Make my own GUI Version

```
%% Create Processing Parameters Data Structure for IBII code
% Authors: Lloyd Fletcher, Jared Van-Blitterswyk
% PhotoDyn Group, University of Southampton
% Date Created: 21/11/2018
% Date Edited: 9/6/2019 - v1.0b
%
% Produces the 'processingParameters.mat' file required to run the IBII
% processing code. The main parameters that need to be specified by the
% user include:
% - The material model - globalOpts.matModel
% - The grid pitch / sampling - grid.pitch / grid.pxPerPeriod
% - The frame rate - time.frameRate
% - The material density - material.rho
% The user should also turn on smoothing (globalOpts.smoothingOn) and
% specify smoothing kernels (smoothingOpts) when processing image
% deformation data with noise or experimental data. Units are SI with↙
the
% following convention: length in m, time in s, force in N, mass in kg.
% For example: density should be specifed in kg/m^3.
%
% If there are any issues with this code please contact:
% Lloyd Fletcher: l.c.fletcher@soton.ac.uk / lloydcolinfletcher@gmail.↙
com
%
%↙
///////////////////////////////////////////////////////////////////////↙
//
% This is an example file for the isotropic material model used to↙
process
% image deformation data for validation. The FE model used to generate↙
the
% synthetic images was produced in Abaqus. The model is a 2D rectangular
% plate of tungsten carbide impacted by a triangular pressure pulse on a
% short edge.
%↙
///////////////////////////////////////////////////////////////////////↙
//

clc
clear all
close all
```

```
fprintf('--------------------------------------------------------------↙
\n')
fprintf('INITIALISE PROCESSING PARAMETERS: IBII Processing, v1.0b\n')
fprintf('--------------------------------------------------------------↙
\n')

%% Create the paths for saving the processing parameters
% By default the processingParameters file is saved in the same↙
directory
% as this '.m' file.
fprintf('Specifying paths to save processing parameter file.\n')
savePaths = {pwd};
saveFile = 'processingParameters.mat';

%% INITALISE - Test Data Structures
fprintf('Creating data structures for processing constants.\n')

% Information about the test, not used in the calculation. This data↙
struct
% can be used to store information like the test date / projectile type↙
etc
test.description = 'WC_Iso_ImageDeformation';

%↙
------------------------------------------------------------------------↙
--
% Global Processing Options - Controls program behaviour and modules↙
that
% will run
globalOpts.imageDefAddNoise = true;    % Allows addition of noise to↙
image for processing image deformation test cases
globalOpts.smoothingOn = true;         % Turns smoothing on/off, useful↙
for processin image deformation data with no noise
% Possible material models: 'isotropic' or 'orthotropicReduced'
globalOpts.matModel = 'isotropic';
% Which kinematic field components are plotted and calculated
globalOpts.fieldComponents = 'all';     % 'all' plot/calc x,y,s↙
components or 'xOnly', xonly option is useful for speeding up↙
calculation for redOrtho model
% Allows user to select free edge location or take it from the↙
```

```
processing parameters file
globalOpts.hardCodeFreeEdge = true;
globalOpts.freeEdge = 'Left';           % Free edge on the 'Left' or↙
'Right', if 'Right' fields are flipped such that the co-ordinate system↙
is consistent
% Flags controlling which parts of the post-processing are run
globalOpts.plotImageSeqs = 'prompt';       % Plot videos or not,↙
Options: 'yes','no','prompt','diagnostic'
globalOpts.processOptVF = true;         % Run piecewise optimised↙
virtual fields identification
globalOpts.identifyStrength = true;     % Run strength indentification↙
code
globalOpts.plotPresentationImages = true; % Plot image sequences of SS↙
curves and fields for presentations
globalOpts.reduceMemory = true;         % Flag used to maintain only↙
necessary data fields, reducing memory use and save file size.
