

DuoDIC

Instruction Manual

Version 1.1.0

February 1, 2022

Dana Solav (danas@technion.ac.il)

Technion - Israel Institute of Technology

Table of Contents

1	Overview	3
2	Installation	3
2.1	Operating system requirements	3
2.2	MATLAB requirements	3
2.3	MEX Setup	3
2.4	Additional included packages	4
2.5	Installation	4
3	Preparation	4
3.1	Images of a flat checkerboard pattern	5
3.1.1	Preparing the checkerboard pattern	5
3.1.2	Capturing images of the pattern	6
3.2	Images of a speckled test object	7
3.2.1	Speckling the test object	7
3.2.2	Capturing images of the test object	7
3.3	File Naming and Storing	7
3.3.1	Stereo calibration Checkerboard images	7
3.3.2	Speckle images	7
4	DuoDIC Main Scripts	8
4.1	Step 1: Stereo Camera Calibration	8
4.1.1	Run DuoDIC_STEP1_Stereo_calibration	8
4.2	Step 2: 2D-DIC Using Ncorr	9
4.2.1	Run DuoDIC_STEP2_2DDIC_using_Ncorr	9
4.3	Step 3: DuoDIC 3D Reconstruction	14
4.3.1	Run DuoDIC_STEP3_3D_reconstruction	14
4.4	Step 4: Post Processing	14
4.4.1	Run DuoDIC_STEP4_Post_processing	15
5	Viewing and manipulating figures and exporting animated GIFs	16
6	References	18

1 Overview

DuoDIC (Duo Digital Image Correlation) is an open-source MATLAB toolbox by Dr. Dana Solav. Three-dimensional (stereo) Digital Image Correlation (3D-DIC) is an important technique for measuring the mechanical behavior of materials. DuoDIC was developed to allow easy-to-use and straightforward DIC processing which can be used with any setup of two cameras with an overlapping field of view. DuoDIC integrates the 2D-DIC subset-based software [Ncorr](#) with MATLAB's camera calibration algorithms to reconstruct 3D surfaces from stereo image pairs. Moreover, it contains algorithms for computing and plotting displacement, deformation and strain measures. High-level scripts allow users to perform 3D-DIC analyses with minimal interaction with MATLAB syntax, while proficient MATLAB users can use stand-alone functions and data-structures to write custom scripts for specific experimental requirements. Comprehensive documentation, user guide, and sample data are included.

2 Installation

2.1 Operating system requirements

DuoDIC was developed on 64-bit Windows 10 and has not yet been tested on other operating systems.

2.2 MATLAB requirements

DuoDIC was developed on MATLAB version R2019b, and has not yet been tested on other versions.

MATLAB toolbox requirements:

- Image Processing Toolbox
- Statistics and Machine Learning Toolbox
- Computer Vision Toolbox

2.3 MEX Setup

DuoDIC includes the 2D-DIC software Ncorr, which requires a MEX (C++) compiler. More details can be found in the Ncorr instruction manual, which is included in DuoDIC\lib_ext\ncorr_2D_matlab-master, and also in the following links: <http://www.ncorr.com/> and https://github.com/justinblaber/ncorr_2D_matlab.

First, you have to make sure that a supported C++ compiler is installed on your computer. Find a C++ compiler that is supported for your MATLAB version here: <https://www.mathworks.com/support/requirements/supported-compilers.html>. If you're not using the most current MATLAB version, click on 'View compiler support for previous releases' on the top right side of the page to access the correct version.

After downloading a supported compiler, make sure MEX has been set up properly in MATLAB. Type `mex -setup` in the MATLAB Command Window. The output should be something similar to:

MEX configured to use 'MinGW64 Compiler (C++)' for C++ language compilation.

(The compiler name 'MinGW64' may be replaced by the name of your installed compiler).

The free MinGW C++ compiler, which is available for download in the above link, works as long as you only use the single threaded version of Ncorr. If you wish to use Ncorr's multithreading option, you will require a compiler that supports OpenMP (see Ncorr manual for further details).

2.4 Additional included packages

DuoDIC uses functions from GibbonCode, the Geometry and Image-Based Bioengineering add-On: <https://www.gibboncode.org/>. All the necessary functions from GibbonCode are already included in DuoDIC, however you are encouraged to check out what other capabilities GibbonCode has to offer (finite element analysis, meshing tools, image segmentation, and more). DuoDIC also uses the following packages and codes: [export fig](#), [findjobi](#), [inputsdlg](#), [numsubplots](#), [selectdata](#), [subtightplot](#), [uipickfiles](#).

2.5 Installation

After MEX is set up correctly, in MATLAB, navigate to the directory where you saved DuoDIC, and type `installMultiDIC` in the MATLAB terminal. This will compile all the necessary files for Ncorr and will save the Matlab MEX files in the Ncorr folder. It will also add all the necessary files to MATLAB search path. It has to be done only the first time. If this is the first time you install, you will get a dialogue box warning you about missing compiled files. Click OK to continue. Then, another dialogue box will appear regarding OpenMP support, as shown in Figure 1:

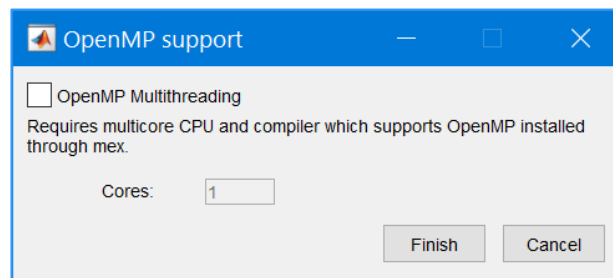


Figure 1. OpenMP support MATLAB dialogue box

Click the checkbox for OpenMP if you have a multicore processor, want multicore support, and have installed a supported compiler with OpenMP support. Enter the number of cores on your system (which can be determined by checking the Performance tab in Windows' task manager). If you choose not to use multithreading, then just leave OpenMP support unchecked and click finish. The files should compile and display in the Command Window, similarly to what is shown below:

```
MEX completed successfully.  
Installing ncorr_alg_dispgrad... Please wait  
Building with 'MinGW64 Compiler (C++)'.
```

If all the files compiled properly, a Graphic User Interface (GUI) for Ncorr should appear, as shown in Figure 2. If this is the case, Ncorr has most likely installed correctly. The next time you open MATLAB, you will not have to repeat the installation process, as long as you do not delete the compiled MEX files that were saved in the Ncorr folder. If this is the case, Ncorr has most likely installed correctly. The next time you open MATLAB, you will not have to repeat the installation process, as long as you do not delete the compiled MEX files that were saved in the Ncorr folder.

3 Preparation

The two following sets of images are required to complete a 3D-DIC analysis:

- A set of stereo calibration images, in which a flat checkerboard is imaged by both cameras simultaneously.

