**pFIRE user documentation**

Dr Ivan Benemerito

**Section Introduction**

pFIRE (parallel Framework for Image REgistration) is an open source software for elastic image registration developed by Phil Tooley at The University of Sheffield (USFD) within the framework of the European project CompBioMed (CBM). pFIRE has been developed to overcome the limitations of ShIRT (Sheffield Image Registration Toolkit), which is currently used for performing elastic image registration at USFD. The main limitation is due to the serial nature of ShIRT: the code is written to use a single CPU, which restricts the size of the largest images that can be treated.

pFIRE uses modern C++ and libraries for parallel computing (PETSc) to overcome this limitation and distribute the data across several processors to perform the registration. pFIRE is released under Apache-2.0 licence and is available on GitHub at [this](https://github.com/INSIGNEO/pFIRE) address.

This document presents the dataset and methodology used for testing pFIRE and comparing its outputs against ShIRT. Throughout this document, ShIRT is considered to be the gold standard for elastic registration against which we benchmark the performance of pFIRE.

Two versions of pFIRE are tested: @0.4.0 and @devel. Simulations are run on ShARC using the versions of pFIRE made available through the RSE.

The dataset used for testing can be found [here](https://drive.google.com/open?id=1rdPcJ3ifOcEvtyppUyqNSGSey9HKuSVo): it consists of 2D and 3D images. 2D images representative of simple displacement fields:

* Compression along one direction, the other is constrained to zero
* Shear along one direction, the other is constrained to zero
* Isotropic expansion and translation

The first two 3D images are obtained after CT scans of real specimens and represent a variety of displacement fields, whereas the last two have been created in MATLAB:

* Rigid translations along three axis
* Compression along one direction, the other two are not constrained
* Compression along one direction, the other two are constrained to zero
* Isotropic expansion

**Subsection Preliminary**

pFIRE accepts as input files in format .dcm and .image. Throughout this document simulations are run using .image file. .image files can be obtained by .dcm using the Python library [flannel](https://github.com/ptooley/flannel). When needed, downsampling of the images is performed using the python script dcmdownscale.py, available [here](https://drive.google.com/open?id=1I8zgqcVXw6yQFUSsgQhgub5Mkhbrux3X). The resulting image presents a 90 degrees rotation in the x-y plane which does not affect registration itself.

**Subsection Running pFIRE on ShARC**

The use of pFIRE is extensively documented [here](https://insigneo.github.io/pFIRE/docs.html), this section will cover only the basic procedure for submitting a job. Submission scripts for pFIRE are available [here](https://drive.google.com/open?id=1W_DJUJlvm9OKIZP9JsN2YawqTKGAJn9D). pFIRE is run in batch by submitting the file pfire\_example.qsub through the command

$ qsub pfire\_example.qsub

The submission script automatically loads all the dependencies pFIRE needs for running. By modifying lines 2, 3 and 4 the user can tune the request of cores, memory and time. Line 5 asks for exclusive access to the node, line 6 sets the name of the batch job. Lines 7 and 8 submit the job to the IMSB node and IMSB queue (see [here](http://insigneo-tech.group.shef.ac.uk/master/compute/sharc.html" \l "using-these-nodes)): if the user does not have access to such resources, those lines must be deleted. Line 17 load pfire as a spack module and allows the user to choose the version (pfire@0.4.0 and pfire@devel). Line 19 launches pfire through an mpi job using the content of run\_pfire.conf as input. The user can edit run\_pfire.conf to choose the fixed and moved input images, the nodal spacing, the smoothing parameter and whether the program should save the intermediate results from the simulation. However, saving intermediate results increases greatly the amount of time that will be needed for the simulation to complete.

**Subsection Postprocessing of pFIRE results**

pFIRE produces the following output files:

* map.xdmf
* map.xdmf.h5
* registered.xdmf
* registered.xdmf.h5

The file map.xdmf contains the following groups:

* nodes\_x
* nodes\_y
* nodes\_z
* x
* y
* z

The first three groups are the x, y and z coordinates of the grid nodes along the x, y and z direction, and they are used to build the 3D grid that is superimposed to the images to perform the elastic registration. The groups x, y and z contain the displacements in the three spatial coordinates as three-dimensional matrices.

Post-processing of the results, performed in MATLAB using scripts and functions available [here](https://drive.google.com/open?id=16NE5yiobmncbymzmtwBKhwCIRE6k-asr), produces the lists of nodal position, nodal displacement and element in .txt format, following the ordering criterion of ShIRT. Scripts make use of relative paths, therefore the structure of the directories must be kept as in the Google Drive folder. Nodal position and nodal displacement are expressed in [mm]. A script for extracting the solution from a cubic region of interest is also provided.

pFIRE@0.4.0 performs the elastic registration on a grid with the same number of nodes of ShIRT, and the two outputs are therefore of the same dimension. pFIRE@devel performs the registration on a grid smaller than that used by pFIRE@0.4.0. A MATLAB script is used to extract from ShIRT the subset of the solution for which the comparison with pFIRE@devel is meaningful. The structure of the nodes to be discarded by ShIRT varies according to the size of the image, thus the script must be customised on the image.

