User Documentation for xCorrRBMCorrection.m

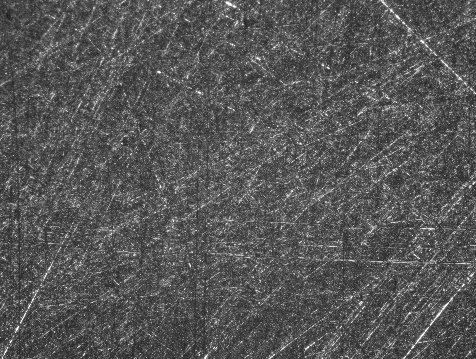
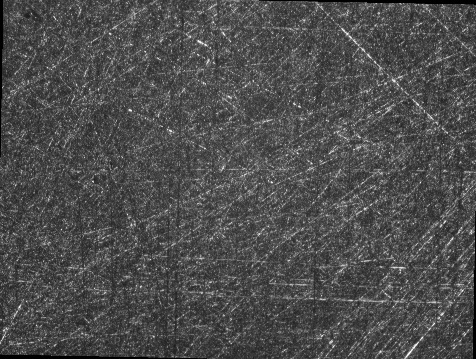
Removal of translational rigid body movement from image sequences.

# Overview

xCorrRBMCorrection.m is a MATLAB program to remove translational rigid body movements from image sequences via a Fourier-space image registration process. This can improve the accuracy of subsequent DIC analysis, especially that which uses the LSM (Least Squares) method. It generates a sequence of ‘corrected’ images where each original image has been cropped such that the field of view of each image is of the same area of the object.

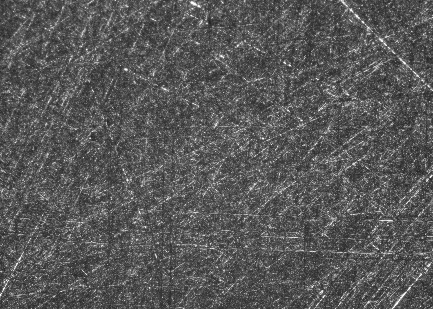
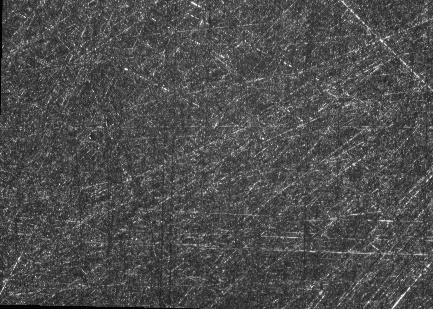
## Inputs

* A single 8-bit .tif image to act as the reference image
* An image / images to perform the rigid body movement correction on. These are known as the ‘test’ images.

## Outputs

* An image / images in which the rigid body movement relative to the reference image has been removed. Each corrected image consists of the intersection of the test image with the reference image, as determined by the Fourier-transform image registration.

* A results table ‘ResultsTable.csv’ which comprises:

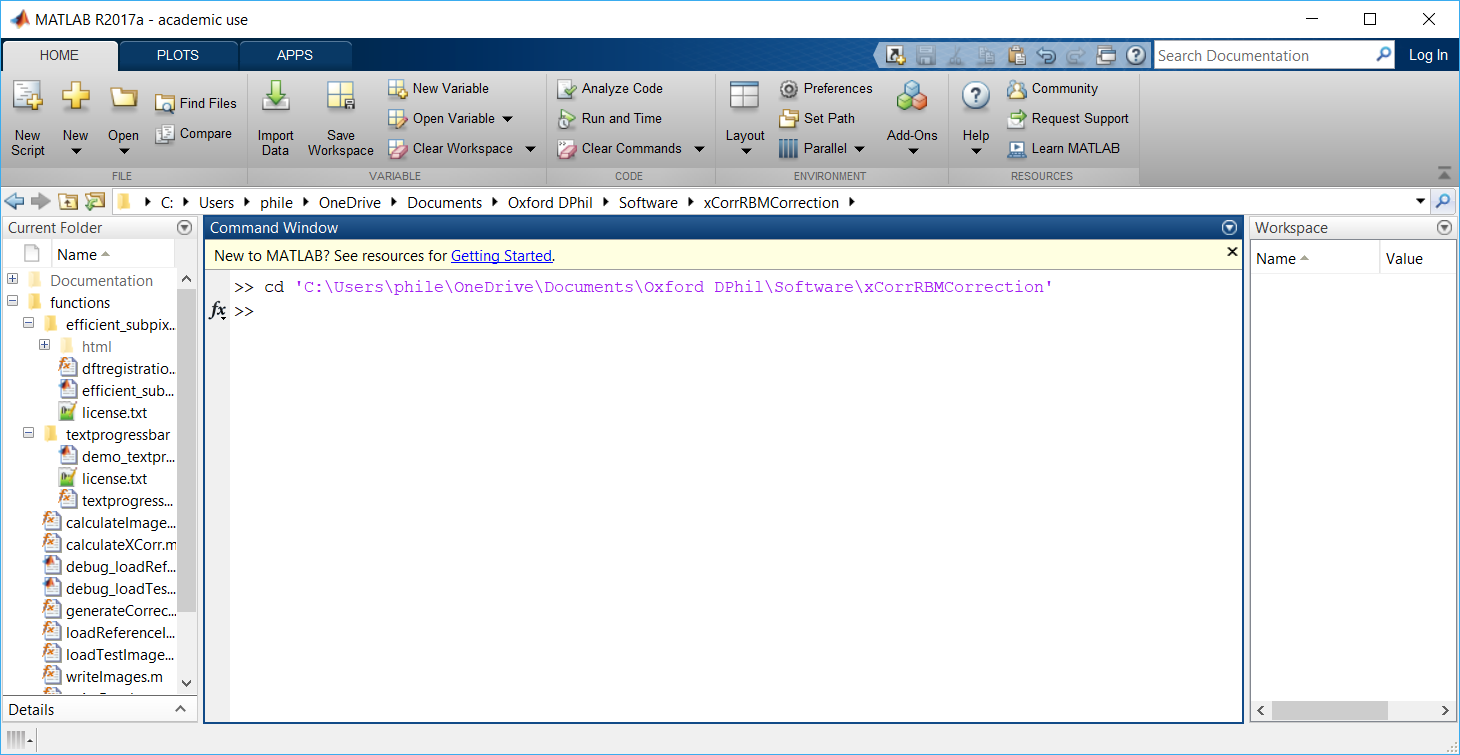
|  |  |  |  |
| --- | --- | --- | --- |
| **Reference Image** | **Test Image** | **ref2TestXShiftPixels** | **ref2TestYShiftPixels** |
| (Filename) | (Filename) | (number of pixels) | (number of pixels) |
|  |  |  |  |

# Preparing MATLAB Environment

* This program has been developed and tested on MATLAB r2017a and r2016b.
* Load MATLAB and navigate to the xCorrRBMCorrection directory containing the function xCorrRBMCorrection.m.
* Identify the files to be used as reference/test images and ensure they are in 8-bit greyscale .tif format. (16/32 bit support coming in a future update)

