Original Software Publication

FM-Track: A fiducial marker tracking software for studying cell mechanics in a three-dimensional environment

Emma Lejeune [a,b], Alex Khang [a], Jacob Sansom [a,c], Michael S. Sacks [a,c,*]

[a] *James T. Willerson Center for Cardiovascular Modeling and Simulation, Oden Institute for Computational Engineering and Sciences and The Department of Biomedical Engineering, The University of Texas at Austin, Austin TX, United States*
[b] *The Department of Mechanical Engineering, Boston University, Boston MA, United States*
[c] *The Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, Austin TX, United States*

ARTICLE INFO

ABSTRACT

Tracking the deformation of fiducial markers in the vicinity of living cells embedded in compliant synthetic or biological gels is a powerful means to study cell mechanics and mechanobiology in three-dimensional environments. However, current approaches to track and quantify three-dimensional (3D) fiducial marker displacements remain ad-hoc, can be difficult to implement, and may not produce reliable results. Herein, we present a compact software package entitled "FM-Track," written in the popular Python language, to facilitate feature-based particle tracking tailored for 3D cell micromechanical environment studies. FM-Track contains functions for pre-processing images, running fiducial marker tracking, and post-processing and visualization. FM-Track can thus aid the study of cellular mechanics and mechanobiology by providing an extensible software platform to more reliably extract complex local 3D cell contractile information in transparent compliant gel systems.

Current code version
Code metadata

Code metadata description	Please fill in this column
Current code version	v1
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX_2019_342
Legal Code License	MIT License
Code versioning system used	git
Software code languages, tools, and services used	python
Compilation requirements, operating environments	py-earth, scikit-learn, pyvista
If available Link to developer documentation/manual	https://github.com/elejeune11/FM-Track/blob/master/README.md
Support email for questions	elejeune@bu.edu

1. Motivation and significance

At present, there is substantial research focused on understanding how cells interact mechanically with their local environment [1–4]. While the field started with cells seeded on two-dimensional surfaces [5–9], emphasis is now moving towards greater realism through the study of the three-dimensional cellular micro-environment, typically using cells embedded in transparent natural or synthetic gels [10–16]. One major class of methods for probing cell mechanics involves tracking the motion of fiducial markers in the vicinity of a given cell [17,18]. For example, use of 3D Traction Force Microscopy (TFM) has grown in recent years [10,11,13–15,19]. In TFM, cells and μm-scale fluorescent beads are co-seeded in gels. The observed deformation of the fluorescent beads in the vicinity of the cell is then used to infer the local forces that the cell is imposing on its local environment given known mechanical properties of the gel [10,20]. However, widespread adoption of TFM and related methods is limited, in part, because the software required for successful bead tracking is

* Correspondence to: 201 East 24th St., Stop C0200, The University of Texas at Austin, Austin, TX 78712-1229, United States.
E-mail address: msacks@oden.utexas.edu (M.S. Sacks).

not trivial to implement and lacks overall standardization [17,18, 21–26]. This includes not only tracking but also features such as microscope imaging stage translation correction and robust post-processing of the resulting displacement data. Such limitations make both the initial displacement measurement and subsequent interpretation difficult. Overall, there is a need for accessible software to enable cell mechanics research and promote standardized technique across different laboratories.

To address these issues, we have developed a software package entitled "FM-Track," an open source software written entirely in the popular Python language. While there are multiple examples of feature based particle tracking algorithms described in the literature [10,18,27], there are limited available open source software resources [23]. With FM-Track, we specifically build on extant methods to introduce an open source particle tracking software written entirely in the Python language. Important novel aspects, to the author's knowledge, are (1) tracking algorithm improvements such as our procedure for checking different permutations of feature matches and (2) a multivariate adaptive regression spline (MARS) based translation correction function that is necessary to overcome the potential microscope hardware limitations of unintended sample translation. In the following, we include a description of how FM-Track works in Section 2 and a minimum working example of running the algorithm presented in Section 3.