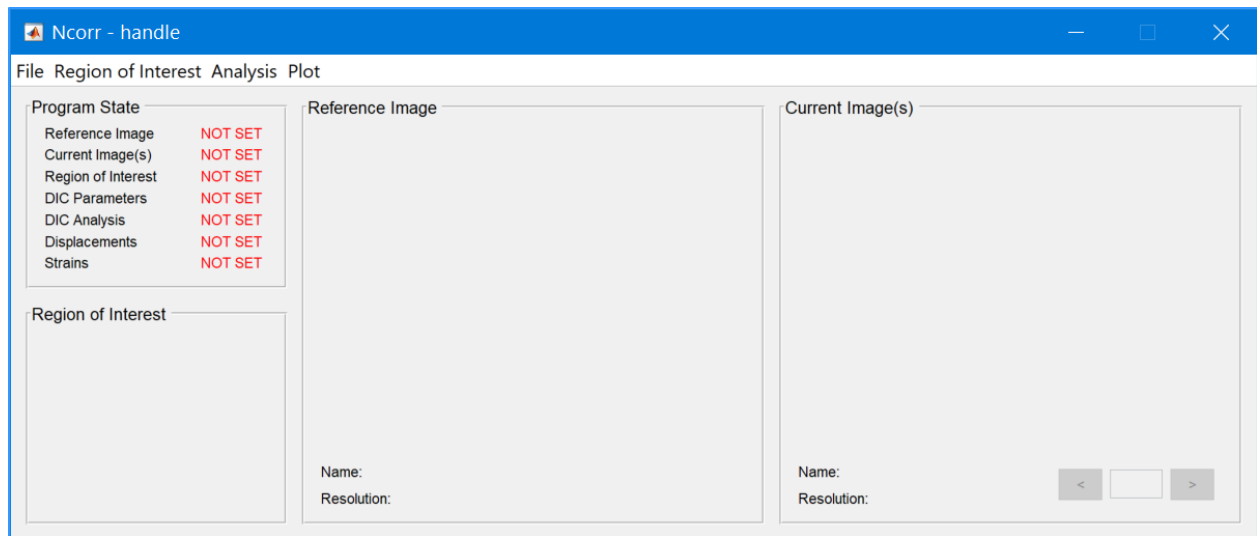


Figure 2. Ncorr MATLAB Graphic User Interface (GUI).

- A set of speckle images, in which a speckled test object is imaged simultaneously by both cameras in a reference (e.g. undeformed) configuration and optionally in one or more deformed configurations.

The preparation steps required for each of these steps are described in detail in the following sections. In addition, you are encouraged to read about [DIC good practices guide](#) (free) and take a DIC course (such as the [one offered by iDICs](#) for example) in order to improve your experimental setup and processing.

3.1 Images of a flat checkerboard pattern

3.1.1 Preparing the checkerboard pattern

For most applications it is sufficient to use a flat calibration target made by attaching a pattern printed on adhesive paper onto a flat surface (such as a glass frame).

Print an asymmetric checkerboard image (odd number of rows and even number of columns, for example) with (accurately) known square size.

To print MATLAB's standard pattern, type `open checkerboardPattern.pdf` in the command window to print MATLAB's pattern.

If you want to print a different size pattern (for example a different number of squares), you can use the DuoDIC function `createCheckerBoardImage` to create an image for printing. To do this, type:

```
createCheckerBoardImage(Nrows, Ncol, squareSize, resolution, fileName)
```

in the command window, where `Nrows` and `Ncol` are your selected number of rows (odd number) and columns (even number), `squareSize` is the size of each square in meters, `resolution` is the required image resolution in pixels/meter (for example 600 dpi is equivalent to ~23622 pixels/meter), `fileName` is the name of the file for saving the checkerboard image (including the path directory unless you want it to be saved in the current directory).

Alternatively, you can use this free app: <https://calib.io/pages/camera-calibration-pattern-generator>.

It is important to make sure that the printer prints the checkerboard in the correct scale and does not shrink or stretch it. Make sure to accurately measure the printed checkerboard to make sure it is printed correctly.

Attach the image on a flat surface (a simple solution is printing the checkerboard pattern on a [sticker paper](#) and sticking it on a glass picture frame).

The size of your checkerboard has to fit your specific experimental setup. The board has to be placed in the approximate location where the speckled test object is placed, and the entire pattern should be viewed in the image (images where only part of it is visible will be discarded).

3.1.2 Capturing images of the pattern

Take a few dozen (at least 20, preferably 50 or more) images with the checkerboard positioned in different positions and orientations (tilting angles) in the field of view of the cameras, covering the entire field of view of the cameras. See for example Figure 3.

The images should be taken simultaneously by both cameras. Any movement of the pattern between the times the cameras are taking the image will impair the calibration.

Keep the pattern in focus, but do not use autofocus or change the zoom between images. The focus, zoom, and all other camera settings and lighting have to remain the same for all the images (both calibration images and speckle images).

Use uncompressed images or images in lossless compression formats such as PNG.

The program accepts either grayscale (size $m \times n$) or RGB (size $m \times n \times 3$) images. If you use color images in TIF format that have a 4th transparency channel, you should convert them to a different format, or delete the 4th channel, for example using `img(:, :, 4) = []`; in MATLAB.

Delete images where the board is cut (not entirely visible in both images), blurry/unfocused images, etc. It is not mandatory to do so, as the program will automatically discard those images, but it will take more time to run. If you delete an image from one camera, you must delete also the corresponding image from the other camera, even if the

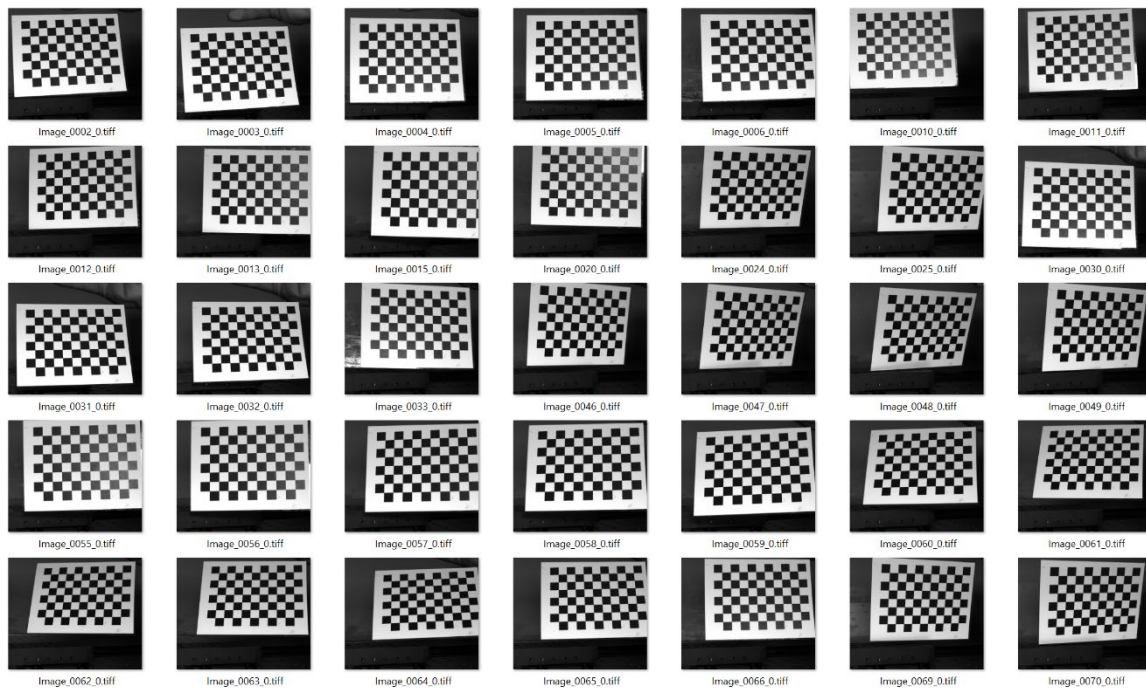


Figure 3. Example images of a checkerboard calibration target.

image looks good on the other camera. This is because only images acquired by both cameras can be used towards stereo calibration.

More details can be found in the [MATLAB Stereo Camera Calibration App page](#).