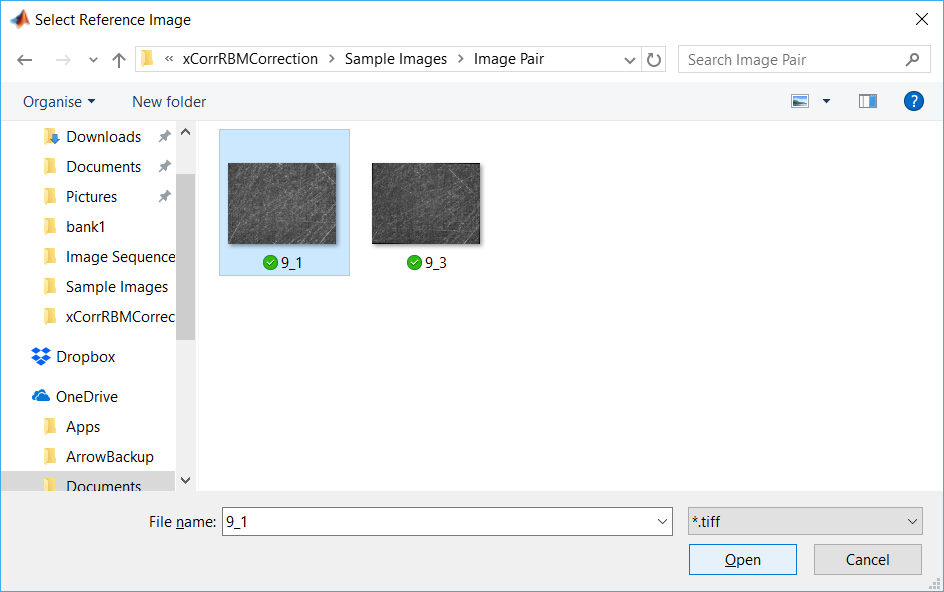
# Example

Navigate to xCorrRBMCorrection directory:

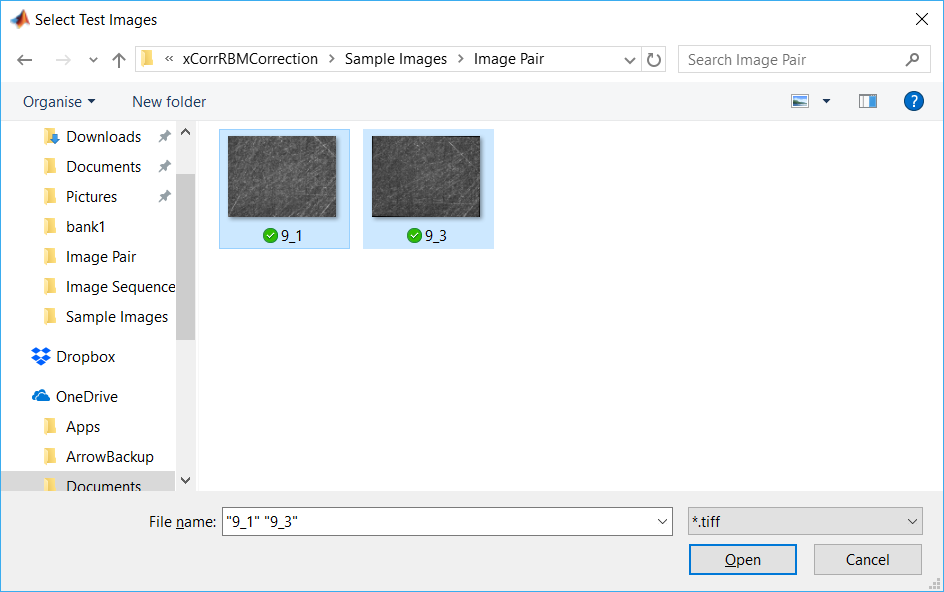


Run xCorrRBMCorrection.m

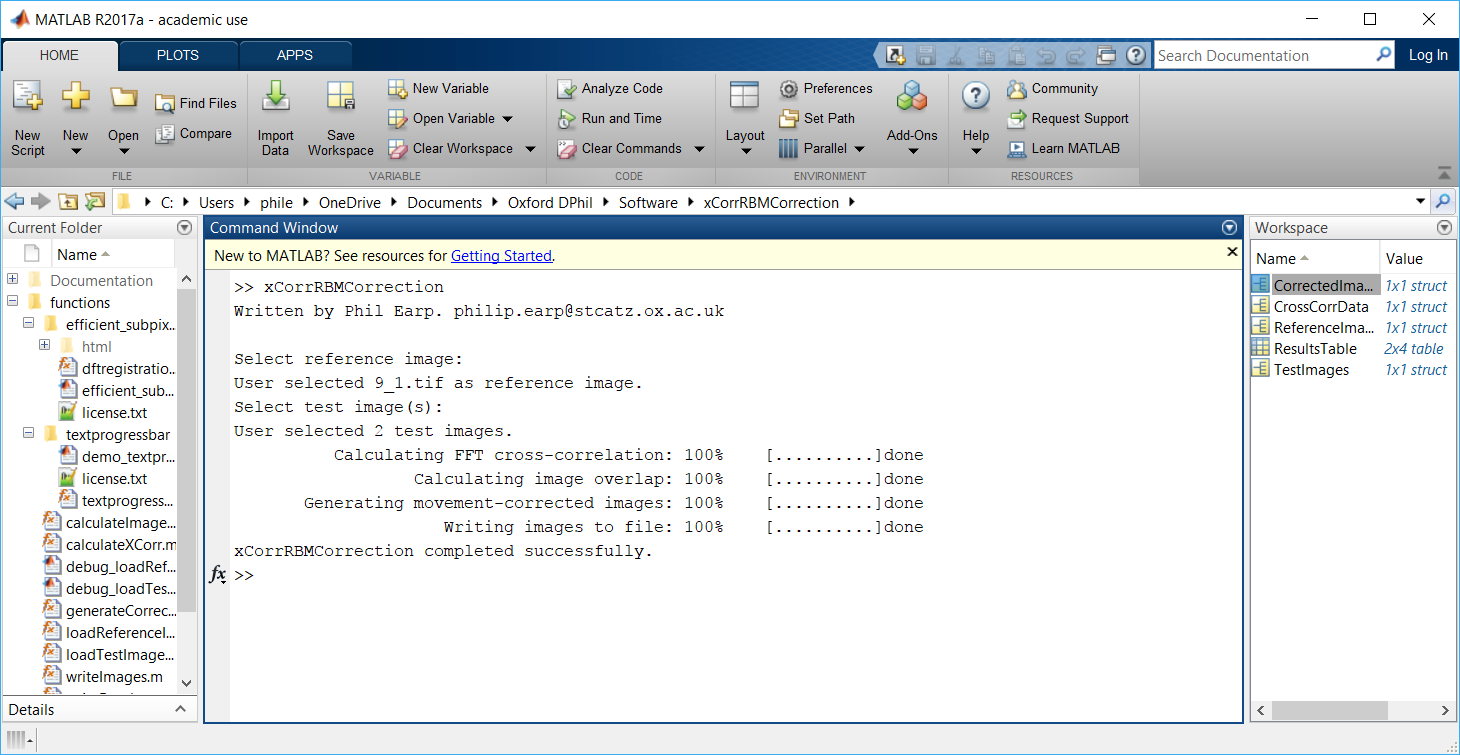
Select Reference Image:



Select Test images:

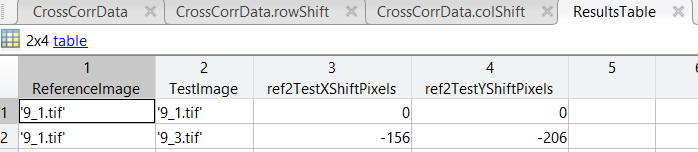


Program runs to completion:



# Results Analysis

The results table indicates the shift in pixels required to shift the *reference* image such that it lies above the *test* image. In the above example, the reference image would have to be shifted 156 pixels to the left (-ve x direction) and 206 pixels down (-ve y direction) to bring the images into coincidence. The shift required to map the *test* image onto the *reference* image would be the negative of this.



# Future Changes

* Currently the program does not check that all the reference and test images have the same dimensions. This check is necessary to ensure correct calculation of the corrected image stacks. An update will cause the program to give an error in this case.
* Test the program with 16-bit images