globalOpts.autoSaveProcData = false;    % Automatically save processed↙
data without prompt

%↙
------------------------------------------------------------------------↙
--
% Time Data Structure
% NOTE: frame rate needs to be define correctly for each test as the
% time.step variable is used in the acceleration calculation
time.frameRate = 5e6;           % Frame rate in frames/second
time.step = 1/time.frameRate;   % Interframe time in seconds
time.cutFrames = 5;             % Frames to cut from the end of the↙
accel/stress to avoid numerical diff effects
time.numFrames = 128;           % Shimadzu HPV-X camera has 128 frame↙
memory
time.vec = 0:time.step:time.step*(time.numFrames-1);

%↙
------------------------------------------------------------------------↙
--
% Material Data Structure - Nominal Properties
% NOTE: density is used for stress calculation, the other input↙
parameters
% are not used in the calculation procedure by default, they are only↙
used
```

```
% for comparison to the identified values on diagnostic plots
material.name = 'WC_ID';
material.rho = 14500;   % Density should be given in kg/m^3
material.Exx = 550e9;   % Stiffness values for comparison should be↙
given in Pa
material.nuxy = 0.24;
if strcmp(globalOpts.matModel,'orthotropicReduced')
    material.Q = 1/(1-material.nuxy*material.nuyx).*...
            [material.Exx  material.nuyx*material.Exx 0;...
             material.nuxy*material.Eyy material.Eyy 0;...
             0 0 material.Gxy*(1-material.nuxy*material.nuyx)];
    material.Qxx = material.Q(1,1);
    material.Qxy = material.Q(1,2);
else
    material.Q = material.Exx/(1-material.nuxy^2).*...
                [1 material.nuxy 0;...
                 material.nuxy 1 0;...
                 0 0 (1-material.nuxy)/2];
    material.Qxx = material.Q(1,1);
    material.Qxy = material.Q(1,2);
end

%↙
------------------------------------------------------------------------↙
--
% Specimen Data Struct
% Nominal geometry only, updates after grid method processing based on↙
FOV
% All geometry is specified in meters
specimen.length = 70e-3;
specimen.height = 44e-3;
specimen.thickness = 4e-3;
specimen.volume = specimen.length*specimen.height*specimen.thickness;
specimen.mass = specimen.volume*material.rho;
% NOTE: determines which side of the image the free-edge is on, this is
% required for specifying which edge the stress-gauge calculation is
% performed from and which edge is the impact edge for the VFM.
specimen.freeEdge = globalOpts.freeEdge;

%↙
------------------------------------------------------------------------↙
```

```
--
% Grid Data Struct
grid.name = '09mm_5pxpp';
grid.pitch = 0.9e-3;         % Grid pitch in meters
grid.pxPerPeriod = 5;        % Grid sampling in pixels/period
grid.rotAngle = 0;           % Used to rotate displacements for↙
misaligned grids
grid.length = specimen.length;
grid.height= specimen.height;
grid.mPerPx = grid.pitch/grid.pxPerPeriod;
grid.numXPeriods = grid.length/grid.pitch;
grid.asymmPitch = false;     % Useful for printed grids with slight pitch↙
asymmetries
grid.pitchX = grid.pitch;
grid.pitchY = grid.pitch;

% Struct to control how the grid images are processed
gridMethodOpts.windowFlag = 1;          % 0 = gauss, 1 = bi-triang, for↙
the HPV-X option 1 is recommended
gridMethodOpts.windowWidth = 1;         % Multiplier for grid pitch to↙
set window width
gridMethodOpts.dispCalcMethod = 2;      % 1) Calculate using phase↙
subtraction or 2) iterative method
gridMethodOpts.temporalUnwrap = true;   % Turns on temporal unwrapping
gridMethodOpts.debug  = false;          % Plots diagnostic figure to↙
debug temporal unwrapping
gridMethodOpts.hardCodeRotAngle = true; % Stops GUI prompt and uses↙
rotation angle in the grid struct
gridMethodOpts.autoLoadProccessedDataFile = false; % Allows the code to↙
load displacement fields from a previous run