**Subsection Running ShIRT on ShARC**

ShIRT is installed on ShARC but is not openly available. Access to ShIRT must be required and granted.

**Section Summary of the results and current situation**

pFIRE@0.4.0 has bad wallclock and memory performance. It performs perfectly in 2D, good on zero strain and extremely bad on virtual tibia.

pFIRE@devel performs bad on all the images. Performances are better in terms of memory, and slightly worse in terms of wallclock.

**Section 2D TEST IMAGES**

All images have been run with voxel size 1 and nodal spacing 20. pFIRE@devel produces results on a smaller grid: however, since this version does not work correctly, this is not taken into account in the analysis.

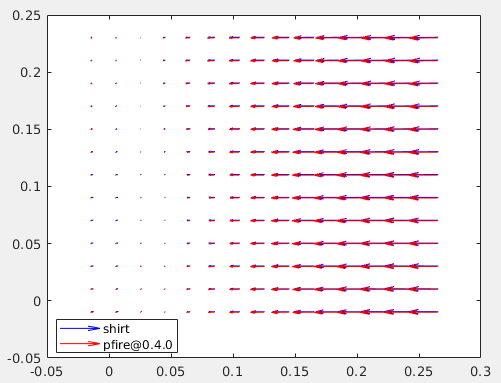
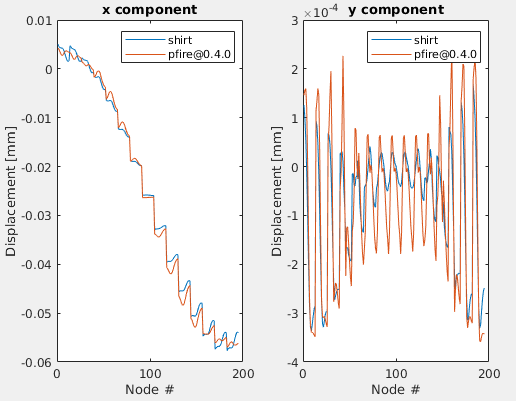
**Section compression\_x**

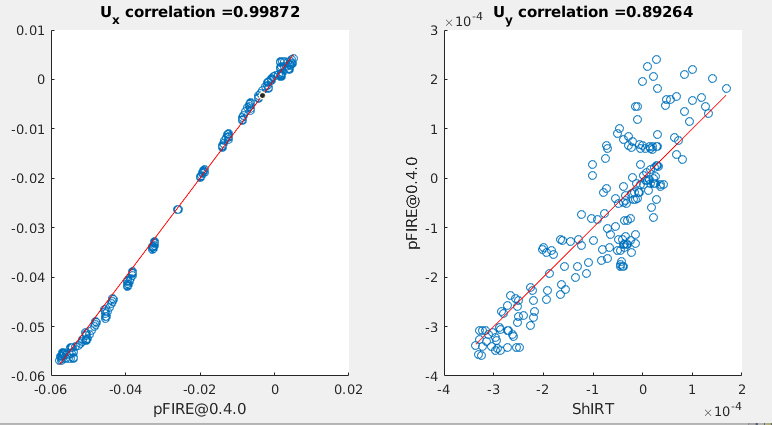
Raw data and processed results are available [here](https://drive.google.com/open?id=1NLfUqzu01XKVA5VVrSikL5510mIuNxdr). The image size is 220x250. The white rectangle has original size of 180x80, and is compressed to 70% of its original length along the longest edge. The expected compressive displacement is 54 um on the extremal edge of the rectangle.



**Subsection ShIRT and pFIRE@0.4.0**

ShIRT and pFIRE@0.4.0 predicts the expected compressive displacement. The agreement is excellent, as shown by the plot of nodal displacements, the quiver plot of the displacement and the linear correlation coefficient.

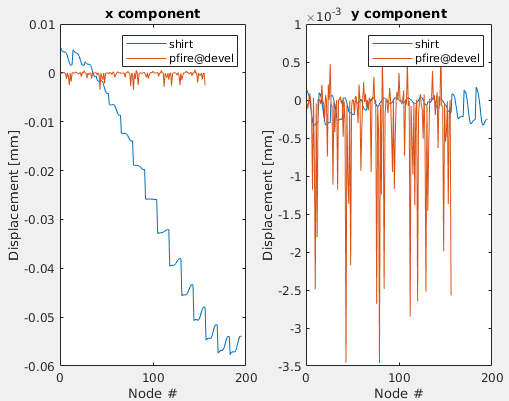




Points in the y direction are more scattered and consequently the correlation is lower. The displacement along y is theoretically 0, and the two methods predict values around 1e-4 mm which is practically zero.

