Universidade Estadual de Campinas – Unicamp

Faculdade de Engenharia Elétrica e Computação – FEEC

Tutorial de Reconstrução de Imagem de Ultrassom Utilizando o Matlab

IA751 – Instrumentação Biomédica Avançada

Prof Dr. Eduardo Tavares Costa

Novembro de 2019

Campinas -SP

**Exemplo de código para reconstrução de imagem de ultrassom usando o Matlab**

close all

clear all

**Função utilizada para ler os dados de RF adquiridos com a ultrassonix, substituir a parte em amarelo pelo nome do seu arquivo com extensão “.rf”**

[x header params actual\_frames] = load\_ux\_signal('14-38-54.rf',1,100);

**Passamos os dados para uma variável chamada data e depois verificamos seu tamanho para sabermos o número de linhas, colunas desses dados (utilizamos a função size sem “;” para que apareça na tela do matlab esses valores)**

data = x;

size(data)

**O passo a passo para uma única linha é como o feito a seguir. Se eu pegar todas as amostras (usando “:”) de uma coluna (equivalente a todas amostras de um elemento do transdutor), posso fazer a detecção de envoltória a partir da transformada de Hilbert, como vimos em aula:**

Aline = data(:,95,1); %para pegar todas as amostras de um único elemento

envelope = abs(hilbert(Aline)); %pega-se apenas a amplitude do sinal analitico obtido pela transformada de Hilbert

**É interessante plotar o gráfico de todas as amostras de um elemento sobrepondo a curva antes e depois de passar pela transformada de Hilbert:**

%Plota Figure\_1

figure; plot(Aline)

hold on

plot(envelope,'r');

hold off

saveas(gcf, 'IA751ScanlineEnvelope.jpg');

**Para reconstruir a imagem, devemos utilizar todas as amostras de todos os elementos**

I = data (:,:,1); %Para pegar todas as amostras de todos os elementos de um único frame

**Depois são feitas todas as etapas de processamento do sinal vistas em sala de aula:**

* **Transformada de Hilbert**

H = hilbert(I); %para obter o sinal analítico

Hm = abs(H); %utiliza-se apenas a magnitude do sinal

**Aqui estamos só verificando o maior e o menor valor das amostras de um elemento, novamente, não utilizamos “;” porque queremos que seja mostrado no Matlab esses valores. Geralmente há uma diferença muito grande na ordem desses valores, o que justifica a utilização da próxima etapa, a compressão logaritmica.**

max(Hm(:)) %valor máximo das amostras

min(Hm(:)) %valor minimo das amostras

* **Compressão Logarítmica**

Hm = log10(Hm); %compressao dos dados em escala logarítmica

**Aqui encontramos o menor e o maior valor dentre todas as amostras de todos os elementos e normalizamos a nossa matriz de dados**

Hm = Hm - min(min(Hm)); %normalizando os dados

Hm = Hm./max(max(Hm));

**Nesta etapa, se plotarmos a imagem reconstruída, ela terá dimensões que não correspondem ao objeto scanneado na prática**

%plota Figure\_2

figure, imshow(Hm);

saveas(gcf, 'IA751\_recLinear.jpg')

%%

**Até aqui, o código é idêntico para os 3 tipos de transdutores utilizados. Os seguintes passos serão divididos entre transdutor: LINEAR, PHASED ARRAY E CONVEXO**

%----------------------------------------------------------------------------------------------------------------------------

% BEGIN CODE LINEAR

%----------------------------------------------------------------------------------------------------------------------------

**Como sabemos a dimensão da imagem (para o transdutor linear, por exemplo é a largura do transdutor pela profundidade de penetração do ultrassom no tecido) conseguimos ajustar as dimensões da imagem reconstruída e plotar essa imagem novamente**

H2=imresize(Hm, [2080/10 191]); %ajuste do tamanho da imagem necessário devido a diferença do numero de amostras e de elementos

H3 = imadjust(H2);

%plota Figure\_3

figure, imshow(H3);

saveas(gcf, 'IA751\_modoB\_Linear.jpg')

%----------------------------------------------------------------------------------------------------------------------------

% END CODE LINEAR

%----------------------------------------------------------------------------------------------------------------------------

%%

%----------------------------------------------------------------------------------------------------------------------------

% BEGIN CODE CONVEXO

%----------------------------------------------------------------------------------------------------------------------------

**Para o transdutor convexo é necessária a etapa de conversão de varredura, devido a disposição física dos seus elementos. Esse código foi copiado do artigo cuja leitura foi indicada.**