2. Software description

Many open source particle tracking algorithms are designed for systems with high time resolution, so that the change in particle positions between images is small [28]. Herein, we focus exclusively on approaches that use paired images, one initial image and one final image, with inert fiducial markers (typically fluorescent beads) and no specific limitations on the displacement magnitude. In this scenario, a successful software implementation must be able to accommodate large deformations between time frames, but computational efficiency is less critical given that the algorithm will only have to run once per image set and will never run in real time. The software written to easily accommodate different sets of input parameters. To operate the software, the user will feed variables (both file paths and tunable parameters) into the "InputInfo" class. The file "run_tracking_all_steps.py" is set up such that the user can run the algorithm on a set of images by calling the "run_tracking_all_steps()" function once.

2.1. Software architecture

Work flow in FM-Track is broken down into the following three steps: (1) pre-processing, (2) tracking, and (3) post-processing. These functions are defined in three separate files and are all called from the main function "run_all_track()". User-specified input parameters and paths to image files are identified in a separate class "InputInfo". The flow chart of FM-Track is illustrated in Fig. 1.

2.2. Software functionalities

The pre-processing file contains three functions. First, a function to import the three dimensional image stacks taken with a confocal microscope in the correct order as a numpy array. Second, a function to identify the coordinates of the center of every bead in a given image. Third, a function to identify the approximate volume, center, surface mesh, and surface mesh normals of a single cell in a given image. If there are multiple cells in the image, the function in its current state will only return the largest cell by volume.

The tracking file contains a main function "track_main-_call()" to perform bead tracking given a specified first and second set of input images. If tracking type 1 is specified, the tracking function will run the tracking algorithm once. If tracking type 2 is specified, the tracking function will run the tracking algorithm, run a translation correction algorithm, and then run the tracking function again. The translation correction algorithm will correct for any translation that can be described as a function of $z-$position (Fig. 1). For clarity, a general outline of the particle matching algorithm is also provided (Fig. 2).

Briefly, a feature vector is computed for each particle. The feature vector contains the relative position of each bead and a fixed number of neighbors (the default selection is 5 neighbors). Each particle's feature vector will then be compared to a fixed number of particles and their feature vectors in the other image configuration (the default selection is 15 comparison particles). The comparison particles are the particles closest to the original particle measured by center to center distance, with the equation for comparing feature vectors is given in Fig. 2. Essentially, the distance between each entry of the two feature vectors is computed. We also compare the permutations of the feature vector and select the lowest score among all of the possible permutations. Then, the candidate particle with the lowest score is considered the best match. If there is a "clash" the pair with the lowest score is selected and the other pair is assigned its next best match. We perform the clash checking procedure iteratively. This algorithm is run from configuration 1 to configuration 2, and then from configuration 2 to configuration 1. Matches that are not identical in both directions are discarded. This approach is an adaptation of other algorithms proposed in the literature [10,23]. We note that future users of FM-Track can take advantage of our modular software design to implement an alternative feature comparison equation. Once the tracking step is complete, the initial coordinates and displacement vector are recorded for every bead that was successfully matched. The tracking file also contains a similar function called "track_no_cell()" that is suitable for running the algorithm when there is no cell present.

The post-processing functions are included for easy visualization of the bead tracking results. Several different types of plots of the raw bead displacements are available (see Figs. 3 and 4). Furthermore, the post processing step also includes a function to interpolate the raw bead displacements using Gaussian Process Regression (GPR) "create_GP_model" and a function to produce filled color plots of the interpolations "plot_gp_model". Further information on Gaussian Process Regression is available in the literature [29,30].