**Subsection pFIRE@devel**

x and y components of the displacement are identical and practically zero. Comparing against ShIRT does not yield good results.



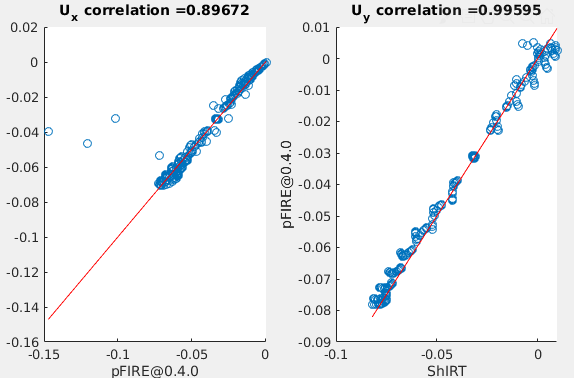
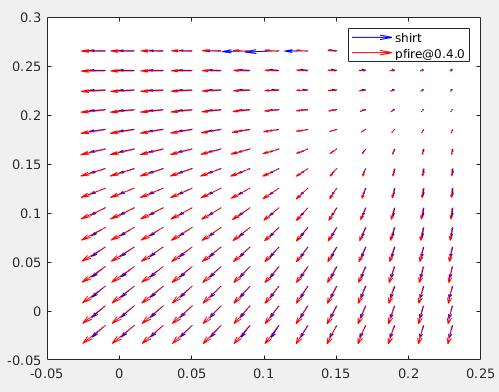
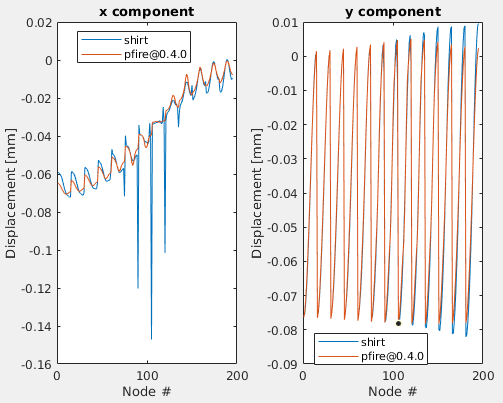
**Section 2D\_expansion**

The image size is 220x250.



**Subsection ShIRT and pFIRE@0.4.0**

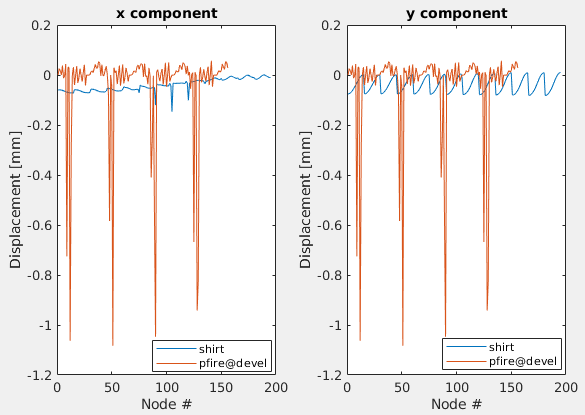
ShIRT and pFIRE@0.4.0 predicts the expected displacement. The agreement is excellent, as shown by the plot of nodal displacements, the quiver plot of the displacement and the linear correlation coefficient.



The correlation coefficient along x drops because in few points, clearly identifiable from the nodal displacement and quiver plots, the displacement given by ShIRT differs considerably by that of their neighbours.

**Subsection pFIRE@devel**

Vague resemblance to the expected displacement. It does not mean that pFIRE@devel performs well on this image. The resemblance is a consequence of pFIRE@devel predicting the same displacement along x and, and in this case the expected displacement has x and y components which are pointwise similar in magnitude and direction.