3.2 Images of a speckled test object

3.2.1 Speckling the test object

As a rule of thumb, speckles should be at least 3-5 pixels in size, have good contrast, equal size black and white areas, and no directionality (random pattern) [1]. Various methods for applying the speckles onto the surface exist, such as spray painting, stamping, printing, airbrushing, etc. A recent review on speckle pattern fabrication can assist you to select a preferable method according to your application [2].

3.2.2 Capturing images of the test object

- Make sure the cameras position, orientation, and settings, are exactly the same as they were in the calibration images. Any movement or difference in the settings will cause errors in the 3D reconstruction of the surface.
- The images of the speckled object must be captured simultaneously, unless the imaged object does not change at all between the times of the different captures.

3.3 File Naming and Storing

3.3.1 Stereo calibration Checkerboard images

Store the set of images from each camera in a folder named with the camera number at the end (after an underscore, or just a number, for example “cam_07”, “Pair_1_camera_7”, “007”, or “9”, but not “cam05”). The names of the image files do not matter but each two cameras in a stereo pair must have the same number of images (taken simultaneously) and their names must be ordered the same way, because MATLAB will determine the pairs of images based on their order, according to the names of the files in each folder.

3.3.2 Speckle images

Save the images in a folder named with the camera number at the end (after an underscore, or just a number, for example “cam_07”, “Pair_1_camera_7”, “007” or “9” but not “cam05”). The camera numbers should match those of the calibration steps. Inside each folder, the images should be named with the time frame or index after an underscore, in a way that will determine their order (For example cam01image_001, cam01image_002... or im_01, im_03...). The first image is considered as the “reference” image. The rest of the images will be ordered according to their numbers but they do not have to be consecutive.

Note: MATLAB sorts the numbers in the file names digit-by-digit, which means that if you name your files this way:

im_1.jpg, im_2.jpg, ..., im_12.jpg

They will be ordered: 1, 10, 11, 12, 2, 3, 4, 5, 6, 7, 8, 9

If you want them to be ordered the natural way, add enough zeros such that all names have at least as many digits as the largest number:

im_01.jpg, im_02.jpg, ..., im_12.jpg

4 Running DuoDIC Main Scripts

General note: When a user selection window will open, the instruction will usually be written in the header of the window.

4.1 Step 1: Stereo Camera Calibration

DuoDIC_STEP1_Stereo_calibration is the main script for calibrating the two cameras and calculating the camera parameters needed for performing the 3D reconstruction of the measured speckled surface. This step uses the checkerboard images from two cameras to calculate the camera parameters. The results of the calibration can be used for multiple tests as long as the cameras have not been moved and the camera settings have not been changed (focus, zoom, etc.). If the cameras are moved or adjusted, STEP1 must be repeated.

To learn more about how the stereo calibration works, refer to this [MATLAB documentation](#). The calibration algorithms are based on the works of Zhang [3], Heikkila and Silven [4], Bouguet [5], and Bradski and Kaehler [6].

4.1.1 Run DuoDIC_STEP1_Stereo_calibration

- a. Select the folder with the checkerboard images from the first camera (the order does not matter but it must be the same for the calibration and speckle images).
- b. Select the folder with the checkerboard images from the second camera.
- c. Enter the checkerboard square size in mm.
- d. The Stereo Camera Calibrator GUI will appear. Wait for the images to upload and for the checkerboard corner points to be detected.
- e. Make sure that all the images in the Data Browser are matched correctly between the two cameras and that the pattern points appear to have been detected correctly.
- f. Click on “Calibrate” (green triangle). When calibration is done, the window will look like in Figure 4.
- g. The bottom right window shows the positions and orientations of the cameras. The coordinate system is right-handed, and defined with its origin at the optical center of first camera, the x-axis points to the right, the y-axis points down, and the z-axis points away from the camera, as shown in the bottom right window in Figure 4.
- h. Examine the Reprojection Errors (bottom left window). These errors represent the distance between the detected and reprojected points (in pixels). If you identify images with much larger errors, you can exclude them by dragging down the red bar to the desired level. This will select the “outlier” images. Then, in the Data Browser on the left, right click on the selected images and click on “Remove and Recalibrate”. You can repeat this step with higher or lower threshold until you are satisfied with the result.
- i. When you finish, save the session results (calibrationSession.mat file) by clicking on “Save Session”.
- j. Return to MATLAB Command Window and press any key to finish STEP1.

Note: The calibrationSession file generated in STEP1 can be used to reconstruct 3D points from speckle images captured in multiple tests. You do not need a new calibration session for each test, as long as the cameras were not moved or re-adjusted in any way.

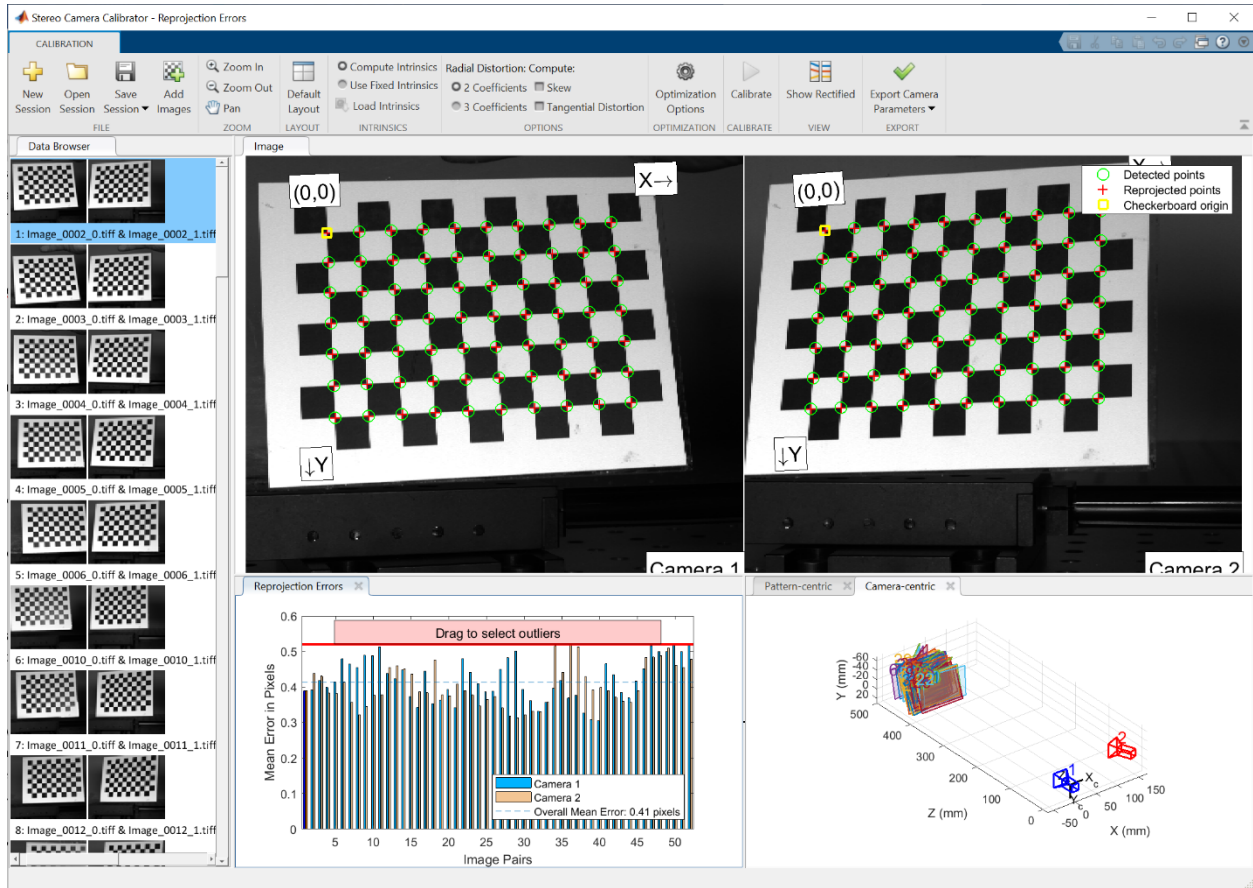


Figure 4. Stereo calibration results. The top windows show the detected vs. the reprojected points. The bottom right window shows the calculated position and orientation of each camera with respect to the checkerboard patterns. The bottom left window shows the mean error for each image from both cameras. The red bar can be dragged down to select outliers with large errors, which are then highlighted in blue and can be removed from calibration to improve the results.