%----------------------------------------------

% Rotina de conversão de varredura CONVEXO

%----------------------------------------------

tic; %inicia temporizador

% Definição de parâmetros de entrada

ne = 128; % numero de elementos

i=1:ne; % indice dos elementos

kerf = 115e-6; % kerf

d = 0.41e-3; % largura do elemento

R = 58e-3; % raio do transdutor

N = 4680; % numero de amostras

n = 1:N; % indice das amostras

c = 1540; % velocidade do som no tecido mole

fs = 20e6; % frequência de amostragem

t = (n/fs)'; % escala de tempo

%----------------------------------

% Ângulo de abertura do transdutor convexo

%----------------------------------

c\_transdutor = 2\*pi\*R;

c\_abertura = (ne-1)\*(kerf + d);

theta\_convexo = 360\*c\_abertura/c\_transdutor;

%----------------------------------

% Ângulo de abertura do transdutor convexo

% de cada elemento

%----------------------------------

theta\_convexo\_i = (i-(ne+1)/2)\*theta\_convexo/(ne-1);

%----------------------------------

% Escala de profundidade

%----------------------------------

z = c\*t/2+R;

%----------------------------------

% Buffer de dados para simulação

%----------------------------------

%valor\_final\_Env\_Log = (rand(N,ne)-2)\*25;

%----------------------------------

% Dados das inclusões para teste

%----------------------------------

%a = 200; b = 10;

%----------------------------------

% Equação das inclusões

%----------------------------------

% valor\_final\_Env\_Log ((N/4)-a:(N/4)+a,ne/4-b:ne/4+b)= 0;

% valor\_final\_Env\_Log ((N/4)-a:(N/4)+a,ne/2-b:ne/2+b)= 0;

% valor\_final\_Env\_Log ((N/4)-a:(N/4)+a,ne\*3/4-b:ne\*3/4+b)= 0;

%

% valor\_final\_Env\_Log ((N/2)-a:(N/2)+a,ne/4-b:ne/4+b)= -25;

% valor\_final\_Env\_Log ((N/2)-a:(N/2)+a,ne/2-b:ne/2+b)= -25;

% valor\_final\_Env\_Log ((N/2)-a:(N/2)+a,ne\*3/4-b:ne\*3/4+b)= -25;

%

% valor\_final\_Env\_Log ((N\*3/4)-a:(N\*3/4)+a,ne/4-b:ne/4+b)= -50;

% valor\_final\_Env\_Log ((N\*3/4)-a:(N\*3/4)+a,ne/2-b:ne/2+b)= -50;

% valor\_final\_Env\_Log ((N\*3/4)-a:(N\*3/4)+a,ne\*3/4-b:ne\*3/4+b)= -50;

[THETA,RHO] = meshgrid(degtorad(theta\_convexo\_i),z);

[xc,yc] = pol2cart(THETA,RHO);

%----------------------------------

% Imagem simulada antes da conversão de varredura

%----------------------------------

figure;

colormap(gray);

imagesc(theta\_convexo\_i,(z-R),H3);

xlabel('\theta\_c\_o\_n\_v\_e\_x\_o [\circ]');

ylabel('Profundidade z[mm]');

axis tight; colorbar;

%----------------------------------

% Imagem simulada após a conversão de varredura

%----------------------------------

figure;

colormap(gray);

h=surf((xc-R)\*1e3,(yc)\*1e3,Hm,'edgecolor','none');

view(90,90);

xlabel('Profundidade z[mm]');

ylabel('Eixo x[mm]');

axis image;

saveas(gcf, 'IA751\_modoBScanConverted.jpg')

toc

% finaliza temporizador

%----------------------------------------------

%----------------------------------------------------------------------------------------------------------------------------

% END CODE CONVEXO

%----------------------------------------------------------------------------------------------------------------------------

%%

%----------------------------------------------------------------------------------------------------------------------------

% BEGIN CODE PHASED ARRAY

%----------------------------------------------------------------------------------------------------------------------------

**Para o transdutor phased array também é necessária a etapa de conversão de varredura. Neste caso, é possível utilizar uma função da *toolbox* k-Wave do Matlab chamada** scanConversion. **Deve-se passar os seguintes parâmetros (que devem ser ajustados de acordo com os SEUS dados)**

ang\_min = -45; % graus

ang\_max = 45; % graus

N\_lines = 191;

depth = 0.16; % m

width = 0.16; % m

fSampling = 20e6; % Hz

Hm=Hm';

%Hm = reshape(Hm, [191, 2080]);

figure, imagesc(scanConversion(Hm,linspace(ang\_min,ang\_max,N\_lines),[depth width],1540,1/fSampling)); colormap gray; % imagem modo-B

title('Out Scan Conversion');

saveas(gcf, 'IA751\_modoBScanConvertedPhased.jpg')

%----------------------------------------------------------------------------------------------------------------------------

% END CODE PHASED ARRAY

%----------------------------------------------------------------------------------------------------------------------------