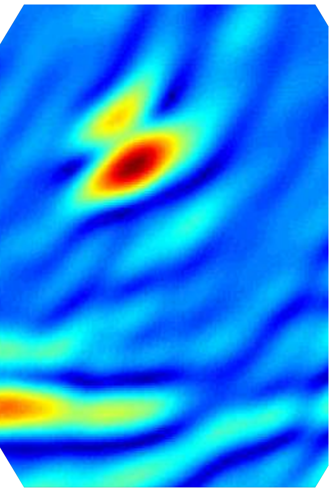
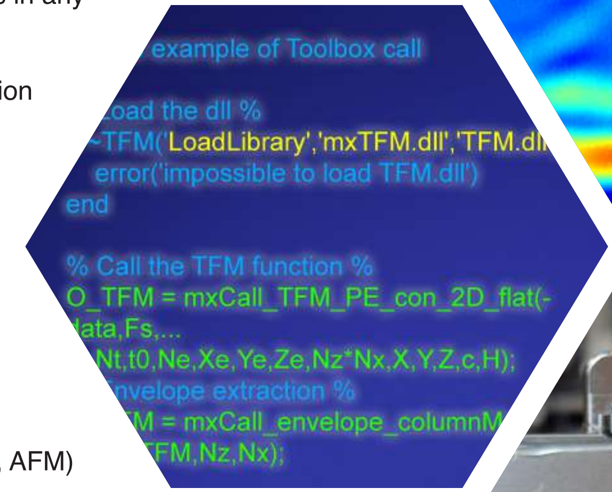
TFM Toolbox

OEM Functions for Advanced Imaging

Uses:

- Toolbox for programming (C, C++, C#, Matlab, Python, etc.), in the shape of a DLL
- Fast imaging using Nvidia GPU parallel computing (N.B. Nvidia GPU is mandatory)
- Usable with real-time, open phased-array hardware (typically AOS OEM-PA)
- Usable without hardware for post processing analysis in any Windows environment
- Compatible with various, constantly evolving acquisition schemes(SAFT, FMC, etc.)
- Perform your own FMC/TFM with the best featured program using this DLL and the highest performance on the market

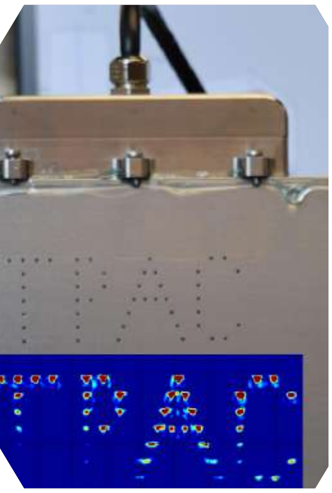


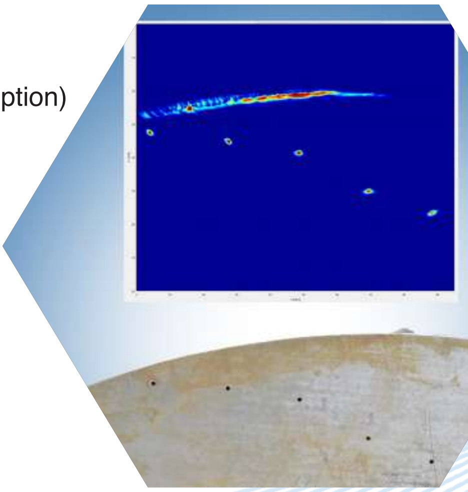


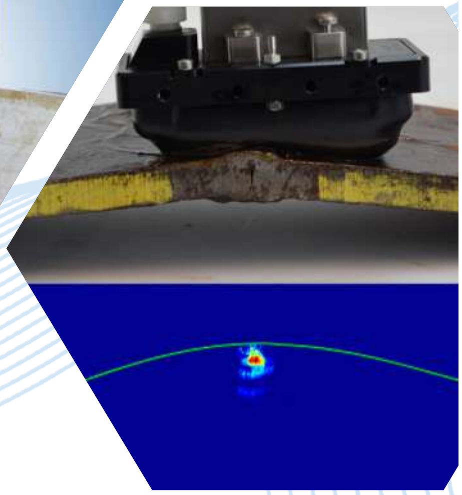
```
 example of Toolbox call

Load the dll %
~TFM('LoadLibrary','mxTFM.dll','TFM.dll
  error('impossible to load TFM.dll')
end

% Call the TFM function %
O_TFM = mxCall_TFM_PE_con_2D_flat(-
data,Fs,...
  Nt,t0,Ne,Xe,Ye,Ze,Nz*Nx,X,Y,Z,c,H);
 nvelope extraction %
 M = mxCall_envelope_columnM
 TFM,Nz,Nx);
```

Methods:

- Delay and Sum (DAS) imaging methods (SAFT, TFM, AFM)
- Migration imaging methods (SAFT, TFM, AFM)
- Surface adaptive methods (in option)
- High-level inverse methods: SAFTp, TFMp, AFMp (in option)
- Envelope functions







Features:

- No image size limitation
- 2D/3D computations
- Free selection of the resolution of the pixel/voxel
- Linear/matrix probes
- Pulse-echo/pitch-catch configurations
- Flat/cylindrical pieces (any for Adaptive TFM)
- Direct/Indirect/Corner modes (LL, L-LL, LL-LL, TT, T-TT, TT-TT)
- Conversion modes (LTL, LLT, etc.)
- About 100 configurations