4.2 Step 2: 2D-DIC Using Ncorr

DuoDIC_STEP2_2DDIC_using_Ncorr is the main script for analyzing stereo images of the speckled object using 2D Digital Image Correlation (2D-DIC). This script utilized the open-source software Ncorr [7]. It can be performed either before or after STEP1. This step has to be performed before the 3D points and surfaces can be reconstructed in STEP3. For the camera pair, the first camera is considered as the *reference* camera and the second camera is considered as the *deformed* camera. This means that images taken by the second camera are analyzed as deformed versions of the images taken by the first camera.

4.2.1 Run DuoDIC_STEP2_2DDIC_using_Ncorr

- Select the folders of the first and second cameras, containing the speckle images.
- Select saving option.
- A figure showing the image pairs from both cameras will appear (see Figure 5). Review the image sets by clicking on the play button, or by scrolling the bottom bar. This figure can help you determine the region of

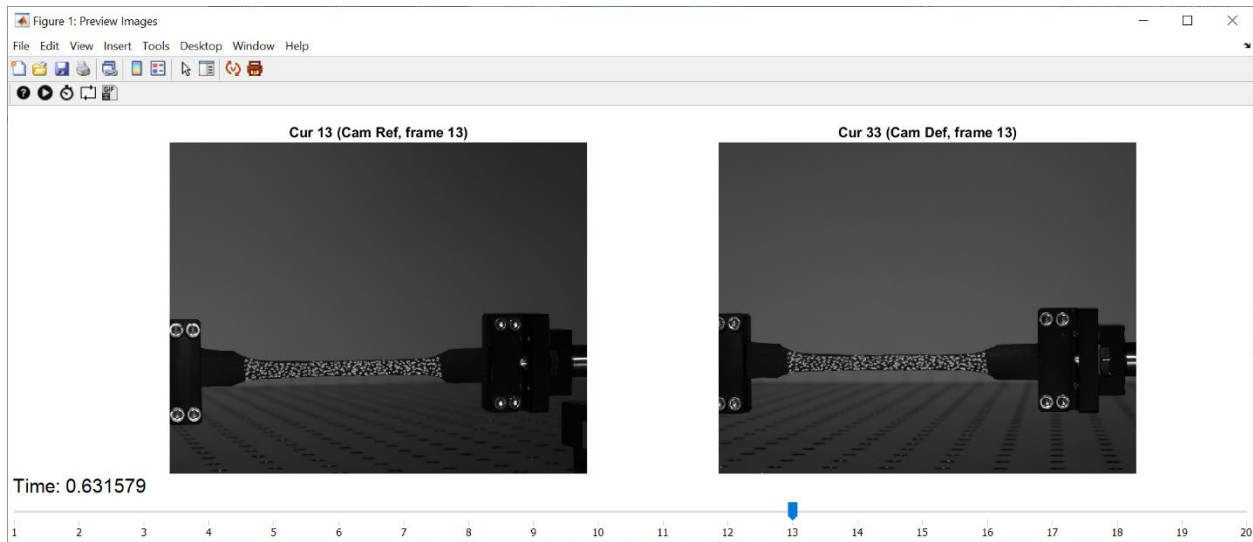


Figure 5. Animated figure displaying all the images from both cameras.

interest (ROI). If you clicked the play button, click the stop button before continuing. To continue, press any key in MATLAB command window.

- d. A new window will appear, where you need to select between the following Region of Interest (ROI) options:
 - **New:** Drawing a new mask by drawing polygons around the ROIs. A window will appear asking for the number of ROIs (number of closed regions), give the amount and press 'OK'. An ROI is defined by drawing a polygon on the reference image (see Figure 6). Make sure that the entire ROI is visible on all the images (from both cameras). Click on the image to select the first vertex and continue placing new vertices by clicking on image points. To close the polygon, click on the first vertex or double-click. Once the polygon is closed, vertices can be moved by dragging them (the cursor will turn to a circle when you hover over a vertex). Also, the entire polygon can be translated by dragging. To finish, double-click on the polygon.
 - **Saved:** If you already have a saved mask and you want to use it again. For example, if you already ran the analysis and now you just want re-run it with different options in Ncorr such as subset size or spacing, but using the same ROI.
 - **Ncorr:** You can choose to draw the ROI in Ncorr. Ncorr has more options for drawing shapes and cutting holes in the ROI, but the GUI is smaller (cannot be resized). In the Ncorr window, go to the "Region of Interest" tab and press on "Set Reference ROI". A window will open. If you want to load an existing ROI press "Load ROI", and if you want to draw a new ROI go to "Draw ROI". Refer to Ncorr instruction manual for additional details if necessary.
- e. Next, the Ncorr window will appear, where the 2D-DIC analysis is performed. If you already set the ROI in MATLAB, the ROI will be displayed. Click Finish. If not, draw an ROI in Ncorr (see previous section).
- f. When the ROI is set, the next step is to set the DIC parameters. Select the Analysis tab on the top menu, then Set DIC parameters. A window will pop up where the Subset radius and Subset spacing can be selected (see Figure 7). These parameters are visualized on the image on the right. These are some points to consider when selecting these parameters:
 - g. The subset radius should be large enough to include at least 2-3 speckles across.

