# Ncorr

## Instruction Manual

Version 1.1
1/6/2014

Justin Blaber (jblaber3@gatech.edu)
Antonia Antoniou (antonia.antoniou@me.gatech.edu)
Georgia Institute of Technology

## Table of Contents

## 1.1 - Installation Requirements

**Version Requirements:**

- *Required:* R2009a+
- Recommended: R2012a+

**NOTE:** Ncorr was developed on MATLAB R2009a and has not been tested on versions before this. It works best on MATLAB R2012a and above, as R2009a has some problems with the stacking order of objects in the plotting tool.

**Toolboxes Requirements:**

- *Required:* Image Processing Toolbox
- *Required:* Statistics Toolbox
- Recommended: Parallel Computing

**NOTE:** Ncorr uses some parallel computing for the seed calculations. These will run in series if the parallel computing toolbox is not present or if the matlabpool is turned off.

**Operating System Requirements**:

- *Required*: Windows or Linux
- Recommended: Windows

**NOTE:** Ncorr was developed on Windows 7. It has also been tested on Linux Ubuntu 11.10. No testing has been done on mac OS.

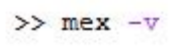
**MEX Compiler Required**:

- Visual Studio 2005 (8.0) +
- GCC 4.2+

**NOTE:** Ncorr has an option to use OpenMP for users with multicore processors. OpenMP requires the compilers listed above; if the use of OpenMP is omitted, then any C++ compiler that supports the STL library should work fine.

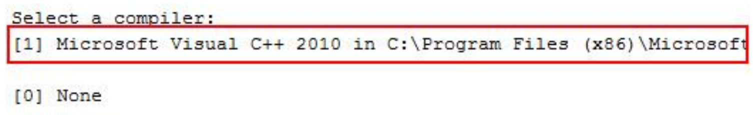
## 1.2 - MEX Setup

Make sure MEX has been set up properly in MATLAB. Type "mex -v" in the MATLAB terminal as shown below:



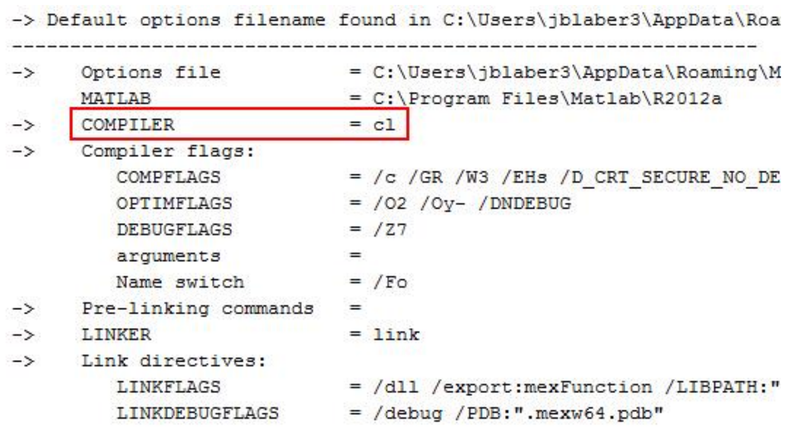
```
>> mex -v
```

The output will either be:



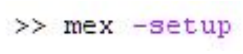
```
Select a compiler:
[1] Microsoft Visual C++ 2010 in C:\Program Files (x86)\Microsoft

[0] None
```

or:



```
-> Default options filename found in C:\Users\jblaber3\AppData\Roa
----------------------------------------------------------------
->    Options file            = C:\Users\jblaber3\AppData\Roaming\M
      MATLAB                  = C:\Program Files\Matlab\R2012a
->    COMPILER                = cl
->    Compiler flags:
         COMPFLAGS            = /c /GR /W3 /EHs /D_CRT_SECURE_NO_DE
         OPTIMFLAGS           = /O2 /Oy- /DNDEBUG
         DEBUGFLAGS           = /Z7
         arguments            =
         Name switch          = /Fo
->    Pre-linking commands    =
->    LINKER                  = link
->    Link directives:
         LINKFLAGS            = /dll /export:mexFunction /LIBPATH:"
         LINKDEBUGFLAGS       = /debug /PDB:".mexw64.pdb"
```

In the first case, MEX has not been set up so, so choose Microsoft Visual C++ (or GCC on Linux systems) and install it. In the second case, MEX has already been set up, but check to make sure the correct compiler is installed. If "COMPILER = cl" or "COMPILER = gcc," then MEX is set up with the correct compiler. If a different compiler is installed, rerun the MEX setup by typing "mex -setup" in the MATLAB terminal as shown below:

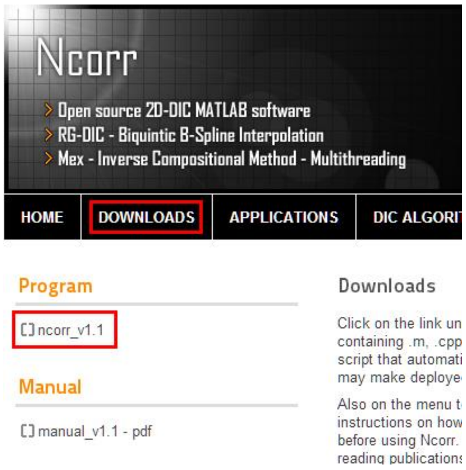


```
>> mex -setup
```

And select either Microsoft Visual C++ (or GCC on Linux systems) from the list that appears.

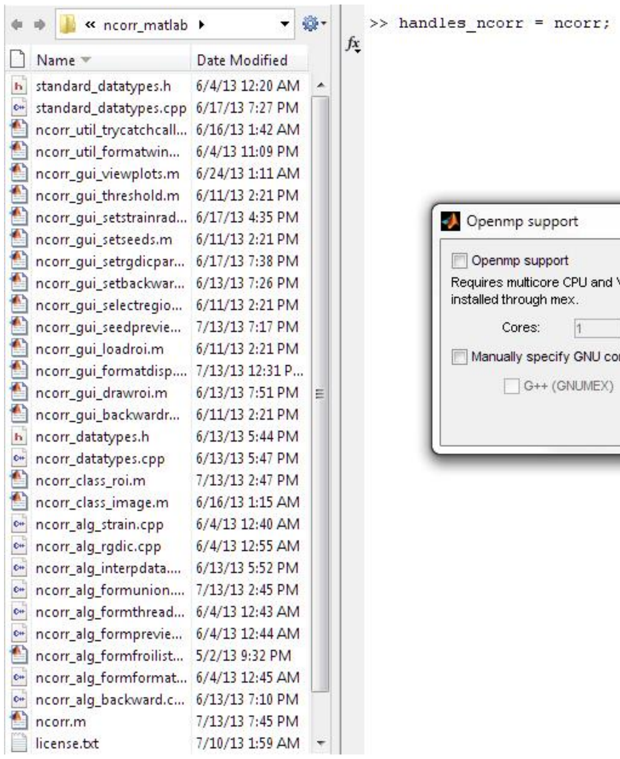
## 1.3 - Automatic Installation

After MEX is set up correctly, get the latest version of Ncorr from the ncorr.com website downloads section, as shown below:


Ncorr
> Open source 2D-DIC MATLAB software
> RG-DIC - Biquintic B-Spline Interpolation
> Mex - Inverse Compositional Method - Multithreading
HOME
DOWNLOADS
APPLICATIONS
DIC ALGORI
Program
ncorr_v1.1
Manual
manual_v1.1 - pdf
Downloads

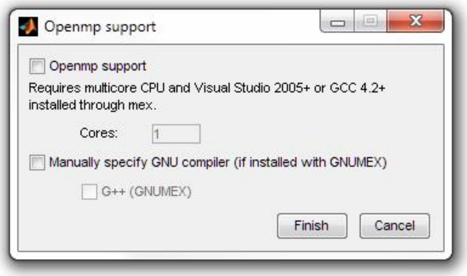
Click on the link un
containing .m, .cpp
script that automati
may make deploye
Also on the menu t
instructions on how
before using Ncorr.
reading publications

Then, navigate to the directory where you saved Ncorr. Make sure none of the files have been moved or altered. From there, type "handles_ncorr = ncorr" in the MATLAB terminal as shown below:



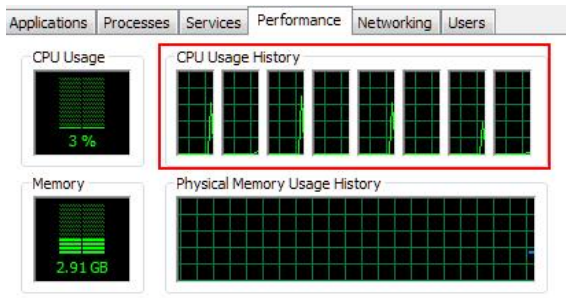
| Name | Date Modified |
|---|---|
| standard_datatypes.h | 6/4/13 12:20 AM |
| standard_datatypes.cpp | 6/17/13 7:27 PM |
| ncorr_util_trycatchcall... | 6/16/13 1:42 AM |
| ncorr_util_formatwin... | 6/4/13 11:09 PM |
| ncorr_gui_viewplots.m | 6/24/13 1:11 AM |
| ncorr_gui_threshold.m | 6/11/13 2:21 PM |
| ncorr_gui_setstrainrad... | 6/17/13 4:35 PM |
| ncorr_gui_setseeds.m | 6/11/13 2:21 PM |
| ncorr_gui_setrgdicpar... | 6/17/13 7:38 PM |
| ncorr_gui_setbackwar... | 6/13/13 7:26 PM |
| ncorr_gui_selectregio... | 6/11/13 2:21 PM |
| ncorr_gui_seedprevie... | 7/13/13 7:17 PM |
| ncorr_gui_loadroi.m | 6/11/13 2:21 PM |
| ncorr_gui_formatdisp.... | 7/13/13 12:31 P... |
| ncorr_gui_drawroi.m | 6/13/13 7:51 PM |
| ncorr_gui_backwardr... | 6/11/13 2:21 PM |
| ncorr_datatypes.h | 6/13/13 5:44 PM |
| ncorr_datatypes.cpp | 6/13/13 5:47 PM |
| ncorr_class_roi.m | 7/13/13 2:47 PM |
| ncorr_class_image.m | 6/16/13 1:15 AM |
| ncorr_alg_strain.cpp | 6/4/13 12:40 AM |
| ncorr_alg_rgdic.cpp | 6/4/13 12:55 AM |
| ncorr_alg_interpdata.... | 6/13/13 5:52 PM |
| ncorr_alg_formunion.... | 7/13/13 2:45 PM |
| ncorr_alg_formthread... | 6/4/13 12:43 AM |
| ncorr_alg_formprevie... | 6/4/13 12:44 AM |
| ncorr_alg_formfroilist... | 5/2/13 9:32 PM |
| ncorr_alg_formformat... | 6/4/13 12:45 AM |
| ncorr_alg_backward.c... | 6/13/13 7:10 PM |
| ncorr.m | 7/13/13 7:45 PM |
| license.txt | 7/10/13 1:59 AM |

```
>> handles_ncorr = ncorr;
```



☐ Openmp support
Requires multicore CPU and Visual Studio 2005+ or GCC 4.2+ installed through mex.
Cores: 1
☐ Manually specify GNU compiler (if installed with GNUMEX)
☐ G++ (GNUMEX)

A dialogue box will appear about OpenMP support. Click the checkbox for OpenMP if you have a multicore processor and want multicore support. The number of cores on your system is most easily determined by checking the task manager on Windows (or system processes on Linux) as shown below:



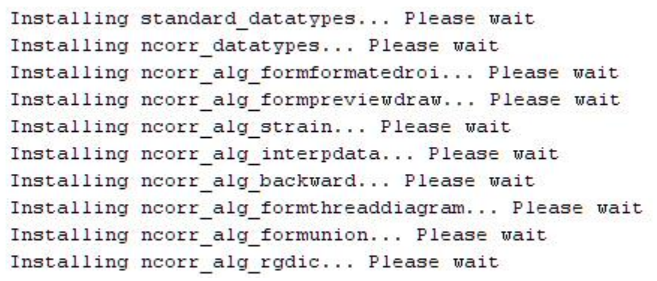
In this case, my computer has 8 logical cores that can be utilized for computation.