3. Illustrative example

To illustrate the functionality of FM-Track, we present an example of a porcine aortic valve interstitial cell (AVIC) embedded in a PEG hydrogel containing suspended 0.5 μm microspheres serving as fiducial markers [31]. Details of the methodology regarding AVIC encapsulation within a PEG hydrogel have been previously presented [31]. We note that the microsphere diameter was chosen to be large enough such that beads can easily be registered but small enough such that a large number of beads can be included in the gel without mutually interfering or causing cell death. AVICS were imaged under normal conditions and imaged after it is treated with Cytochalasin-D, a chemical that inhibits actin polymerization and therefore shuts down the contractile machinery of the cell. By comparing the treated cell to the normal (untreated) cell, we are able visualize how the cell actively deforms the surrounding gel. A brief overview
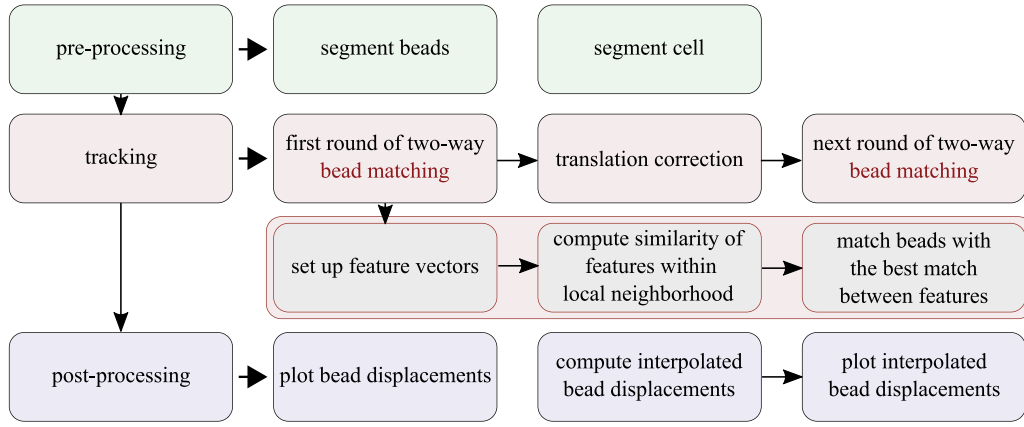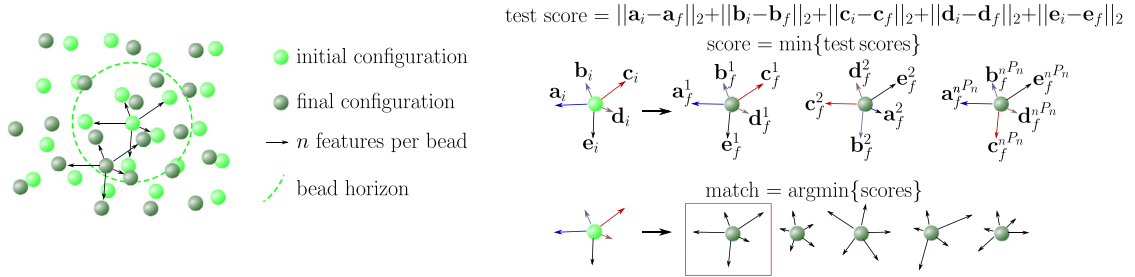
Fig. 1. Basic flow chart of FM-Track.

Fig. 2. Schematic of the algorithm for feature based bead matching. The $_nP_n$ permutations of features for each bead within a defined horizon are compared, and the matched bead is the bead with the best score. Clashes are resolved by assigning the match to the pair with the better score. Two-way tracking means that the match must hold when the algorithm is run from initial configuration to final configuration and from the final configuration to the initial configuration.

of the experimental protocol used to generate these images is included in Appendix A. In Appendix B, we also included two additional examples with synthetic data derived from a finite element model of a block undergoing uniaxial extension. We note that the synthetic data is useful for evaluating the performance of our tracking algorithm. On synthetic data, our tracking algorithm performs quite well on the relevant bead densities up to realistic expected levels of deformation.

To run the code, first install FM-Track using the installation instructions specified in the "README.md" file located in the "FM-Track" repository. Next, move the sample data folder and the "test.py" script to the same directory. An example data set and the "test.py" script are both found in the examples folder of the repository. Finally, change the name of the "root_directory" variable at the top of "test.py" and run it using either a Python IDE or the terminal. Further instructions on how to run the script are found in the "README.md" file.