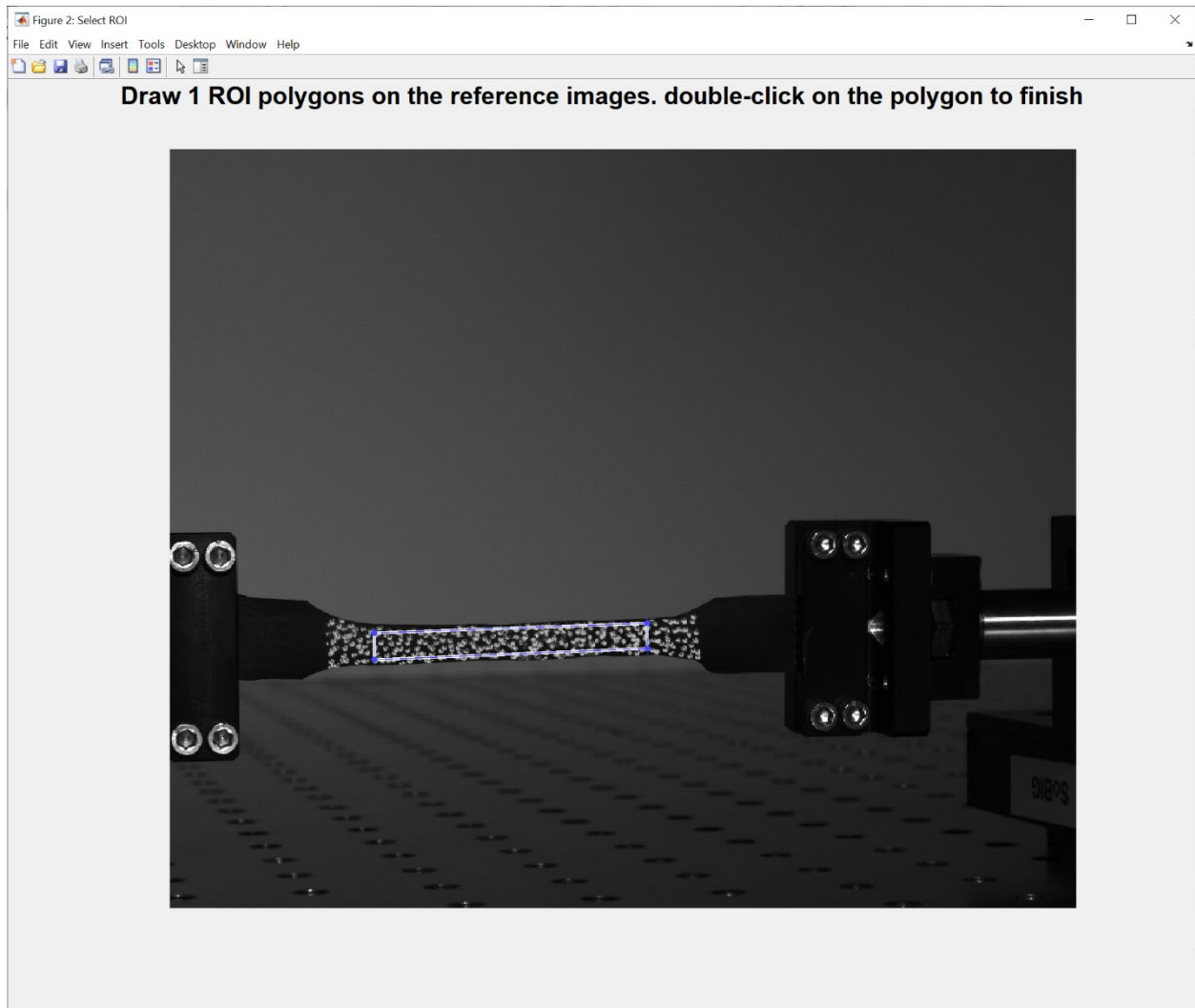


Figure 6. Select a ROI by drawing a polygon on the reference image.

- The subset radius should be small enough to satisfy the assumption that the deformation is homogeneous inside the subset.
 - The subset spacing determines the distance between data points. The smaller Subset Spacing is, the denser the grid will be. A denser grid will result in a longer analysis time, but the resolution of the 3D reconstructed surface will be higher. A smaller subset spacing (step size) also means the deformations and strains will be computed on a smaller area.
 - More information on subset size and spacing selection can be found in [1], [8].
- h. Select the desired parameters and click Finish. There is usually no need to change the other options. A window showing all the selected parameters will pop up. Click Yes.
 - i. Select the Analysis tab, then Perform DIC analysis. A Select region window will pop up. Click on Select region and then on one of the white regions on the image (if the ROI is composed of only one region, then you will have only one option).
 - j. After selecting a region, a Set seeds window will pop up. Click on Set seeds and then click on a point inside the ROI. Select a point that is clearly visible from both views, and that is not too close to the edges of the

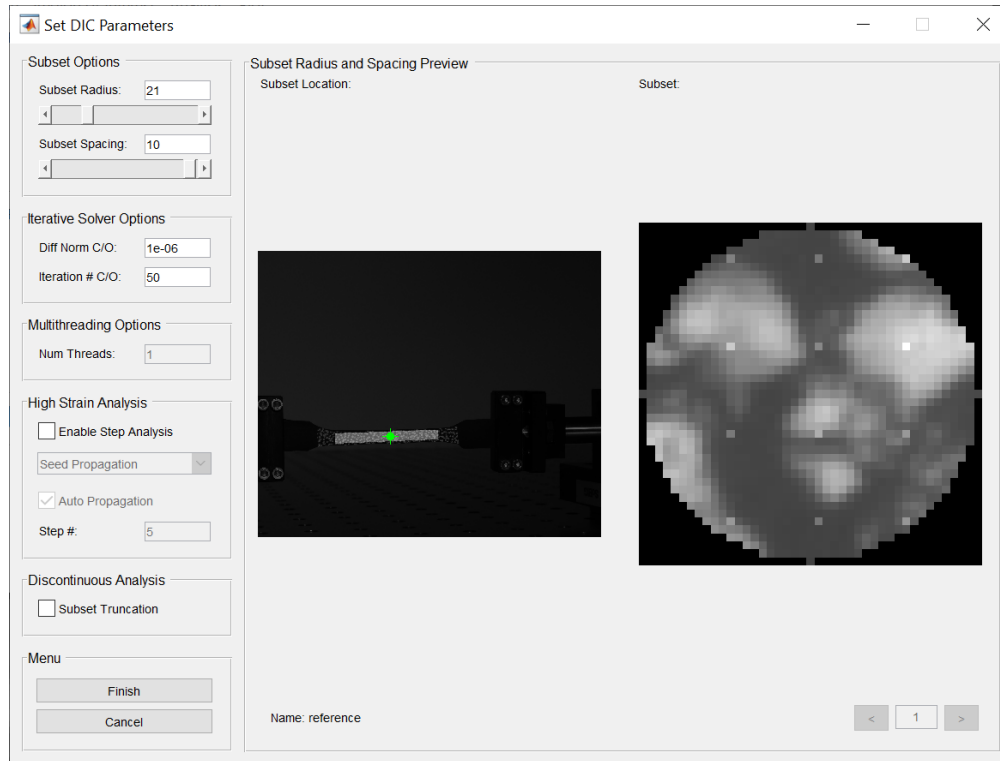


Figure 7. Subset radius and Subset spacing selection in Ncorr. Select a subset radius such that the circle contains at least 2-3 well defined speckles. The green cursor can be dragged over the ROI to inspect speckles in different regions.

ROI. See Ncorr documentation for more details on optimal seed placement, if necessary. To continue, click Finish.

- k. When *processing seeds* is finished, a *Seed Preview* window will pop up (see Figure 8). Scroll through all the images using the arrows on the bottom right to make sure that the seed point was detected correctly in all of them. If they look correct, click Finish. If the correlation coefficient between the subsets around the detected seeds is too high, an error prompt will appear. This usually means that the seed point was not detected correctly on at least one of the images. If this is the case, click cancel, move the seed to a better position, and try again until seed placement is successful. If seed placement fails again and again, try increasing the subset size.
- l. When the seeds are properly placed and you click Finish, the DIC analysis will start running, propagating from the seed points to the rest of the ROIs. When the analysis is done, a message will pop up stating that DIC analysis completed successfully. Press OK to finish.
- m. Click on Analysis, and select Format Displacements. There is no need to select anything in this window or change the settings. Just click Finish and then Yes. Since this is a 2D analysis of stereo images, the displacements here do not have a direct physical meaning. Only after the 3D reconstruction step, you will be able to view the 3D displacements.
- n. At this point the Ncorr analysis is complete. There is no need to run the Calculate Strains part. Go back to MATLAB main window without closing the Ncorr window and press any key in the command window. Then, the results will be imported from Ncorr, and the results structure will be saved in a file named DIC2DpairResults.mat in the folder you selected at the beginning of this step. You might get a warning from