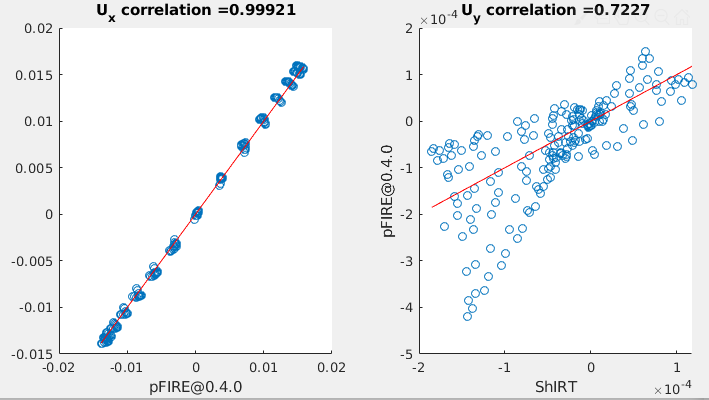
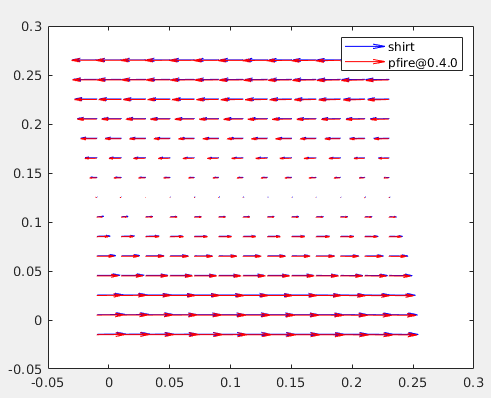
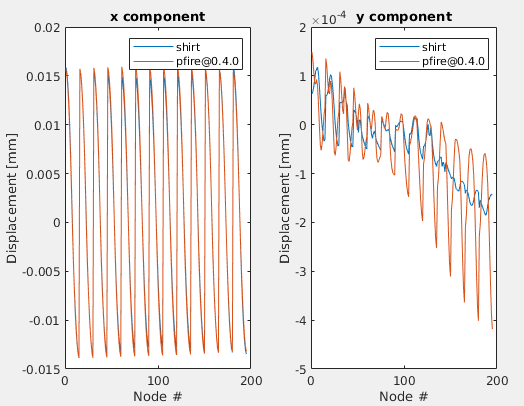
**Section 2D\_shear**

The image size is 220x250.



**Subsection ShIRT and pFIRE@0.4.0**

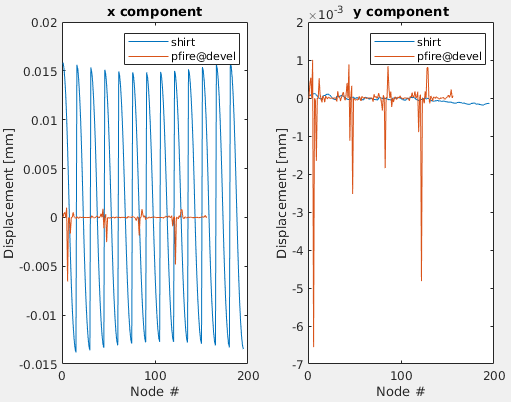
The agreement between ShIRT and pFIRE@0.4.0 is excellent, as shown by nodal and quiver displacement, and linear regression.



Points in the y direction are more scattered and consequently the correlation is lower. The displacement along y is theoretically 0, and the two methods predict values around 1e-4 mm which is practically zero.

**Subsection pFIRE@devel**

x and y components of the displacement are identical and practically zero. Comparing against ShIRT does not yield good results.



**Section 2D\_brain**

Nodespacing = 10

Voxel size = 1

**Section ET3**

Images in .dcm and .image format can be found [here](https://drive.google.com/open?id=1FeKS70dv0u8D6OCul7vs8mCgcejVglXl). It is a cube of trabecular bone, scanned twice without any deformation being applied (zero strain). The dimension of the dicom image are 210x210x210 voxels, and each voxel has a dimension of 9.96 um, resulting in a total dimension of approximately 2x2x2 mm^3. Simulations are run with nodespacing 20.

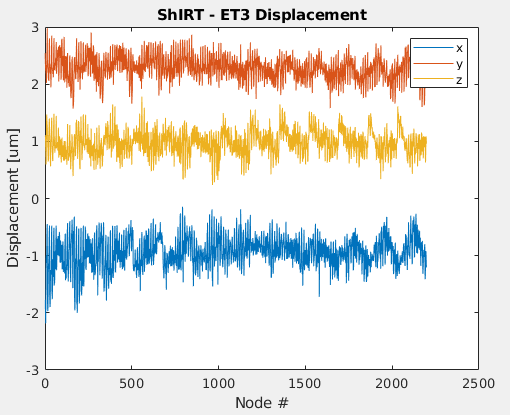
**Subsection ShIRT**

The solution computed by ShIRT is available [here](https://drive.google.com/open?id=1cFYec0OL2SydCD17YU9dfjiPqFLn6Ehm). ShIRT predicts small displacements, uniform over the grid along the three directions. The displacements are clearly divided into bands as in figure, suggesting that they are micro-translations occurred due to the repeated scanning. Average values are