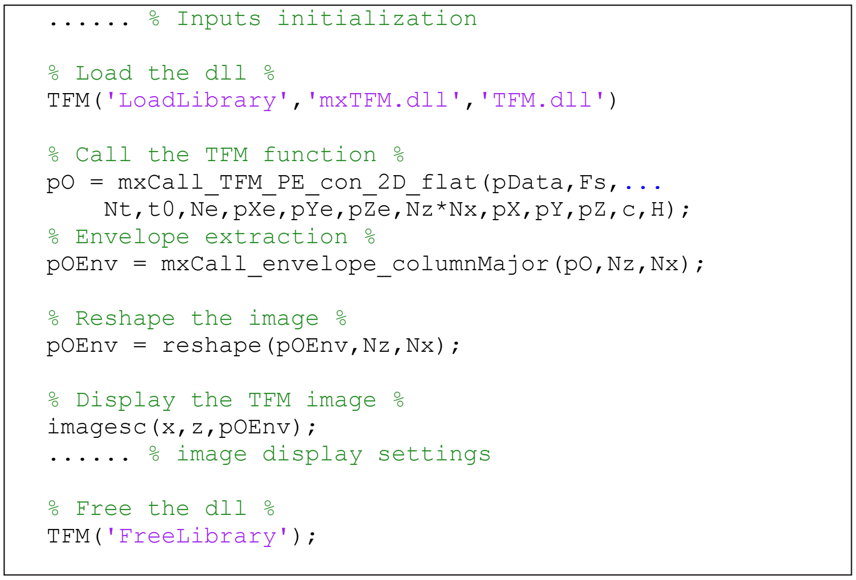
python™ MATLAB

TPAC The Phased Array Company

www.thephasedarraycompany.com
info@thephasedarraycompany.com

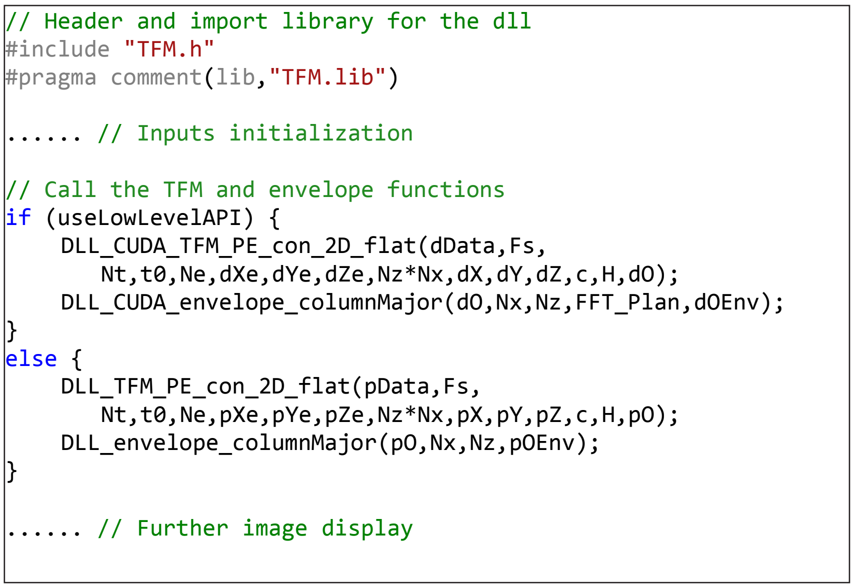
TECHNICAL SIDE OF DLL

Matlab Example:



```
...... % Inputs initialization

% Load the dll %
TFM('LoadLibrary','mxTFM.dll','TFM.dll')

% Call the TFM function %
pO = mxCall_TFM_PE_con_2D_flat(pData,Fs,...
    Nt,t0,Ne,pXe,pYe,pZe,Nz*Nx,pX,pY,pZ,c,H);
% Envelope extraction %
pOEnv = mxCall_envelope_columnMajor(pO,Nz,Nx);

% Reshape the image %
pOEnv = reshape(pOEnv,Nz,Nx);

% Display the TFM image %
imagesc(x,z,pOEnv);
...... % image display settings

% Free the dll %
TFM('FreeLibrary');
```

C/C++ example:



```
// Header and import library for the dll
#include "TFM.h"
#pragma comment(lib,"TFM.lib")

...... // Inputs initialization

// Call the TFM and envelope functions
if (useLowLevelAPI) {
    DLL_CUDA_TFM_PE_con_2D_flat(dData,Fs,
       Nt,t0,Ne,dXe,dYe,dZe,Nz*Nx,dX,dY,dZ,c,H,dO);
    DLL_CUDA_envelope_columnMajor(dO,Nx,Nz,FFT_Plan,dOEnv);
}
else {
    DLL_TFM_PE_con_2D_flat(pData,Fs,
       Nt,t0,Ne,pXe,pYe,pZe,Nz*Nx,pX,pY,pZ,c,H,pO);
    DLL_envelope_columnMajor(pO,Nx,Nz,pOEnv);
}

...... // Further image display
```

These code examples demonstrate how to use the TFM DLL in Matlab and C/C++. Our TFM DLL is designed to have the following properties:

- Extremely fast image reconstruction. The GPU-accelerated library is fully optimized to allow decent frame rate in real-time applications.
- User-friendly APIs. The library brings easy-to-use APIs in Matlab and C/C++. Developers can employ our DLL in their existing applications with minimum modifications.
- Complete configuration support. Our library provides full support of conventional scan and imaging configurations, as well as new migration-based imaging and adaptive imaging for complicated/unknown surface geometry. Low level APIs are also provided for developers with knowledge in GPU computing.

You can use any TFM software you want. We have developed various and very effective, highly optimized TFM algorithms for you to implement. Enjoy!