To implement the code on a new data set, it will be necessary to modify the following variables:

- "root_directory": used to specify the path to the data folder
- "filenames_cell": used to specify the path to all cell files (the images must be sequentially numbered, with 4 digits padded by zeros)
- "filenames_beads": used to specify the path to all bead files
- "savefnames": used to specify the unique filename that segmented cell and bead files will be saved as
- "tracking_pairs": used to specify the tracking pairs, (these must match the unique filenames specified in "savefnames")
- "fov_dims": used to specify the microscope field of view dimensions

If necessary, changes can readily be made to: the specified fluorescent color channels in the RGB input images "cell_channel" and "bead_channel"; the threshold for segmenting the cell image "cell_thresh"; the tracking parameters "num_feat", "num_nearest", "buffer_cell_tracking", and "track_type"; the translation correction parameters "buffer_cell_translation"; and the type(s) of figure to save post processed results as "figtype_list".

```
1   root_directory = '/Users/<username>/Desktop/data'
2
3   # section (1)
4   # import the necessary modules to use FM-Track
5
6   from fmtrack.inputinfo import InputInfo
7   import numpy as np
8
9   # section (2)
10  # initialize all variables that will be passed into InputInfo
11
12  filenames_cell = [ \
13      'CytoD/Cell/Gel 2 CytoD%s.tif',\
14      'Endo1/Cell/Gel 2 Endo1%s.tif',\
15      'Normal/Cell/Gel 2 Normal%s.tif']
16  filenames_beads = [ \
17      'CytoD/Beads/Gel 2 CytoD%s.tif',\
18      'Endo1/Beads/Gel 2 Endo1%s.tif',\
19      'Normal/Beads/Gel 2 Normal%s.tif']
20
21  savefnames = [ \
22      'CytoDSave',\
23      'Endo1Save',\
24      'NormalSave']
25
26  tracking_pairs = [ \
27      ['CytoDSave', 'Endo1Save'],\
28      ['CytoDSave', 'NormalSave'],\
29      ['Endo1Save', 'NormalSave']]
30
31  fov_dims = np.array([149.95, 149.95, 140.0])
32
33  # section (3)
34  # create an InputInfo object
35
36  info = InputInfo(root_directory)
37  info.set_inputs(filenames_cell, filenames_beads, savefnames,
        tracking_pairs, fov_dims)
38
39  # section (4)
40  # additional, optional steps of the script
41
42  #out_folder_cell = 'Gel_cell_coords'