**NOTE:** If you use all your cores during computation, generally any other applications on the computer will slow down drastically (such as browsing the internet), so if you plan on using your computer during computation, then it might be best to leave out a core or two.

The manual compiler specification option is there for the case that GCC is installed on Windows through gnumex. This is probably won't be the case for most users, so when in doubt leave it unchecked.

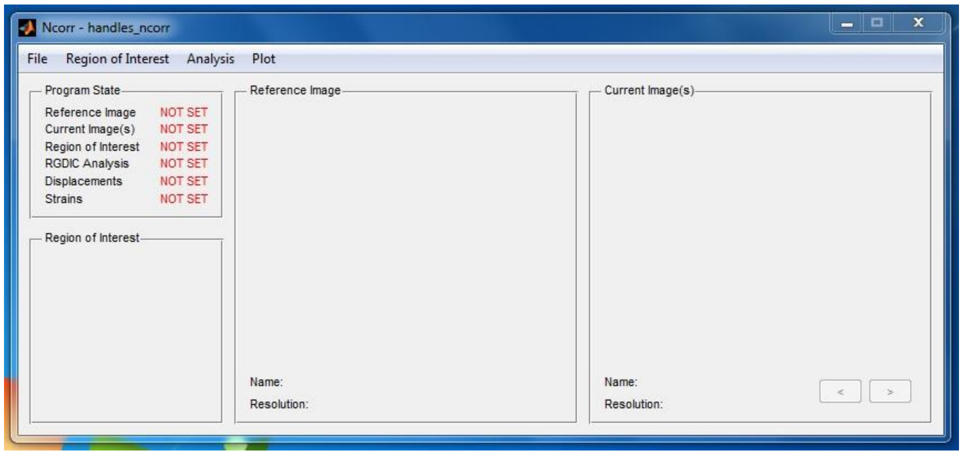
During installation, a pop-up for setting the path will appear. The path needs to be set or else Ncorr can fail if you change directories. This is only a temporary solution and the dialogue box will reappear if MATLAB is closed and reopened. For a more permanent solution, you can set the path manually through: *File > Set Path > Add Folder*.

If the installation proceeds as normal, the following should appear in the terminal:



```
Installing standard_datatypes... Please wait
Installing ncorr_datatypes... Please wait
Installing ncorr_alg_formformatedroi... Please wait
Installing ncorr_alg_formpreviewdraw... Please wait
Installing ncorr_alg_strain... Please wait
Installing ncorr_alg_interpdata... Please wait
Installing ncorr_alg_backward... Please wait
Installing ncorr_alg_formthreaddiagram... Please wait
Installing ncorr_alg_formunion... Please wait
Installing ncorr_alg_rgdic... Please wait
```

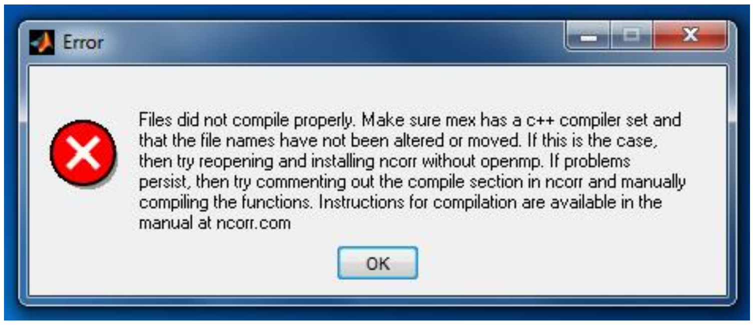
And then a GUI for Ncorr should appear:



| | |
|---|---|
| Reference Image | NOT SET |
| Current Image(s) | NOT SET |
| Region of Interest | NOT SET |
| RGDIC Analysis | NOT SET |
| Displacements | NOT SET |
| Strains | NOT SET |

If this is the case, Ncorr has most likely installed correctly and the installation should be complete. The next time you open MATLAB, you can open Ncorr again by typing "handles_ncorr = ncorr" into the MATLAB terminal and the GUI above should appear without having to repeat the installation process.

If compilation fails, then the following error message will appear:



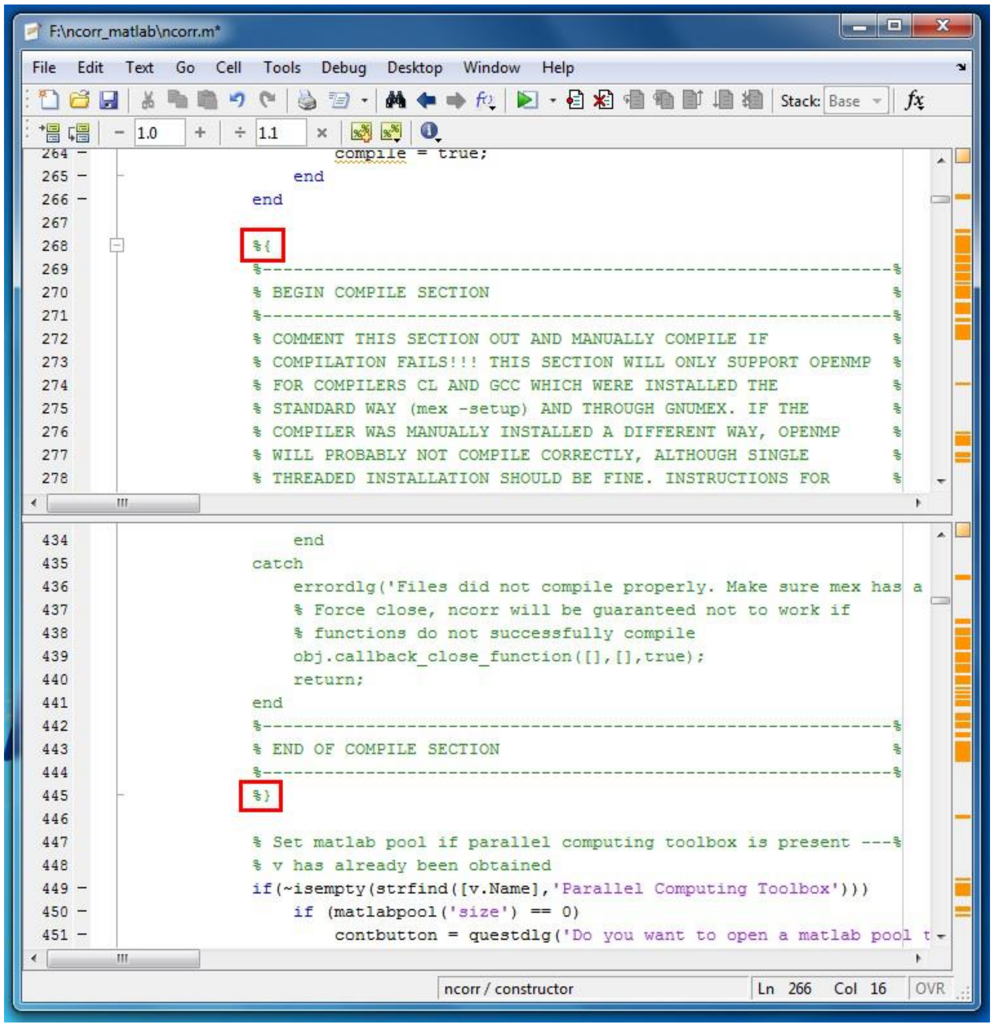
```
Files did not compile properly. Make sure mex has a c++ compiler set and
that the file names have not been altered or moved. If this is the case,
then try reopening and installing ncorr without openmp. If problems
persist, then try commenting out the compile section in ncorr and manually
compiling the functions. Instructions for compilation are available in the
manual at ncorr.com
```

If all the above steps have been followed correctly, then first try to install Ncorr without OpenMP. If compilation still fails, then you may need to complete the installation manually as described in the next section. If the installation proceeded correctly, then you can skip the rest of the installation section and move onto the user guide.

## 1.4 - Manual Installation

These steps show how to perform the manual installation of Ncorr, which involves the compilation of MEX files as well as the creation of an "ncorr_installinfo.txt" file which contains information about multicore support.

The first step is to comment out the automatic compile section in the **ncorr.m** file. Add "%{" before the compile section and then "%}" after it as shown below:



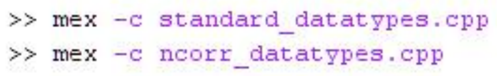
```
        compile = true;
    end
end

%{
%--------------------------------------------------------------%
% BEGIN COMPILE SECTION                                        %
%--------------------------------------------------------------%
% COMMENT THIS SECTION OUT AND MANUALLY COMPILE IF             %
% COMPILATION FAILS!!! THIS SECTION WILL ONLY SUPPORT OPENMP   %
% FOR COMPILERS CL AND GCC WHICH WERE INSTALLED THE            %
% STANDARD WAY (mex -setup) AND THROUGH GNUMEX. IF THE         %
% COMPILER WAS MANUALLY INSTALLED A DIFFERENT WAY, OPENMP      %
% WILL PROBABLY NOT COMPILE CORRECTLY, ALTHOUGH SINGLE         %
% THREADED INSTALLATION SHOULD BE FINE. INSTRUCTIONS FOR       %
```

```
    end
catch
    errordlg('Files did not compile properly. Make sure mex has a
    % Force close, ncorr will be guaranteed not to work if
    % functions do not successfully compile
    obj.callback_close_function([],[],true);
    return;
end
%--------------------------------------------------------------%
% END OF COMPILE SECTION                                       %
%--------------------------------------------------------------%
%}

% Set matlab pool if parallel computing toolbox is present ---%
% v has already been obtained
if(~isempty(strfind([v.Name],'Parallel Computing Toolbox')))
    if (matlabpool('size') == 0)
        contbutton = questdlg('Do you want to open a matlab pool t
```

The next step is to manually compile all the necessary libraries and MEX files. At the time of writing this manual, there are two basic library .cpp files, and eight MEX files which are listed in the table below:

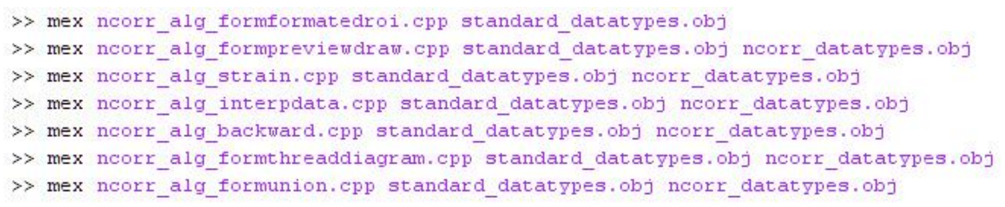
| | |
|---|---|
| Library Files | standard_datatypes.cpp<br>ncorr_datatypes.cpp |
| MEX Files | ncorr_alg_formformatedroi.cpp<br>ncorr_alg_formpreviewdraw.cpp<br>ncorr_alg_strain.cpp<br>ncorr_alg_interpdata.cpp<br>ncorr_alg_backward.cpp<br>ncorr_alg_formthreaddiagram.cpp<br>ncorr_alg_formunion.cpp<br>ncorr_alg_rgdic.cpp |

Start by compiling "standard_datatypes.cpp" and "ncorr_datatypes.cpp" as object files by using the "-c" flag with MEX as shown below:



```
>> mex -c standard_datatypes.cpp
>> mex -c ncorr_datatypes.cpp
```

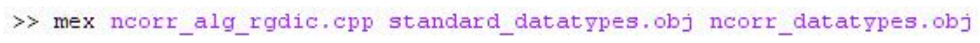
Next, install the following MEX files with:



```
>> mex ncorr_alg_formformatedroi.cpp standard_datatypes.obj
>> mex ncorr_alg_formpreviewdraw.cpp standard_datatypes.obj ncorr_datatypes.obj
>> mex ncorr_alg_strain.cpp standard_datatypes.obj ncorr_datatypes.obj
>> mex ncorr_alg_interpdata.cpp standard_datatypes.obj ncorr_datatypes.obj
>> mex ncorr_alg_backward.cpp standard_datatypes.obj ncorr_datatypes.obj
>> mex ncorr_alg_formthreaddiagram.cpp standard_datatypes.obj ncorr_datatypes.obj
>> mex ncorr_alg_formunion.cpp standard_datatypes.obj ncorr_datatypes.obj
```

**NOTE:** When installing the MEX files with linux, the standard_datatypes and ncorr_datatypes libraries will be compiled with an ".o" extension instead of an ".obj" extension. Also note that "ncorr_alg_formformatedroi.cpp" is the only file that doesn't use the "ncorr_datatypes.obj" library.