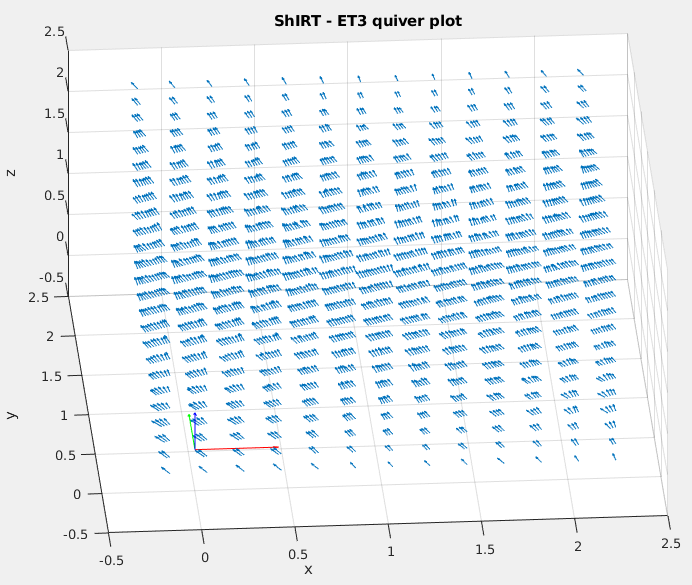
x = -0.93±0.27 um

y = 2.28±0.2 mm

z = 0.98±0.21 um

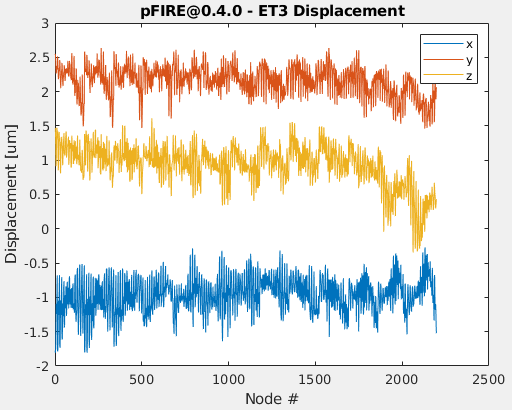


A quiver plot shows clearly that the displacements are homogenous along the three directions.



**Subsection pFIRE@0.4.0**

Results from pFIRE@0.4.0 are available [here](https://drive.google.com/open?id=1QFqirApfxmXHaGzkNAV1wfUmvDuqFRgB). The trend is the same as ShIRT’s, with the displacement clearly divided into three bands.



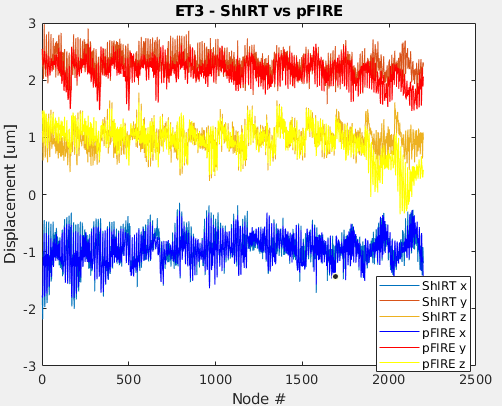
Average values are

x = -0.96±0.25 um

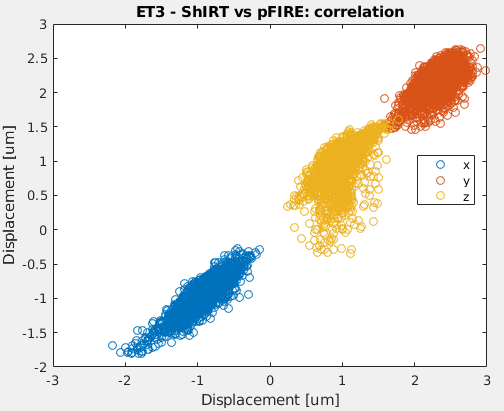
y = 2.16±0.21 mm

z = 0.93±0.3 um

Superimposing of the solution for ShIRT and pFIRE seems to show a moderate agreement,

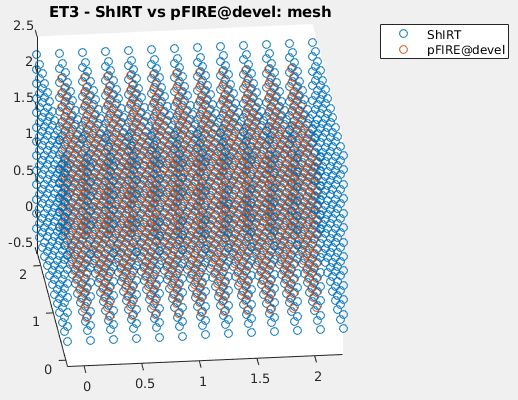


however point-to-point comparison ([Oliviero et al., 2018](https://www.sciencedirect.com/science/article/pii/S175161611830328X)) and scatter plots show low values of correlation.