```
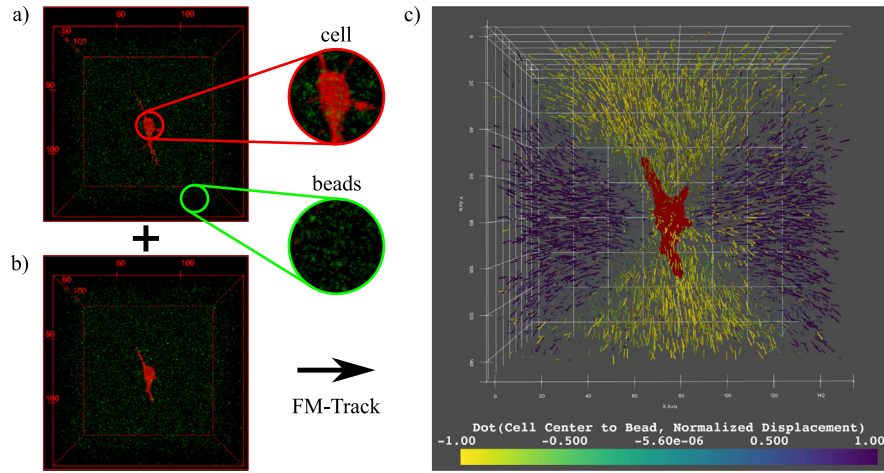
Fig. 3. (a) 3D image stack of an AVIC (red), wherein cytochalsin D has been applied to suppress the cell's contractile machinery, surrounded by fluorescent beads (green) visualized with ImageJ software; (b) 3D image stack of a cell in a normal state surrounded by fluorescent beads visualized with ImageJ software; (c) superimposed image of both cells with bead displacements illustrated visualized with ParaView software.
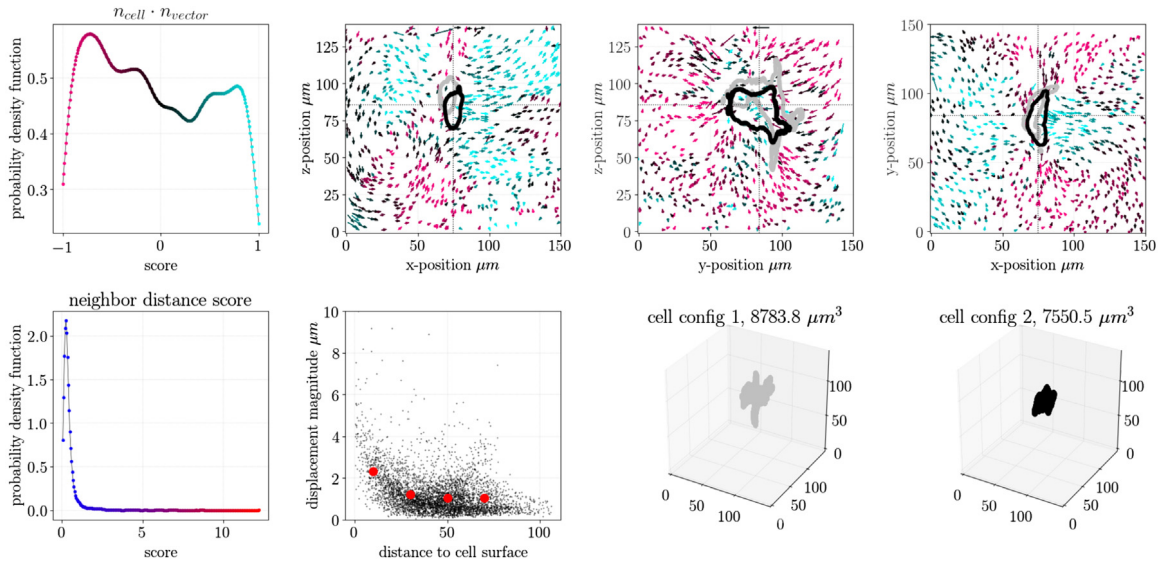
Fig. 4. Summary of the results from FM-Track. Upper left: distribution of $n_{cell} \cdot n_{vector}$ where n_{cell} is the outward facing unit normal at the closest point on the cell surface and n_{vector} is the unit normal vector direction of bead motion; Upper right: projections of bead displacement through plane slices, where the gray outline is the cell in the initial configuration, the black outline is the cell in the final configuration, and bead displacements are color coded by direction compared to the cell surface; Lower left: comparison of bead displacements to their immediate neighbors (beads with high scores may be spurious); Lower center: bead displacement magnitude with respect to distance from cell surface; Lower right: three-dimensional plots of the cell in each configuration.

```
43  #out_folder_beads = 'Gel_bead_center_coords'
44  #out_folder = 'Post_proc_summary'
45  #info.set_out_folder_cell(out_folder_cell)
46  #info.set_out_folder_beads(out_folder_beads)
47  #info.set_out_folder(out_folder)
48
49  #info.cell_channel = 0 #CellBrite Red
50  #info.bead_channel = 1 #Green fluorescent beads
51  #info.cell_thresh = 1.0
52  #info.num_feat = 5
53  #info.num_nearest = 15
54  #info.buffer_cell_tracking = 0
55  #info.track_type = 2 # type 1 will NOT perform translation
        correction, type 2 will
56  #info.buffer_cell_translation = 30
57  #info.figtype_list = ['.png']
58  #info.plot_type = 6.0
59  #info.run_GP = False
60  #info.use_corrected_cell = True
61  #info.should_plot = True # toggles 3D plotting using PyVista
62  #info.print_progress = True # toggles all printing in algorithm
63
64  # section (5)
65  # run all steps of the program
66
67  info.run_tracking_all_steps(True,True,True)
```