The last file that needs to be compiled is "ncorr_alg_rgdic.cpp." This file utilizes OpenMP and thus needs to be compiled with certain compiler flags. If multithreaded support is not desired, then the file can be compiled like the other MEX files in the following way:



```
>> mex ncorr_alg_rgdic.cpp standard_datatypes.obj ncorr_datatypes.obj
```

Otherwise, flags need to be passed to the compiler. The required flags are shown in the table below:

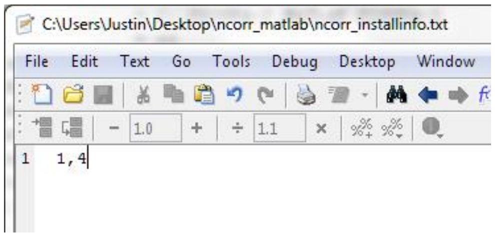
| OS/Compiler | Flags |
|---|---|
| Windows with Visual Studio | COMPFLAGS="$COMPFLAGS /openmp /DNCORR_OPENMP" |
| Windows with GCC (gnumex) | COMPFLAGS="$COMPFLAGS -fopenmp -DNCORR_OPENMP"<br>GM_ADD_LIBS="$GM_ADD_LIBS -lgomp" |
| Linux with GCC | CXXFLAGS="\$CXXFLAGS -fopenmp -DNCORR_OPENMP" CXXLIBS="\$CXXLIBS -lgomp" |

An example of compilation with the appropriate flags is shown below for a system with Windows and Visual studio:

```
>> mex ncorr_alg_rgdic.cpp standard_datatypes.obj ncorr_datatypes.obj COMPFLAGS="$COMPFLAGS /openmp /DNCORR_OPENMP"
```

The important thing to note here is to make sure you use the correct name and format for the compiler flags, which **depend on both the operating system and the compiler**. If compilation still fails or more assistance is needed then please email the author of Ncorr at jblaber3@gatech.edu.

At this point, all the MEX files should be compiled correctly. The next step is to create a file called "ncorr_installinfo.txt" and then fill it with the following:

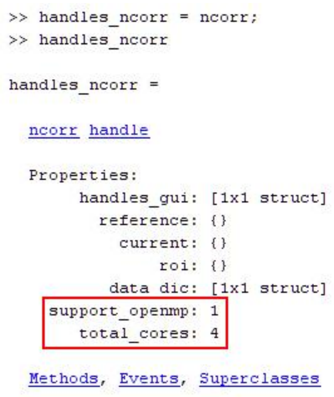

C:\Users\Justin\Desktop\ncorr_matlab\ncorr_installinfo.txt
File Edit Text Go Tools Debug Desktop Window

```
1,4
```

The first number represents whether or not OpenMP support exists, and should be either 0 or 1 (0 = no OpenMP support; 1 = OpenMP support). The second number represents how many threads you want the DIC analysis to run on (this should be a number greater than or equal to 1).

**NOTE:** Make sure to separate the numbers by a comma; the format should be: "#,#".

After the above is complete, type "handles_ncorr = ncorr" into the MATLAB terminal to bring the GUI up. If no error messages appear, then installation should be complete. As a double check, make sure the options you specified in the "ncorr_installinfo.txt" file were loaded correctly by typing "handles_ncorr" in the MATLAB prompt to view its properties:



```
>> handles_ncorr = ncorr;
>> handles_ncorr

handles_ncorr =

  ncorr handle

  Properties:
       handles_gui: [1x1 struct]
         reference: {}
           current: {}
               roi: {}
          data_dic: [1x1 struct]
    support_openmp: 1
       total_cores: 4

  Methods, Events, Superclasses
```

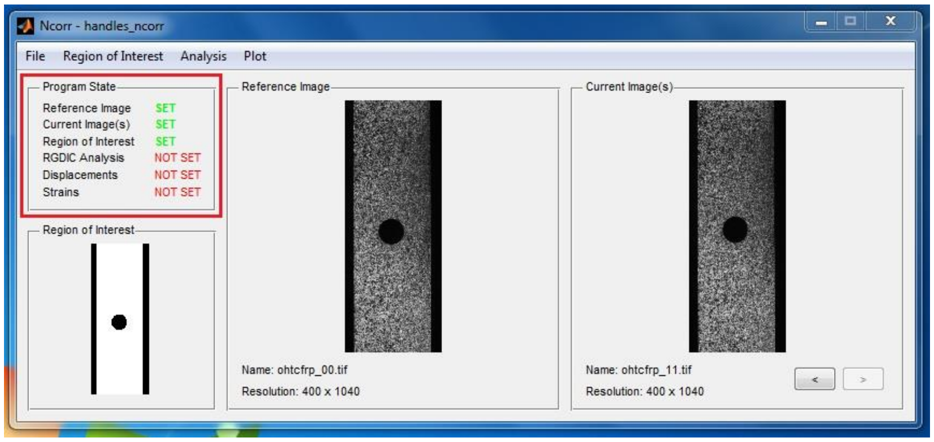
Lastly, I suggest following the user guide and trying all Ncorr options with example images to ensure everything is working properly. If not, then try recompiling the files.

## 2.1 - Program Flow

At this point, it is assumed Ncorr has been successfully compiled and installed on the user's computer. The work flow of Ncorr is as follows:

1. Set Reference Image
2. Set Current Image(s)
3. Set Region of Interest (ROI)
    - *Dependencies:* Requires reference image to be set first
4. Perform RG-DIC Analysis
    - *Dependencies:* Requires reference image, current image(s), and region of interest to be set first
5. Format Displacements
    - *Dependencies:* Requires RG-DIC Analysis to be run first
6. Calculate Strains
    - *Dependencies:* Requires displacements to be formatted first.

Any of these steps can be altered at any stage, so long as the dependencies have been met. The state of the program is visible at the top left corner of the main Ncorr GUI as shown below:



| Program State | |
|---|---|
| Reference Image | SET |
| Current Image(s) | SET |
| Region of Interest | SET |
| RGDIC Analysis | NOT SET |
| Displacements | NOT SET |
| Strains | NOT SET |

Name: ohtcfrp_00.tif
Resolution: 400 x 1040

Name: ohtcfrp_11.tif
Resolution: 400 x 1040

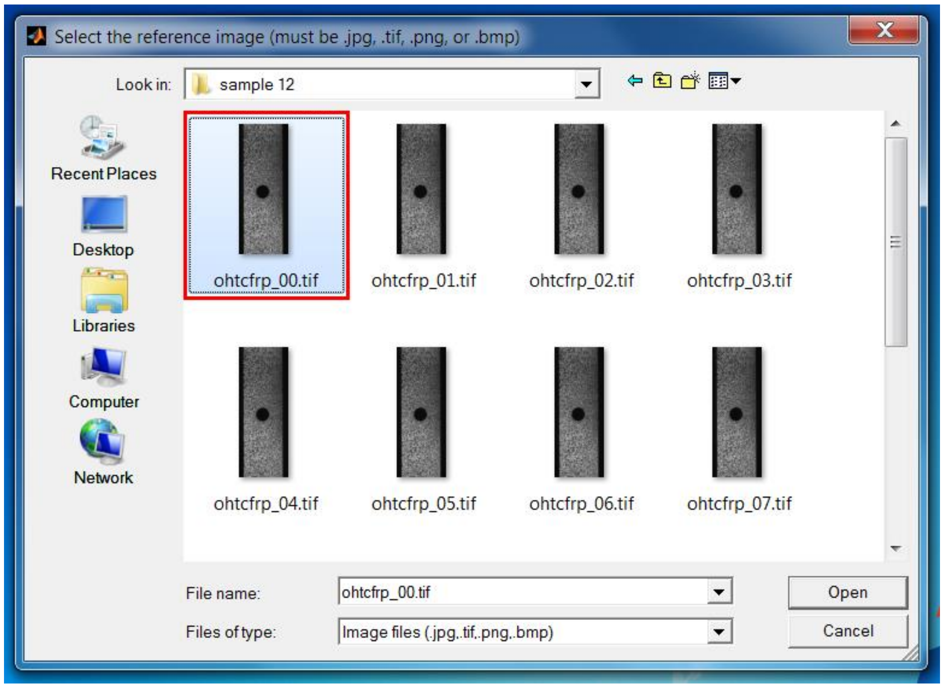
This lets the user know which stage they are currently on and which should come next. The state is listed in sequential order for convenience. In the example above, the state lets the user know that the reference image, current image, and ROI are loaded and that RG-DIC analysis needs to proceed next.

Additionally, throughout this manual, I have tried to make all the examples consistent throughout. If a reader is trying out Ncorr for the first time, he/she can download sample data through: http://www.sem.org/images/dic-challenge/Sample12.zip (or from the http://www.ncorr.com website), and follow along with the examples if he/she desires.

## 2.2 - Setting the Reference Image

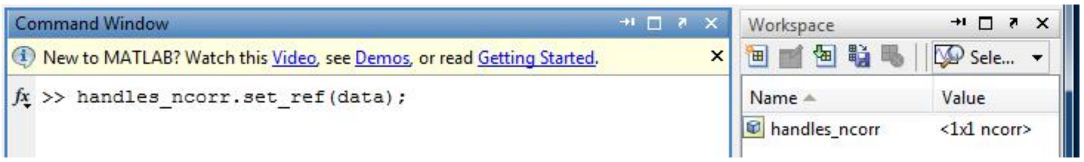
There are two main ways to set the reference image. The first way is to use the GUI by going to *File > Load Reference Image* on the main Ncorr GUI which results in the following menu:

**NOTE:** The reference image means the initial or first image you take of the sample, before any deformation occurs.



**NOTE:** The images that are loaded in this way need to have an extension of ".jpg", ".tif", ".png", or ".bmp," so if your images aren't appearing, it is most likely because they are an unsupported format.

The other way to load a reference image is to do it through the MATLAB terminal. You can set the reference image by typing "handles_ncorr.set_ref(data)," where "data" is a 2D matrix (of type double, uint16, or uint8) containing image data, as shown below:

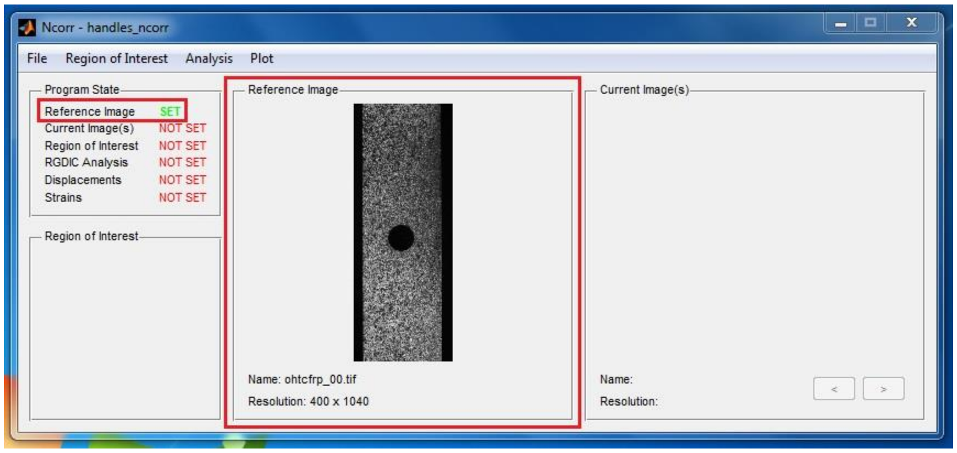

Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Workspace
Sele...
Name
Value
handles_ncorr
<1x1 ncorr>

```
>> handles_ncorr.set_ref(data);
```

**NOTE:** If you clear the workspace and delete "handles_ncorr" on accident, you can go to *File > Set Handle* to place another handle in the base workspace.

The reason for allowing uploading the reference image through the MATLAB terminal directly is because sometimes the reference image is obtained through a series of reference images which are averaged together. If this is the case, the reference image will be stored as a matrix in the base workspace, and it's much more convenient to load it directly than being required to save it as an image, which can also lead to loss of data through image compression or binning of gray scale values.