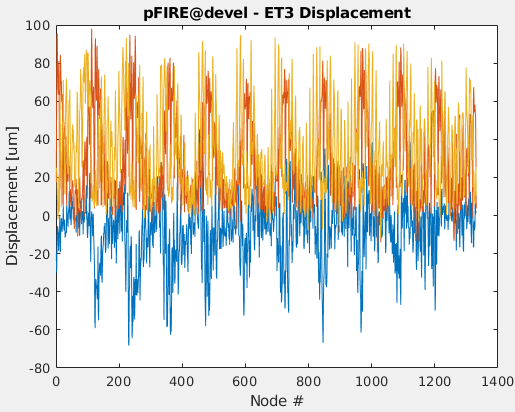


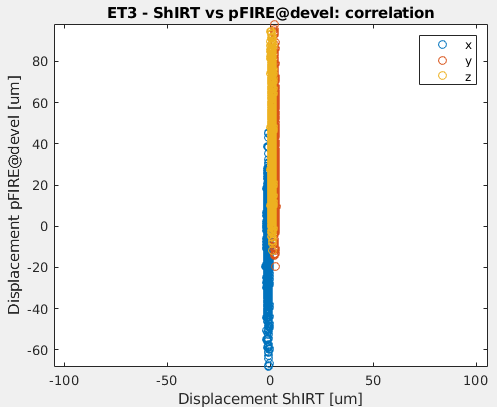
**Subsection pFIRE@devel**

Results from pFIRE@devel are available [here](https://drive.google.com/open?id=16WjgX1GBG2ykJU8g77QkulnsQrD37xrz). The solution is provided on a number of nodes smaller than ShIRT. The figure above shows that the outer layer of nodes must be discarded from ShIRT’s solution.



Displacements computed by pFIRE@devel are considerably larger (about 30 to 100 times) than ShIRT’s

