#Input: 3-d surface file from step 3

#Function: Upload 3D reconstructed points (either as seperate pairs or a stitched surface)

and calculate displacements, deformations, and strains.

Load file

Save results?

Yes:

Qsave3DDICPPresults

No:

Pass

3D reconstruction using Direct Linear Transformation

Get number of images

nImages= numel(DIC3D.Points3D); (in this case is 5)

pre-allocate 3D-DIC result variables

*DIC3D.xxxxx=cell(1,nImages)*

Cell is an array that can collect data of different type

Cell(1,5) is to create a empty 1X5 cell

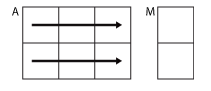
#Calculate Displacements:

loop over images (ii = 1-5):

Face correlation coefficient (worst)

*DIC3D.FaceCorrComb{ii}=max(DIC3D.corrComb{ii}(F),[],2);*

Max ( (F),[],2 ) return max value of F @ dim 2 (each row)



compute face centroids

DIC3D.FaceCentroids

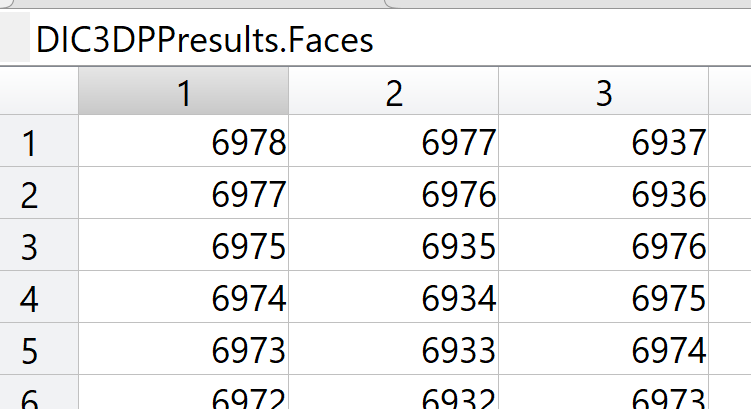
Loop over each face (iface = 1-3):

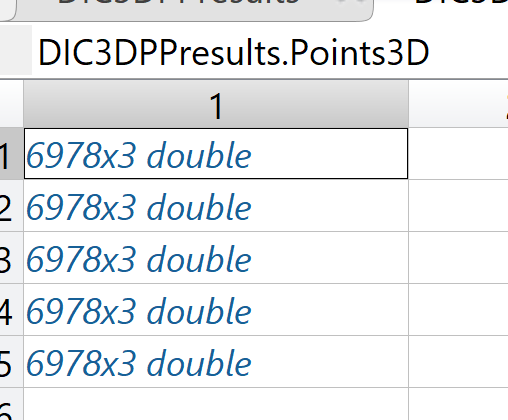
*DIC3D.FaceCentroids{ii}(iface,:)=mean(DIC3D.Points3D{ii}(F(iface,:),:));*

Returns a mean value of row number--F(iface)

(the max value of F = instances number of Points3D)

(just to understand, not a part of pseudo code)





Compute displacements between frames (per point)

*DispVec = DIC3D.Points3D{ii}-DIC3D.Points3D{1};*

*DIC3D.Disp.DispVec{ii} =DispVec* #Nth column - 1st column

*DIC3D.Disp.DispMgn{ii}=sqrt(DispVec(:,1).^2+DispVec(:,2).^2+DispVec(:,3).^2);*

#that’s DispVec each column's value ^2 and sum up, then sqrt.

Compute rigid body transformation between point clouds

 rigidTransformation

*DIC3D.RBM.RotMat{ii},DIC3D.RBM.TransVec{ii},DIC3D.Points3D\_ARBM{ii}*

These three are values returned from rigidTransformation function, py code available

The input of this function are *'DIC3D.Points3D{ii}'and 'DIC3D.Points3D{1}*'

Compute displacements between sets - after RBM

DIC3D.Points3D\_ARBM, needs data produced by previous lines.

Doing same things as "Compute displacements between frames (per point)" by using DIC3D.Points3D\_ARBM instead of DIC3D.Points3D

Compute face centroids - after transformation

DIC3D.FaceCentroids\_ARBM doing same thing as "compute face centroids" by using DIC3D.Points3D\_ARBM instead of DIC3D.Points3D

#compute deformation and strains (per triangular face)

 triSurfaceDeformation (functions need to be discussed separately)

Run "triSurfaceDeformation" twice, with input as:

(F,DIC3D.Points3D{1},DIC3D.Points3D) and

(F,DIC3D.Points3D\_ARBM{1},DIC3D.Points3D\_ARBM)

Assign them to :

DIC3D.Deform and

DIC3D.Deform\_ARBM

compute triangle regularity (isotropy index)

 faceIsotropyIndex

Loop over images (ii 1-5):

*DIC3D.FaceIsoInd = faceIsotropyIndex(F,DIC3D.Points3D{ii});*

save results

Plot results