Lastly, when the reference image is loaded it should appear as shown below:



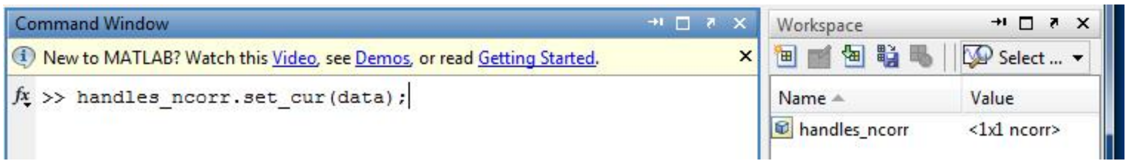
| | |
|---|---|
| Reference Image | SET |
| Current Image(s) | NOT SET |
| Region of Interest | NOT SET |
| RGDIC Analysis | NOT SET |
| Displacements | NOT SET |
| Strains | NOT SET |

Name: ohtcfrp_00.tif
Resolution: 400 x 1040

Name:
Resolution:

## 2.3 - Setting the Current Image(s)

If a single current image is to be uploaded, then setting it through the GUI and MATLAB terminal follow the same procedure as the reference image. The only difference is that you must go to *File > Load Current Image(s)* to upload it through the GUI or type "handles_ncorr.set_cur(data)" through the MATLAB terminal, as shown below:


Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Workspace
Select ...
Name
Value

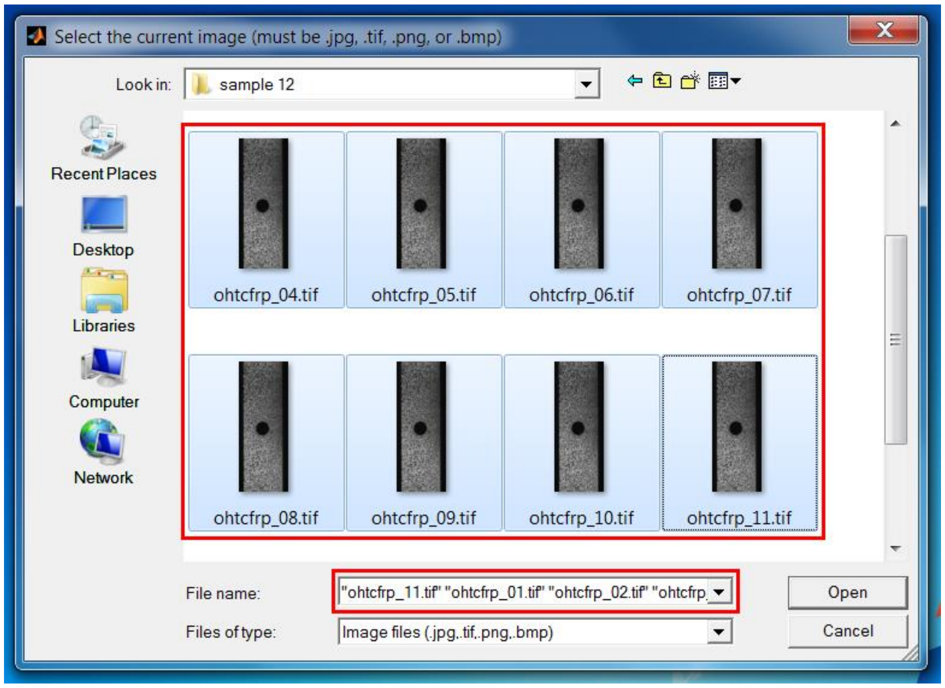
```
>> handles_ncorr.set_cur(data);
```

| Name | Value |
|---|---|
| handles_ncorr | <1x1 ncorr> |

If multiple current images are to be set, then they must be set through the GUI (for now) and must follow a specific naming convention as shown below:

name_#.ext

Where "name" is the name of the image, "#" is a number associated with the image, and "ext" is the image extension, which must be either ".jpg", ".tif", ".png," or “.bmp.” Furthermore, there must only be a **single underscore**. To select multiple images, go to *File > Load Current Image(s)* and simply click and drag your cursor over the images to select more than one current image as shown below:

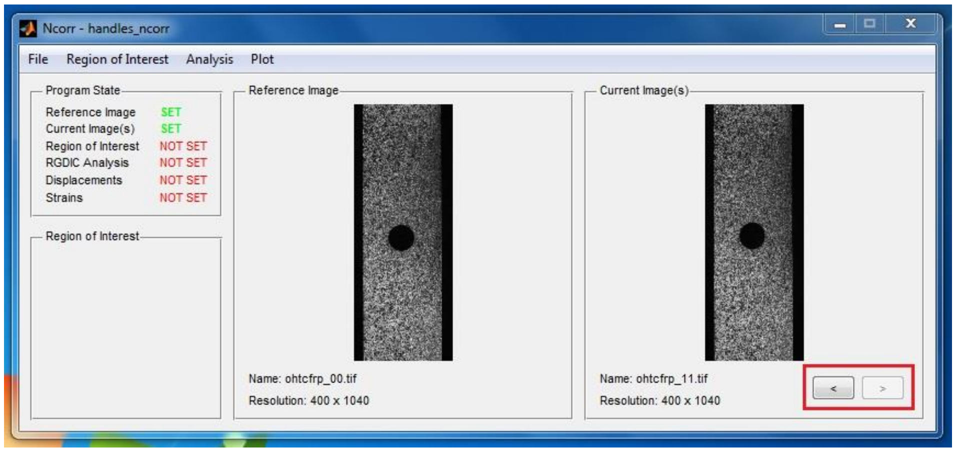


File name: "ohtcfrp_11.tif" "ohtcfrp_01.tif" "ohtcfrp_02.tif" "ohtcfrp

Files of type: Image files (.jpg,.tif,.png,.bmp)

**NOTE:** Take note of the naming convention above. All of the current images follow the "name_#.ext" format, where name = "ohtcfrp" and ext = "tif".

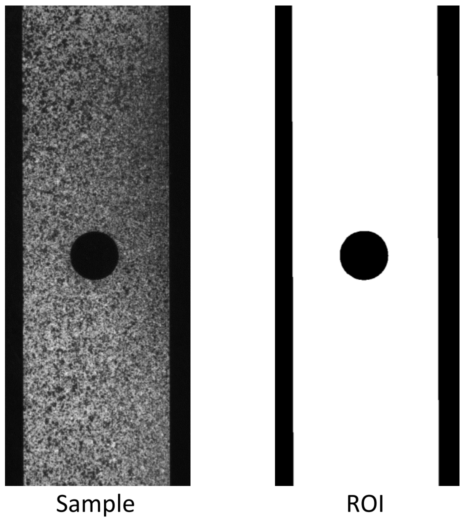
After the current images are uploaded, there are buttons on the bottom right of the Ncorr GUI which become enabled and allow you to scroll through the images as shown below:



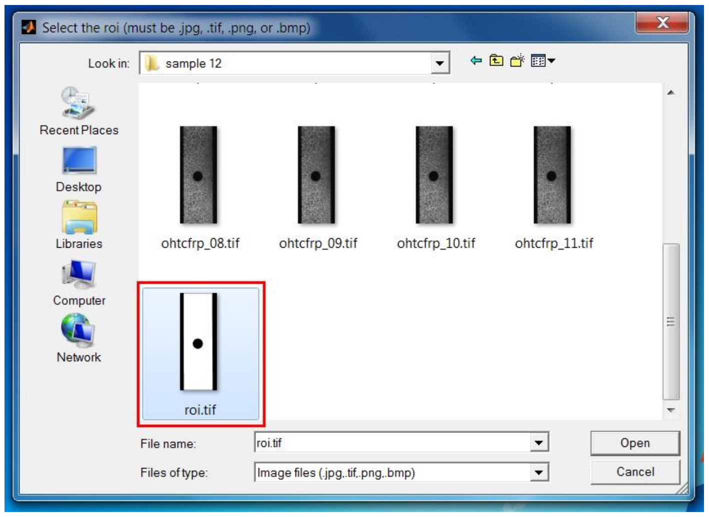
Lastly, the current images should be ordered properly according to their number, and the last image in the sequence will appear by default.

## 2.4 - Setting the Region of Interest

There are four ways to set the region of interest (ROI); three are through the GUI and one is through the MATLAB terminal. The most important thing to note here is that the ROI should be an array of the **same size as the reference image**. Furthermore, **white regions represent the ROI**, and **black regions are outside the ROI**. It is also recommended to **leave a border** (on the order of a couple pixels) between the edge of the sample and ROI. An example ROI is given below:



**Loading:** The first way to set the ROI is to load it from an image using *Region of Interest > Load ROI*. This is the recommended way to set a ROI because you can use a program like Photoshop to accurately trace an outline of the sample in order to obtain the best ROI. The only problem with this method is that it is more tedious, so other methods should be considered first if quick results are desired. An example is shown below:

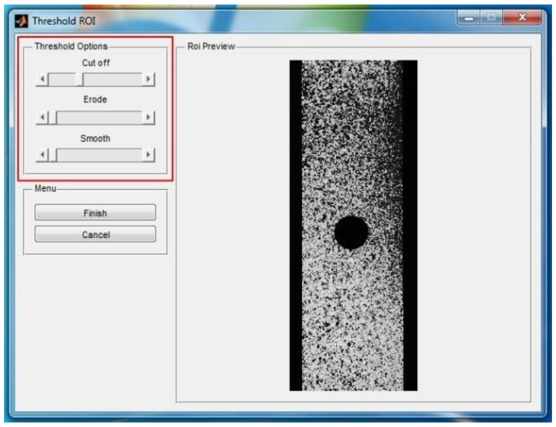

Select the roi (must be .jpg, .tif, .png, or .bmp)

Look in: sample 12

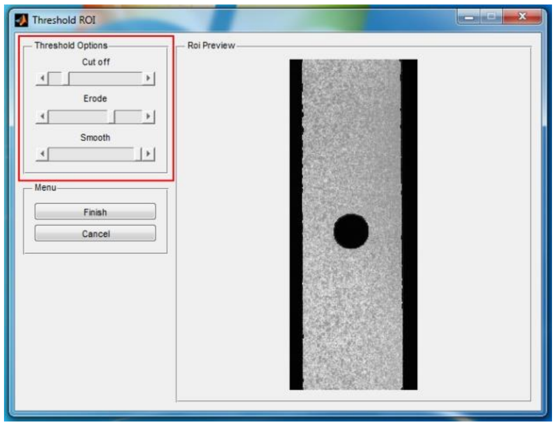
File name: roi.tif

Files of type: Image files (.jpg,.tif,.png,.bmp)

**Thresholding:** The second way to set the ROI is to threshold the reference image by going to *Region of Interest > Threshold*. This is a simple and fast way to set the ROI but can be inaccurate, so it is only recommended if there is good contrast between the sample and background and if quick results are desired. The best way to utilize this method is by setting the **cutoff** slider first so that most of the sample is visible but the background is black. Some black speckles may also be thresholded as well. To account for this, use the **smooth** slider to obtain the whole sample. After this, part of the boundary might also be included in the ROI which should be avoided. Use the **erode** slider last to create a space between the ROI and sample. An example of default vs the optimal slider positions are shown below:

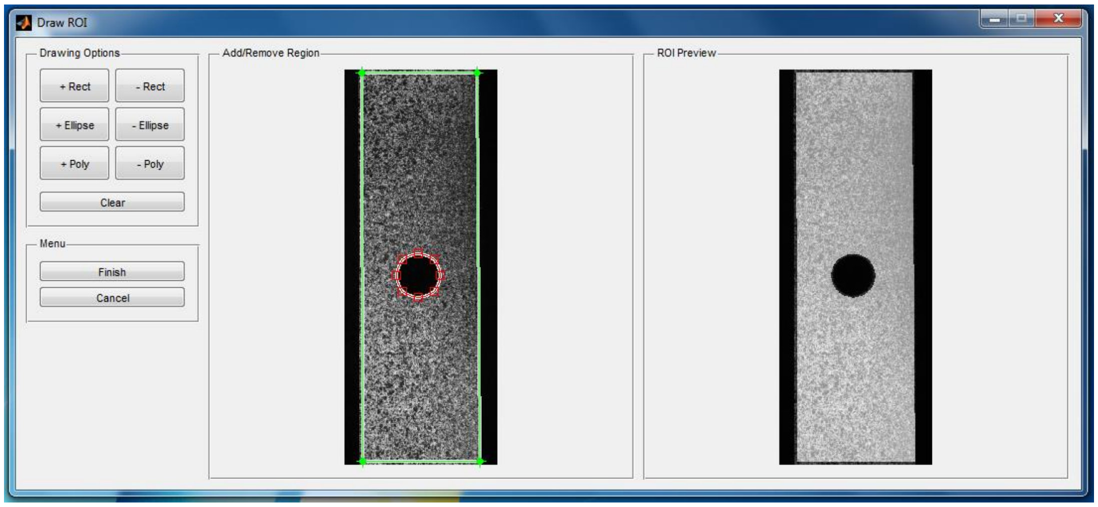


Default Parameters

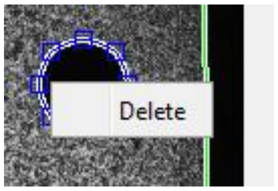


Optimal Parameters

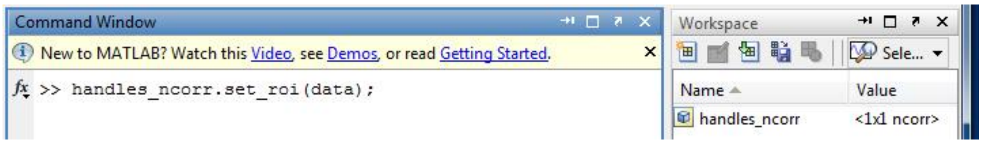
**Drawing:** The third way to set the ROI is to draw the ROI directly in MATLAB by going to *Region of Interest > Draw*. This is the preferred method for fast analysis because it can be done quickly and is more robust than the thresholding method. An example ROI is shown below and was constructed using "+ Poly" and "- Ellipse," where the "+" prefix indicates adding portions to the ROI; "-" subtracts regions:



**NOTE:** The green shapes represent regions which add to the ROI; the red shapes represent regions that subtract from the ROI. Furthermore, it is also possible to delete specific shapes by right clicking them and selecting "Delete," as shown below:



**MATLAB Terminal:** Once again, similar to loading the reference and current image through the MATLAB terminal, typing "handles_ncorr.set_roi(data)" will set the ROI using the 2D array "data" as shown below:


Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Workspace
Sele...
Name
Value

```
>> handles_ncorr.set_roi(data);
```

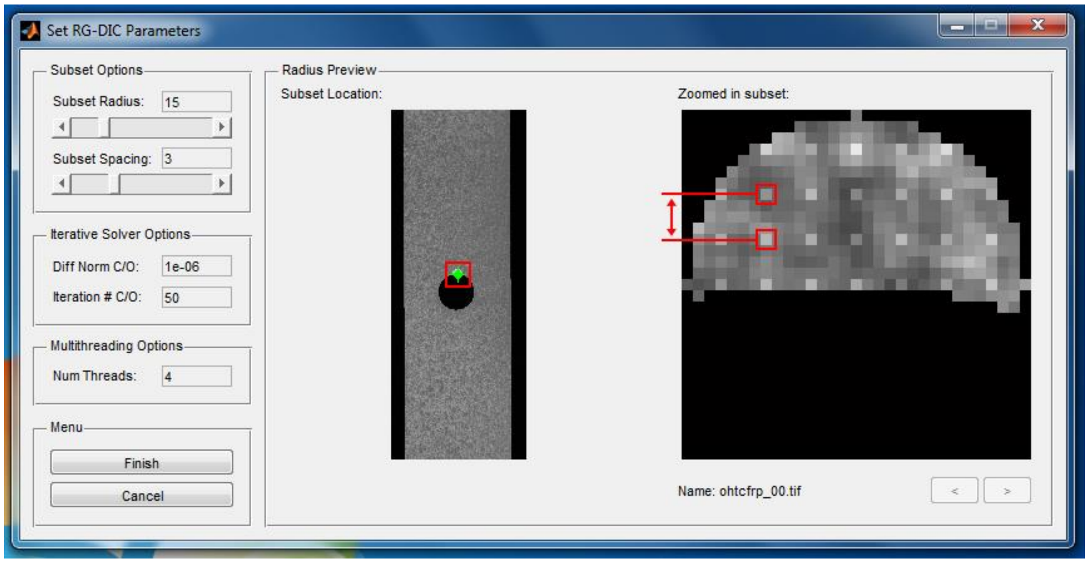
| Name | Value |
|---|---|
| handles_ncorr | <1x1 ncorr> |

## 2.5 - RG-DIC Analysis

The RG-DIC analysis begins by going to *Analysis > Perform RG-DIC Analysis > Forward Analysis*. The DIC analysis used in this program is based off of Bing Pan's RG-DIC framework, as well as his work on DIC for complex shapes. More specifically, Ncorr uses circular and contiguous subset. Overall, RG-DIC is highly robust and relatively computationally efficient as well. More details can be found in Bing Pan's paper on RG-DIC as well as the "DIC Overview" section on the ncorr.com website.

### RG-DIC Parameters

For now, this guide will only discuss the **forward** analysis which uses standard DIC techniques. The **backward** analysis is used for discontinuous displacement fields in the case where the discontinuities are visible in the current configuration. A guide for its use will be implemented in a future revision of the manual. Anyway, a GUI should appear as shown below:



Subset Radius: 15
Subset Spacing: 3
Diff Norm C/O: 1e-06
Iteration # C/O: 50
Num Threads: 4
Name: ohtcfrp_00.tif

**NOTE:** A subset spacing of 3 is shown above to better illustrate the spaced points on the zoomed in subset, but for the example analyzed in this manual, a subset spacing of 1 was used to obtain higher resolution results.

There are several key components to this GUI. The first is obviously the menu on the left, but it is also important to note that the subset preview is interactive. A green *impoint* (highlighted by a red square) is placed in the axes labeled "Subset Location." This point is draggable and is the center point of the subset shown on the right. The subset on the right gives an idea of what the subset spacing (space between the two dots within the red squares) and subsets will appear like. It's important to note that these highlighted points are where the subset locations will be, and not part of the speckle pattern in the uploaded image. An explanation of the menu components are given below:

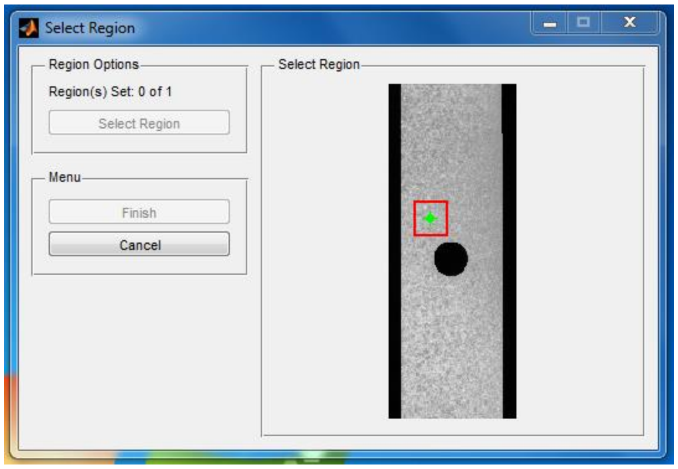
**Subset Options:** These options are the main components of DIC analysis. They dictate how large the subsets should be and the spacing between them. The spacing component is purely for reducing computational load. There are default guesses for both options but it's up to the user to select the most optimal settings. The most important option to get correct is the subset radius. There is a decent amount of literature available for the selection of the subset size as well the effects of subset size on DIC analysis, but most of the conclusions from these studies are often based on heuristics and empirical observations. Overall, the main idea is to select **the smallest subset possible which does not result in noisy displacement data** (as larger subsets tend to have a smoothing effect). Some iteration may be required to get this option right.

**Iterative Solver Options:** The iterative solver used in the DIC analysis is the inverse compositional image alignment technique (A good paper on this topic is Baker et al and Bing Pan's paper on his adaptation for this technique for DIC). The exit criteria for this iterative solver are the norm of the difference vector as well as the number of iterations. The default options for the iterative solver are actually pretty strict, but they can be relaxed if faster analysis is desired by reducing the iteration number cutoff as well as increasing the norm of the difference vector cutoff.

**Multithreading Options:** Ncorr has the ability to use multithreading to speed up the computation process. The default number of threads is the number of cores specified by the user during installation, although I have left the option to alter this number.

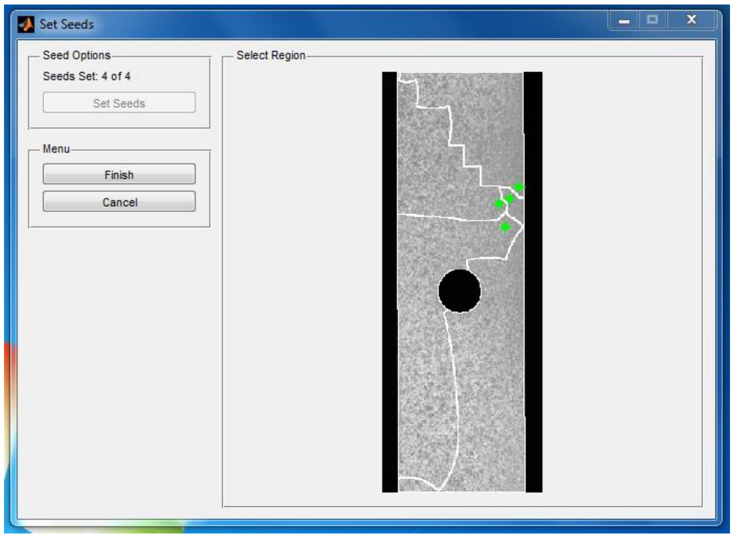
## Select Region

The next step in the analysis is to select the contiguous region to process. In this example, only one region will be processed (because only one is present), but it is possible to have more than one region when forming the ROI. Press the "Select Region" button and place the *impoint* anywhere within the region you want to select seeds for, as shown below:

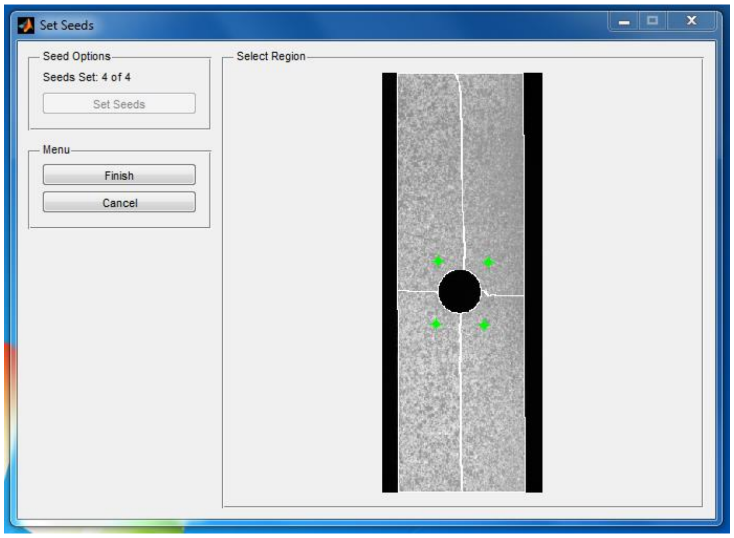


## Seed Placement

The seed selection process basically serves two purposes: it provides initial guesses for the RG-DIC analysis, and it also partitions the ROI so that each partition can be calculated in parallel. The placement of the seeds is somewhat important, because poor seed placement can lead to a failure in analysis. It's important for the region surrounding the seed point to be "full" because it maintains the robustness of the RG-DIC analysis. In addition, the seeds should be placed so that the regions are evenly sized so that the computation for each thread completes at about the same time.