Listing 1: Example implementation file.

Once the parameters are specified in `test.py`, we run the script. This file is shown in Listing 1. The script calls the FM-Track function "`run_tracking_all_steps()`", which runs the software on the specified data. The components of this function are as follows:

- Pre-process: This step takes the two-channel image stacks and outputs the information needed to run the tracking algorithm. The key steps are getting the bead center positions "`get_bead_centers()`", and getting the cell surface meshes "`get_cell_surface()`". The bead centers are identified by first applying a threshold to the fluorescent bead channel and then finding the center of volume of each connected voxel group in the binary image. The cell mesh is identified by first applying a threshold to binarize the fluorescent cell channel and then using the marching cubes algorithm to identify the surface mesh. A Gaussian filter is applied to both image channels before processing [32].

- Track: This step runs the tracking step "`track_main_call()`", where the result of running this step are text files that contain the x, y, and z coordinates of each bead in the first configuration, and text files that contain the displacement of each bead u, v, and w that show how it moves from the first configuration to the second configuration. The fundamental components of the tracking algorithm are explained in Section 2.2.
- Post-process: This step plots the raw data "`call_plot_main()`", with an option to generate a Gaussian Process Regression (GPR) model to interpolate bead displacements "`create_GP_model()`", and subsequently plot the GPR model "`plot_gp_model()`" (note: generating the Gaussian Process model may be slow and the saved model may take up a large amount of file space). An example of a summary sheet generated by the post processing script is shown in Fig. 4.

The results of running FM-Track (pre-processing, tracking, and post-processing) are shown in Figs. 3 and 4. Fig. 3a and b show the initial paired image stacks, visualized in ImageJ, and Fig. 3c shows a final three-dimensional plot in ParaView that shows how the fluorescent beads move between the two images. The color scheme in Fig. 3c is chosen such that beads moving towards the cell surface are yellow, while beads moving away from the cell surface are blue. Fig. 4 shows an example of an automatically generated post-processing summary sheet created with matplotlib [33]. Notably, the summary sheet shows bead movement in planar slices through the cell center, magnitude of bead displacement as a function of distance from the cell surface, and an illustration of the cell mesh in both the initial and final configuration.

4. Impact and conclusions

The main benefit of FM-Track is that it provides an open source and simple to implement feature based bead tracking algorithm in the Python framework. Users have the option to implement the code directly as specified in the working example, use it with altered input parameters, or build on the functions and framework presented here in a manner that is entirely specific to their own needs. Furthermore, the modular nature of this code allows it to be adapted to accommodate alternate types of fiducial markers, even potentially cell centers or nuclei [34,35]. Important novel aspects of our code that are, to the author's knowledge, not found elsewhere in the literature include: (1) minor aspects of the tracking algorithm (2) the MARS based translation correction function, and (3) the open source Python based framework. Our software is simple to use and modify, even for researchers with limited programming experience. Looking forward, we anticipate that FM-Track will enable research in Traction Force Microscopy and related techniques.

5. Further information

FM-Track software can be accessed at https://github.com/elejeune11/FM-Track. The GitHub repository contains the most up to date information on installing, running, and adapting FM-Track.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Experimental protocol used to generate the data used in Section 3

A.1. Sample preparation

Aortic heart valve leaflets were dissected from porcine hearts obtained from a local abattoir. Aortic valve interstitial cells (AVICs) were extracted from the leaflets via previously published methods [36]. AVICs were seeded at a density of 500,000 cells/mL along with 0.5 μm yellow–green fluorescent polystyrene microbeads (Polysciences) suspended at a density of 3×10^9 beads/mL within peptide-modified, poly (ethylene glycol) hydrogels comprised of MMP degradable peptide crosslinks (Bachem), CRGDS adhesive peptides (Bachem), 8-arm 40 kDa PEG-norbornene (Jenkem), lithium phenyl- 2, 4, 6 – trimethylbenzophosphinate (LAP) photoinitiator (Sigma-aldrich), and PBS (Sigma-aldrich). The solution was pipetted into a petri dish with a 12 mm glass insert (Fisher Scientific) and polymerized underneath UV light (365 nm, 2.5 mW/cm^2) for 3 min. After that, the hydrogel construct was incubated at standard conditions in growth media for 72 h before any experimentation was performed.

