Contents

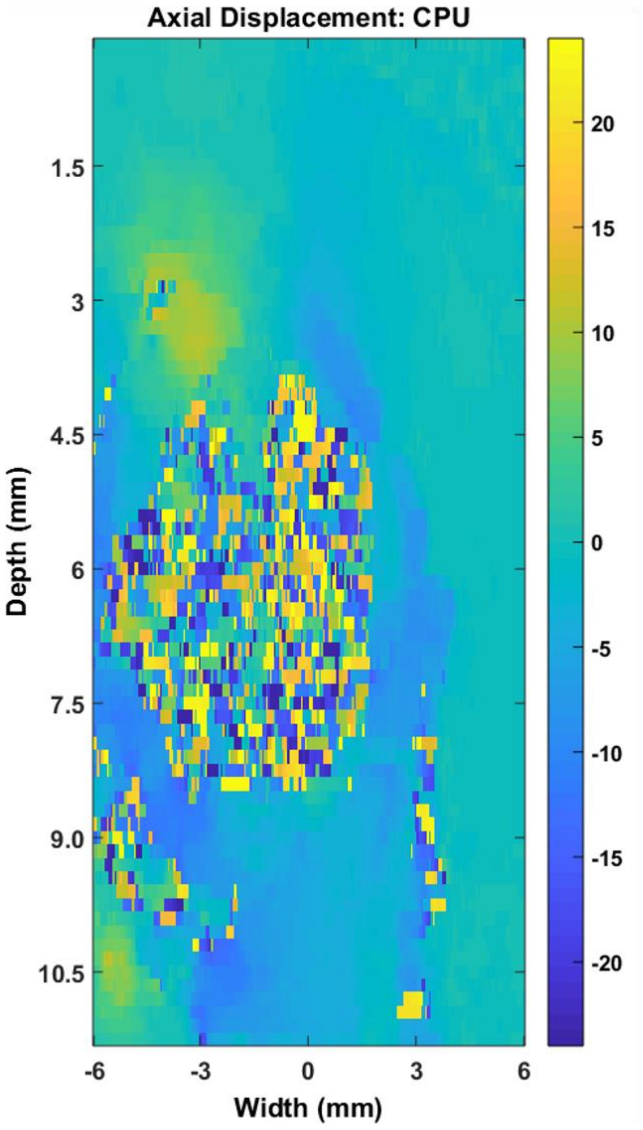
```
% Compiled by M. Turney, R. Mukaddim, R. Pohlman on 12/20/2017

% This script is a demo of the CUDA-MATLAB approach for accelerating
% ultrasound elasticity imaging. This algoithm is developed and improved in
% Varghese Ultrasound Lab, Department of Medical Physics, University of
% Wisconsin, Madison
% Email: tvarghese@wisc.edu

% This code is provide in support of the final project for partial
% completion of ME 759 Fall 2017.
% For permission to use otherwise, please contact tvarghese@wisc.edu.


load('rfData.mat'); % Load pre/post images and imaging acquisition parameters

load('Multilevel_Algorithm_Parameters'); % Load Algorithm Parameters

% Boundary Considerations and Padding

rangeY = round(searchParams.kernelY*.15);
rangeX = [8,4,4,4];

halfY = (searchParams.kernelY - 1)/2;
halfX = (searchParams.kernelX - 1)/2;

rfX = size(pre,2);

maxRangeY = sum(rangeY(1));
maxRangeX = sum(rangeX(1));
stepY = round(searchParams.kernelY*(100-searchParams.overlap)/100 );

pad_level_y = maxRangeY + halfY(1);
pad_level_x = maxRangeX + halfX(1);

% Padded data for 2-D NCC
pad_pre_rf=padarray(pre,[pad_level_y,pad_level_x]);
pad_post_rf=padarray(post,[pad_level_y,pad_level_x]);

startYrf_pad = 1 + maxRangeY + halfY(1);
startX_pad = 1 + maxRangeX + halfX(1);
stopYmax_pad = size(pad_pre_rf,1) - halfY(1) - maxRangeY;

numY = floor( (stopYmax_pad - startYrf_pad) / stepY(1) ) + 1;

stopYrf_pad = startYrf_pad + stepY(1)*(numY-1);
stopX_pad = size(pad_pre_rf,2) - halfX(1) - maxRangeX;

numX = stopX_pad - startX_pad + 1;

kernCenterY_pad = startYrf_pad:stepY:stopYrf_pad;
kernCenterX_pad = startX_pad:stopX_pad;

% Original Kernel locations for interpolation
startYrf=1;
stopYrf = 1 + stepY*(numY-1);
startX=1;
stopX=rfX;

% Coordinate points for imaging and further interpolation purpose
geomRf = struct( 'stopY', stopYrf, 'startY', startYrf, 'stepY', stepY, 'startX', startX, 'stopX', stopX );
geomRf_pad = struct( 'stopY', stopYrf_pad, 'startY', startYrf_pad, 'stepY', stepY, 'startX', startX_pad, 'stopX', stopX_pad );
```