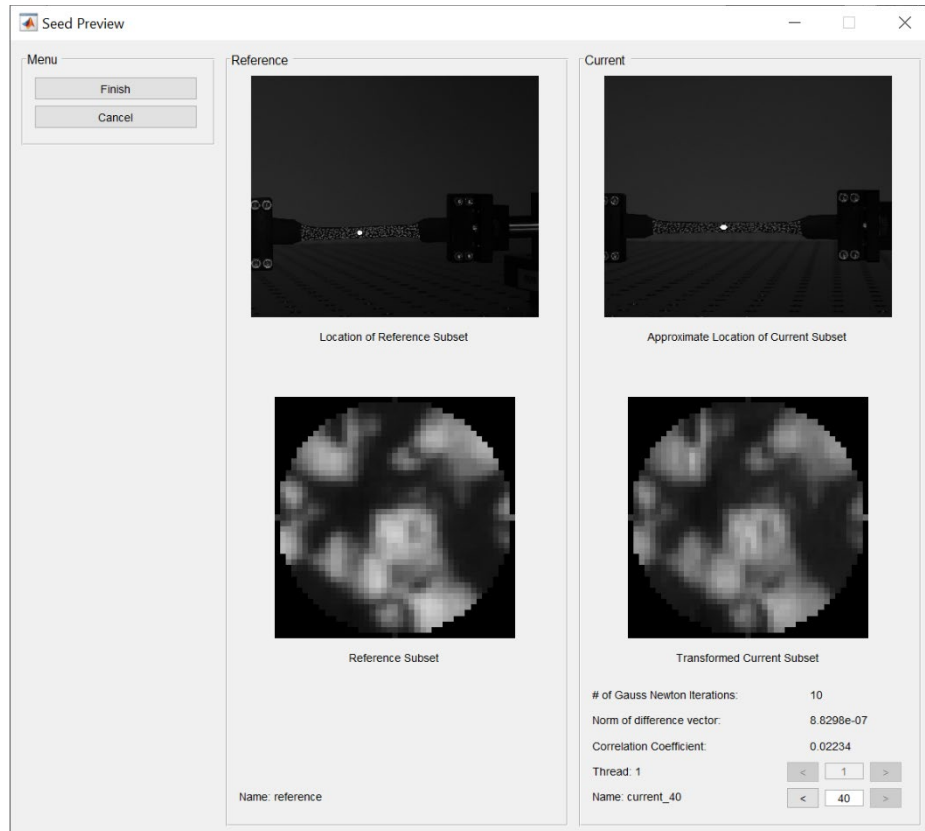


Figure 8. Seed preview window. Scroll through all images using the arrows at the bottom right of the window, and ensure that the seeds were placed correctly by inspecting their positions on the images, and the value of the correlation coefficient. If the seed point is placed in a region which is highly distorted due to the angled view, the seed placement might occasionally be wrong and the correlation coefficient too high. In this case, place the seed in a better position and try again.

Ncorr: prior DIC has been detected and will be deleted. You can click Yes, as all the necessary analysis results for 3D-DIC are saved outside Ncorr.

- o. Select if you want to plot the results now. If you select Yes, you will be requested to select if you want to change the limits of the correlation coefficients for display (leave this blank to use the default limits, which are between 0 and the maximum value of the correlation coefficient found in this analysis. Three animation figures will appear (in each figure you can animate by pressing the play button or by moving the bottom scroll bar):
 - Figure 1 plotting the reference image on the left and all the current images on the right (from both views). Corresponded points are displayed on the images with the color depicting the value of the correlation coefficient (see Figure 9).
 - Figure 2 plotting the same as 1, but the images from the two views are plotted on the left and right subplots separately.
 - Figure 3 plotting the same as 2, but instead of points, the triangular faces are plotted, and the face colors represent the combined (maximal) correlation coefficient of the three vertices.

There are special buttons on the figure interface to help examine the results. See details in Section 5.

Note: You can also run the function `plotNcorrPairResults` separately, after the results are stored. If you run `plotNcorrPairResults` without any input, you will be requested to select a `DIC2DpairResults` structure by

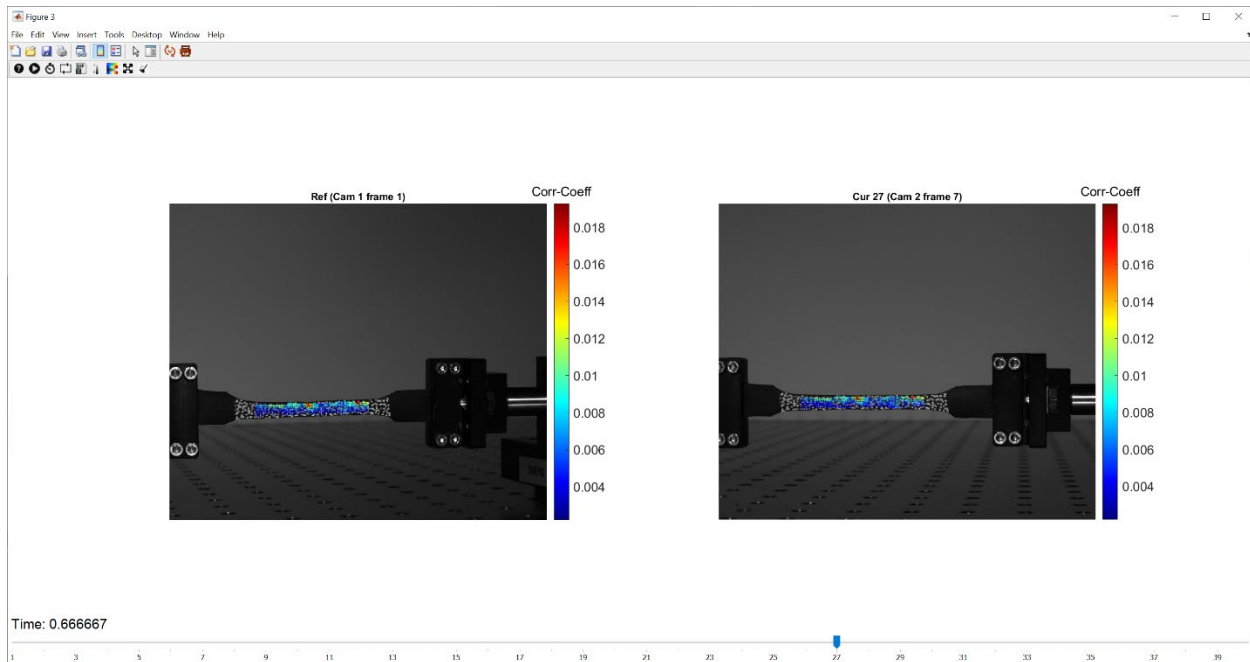


Figure 9. 2D-DIC results plotted as an animated figure. The image on the left is the reference image (first camera, first time frame), and the image on the right is a deformed image (second camera, seventh time frame). Corresponding points are plotted with colors depicting the value of the correlation coefficient of the matching between each pair of points.

browsing. Otherwise, if the DIC2DpairResults is already in the workspace, you can give it as an input to the function: `plotNcorrPairResults (DIC2DpairResults)`.

4.3 Step 3: DuoDIC 3D Reconstruction

DuoDIC_STEP3_3D_reconstruction is the main script for transforming pairs of corresponding image points from the speckle images obtained in STEP2 into 3D points and surfaces, using the calibration parameters computed in STEP1.

4.3.1 Run DuoDIC_STEP3_3D_reconstruction

- Select the 2D-DIC results file, which was the output of STEP2.
- Select a calibration session results file, which was the output of STEP1.
- Select whether or not you want to save the 3D-DIC results and the folder in which you would like to save it.
- The script uses the calibration parameters to reconstructs the 3D positions of the points which were matched on the speckle images.
- If saving was selected, a file named DIC3D_stereo will be saved in the selected location.
- An animated figure will appear, showing the reconstructed 3D points and triangular mesh representing the surface. The colors represent the values of the correlation coefficients from the matching process in STEP2.