A.2. 3D traction force microscopy experiment

Image stacks (150 $\times$ 150 $\times$ 140 μm at 0.8 μm z-spacing) of a single AVIC and the immediately surrounding fluorescent microbeads were captured using a Zeiss LSM 710 inverted confocal microscope with a 63x, 1.2 numerical aperture water-immersion objective lens (Zeiss). Two total image stacks were taken for every cell under normal and non-contractile conditions. For the normal case, the sample was incubated for 40 min within Tryode's Salt Solution (TSS, Sigma Aldrich) before the first image stack was acquired. After that, a non-contractile state was induced by adding a stock solution of Cytochalsin D (dissolved in DMSO, Sigma Aldrich) to achieve a final concentration of 4 μM. A second image stack of the same FOV was acquired after 40 min of incubation.

Appendix B. Testing FM-track performance on synthetic data

B.1. Note on synthetic data

Synthetic data is generated by randomly identifying points in the reference configuration of finite element simulations. The initial fiducial marker locations are the randomly identified points in the reference configuration, and the final fiducial marker locations are the randomly identified points in the current configuration after some boundary conditions or loading has been applied in the finite element setting. Notably, we shuffle the order of the fiducial markers in the final configuration in our synthetic data set.
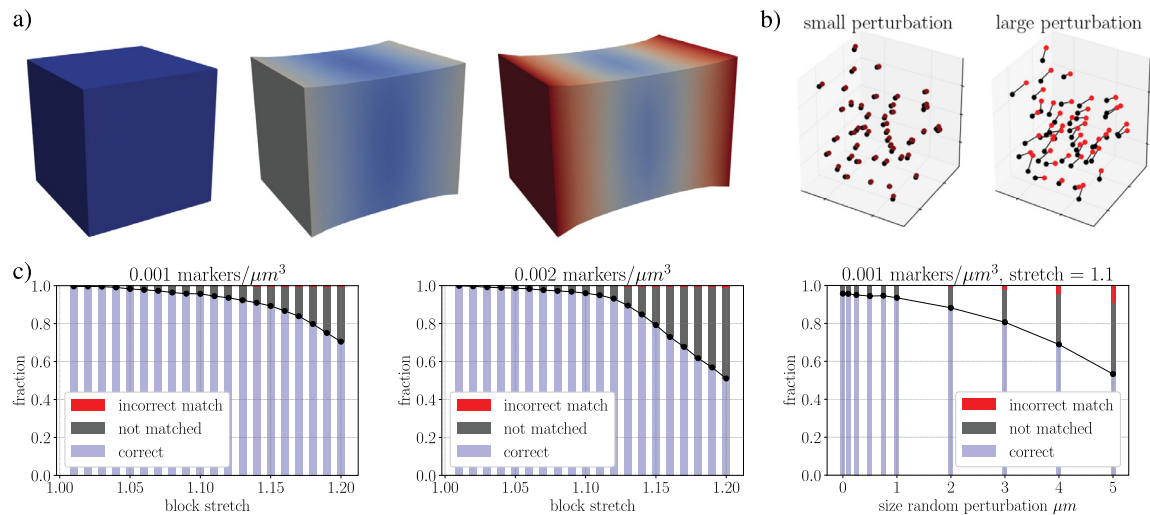
Fig. A.1. (a) Illustration of a finite element model of a $100\,\mu m \times 100\,\mu m \times 100\,\mu m$ block under uniaxial displacement controlled extension; (b) illustration of random perturbations to the bead positions, where the size of the random perturbation indicates the maximum perturbation in a single direction; (c) summary of FM-Track performance on this synthetic data set.

B.2. Performance on synthetic data

Here we show the performance of our tracking algorithm on a synthetic data set. Fig. A.1a shows the finite element simulation used to generate the synthetic data, and Fig. A.1b shows the method for adjusting the synthetic data set to contain noise. In Fig. A.1c, we show the performance of FM-Track at two different marker densities, and on a data set with progressively higher levels of random perturbation. Notably, FM-Track performs quite well on block stretch up to 1.10, and with small random perturbations, and when it fails it tends to fail by not matching beads rather than by reporting incorrect matches.