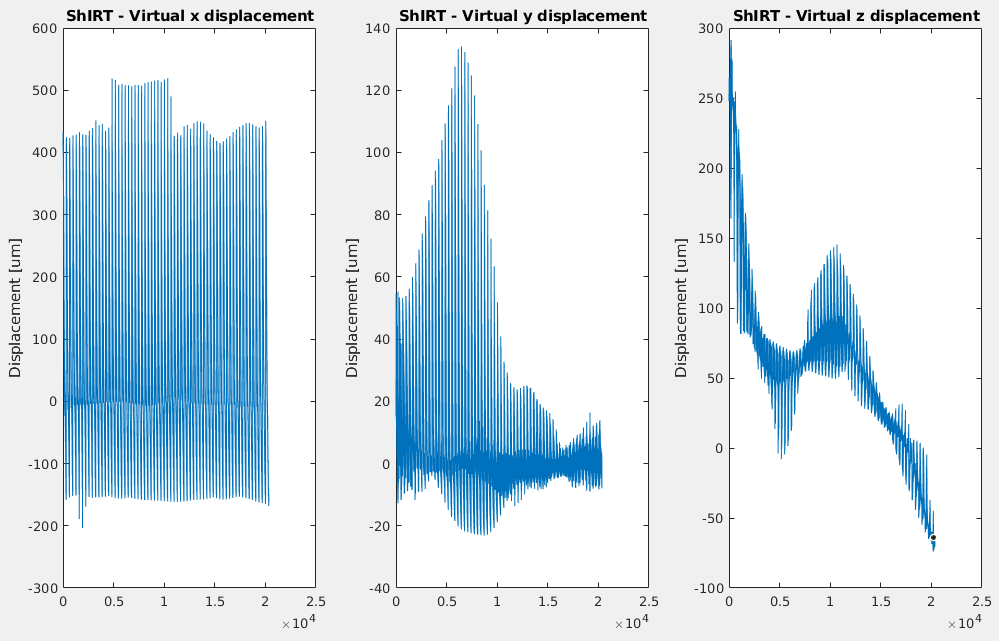




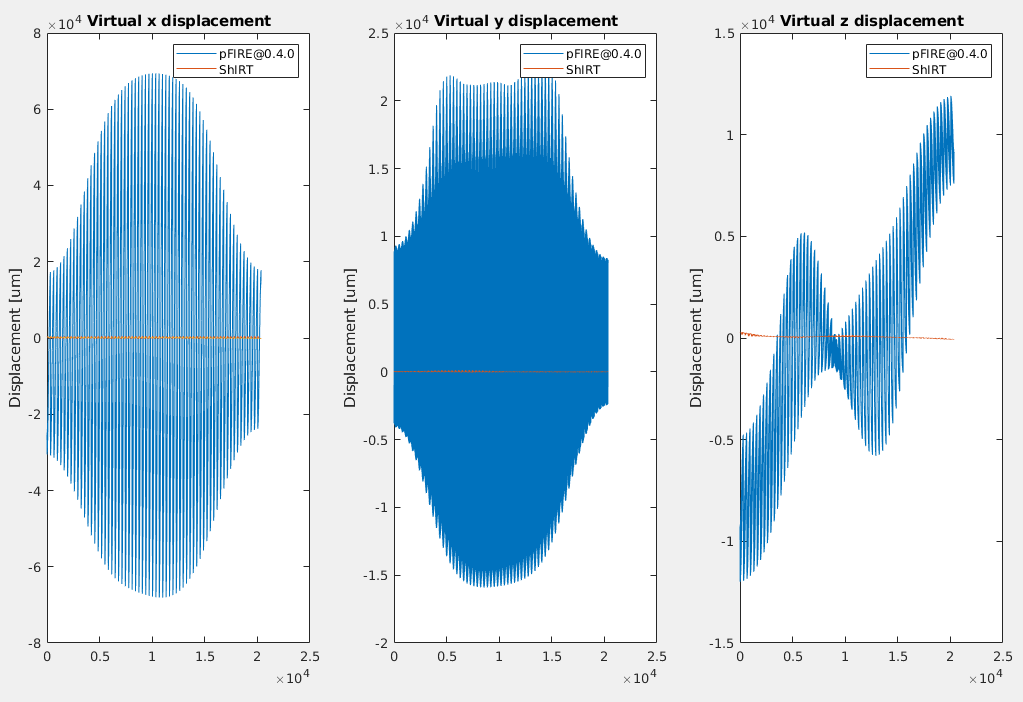
**Section TIBIA (virtual)**

The original image has size 454x432x1799 voxels and voxel size 10.4 um, resulting in size 4.7x4.49x18.7 mm3. For reasons of speed and memory simulations are run on a downsampled image of size 144x172x616, with voxel size 31.2 um and nodal spacing 10. The voxel size is altered to keep the same physical dimension after the downsampling. The image contains a mouse tibia, which has been artificially deformed in Amira by applying a displacement of 94 um (~ 3 voxels) on the most proximal part of the bone. The displacement at the distal part is zero. Displacements in the transverse plane are not constrained.

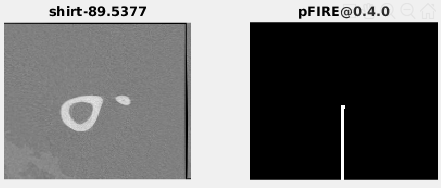
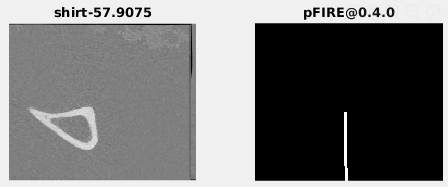
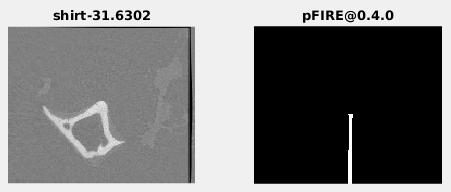
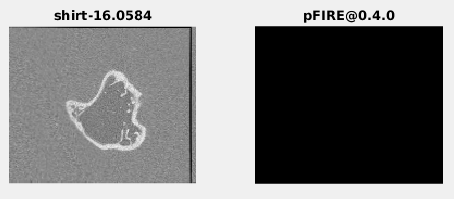
**Subsection ShIRT**