%↙
-----------------------------------------------------------------------↙
--
% Image Noise Structure
% Used for adding noise to image deformation simulations
if globalOpts.imageDefAddNoise
    imageNoise.addNoise = true;
    imageNoise.pcNoise = 0.4;      % Grey level noise as % of the↙
dynamic range
    imageNoise.bits = 16;           % Bit range of the image
    imageNoise.convToUInt16 = true; % Scales the image to 16bit int
else
    imageNoise.addNoise = false;
end

%% INITALISE - Post-Processing Options
% Create smoothing and edge extrapolation options data structures
fprintf('Creating data structures for smoothing and extrapolation↙
options.\n')

%↙
------------------------------------------------------------------------↙
--
% Differentiation Options
% NOTE: the default options set here are suitable for most applications
% Select method for numerical differentiation
diffOpts.method = 'gradient';   % Options 'gradient' uses gradient↙
twice, 'cDiff' centred difference to obtain accel directly
diffOpts.temporalPad = false;   % Pad the data at the start to account↙
for forward/back diff at edges
diffOpts.temporalPadFrames = 3; % Number of frames to pad
diffOpts.temporalPadMethod = 'replicate';

%↙
------------------------------------------------------------------------↙
--
% Smoothing Options
% NOTE: the key parameters here are the spatial and temporal smoothing
% kernels: smoothingOpts.spatialKernal, smoothingOpts.WATemporalKernal↙
and
% smoothingOpts.FFTemporalKernal. The remaining parameters can be left↙
as
% the default options.

% Turn all smoothing on/off with the global flag, useful for processing
% image deformation data without noise.
if globalOpts.smoothingOn
    smoothingOpts.spatialSmooth = true;
    smoothingOpts.WATempSmooth = true;
    smoothingOpts.FFTempSmooth = true;
else
```

```
    smoothingOpts.spatialSmooth = false;
    smoothingOpts.WATempSmooth = false;
    smoothingOpts.FFTempSmooth = false;
end

% Spatial Smoothing Options
smoothingOpts.spatialFilt = 'gauss';
% For gauss filter kernal length is specified giving: KernLeng = 2*ceil↙
(2*sigma)+1
% An asymmetric kernal can be specified: [LKernX,LKernY]
smoothingOpts.spatialKernal = [41,41];          % Specify kernal size in↙
px, must be odd
smoothingOpts.spatialEdgeMode = 'symmetric';    % Option for edge↙
padding with Matlab gauss filter

% Full-Field Temporal Smoothing Options
smoothingOpts.FFTemporalFilt = 'sgolay';    % No other options are↙
implemented yet
smoothingOpts.FFTemporalKernal = [11,3];    % [numFrames,filterOrder]
smoothingOpts.FFTemporalPad = false;        % Pads the start of the data↙
by half a smoothing kernel
smoothingOpts.FFTemporalPadFrames = 3;      % Extra frames to pad in↙
addition to the half smoothing kernel
smoothingOpts.FFTemporalPadMethod = 'replicate';
% Width Averaged Temporal Smoothing Options
% If true, average displacements over width and then apply temporal
% smoothing, if false these parameters are not used and the width↙
average is
% calculated from the full-field acceleration.
smoothingOpts.WATemporalAvgFirst = false;
smoothingOpts.WATemporalFilt = 'sgolay';    % No other options are↙
implemented yet
smoothingOpts.WATemporalKernal = [11,3];    % [numFrames,filterOrder]
smoothingOpts.WATemporalPad = false;        % Pads the start of the data↙
by half a smoothing kernel
smoothingOpts.WATemporalPadFrames = 3;      % Extra frames to pad in↙
addition to the half smoothing kernel
smoothingOpts.WATemporalPadMethod = 'replicate';

%↙
-----------------------------------------------------------------------↙
```

```
--
% Data Extrapolation Options
% NOTE: the default options specified here should be suitable for most
% applications

% Options for replacing any NaNs that appear after grid processing
% This may occur when using the iterative displacement calculation with↙
the
% grid method code.