GPU Implementation - Final Version

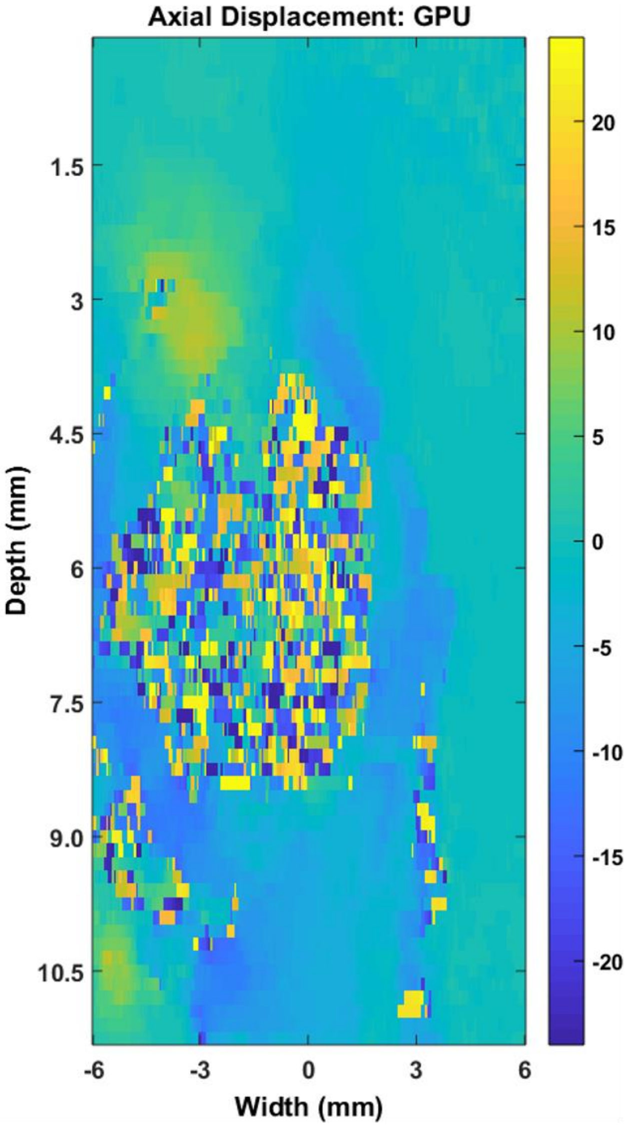
```
tic;
% Defining input parameters
kerX = searchParams.kernelX(1);
kerY = searchParams.kernelY(1);
sX = 2*rangeX(1)+1;
sY = 2*rangeY(1)+1;
overlap = 50.0;

% Type cast to single
pad_pre_rf=single(pad_pre_rf);
pad_post_rf=single(pad_post_rf);

[quality_GPU, dpX_GPU, dpY_GPU] = Corr2GPUMex_Final(pad_pre_rf,pad_post_rf,sX,sY,kerX,kerY,numX,numY,overlap);

% Reshape output
quality_GPU=reshape(quality_GPU,[numX numY])';
dpX_GPU=reshape(dpX_GPU,[numX numY])';
dpY_GPU=reshape(dpY_GPU,[numX numY])';

toc;
```

```
Search_x = 17
Search_y = 49
Kernel_x = 9
Kernel_y = 161
numX = 220
numY = 75
Overlap = 50.000000
Elapsed time is 1.574731 seconds.
```

Published with MATLAB® R2013a



Axial Displacement: CPU
Depth (mm)
Width (mm)
1.5
3
4.5
6
7.5
9.0
10.5
-6
-3
0
3
6
20
15
10
5
0
-5
-10
-15
-20



Axial Displacement: GPU
Depth (mm)
Width (mm)
1.5
3
4.5
6
7.5
9.0
10.5
-6
-3
0
3
6
20
15
10
5
0
-5
-10
-15
-20