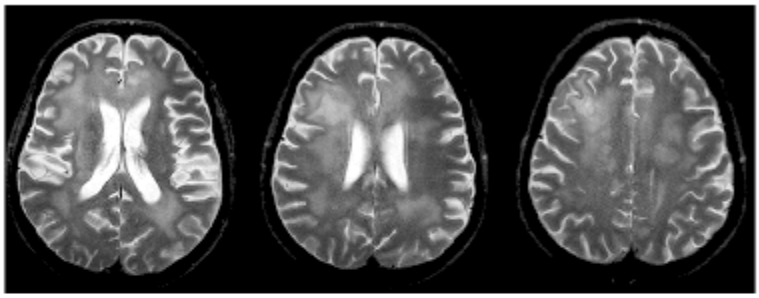
```
H = (1/9)*[ 1 1 1; 1 -8 1; 1 1 1];
K = [0 0 0; 0 1 0; 0 0 0];
HB = ((A-1).*K) +H;
sharpened = imfilter(I, HB);
imshow(I);
figure(4)
imshow(sharpened)

% 5. b)
% O melhor desempenho foi com o filtro com A = 1.15;

% 5. c)
% Detecção de borda utilizando Sobel
figure
imshow(I);
J = edge(I, 'sobel');
figure(5)
imshow(J)

% 5. d)
% Pode somar a imagem resultado do filtro com a imagem inicial
```

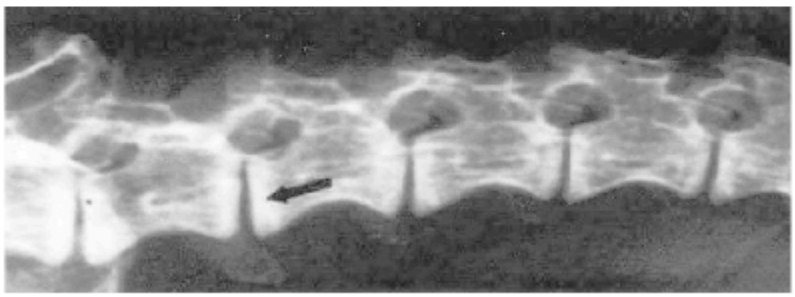


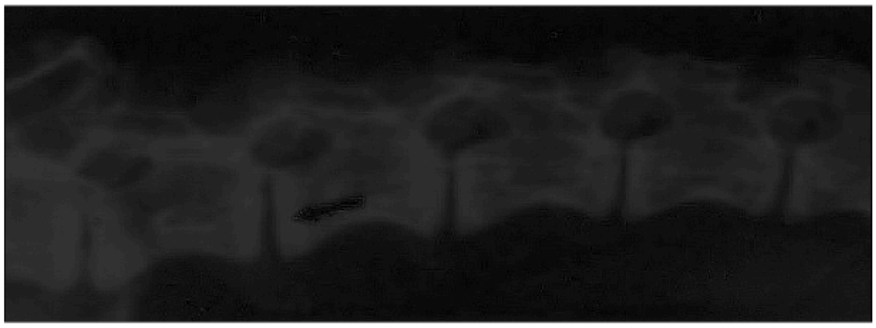


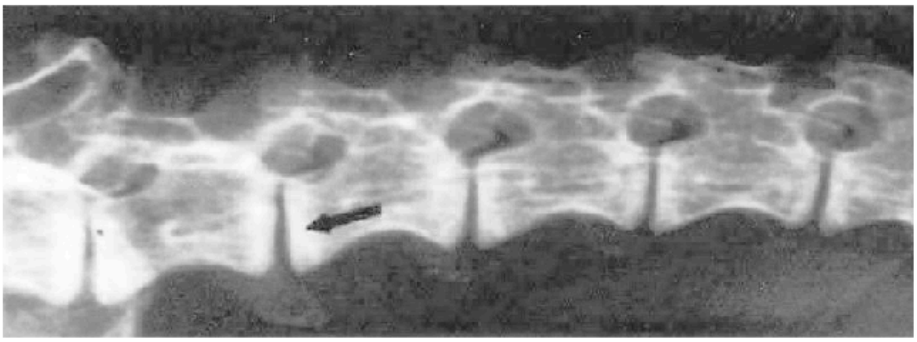


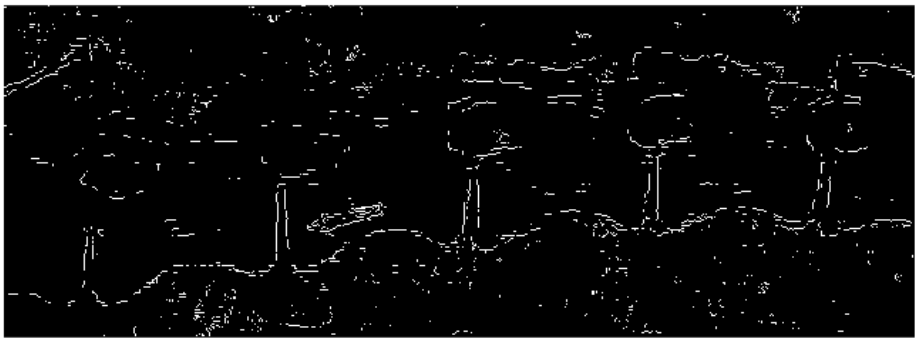












Published with MATLAB® R2018b