extrapOpts.fixNaNs = true;
extrapOpts.fixNaNKernal = 3*grid.pxPerPeriod;    % Radius for averaging↙
to replace NaNs in px

% Options for edge padding method
extrapOpts.imageDef = false;
extrapOpts.padEdgeMethod = 'regression';         % options: quadratic,↙
regression (linear fit), linear (default - interp1)
extrapOpts.padFitWindow = grid.pxPerPeriod;      % number of pixels to↙
use for extrapolation fitting (quadratic and regression options only)
extrapOpts.extendPadBeforeSmooth = false;

% Extrapolation options for width averaged data used with the stress-↙
gauge
extrapOpts.extrapWidthAvgData = true;
extrapOpts.dispPx = grid.pxPerPeriod;
% Extrapolation options for the full-field data used with the VFM
extrapOpts.extrapFullFieldData = true;
extrapOpts.FFDispPx = grid.pxPerPeriod;

%% INITALISE - Stiffness and Strength Identification Options
fprintf('Creating data structures for stiffness and strength↙
identification options.\n')

%↙
------------------------------------------------------------------------↙
--
% Stiffness Identification Options for the VFM
% NOTE: the default options should be suitable, the user may want to
% adjust the temporal range over which the stiffness average is↙
calculated
% using the variable: VFOpts.avgQVsTRange
% Specify the temporal range over which the VFM identification is↙
performed
% at the start of the test there is not enough information so it can be↙
cut
VFOpts.startFrame = 5;
VFOpts.endFrame = time.numFrames;
% Option to move the impact edge boundary condition to remove smoothing
% edge effects from the VFM analysis
VFOpts.cutImpactEdge = false;
VFOpts.cutEdgePx = grid.pxPerPeriod+1;
% Specify the range over which the average stiffness will be identified,
% defaulted to between 20% and 60% of the recording time but should be
% adjusted based on the stable portion of each test
VFOpts.avgStartTime = 0.2*time.vec(end);
VFOpts.avgEndTime = 0.7*time.vec(end);
VFOpts.avgQVsTRange = (round(VFOpts.avgStartTime/time.step)+1):...
    (round(VFOpts.avgEndTime/time.step)+1);
% Number of elements in the piecewise special optimsied VFM, for the
% reduced orthotropic model nElemY should be 1
VFOpts.nElemX = 5;
if strcmp(globalOpts.matModel,'orthotropicReduced')
    VFOpts.nElemY = 1;
else
    VFOpts.nElemY = 4;
end

%↙
-------------------------------------------------------------------------↙
--
% Stress-Gauge and Generalised Stress-Strain Curve Options
% NOTE: the default options will be suitable for most applications here,
% For the isotropic model, if the Poisson's ratio identification with↙
the
% VFM is highly unstable the user may want to specify the value of
% Poisson's ratio as the one from the material struct using:
% stressGaugeOpts.strainCalcNuxy = 'QS'

% Possible choices:
% 1) 'full' - fits whole ss curve
% 2) 'componly' - fits up to the max compressive stress
% 3) 'tensonly' - fits up to the max tensile stress
% 4) 'maxThreshold' - fits up to a specified threshold value
% 5) 'maxabs' - fits up to the absolute max stress tensile/compressive
stressGaugeOpts.fitRangeOpt = 'componly';
% Which variable to use for the min and max threshold, possible options↙
'stress' or 'strain'
stressGaugeOpts.limParam = 'stress';
% Set to zero to disable a min threshold, note that a min threshold can↙
be
% used for all options 1-5 just by setting this to something non-zero
stressGaugeOpts.minThreshold = 0;
% Max threshold value is only used for option 4 'maxThreshold' above
stressGaugeOpts.maxThreshold = 50e6;
% Default range of frames over which to perform the fit
stressGaugeOpts.fitDefaultRange = time.cutFrames:time.numFrames-time.↙
cutFrames;
% Option to use a robust linear least squares fit
stressGaugeOpts.robustFit = false;
% Specifies the range over which the average stiffness is identified,
% defaults to between 25% and 75% of the length from the free edge. This
% avoids the poor identification at the edges
stressGaugeOpts.avgQVsLRangePc = [0.25,0.75];
% For the isotropic model the poisson's ratio is required, this variable
% specifies where the value is taken from. Defaults to the optimised VF.