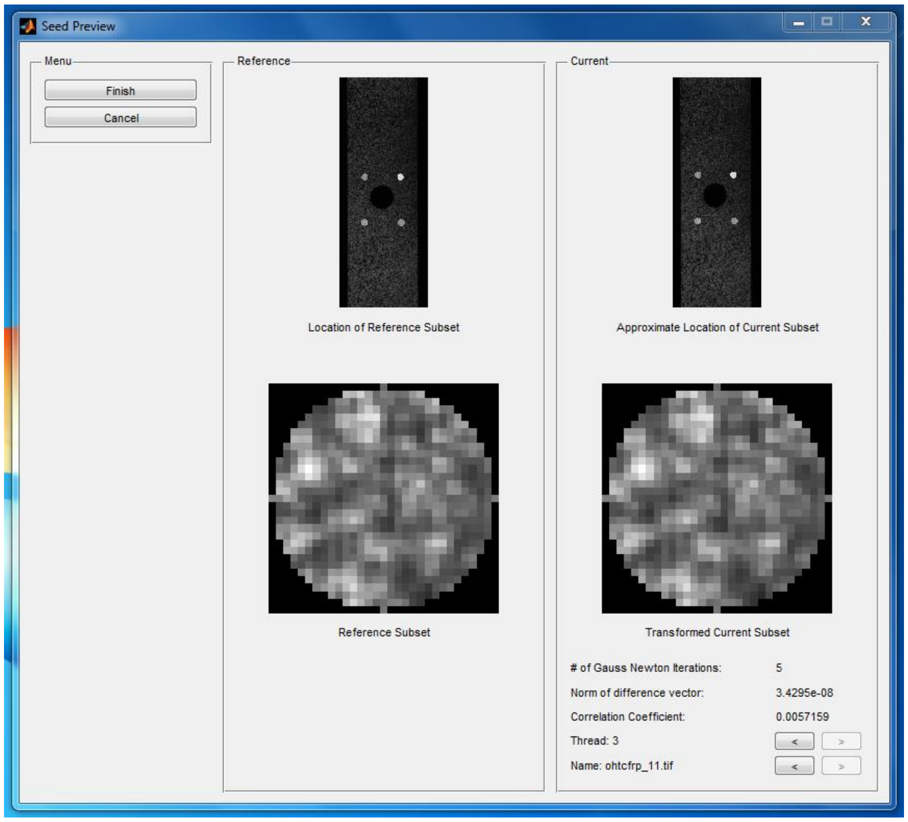


These seed positions are not preferable because the regions aren't evenly sized and they aren't nice "full" shapes.



These seed placements are more preferable because the regions are evenly sized and have nice "full" shapes.

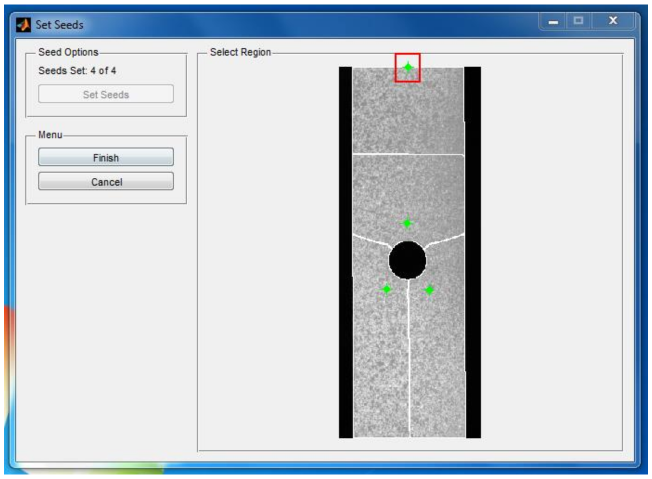
After the seed placement is done, a preview will appear. It's important that you check the seeds through this GUI because it's possible that the seeds can be processed incorrectly. The main reasons for seeds being incorrect can be either through a failure in convergence (this rarely happens as long as there are no large rotations present), or if the seeds travel outside the current image as the sample deforms. An example for proper seed placement is shown below:


Seed Preview
Menu
Finish
Cancel
Reference
Current
Location of Reference Subset
Approximate Location of Current Subset
Reference Subset
Transformed Current Subset

```
# of Gauss Newton Iterations:     5
Norm of difference vector:        3.4295e-08
Correlation Coefficient:          0.0057159
Thread: 3
Name: ohtcfrp_11.tif
```

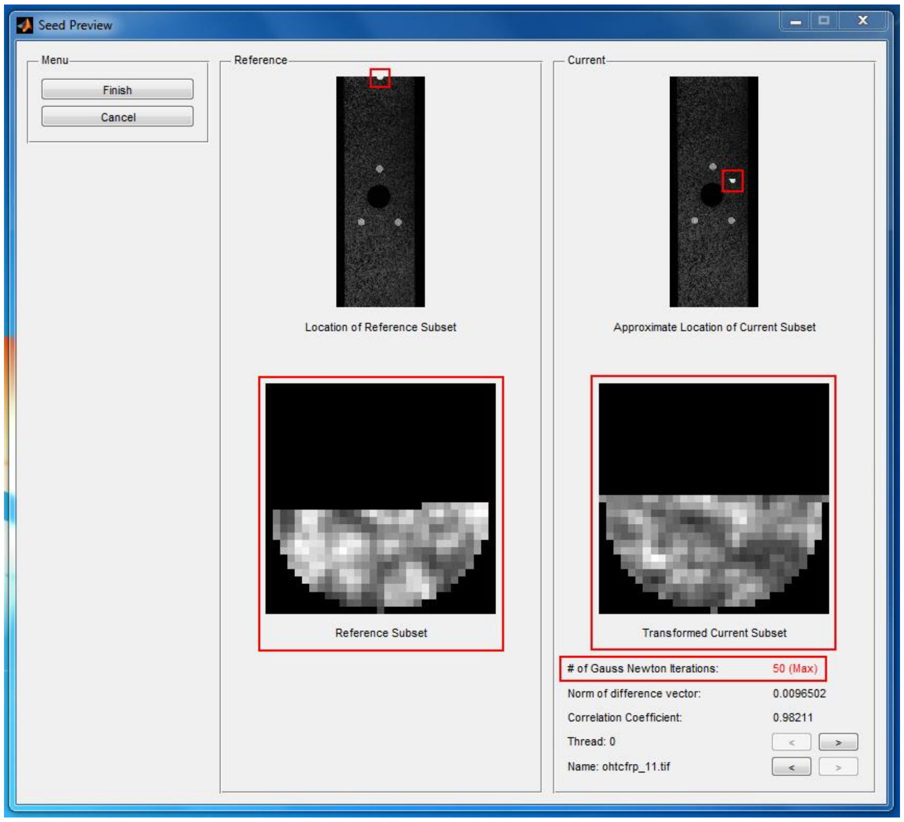
Notice that all the seed locations in the reference subset seem to match appropriately with the locations in the current subset. Furthermore, the reference subset and transformed current subsets look very similar. Lastly, the number of iterations to convergence was way below the cutoff, and also has a relatively low correlation coefficient which implies a good and correct seed placement.

As mentioned before, it's possible for the seed to travel outside the image during deformation and image acquisition. Because this is actually a somewhat common occurrence (especially when the whole sample cannot be imaged within the field of view), an example is given below with tips on how to catch this mistake and correct it.



Seeds Set: 4 of 4

Suppose for instance that the seeds are placed as shown above. It turns out that the sample in the picture undergoes some rigid body translation upwards. The seed highlighted in red actually goes "out of the picture" in some of the current images. In this case, an error message will generally display that notifies the user of a seed point with a high correlation coefficient value. This is an important indicator of a seed that has possibly been processed incorrectly. You can confirm this by viewing the seed preview below:


Seed Preview
Menu
Finish
Cancel
Reference
Current
Location of Reference Subset
Approximate Location of Current Subset
Reference Subset
Transformed Current Subset

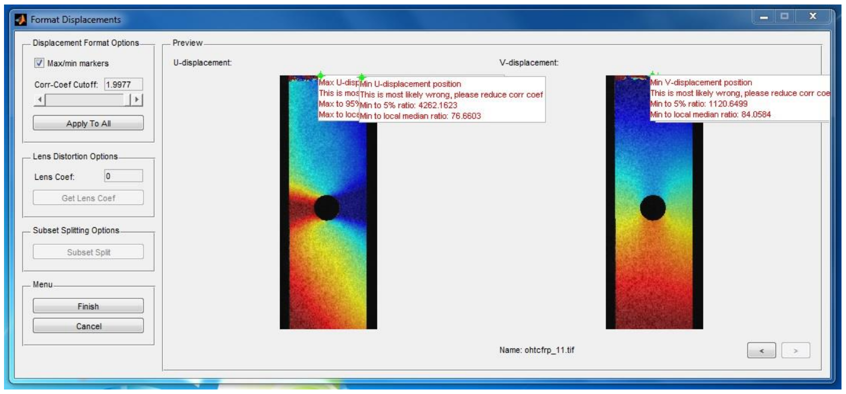
# of Gauss Newton Iterations: 50 (Max)
Norm of difference vector: 0.0096502
Correlation Coefficient: 0.98211
Thread: 0
Name: ohtcfrp_11.tif

All of the highlighted things in the figure above indicate potential problems. In the location pictures, it becomes evident that the location of the top seed in the current image is completely off, whereas the other three seeds are correct. Visual inspection of the enlarged reference and current subsets also show that they look very different. Lastly, the maximum number of iterations (50 for this case) was used before convergence was reached; this is typically another good indicator of an improperly calculated seed. The solution here is to move the seed to a lower location or potentially redraw the ROI to exclude the top region.

Anyway, after the seeds have been properly placed and calculated, the RG-DIC analysis will run until completion. The next step in the process is to format the displacements calculated during the RG-DIC analysis.

## 2.6 - Formatting Displacements

The displacement formatting begins by going to *Analysis > Format Displacements*. In maintaining the consistency of the examples given above, the resulting GUI for formatting the displacements should look something like this:



Max/min markers: ☑
Corr-Coef Cutoff: 1.9977
Lens Coef: 0

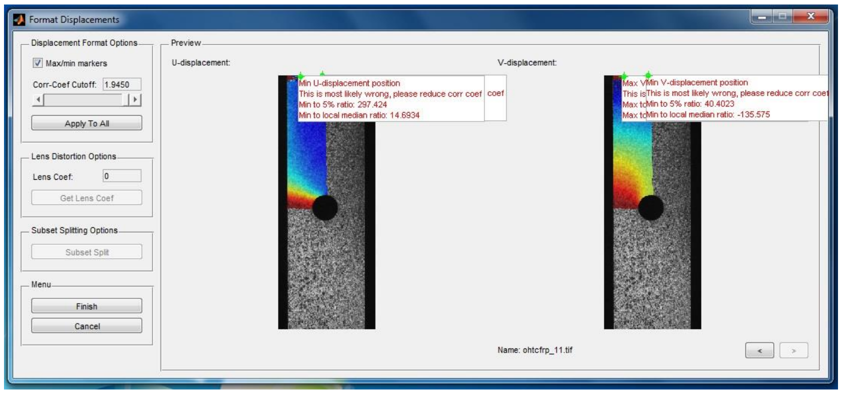
Max U-disp
This is mos
Max to 95%
Max to loca

Min U-displacement position
This is most likely wrong, please reduce corr coef
Min to 5% ratio: 4262.1623
Min to local median ratio: 76.6603

Min V-displacement position
This is most likely wrong, please reduce corr coe
Min to 5% ratio: 1120.6499
Min to local median ratio: 84.0584

Name: ohtcfrp_11.tif

This is a good check to make sure OpenMP is working correctly. If OpenMP installed incorrectly and did not result in a compiler error, the analysis may still run but will result in results that look like this:



Max/min markers: ☑
Corr-Coef Cutoff: 1.9450
Lens Coef: 0

Min U-displacement position
This is most likely wrong, please reduce corr coef
Min to 5% ratio: 297.424
Min to local median ratio: 14.6934

coef

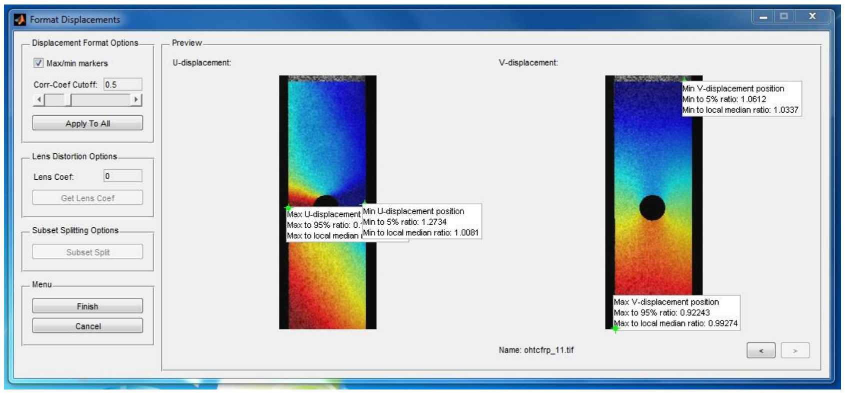
Max V
This is
Max to
Max to

Min V-displacement position
This is most likely wrong, please reduce corr coef
Min to 5% ratio: 40.4023
Min to local median ratio: -135.575

Name: ohtcfrp_11.tif

These results are obtained because only one thread (and thus one seed) was processed. If this happens, go to *File > Reinstall* to recompile the MEX files. If problems persist, try to either install Ncorr manually, or simply use single threaded performance.