4.4 Step 4: Post Processing

DuoDIC_STEP4_Post_processing is the main script for calculating and plotting the full-field displacements, deformations, and strains for each time step from the mesh reconstructed in STEP3.

4.4.1 Run DuoDIC_STEP4_Post_processing

- Select the output file from STEP3 (DIC3D_stereo.mat) that you want to process.
- Select whether or not you want to save the Post Processing results and the folder in which you would like to save it.
- Displacements, rigid body motion, deformation, and strains will be calculated.
- Select whether or not you want to plot the results now. If you select 'Yes', you will be asked which type of plots you want, "Plot 3D meshes" or "Plot on 2D images". In both cases, 3D results are displayed, but in the latter option, they are overlayed on the images as a background.
- Each type of plot will open a menu to select which result parameters you want to plot. Click 'Select All' to plot all the parameters (warning: a large number of figures will be created!). Click 'Select to remove rigid body motion' if you want to visualize the dynamic shape and associated parameters with the rigid body motion subtracted from the positions of the vertices (this option is only available for 3D meshes).
- Each parameter you selected will be plotted in a separate animated figure, where you can examine how the parameter changes between the time frames (see for example the principal stretches and their directions shown as a 3D mesh in Figure 10, and the displacement magnitudes on the images in Figure 11). For detailed explanation on how each measure is computed, refer to the MultiDIC paper [9].
- The results of the post-processing are saved in a structure named DIC3D_PResults.mat.

Note: You can also run the function `plot_DuoDIC_PResults` independently, after the results of STEP4 are stored. If you run the function `plot_DuoDIC_PResults` by typing it in the command window without any input, you will be requested to select a DIC3D_PResults.mat by browsing. Alternatively, if `DIC3DAllPairsResults` is in the MATLAB workspace, you can give it as an input to the function, like this: `plot_DuoDIC_PResults(DIC3D_PResults);`

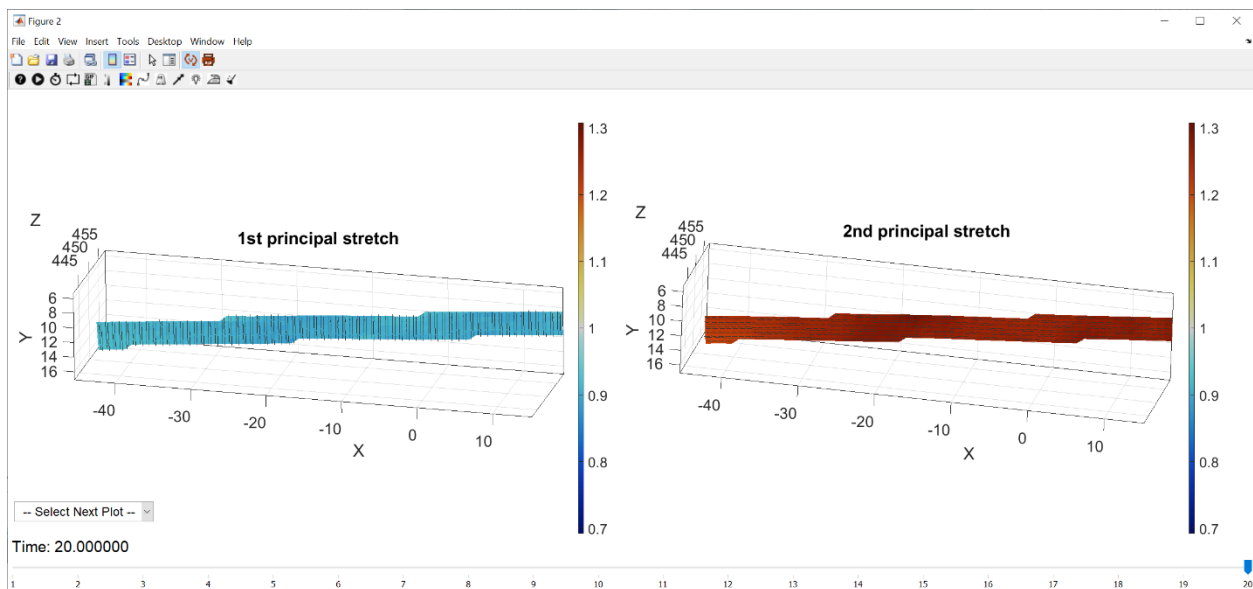


Figure 10. Post-processing results 3D plot animated figure. The first and second principal stretches are plotted as face colors and their associated directions are plotted with black lines for each triangular element.

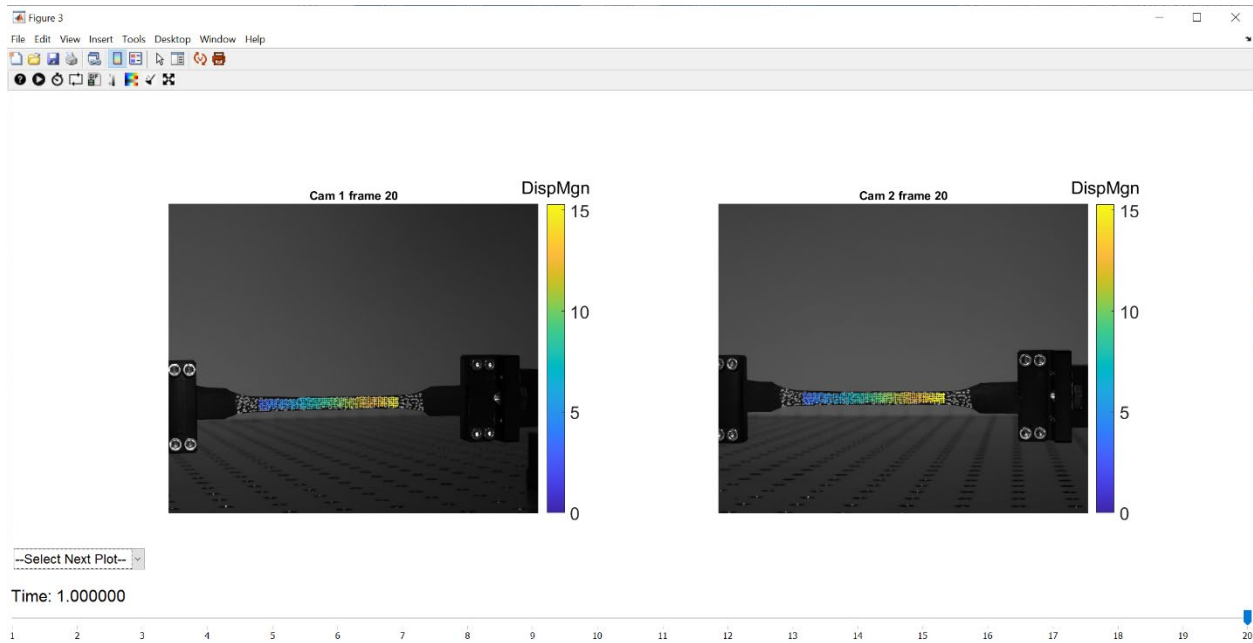



Figure 11. Post-processing results overlayed on the images. The first and second principal stretches are plotted as face colors and their associated directions are plotted with black lines for each triangular element.