stressGaugeOpts.strainCalcNuxy = 'VFMan'; % VFOpt - use average from↙
VFO, VFMan - average from VFMan , QS - take nu from material struct
if globalOpts.processOptVF
    stressGaugeOpts.strainCalcNuxy = 'VFOpt';
end

%↙
-------------------------------------------------------------------------↙
--
% Strength Identification Options
% NOTE: the default options will be suitable for most applications here,↙
if
% the data is not very noisy the size of the virtual gauge may be↙
reduced

% Specify the method for determining the strength
% 'Manual' - allows user to specify strength frame
% 'Automated' - takes the maximum value of the stress gauge in the↙
virtual gauge area
strOpts.method = 'Automated';
% Size of the virtual gauge for strength identification in pixels
strOpts.virtualGauge.Xpx = 20;
strOpts.virtualGauge.Ypx = 30;
% Specify which identified stiffness to to use for stress calc from↙
strain
% 'Avg' - average of VF and SG, 'VF' - use VF value, 'SG' use stress↙
gauge
if globalOpts.processOptVF
    strOpts.stiffnessMethod = 'Avg';
else
    strOpts.stiffnessMethod = 'SG';
end
% If a fracture location has been previously selected then automatically
% load this file if true.
strOpts.autoLoadFractureFile = false;

%% INITALISE - Plot Formatting Options
% NOTE: all options in this section only control cosmetic appearance of
% plots and maps produced by the code.
fprintf('Creating data structures for figure formatting options.\n')

%↙
-----------------------------------------------------------------------↙
--
% Plot Parameters Structure
% Creates data structure for figure formatting using figure size and↙
latex/
% font size that is suitable for journal articles
plotParams.formatType = 'article';
plotParams.saveImageMatFig = false;     % Save a .fig as well as a .png↙
for images and image sequences
plotParams.saveImageVecFig = false;      % Save a vector image file as↙
well as a .png
plotParams.imageAxis = true;            % Fixes the aspect ratio on heat↙
maps to the true aspect ratio of the image using the matlab command↙
'axis image'
plotParams.cAxisLoc = 'eastoutside';    % Specifies the location of the↙
colorbar for heat maps, for long thin samples use 'southoutside'
```

```
% Allows the user to remove the reconstructed edge data from the plotted
% fields (cosmetic only). Full-field including reconstructed data is
% plotted by default
plotParams.cutEdgePx = false;
if plotParams.cutEdgePx
    plotParams.cutPxX = grid.pxPerPeriod+round(smoothingOpts.↙
spatialKernal(1)/2);
    plotParams.cutPxY = grid.pxPerPeriod;
else
    plotParams.cutPxX = 0;
    plotParams.cutPxY = 0;
end
% Options: 'Specified','Auto','MaxQuantileOverCompOnly','MaxQuantile'
% Use 'MaxQuantile' if ranges of each parameter below are unknown, for
% nicer plot formatting use 'Specified' with the limits below
plotParams.cAxisType = 'MaxQuantile';

% Use these axis limits if 'Specified' is selected above
plotParams.cAxisDisp = [-0.06,0.06];
plotParams.cAxisAccel = [-7e6,7e6];
plotParams.cAxisStrain = [-4,4];
plotParams.cAxisRawStrain = [-10,10];
plotParams.cAxisStrainRate = [-1500,1500];
plotParams.cAxisStress = [-2000,1500];

%↙
------------------------------------------------------------------------↙
--
% Stiffness Identification Plot Options

% Options for plotting identified stiffness against time from the VFM,
% manual and optimised
VFPlotOpts.formatType = plotParams.formatType;
VFPlotOpts.plotAvg = true;                           % Plot the average↙
identified stiffness as a horizontal line
VFPlotOpts.plotAvgErrBound = true;                   % Plot error bounds↙
around the identified average as horizontal lines
VFPlotOpts.plotAvgErrBoundPc = 0.1;                  % Used for plotting↙
% bounds around the material.Q value for comparison
VFPlotOpts.plotQS = true;                            % Used for plotting↙
% bounds around the identified average value