**Displacement Format Options:** The next thing to notice on the first figure is the max/min markers. These are *impoints* that display the locations of the maximum and minimum displacements. They also display the max/95%, max/local median, min/5%, and min/local median ratios. The first ratio represents the maximum value to the displacement value that's at the 95th percentile. This value should realistically be somewhat close to ~1 (as displacements are typically smooth). Furthermore, the value of the maximum point to the local median should also be reasonably close to ~1 as well (once again because displacements are typically smooth). Thus, all four values should be in the neighborhood of 1. The numbers displayed on the first image are orders of magnitude off, which typically mean there are incorrectly processed data points that we need to get rid of. These data points generally have a high correlation coefficient as well, which we can use as a way to filter out bad data points. In general, bad data points usually result in very high or very low displacement values, which is why max/min markers are used as a way to way to error check. Anyway, a good way to process these points is to select an appropriate correlation cutoff for the last image (which displays first by default) and then apply this value to all images, although each one can be set individually if the user wishes to. The value selected should be **the smallest value possible which does not result in "good" data being deleted**. An example of an appropriate correlation cutoff is shown below:



[✓] Max/min markers
Corr-Coef Cutoff: 0.5
Lens Coef: 0

Max U-displacement
Max to 95% ratio: 0.
Max to local median

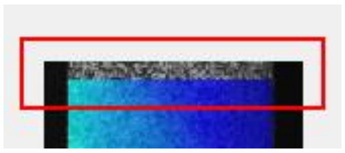
Min U-displacement position
Min to 5% ratio: 1.2734
Min to local median ratio: 1.0081

Min V-displacement position
Min to 5% ratio: 1.0612
Min to local median ratio: 1.0337

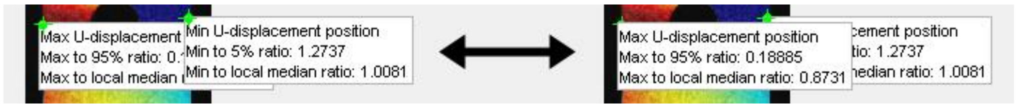
Max V-displacement position
Max to 95% ratio: 0.92243
Max to local median ratio: 0.99274

Name: ohtcfrp_11.tif

Looking at the max/min markers, the aforementioned ratios are all somewhat near ~1. Lastly, by inspecting the top portion of the image as shown below:



The correlation cutoff actually removed points from the top portion of the ROI. These are actually data points which moved outside the image during deformation. Another way to get rid of these points would be to redraw the ROI and then rerun the analysis, but since this boundary isn't known beforehand, using the correlation cutoff is a little more robust and generally yields good results. Lastly, if any of the information from the max/min markers overlaps, they can be clicked on to bring the info to the top.



Max U-displacement
Max to 95% ratio: 0.
Max to local median

Min U-displacement position
Min to 5% ratio: 1.2737
Min to local median ratio: 1.0081

Max U-displacement position
Max to 95% ratio: 0.18885
Max to local median ratio: 0.8731

cement position
tio: 1.2737
nedian ratio: 1.0081

**Lens Distortion Coefficient:** This is an option to correct for errors involving radial lens distortion. The distortion correction follows from eq.6 in “Systematic errors in two-dimensional digital image correlation due to lens distortion,” except the coefficient’s sign is flipped to make it positive in general. The correction is applied assuming the distortion center is at the center of the image; this might not always be true but is a reasonable assumption. If no calibration tests have been done or you’re unsure what the lens coefficient is, then just leave it as 0.

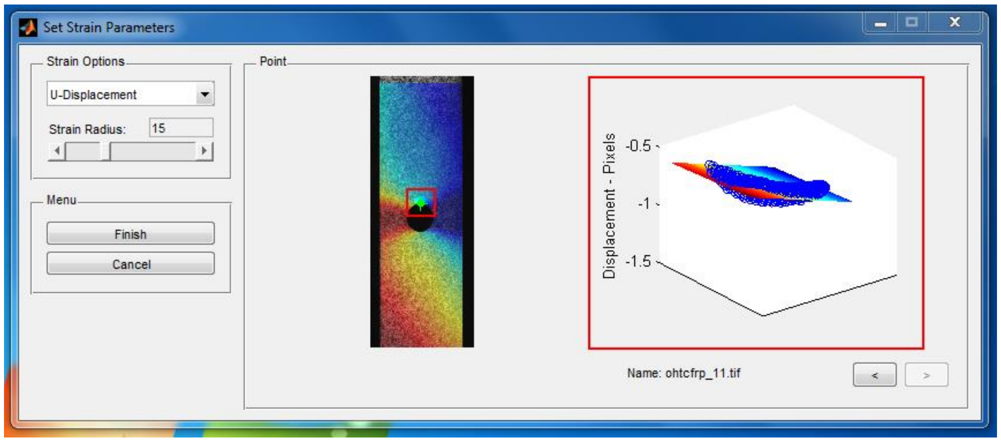
As of now, a lens coefficient can only be applied. Bing Pan provides a method for determining the lens coefficient, but I have yet to integrate this into Ncorr. The process basically requires DIC analysis to be run on images taken during rigid body translation tests and then performing a simple least squares analysis to solve for the lens distortion parameter. But, overall, it should be noted that the error associated with lens distortion is generally small.

**Subset Splitting Options:** This is an option I've left on the GUI to potentially perform a variant of subset splitting in the future. In my opinion, the algorithm used in the subset splitting procedure by Poissant et al doesn't appear to be robust enough to warrant implementing right now. In the future I may implement it though.

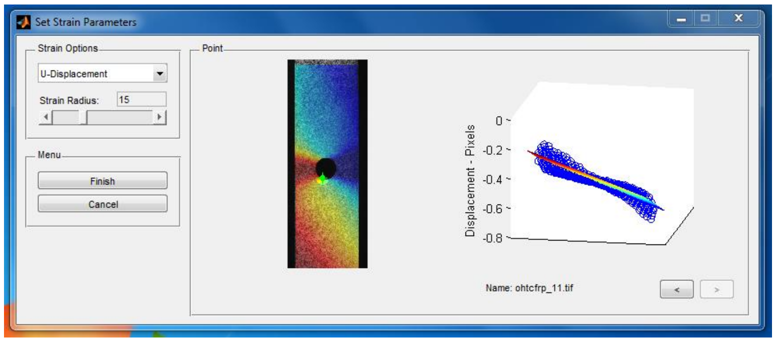
## 2.7 - Strain Analysis

The strain analysis begins by going to *Analysis* > *Calculate Strains*. The strains are calculated from the displacement data by using a least squares plane fit to a local group of data points (in this case a contiguous circle) which is based on Bing Pan's work on strain calculation. The displacements gradients are then found from the plane parameters; these gradients are used to calculate **Green-Lagrangian** strains. A more detailed description is given in the "DIC Overview" section on the ncorr.com website.

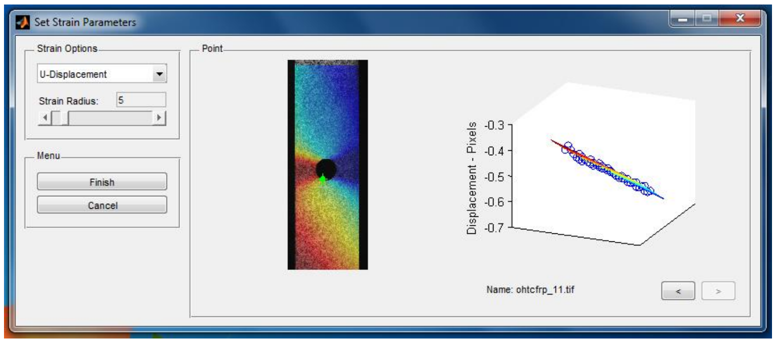
**Strain Options:** The only parameter you can vary here is the strain radius. This is the radius of a circle which selects a group of points to fit a plane to. A preview is provided so the user can visualize the plane fitting. The selection of the ideal strain radius is similar to the selection of the ideal subset radius, in that **the smallest radius is desired which does not result in noisy strain data**. The default radius is set to 15, but it is up to the user to select the most optimal radius for their data. An example is given below:



The *impoint* highlighted in the red box is draggable, and updates the plane fit shown on the right side. The recommended way to use this analysis is to drag the point to areas of high displacement gradients and view if the curve fit is still reasonable. For this example, I've placed the point near the max displacement values and reduced the strain radius until a good plane fit was obtained as shown below:



Visually, it appears the strain radius is too large for this displacement field.

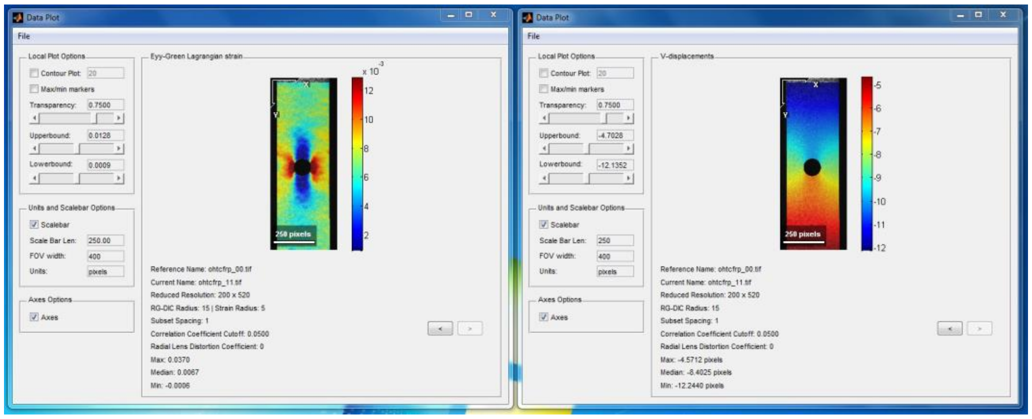


Reducing the strain radius to a value of 5 appears to give a much better fit for the plane.

In future revisions, I may add other metrics (such as $R^2$ values) to assist in the strain radius size selection. Lastly, the strain radius should be selected with regards to the last current image (which is shown first by default), as this image will most likely have the highest displacement gradients.