5 Viewing and manipulating figures and exporting animated GIFs

This toolbox uses [GibbonCode](#)'s functions for plotting 3D points and meshes. It adds the vcw (View Control Widget) which allows users to better manipulate a view in 3D. Click the  button or press 'v' to activate it. If you re-open an existing figure, and the widget doesn't appear, type vcw in the command window to enable it. The widget allows the user to rotate, pan and zoom a figure using key presses and mouse gestures (right mouse button for zoom, left for panning, and middle/scroll for rotating). Press 'i' to show help information, as shown in Figure 12.

The following are brief descriptions of a set of widgets that may also be useful in viewing and analyzing both 2D and 3D figures. When activated, each widget will be applied to the current figure by default, unless a different figure is otherwise specified in the function call.



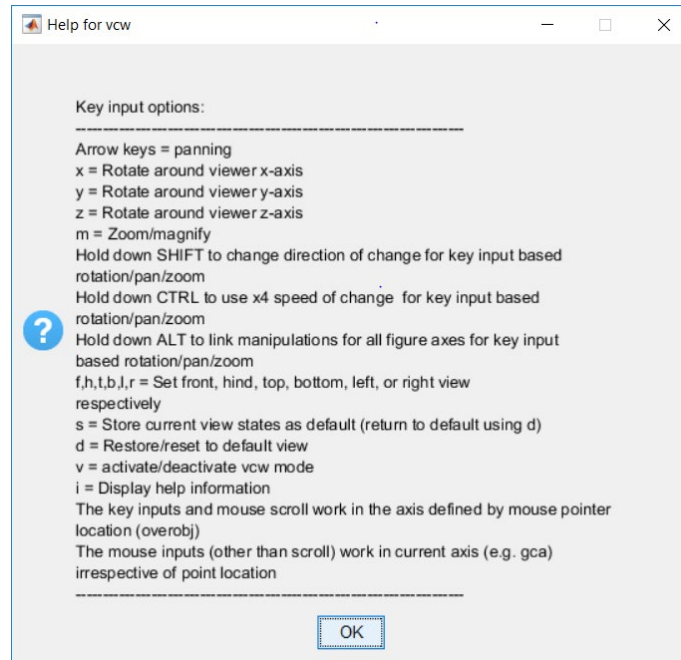



Figure 12. View Control Widget input options.

- **Light:** Click the  button to activate the light widget. This widget will allow you to change the light direction, ambient strength, diffuse strength, specular strength and the face lighting algorithm.

Light: allows the user to project light onto a figure from the left, right, or at the camera position of the current display.

Ambient strength: allows the user to change it to a value between [0,1], which determines the intensity of the ambient component of the light reflected from the object. To change ambient strength, Light must first be activated.

Diffuse strength: allows the user to change it to a value between [0,1], which determines the intensity of the diffuse component of the light reflected from the object. To change diffuse strength widget, Light must first be activated.

Specular strength: allows the user change it to a value between [0,1], which determines the intensity of the specular component of the light reflected from the object. To change specular strength, Light widget must be activated.

Face lighting algorithm: allows the user to view and change the setting, which determines the method used to calculate the effect of the light on the faces of the object: flat (uniform lighting across each face), Gouraud (linear interpolation of the light across each face), and none (turn off lighting). To change face lighting, Light must be activated.



you can select the level of conductivity, meaning to take into account only the first level or both the first and second levels of faces surrounding each face, as demonstrated in

Figure 13. Input the fraction of consideration of the other surfaces (λ_1, λ_2), between 0-1 for each level. Input the number of times you want to do the smoothing (n), which must be an integer.

•


$$C_{smooth} = (1 - \lambda_1)C + \frac{\lambda_1}{m_1} \sum_{i=1}^{m_1} C_1^{(i)}$$

$$C_{smooth} = (1 - \lambda_1 - \lambda_2)C + \frac{\lambda_1}{m_1} \sum_{i=1}^{m_1} C_1^{(i)} + \frac{\lambda_2}{m_2} \sum_{i=1}^{m_2} C_2^{(i)}$$



- Change Figure: Drop down menu on the bottom left. Select the plot you want to see next. The position, smoothness, and light you had in the previous plot.



- Clicking the  button links to the export_fig function where users can specify file names, formats, and more. For more details: <http://www.mathworks.com/matlabcentral/fileexchange/23629-export-fig>.

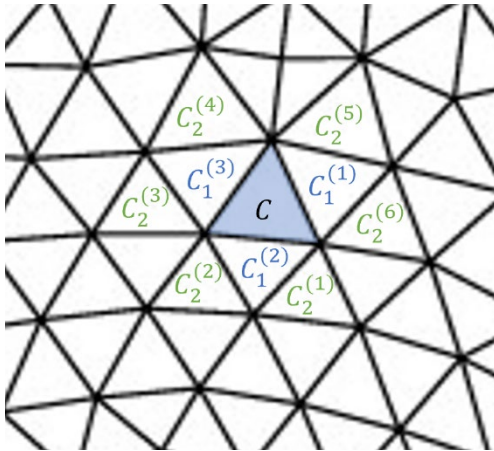


Figure 13. Face connectivity determines the smoothing algorithm. In this example, there are 3 faces in the first level and 6 faces in the second level, such that the face value after smoothing is:

$$C_{smooth} = (1 - \lambda_1 - \lambda_2)C + \frac{\lambda_1}{3} \sum_{i=1}^3 C_1^{(i)} + \frac{\lambda_2}{6} \sum_{i=1}^6 C_2^{(i)}$$

6 References

- [1] M. A. Sutton, "Digital Image Correlation for Shape and Deformation Measurements," *Springer Handb. Exp. Solid Mech.*, pp. 565–600, 2008, doi: 10.1007/SpringerReference_61945.
- [2] Y. L. L. Dong and B. Pan, "A Review of Speckle Pattern Fabrication and Assessment for Digital Image Correlation," *Exp. Mech.*, vol. 57, no. 8, pp. 1161–1181, Oct. 2017, doi: 10.1007/s11340-017-0283-1.
- [3] Z. Zhang, "A Flexible New Technique for Camera Calibration (Technical Report)," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, 2000, doi: 10.1109/34.888718.
- [4] J. Heikkila and O. Silven, "A four-step camera calibration procedure with implicit image correction," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 1106–1112, 1997, doi: 10.1109/CVPR.1997.609468.
- [5] J.-Y. Bougue, "Camera Calibration Toolbox for Matlab: Calibration examples," 2013.

http://www.vision.caltech.edu/bouguetj/calib_doc/htmls/example.html (accessed Jan. 25, 2018).

- [6] G. Bradski and A. Kaehler, *Learning OpenCV*, First Edit. O'Reilly Media, Inc., 2008.
- [7] J. Blaber, B. Adair, and A. Antoniou, "Ncorr: Open-Source 2D Digital Image Correlation Matlab Software," *Exp. Mech.*, vol. 55, no. 6, pp. 1105–1122, 2015, doi: 10.1007/s11340-015-0009-1.
- [8] B. Pan, H. Xie, Z. Z. Z. Wang, K. Qian, and Z. Z. Z. Wang, "Study on subset size selection in digital image correlation for speckle patterns," *Opt. Express*, vol. 16, no. 10, p. 7037, May 2008, doi: 10.1364/OE.16.007037.
- [9] D. Solav, K. M. Moerman, A. M. Jaeger, K. Genovese, and H. M. Herr, "MultiDIC: an Open-Source Toolbox for Multi-View 3D Digital Image Correlation," *IEEE Access*, vol. 6, pp. 30520–30535, May 2018, doi: 10.1109/ACCESS.2018.2843725.