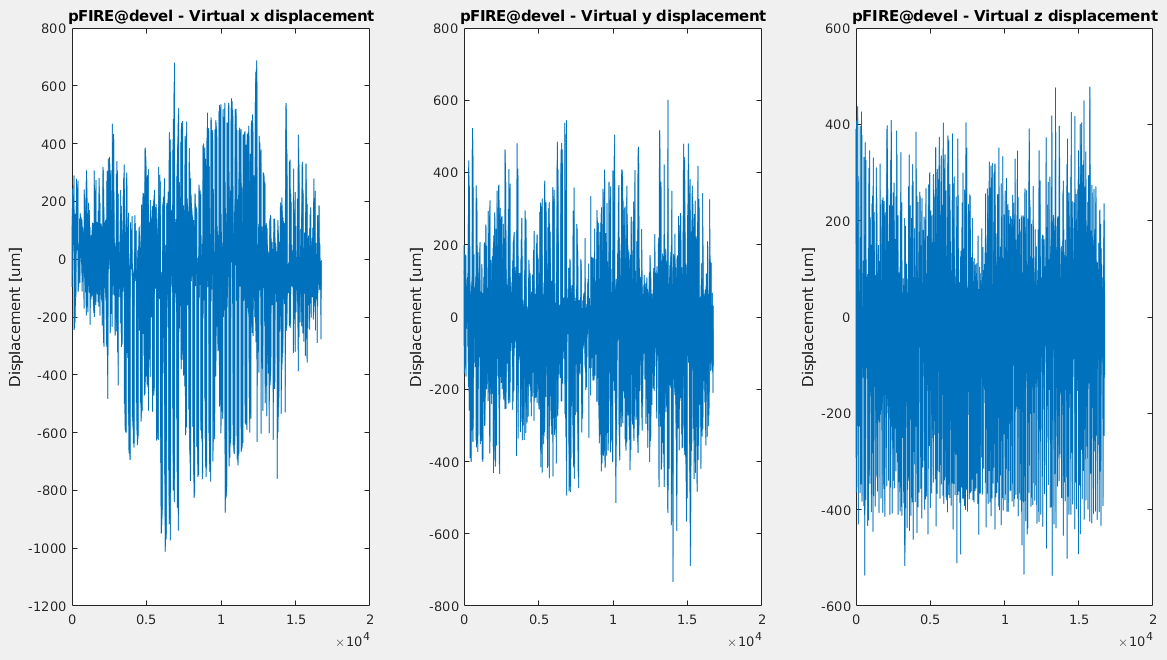
**Subsection pFIRE@0.4.0**



Registered images:



**Subsection pFIRE@devel**



**Section 3D TEST IMAGES**

**Section 3D\_Transl\_y**

The image has been created in MATLAB with a size of 210x190x420. It is a white parallelepiped on black background. It is rigidly displaced along the y direction of 50 voxels. Expected results from a simulation with nodespacing 20 and voxel size 1 is 0 displacement along x and z, and

**Subsection ShIRT**

ShIRT correctly identifies a y-displacement of 0.05 mm and 0 displacement in the other directions.

**Subsection pFIRE@0.4.0**

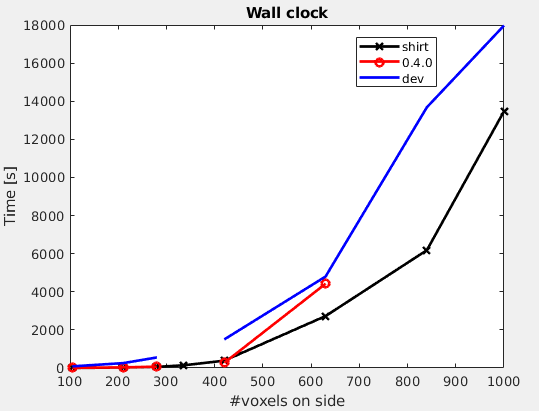
pFIRE@0.4.0 correctly identifies a y-displacement of 0.05 mm and 0 displacement in the other directions.

**Section Benchmarking**

ShIRT, pFIRE@0.4.0 and pFIRE@devel have been benchmarked on wallclock time and maximum memory consumption. Registrations have been performed on scaled versions of ET3, that is a cube of trabecular bone subject to zero strain deformation. The size of the cube’s side were 105, 210, 280, 336, 420, 630, 840 and 1000 voxels. Simulations were run on ShARC with node spacing 20. Both versions of pFIRE were run asking for exclusive access to the IMSB large memory node (node 169).

ShIRT shows the best performance in terms of wallclock and memory consumption. They scale with the third power of the side of the cube or, equivalently, linearly with the total number of voxels in the image.

Wallclock times of [pFIRE@0.4.0](mailto:pFIRE@0.4.0) and [pFIRE@devel](mailto:pFIRE@devel) are comparable when registering images with side smaller than 630 voxels. Because of the excessive memory consumption, [PFIRE@0.4.0](mailto:PFIRE@0.4.0) is not able to treat images larger than this. For reasons yet to identify, the memory consumption of [pFIRE@devel](mailto:pFIRE@devel) does not increase monotonically, and this makes [pFIRE@devel](mailto:pFIRE@devel)able to deal with images with side as big as 1000 voxels.



**Conclusions**

The results presented in this document show that pFIRE@0.4.0 performs better than pFIRE@devel.

pFIRE@0.4.0 shows perfect agreement with ShIRT on 2D images, with values of correlation close to 1. pFIRE@devel, instead, produces an x displacement perfectly identical to the y displacement on 2D images, resulting in values which are completely inaccurate. Furthermore, the code produces a 1D registered image, therefore completely different from the expected one.

In 3D scenarios none of the version behaved satisfactorily. pFIRE@0.4.0 is able to register correctly the zero strain image, but fails on the tibia both in terms of displacement map and registered image. pFIRE@devel, coherently with its overall poor performances, does not register correctly any of the images. It is not clear though why the registered images are not severely deformed.

The following table summarises graphically the results presented in this document.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Registered Image: Visual comparison | | | | | Order or magnitude of displacement | | | | | | | Correlation of displacement | | | | |
| 2D | | | 3D | | 2D | | | | 3D | | | 2D | | | 3D | |
| Compr | Exp | Shear | ET3 | Tibia | Compr | Exp | Shear | ET3 | | Tibia | Compr | | Exp | Shear | ET3 | Tibia |
| pFIRE@0.4.0 |  |  |  |  |  |  |  |  |  | |  |  | |  |  |  |  |
| pFIRE@devel |  |  |  |  |  |  |  |  |  | |  |  | |  |  |  |  |