## 2.8 - Plotting

Plotting can be achieved by going to *Plot > View Displacement Plots* or *Plot > View Strain Plots*. This is the last step of the analysis. An example of the plots used in the examples in this user guide is shown below:



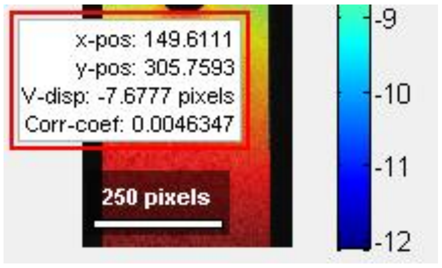
☐ Contour Plot: 20
☐ Max/min markers
Transparency: 0.7500
Upperbound: 0.0128
Lowerbound: 0.0009
☑ Scalebar
Scale Bar Len: 250.00
FOV width: 400
Units: pixels
☑ Axes

Reference Name: ohtcfrp_00.tif
Current Name: ohtcfrp_11.tif
Reduced Resolution: 200 x 520
RG-DIC Radius: 15 | Strain Radius: 5
Subset Spacing: 1
Correlation Coefficient Cutoff: 0.0500
Radial Lens Distortion Coefficient: 0
Max: 0.0370
Median: 0.0067
Min: -0.0006

☐ Contour Plot: 20
☐ Max/min markers
Transparency: 0.7500
Upperbound: -4.7028
Lowerbound: -12.1352
☑ Scalebar
Scale Bar Len: 250
FOV width: 400
Units: pixels
☑ Axes

Reference Name: ohtcfrp_00.tif
Current Name: ohtcfrp_11.tif
Reduced Resolution: 200 x 520
RG-DIC Radius: 15
Subset Spacing: 1
Correlation Coefficient Cutoff: 0.0500
Radial Lens Distortion Coefficient: 0
Max: -4.5712 pixels
Median: -8.4025 pixels
Min: -12.2440 pixels

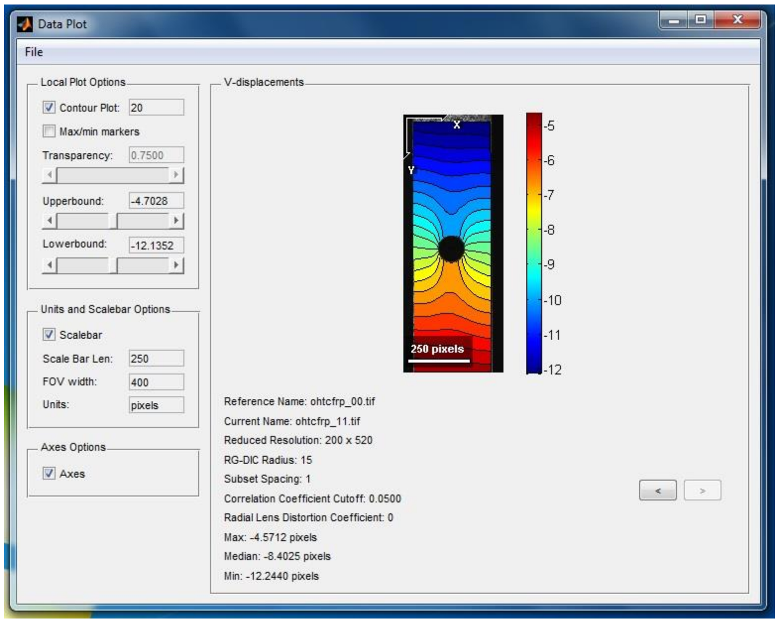
There are a couple basic features available, the first one being that **the plot windows are resizable**. They are the only GUI windows in Ncorr that are resizable at the time of writing this manual. Furthermore, if you hover your cursor over the data, a data cursor appears that informs you of the displacement or strain data at that location as well as the correlation coefficient value, as shown below:



x-pos: 149.6111
y-pos: 305.7593
V-disp: -7.6777 pixels
Corr-coef: 0.0046347

**NOTE:** From my experiences working with the plotting tool on different operating systems and versions of MATLAB, there appear to be inconsistencies between versions with regards to the stacking order of objects (i.e. sometimes the scalebar/axes appear behind the data if a contour plot is present), but I tried to optimize it for Windows on version R2012a. If the axes, scalebar, and contour plot are disabled, the plot appears to work fine on all versions of MATLAB I've tested Ncorr on.

**Local Plot Options:** These options are pretty self-explanatory. The only thing to note here is that these options only affect the plot that is being adjusted (as opposed to other options which affect all plots which are described later). Furthermore, the transparency option is disabled when the contour plot is enabled. This is because transparency does not work well with the contour plot. An example of the contour plot is shown below:



Local Plot Options
Contour Plot: 20 [✓]
Max/min markers
Transparency: 0.7500
Upperbound: -4.7028
Lowerbound: -12.1352

Units and Scalebar Options
Scalebar [✓]
Scale Bar Len: 250
FOV width: 400
Units: pixels

Axes Options
Axes [✓]

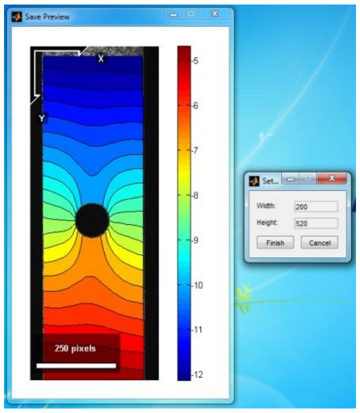
Reference Name: ohtcfrp_00.tif
Current Name: ohtcfrp_11.tif
Reduced Resolution: 200 x 520
RG-DIC Radius: 15
Subset Spacing: 1
Correlation Coefficient Cutoff: 0.0500
Radial Lens Distortion Coefficient: 0
Max: -4.5712 pixels
Median: -8.4025 pixels
Min: -12.2440 pixels

Usually, the contour plot does not work that well with strain plots because strain plots are generally noisier than displacement plots (as derivatives are highly susceptible to noise) which results in jagged contours.

**Units and Scalebar Options:** Once again, these options are pretty self-explanatory. But, one thing to note is that the scalebar options will be consistent among any open plots. Thus, if the U and V displacement plots are both open, then the scalebar will be either present or not present in both of them and any changes in scalebar length in one plot will update the other plot automatically.

**Axes Options:** Like the "units and scalebar options," these options are consistent among any open plot.

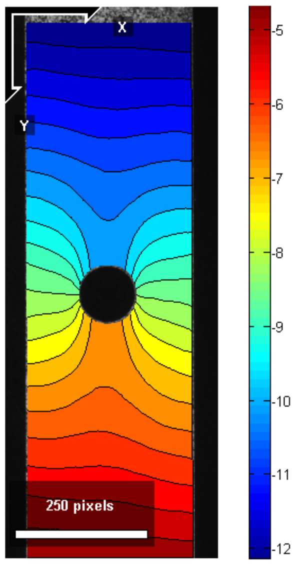
**Saving the Figure:** The figure can be saved by going to *File > Save Image > Save Image Without Info* or *File > Save Image > Save Image With Info* **within the plot window**. This will allow you to save figures for publications. The menu which appears should look like this:



Width: 200
Height: 520

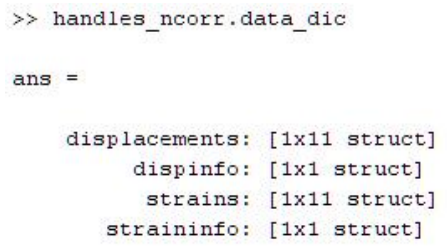
This will allow you to resize the image if you desire. Please note that if a large spacing parameter is used, the image may look blurry if you upsize it because the image and data plot must be the same size when plotting in MATLAB. The best solution for this would be to rerun the analysis with a smaller subset spacing parameter. Also, I recommend **saving the figure as a .tif** because MATLAB appears to compress other image formats which may lead to a loss of quality.

Finally, the final save figure should appear as:



## 3.1 - Directly Obtaining and Altering the DIC Data

The DIC data can be accessed directly through the "handles_ncorr" handle. For example, by typing "handles_ncorr.data_dic" into the MATLAB terminal, you can access the data structure that contains the strain and displacement data as shown below:



```
>> handles_ncorr.data_dic

ans =

    displacements: [1x11 struct]
         dispinfo: [1x1 struct]
          strains: [1x11 struct]
       straininfo: [1x1 struct]
```

Notice that both "dispinfo" and "straininfo" only have one element. This is because they contain parameters which are consistent over the entire analysis. On the other hand, both "displacements" and "strains" have eleven elements. This is because these fields contain information specific to each current image. For the example used throughout this guide, we have uploaded eleven current images, which correspond to eleven elements seen above.

The size of the displacement/strain plots themselves depend on the reference image and the subset spacing utilized. For the general case, the size of the displacement/strain plot will be:

size(reference(1:1+subsetspacing:end,1:1+subsetspacing:end))

where "reference" is the 2D reference image and "subsetspacing" is the subset spacing parameter (where a value of 0 results in no reduction). For example, if a reference image of size 400x1040 and a subset spacing of 1 is used (as is done in the example in this manual), then the displacement/strain plot size will be 200x520.

The displacements with respect to the reference configuration are stored in the plots with the "_ref" suffix. The "_cur" suffix is used in the case that backward DIC analysis is performed, which will be discussed in a later revision of this manual.

For easy reference, each field of "handles_ncorr.data_dic " will be explained in a table format, starting with "handles_ncorr.data_dic.displacements" as shown below:

| **handles_ncorr.data_dic.displacements** | |
|---|---|
| Field | Explanation |
| plot_udisp_ref | Plot containing U displacement data with respect to the reference image |
| plot_vdisp_ref | Plot containing V displacement data with respect to the reference image |
| plot_currcoef_ref | Plot containing correlation coefficient data with respect to the reference image |
| plot_udisp_ref_formated | Plot containing U displacement data with respect to the reference image after the correlation cutoff and lens distortion coefficient have been applied. |
| plot_vdisp_ref_formated | Plot containing V displacement data with respect to the reference image after the |

| | |
|---|---|
| | correlation cutoff and lens distortion coefficient have been applied. |
| plot_udisp_cur | Plot containing U displacement data with respect to the current image |
| plot_vdisp_cur | Plot containing V displacement data with respect to the current image |
| plot_corrcoef_cur | Plot containing correlation coefficient data with respect to the current image |
| plot_udisp_cur_formated | Plot containing U displacement data with respect to the current image after the correlation cutoff and lens distortion coefficient have been applied. |
| plot_vdisp_cur_formated | Plot containing V displacement data with respect to the current image after the correlation cutoff and lens distortion coefficient have been applied. |
| roi_cur | Object containing the ROI information set directly by the user in each current image if doing backward analysis |
| roi_valid | Object containing the ROI information for valid displacement points after the RG-DIC analysis |
| roi_formated | Object containing the ROI information for formatted displacement points (it is a subset of roi_valid due to the correlation cutoff parameter) |
| cutoff_corrcoef | Number representing the correlation coefficient cutoff value; each plot has one although this number is generally the same for each plot |

**handles_ncorr.data_dic.dispinfo**

| Field | Explanation |
|---|---|
| type | String representing whether forward or backward analysis was done; this value can either be 'forward' or 'backward' |
| radius | Integer representing the subset radius |
| spacing | Integer representing the subset spacing |
| cutoff_diffnorm | Number representing the cutoff for the norm of the difference vector in the inverse compositional method |
| cutoff_iteration | Integer representing the cutoff for the maximum number of iterations allowed |
| total_thread | Integer representing the total number of threads specified during the RG-DIC parameter specification stage (can be different than the number of cores specified during installation) |
| lenscoef | Number representing the radial lens distortion coefficient |

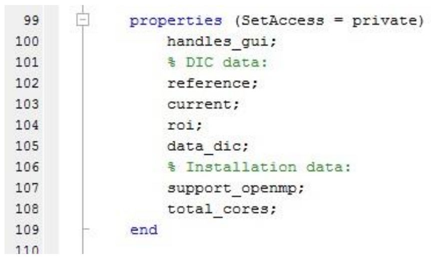
**handles_ncorr.data_dic.strains**

| Field | Explanation |
|---|---|
| plot_exxstrain_ref | Plot containing the Exx Green-Lagrangian strains (with respect to the reference image) |
| plot_exystrain_ref | Plot containing the Exy Green-Lagrangian strains (with respect to the reference image) |
| plot_eyystrain_ref | Plot containing the Eyy Green-Lagrangian strains (with respect to the reference image) |
| roi_formated | Object containing the ROI information for valid points after strain calculation (It's possible for some points to be removed during strain calculation for regions that do not have enough points to form a plane fit) |

**handles_ncorr.data_dic.straininfo**

| Field | Explanation |
|---|---|
| radius | Integer representing the strain radius |

Lastly, any information that the user wants to alter should be copied first to another variable in the main workspace. By default, the properties of Ncorr are set to "(SetAccess = private)" to prevent inadvertent alteration of data. If the user wants to alter the data somehow and still wants to use the Ncorr plotting tools, then the user can go into the **ncorr.m** source code and set "(Access = public)" to  alter these properties directly as shown below:



```
properties (SetAccess = private)
    handles_gui;
    % DIC data:
    reference;
    current;
    roi;
    data_dic;
    % Installation data:
    support_openmp;
    total_cores;
end
```

It may be possible to crash Ncorr by setting inconsistent data (i.e. a ROI larger than the reference image) and thus caution is advised if going this route.