```

```
VFPlotOpts.plotQSErrBound = false;                   % Allows error↙
bounds on the QS value to be plotted for comparison
VFPlotOpts.plotQSErrBoundPc = 0.05;                  % Used for plotting↙
% bounds around the material.Q value for comparison with QS value
VFPlotOpts.axisLimVar = 'avg';                       % Creates the axis↙
scale based the average identified 'Avg' or the 'QS' ref
VFPlotOpts.axisRangeFactor = 5;                      % Specified the↙
axis range for VF stiffness vs time plots
VFPlotOpts.plotAvgTimeRange = true;                  % Plot the window↙
over which the average identified stiffness was taken
VFPlotOpts.printAvgInTitle = true;                   % Plot the average↙
identified stiffness in the title
VFPlotOpts.tRange = 5:time.numFrames;                % Specify the time↙
range over which to plot the VF stiffness vs time data
VFPlotOpts.specifyAxisLims = false;                  % Allows user to↙
hard code axis limits of VF stiffness vs time plots
VFPlotOpts.axisLims = [0,time.vec(end)*10^6,0,0.5];  % Used for plotting↙
bounds of Poisson's ratio which is usually between 0 and 0.5
VFPlotOpts.unitConv = 10^-9;                         % Allows stiffness↙
to be displayed in GPa

% Options for plotting stress-strain curves using the stress-gauge↙
equation
ssCurvePlotParams.formatType = plotParams.formatType;
% Specifies the locations of the stress-strain curves to be plotted as a
% fraction of the specimen length from the free edge.
ssCurvePlotParams.locXPcVec = [0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9];
% Specified the temporal range over which the stress-strain data is↙
plotted
ssCurvePlotParams.tRange = time.cutFrames:time.numFrames-time.cutFrames;
% Number of rows and columns of SS curve plots there must be enough↙
space
% for all the requested plots:
% ssCurvePlotParams.Rows*ssCurvePlotParams.Cols == length↙
(ssCurvePlotParams.locXPcVec)
ssCurvePlotParams.Rows = 2;
ssCurvePlotParams.Cols = 4;

% Options for plotting identified stiffness against the length from
% Stress-strain curves from the standard stress-gauge
ssCurveIdentPlotOpts.formatType = plotParams.formatType;
ssCurveIdentPlotOpts.plotAvg = true;              % Plot the average↙
identified stiffness as a horizontal line
ssCurveIdentPlotOpts.plotAvgErrBound = true;      % Plot error bounds↙
around the identified average as horizontal lines
ssCurveIdentPlotOpts.plotAvgErrBoundPc = 0.1;     % Used for plotting %↙
bounds around the identified average value
ssCurveIdentPlotOpts.plotQS = true;               % Plot a horizontal↙
line indicating the QS ref for comparison
ssCurveIdentPlotOpts.plotQSErrBound = false;      % Allows error bounds↙
on the QS value to be plotted for comparison
ssCurveIdentPlotOpts.plotQSErrBoundPc = 0.05;     % Used for plotting %↙
bounds around the material.Q value for comparison with QS value
ssCurveIdentPlotOpts.axisLimVar = 'avg';          % Creates the axis↙
scale based the average identified 'avg' or the 'QS' ref
ssCurveIdentPlotOpts.axisRangeFactor = 5;         % Specified the axis↙
range for VF stiffness vs time plots this factor is multipled by the↙
error bar range
ssCurveIdentPlotOpts.plotAvgPosRange = true;      % Plot the window over↙
which the average identified stiffness was taken
ssCurveIdentPlotOpts.printAvgInTitle = true;      % Show the average↙
identified stiffness value in the figure title
ssCurveIdentPlotOpts.specifyLRange = true;        % Used for plotting↙
off-axis sample identification where the number of slices is not equal↙
to the specimen length
ssCurveIdentPlotOpts.unitConv = 10^-9;            % Allows stiffness to↙
be displayed in GPa

%% INITALISE - Save all initialisation paramters
fprintf('Saving processing parameters file(s).\n')
for ii = 1:length(savePaths)
    save(fullfile(savePaths{ii},saveFile))
end
fprintf('COMPLETE.\n')
```