References

[1] Aydin O, Aksoy B, Akalin O, Bayraktar H, Alaca B. Time-resolved local strain tracking microscopy for cell mechanics. Rev Sci Instrum 2016;87(2):023905.

[2] Burnette DT, Shao L, Ott C, Pasapera AM, Fischer RS, Baird MA, Der Loughian C, Delanoe-Ayari H, Paszek MJ, Davidson MW, et al. A contractile and counterbalancing adhesion system controls the 3D shape of crawling cells. J Cell Biol 2014;205(1):83–96.

[3] Gjorevski N, Nelson CM. Mapping of mechanical strains and stresses around quiescent engineered three-dimensional epithelial tissues. Biophys J 2012;103(1):152–62.

[4] Roca-Cusachs P, Conte V, Trepat X. Quantifying forces in cell biology. Nature Cell Biol 2017;19(7):742.

[5] Cirka H, Monterosso M, Diamantides N, Favreau J, Wen Q, Billiar K. Active traction force response to long-term cyclic stretch is dependent on cell pre-stress. Biophys J 2016;110(8):1845–57.

[6] Style RW, Boltyanskiy R, German GK, Hyland C, MacMinn CW, Mertz AF, Wilen LA, Xu Y, Dufresne ER. Traction force microscopy in physics and biology. Soft Matter 2014;10:4047–55.

[7] Dembo M, Wang Y-L. Stresses at the cell-to-substrate interface during locomotion of fibroblasts. Biophys J 1999;76(4):2307–16.

[8] Butler JP, Tolić-Nørrelykke IM, Fabry B, Fredberg JJ. Traction fields, moments, and strain energy that cells exert on their surroundings. Am J Physiol Cell Physiol 2002;282(3):C595–605.

[9] Tan JL, Tien J, Pirone DM, Gray DS, Bhadriraju K, Chen CS. Cells lying on a bed of microneedles: An approach to isolate mechanical force. Proc Natl Acad Sci 2003;100(4):1484–9.

[10] Legant WR, Miller JS, Blakely BL, Cohen DM, Genin GM, Chen CS. Measurement of mechanical tractions exerted by cells in three-dimensional matrices. Nature Methods 2010;7(12):969.

[11] Hall MS, Long R, Feng X, Huang Y, Hui C-Y, Wu M. Toward single cell traction microscopy within 3D collagen matrices. Exp Cell Res 2013;319(16):2396–408.

[12] Jaqaman K, Loerke D, Mettlen M, Kuwata H, Grinstein S, Schmid SL, Danuser G. Robust single-particle tracking in live-cell time-lapse sequences. Nature Methods 2008;5(8):695.

[13] Koch TM, Münster S, Bonakdar N, Butler JP, Fabry B. 3D traction forces in cancer cell invasion. PLoS One 2012;7(3). e33476.

[14] Legant WR, Choi CK, Miller JS, Shao L, Gao L, Betzig E, Chen CS. Multidimensional traction force microscopy reveals out-of-plane rotational moments about focal adhesions. Proc Natl Acad Sci 2013;110(3):881–6.

[15] Steinwachs J, Metzner C, Skodzek K, Lang N, Thievessen I, Mark C, Münster S, Aifantis KE, Fabry B. Three-dimensional force microscopy of cells in biopolymer networks. Nature Methods 2016;13(2):171.

[16] Hall MS, Alisafaei F, Ban E, Feng X, Hui C-Y, Shenoy VB, Wu M. Fibrous nonlinear elasticity enables positive mechanical feedback between cells and ecms. Proc Natl Acad Sci 2016;113(49):14043–8.

[17] Cheezum MK, Walker WF, Guilford WH. Quantitative comparison of algorithms for tracking single fluorescent particles. Biophys J 2001;81(4):2378–88.

[18] Feng X, Hall MS, Wu M, Hui C-Y. An adaptive algorithm for tracking 3D bead displacements: application in biological experiments. Meas Sci Technol 2014;25(5):055701.

[19] Song D, Hugenberg N, Oberai AA. Three-dimensional traction microscopy with a fiber-based constitutive model. Comput Methods Appl Mech Engrg 2019;357:112579.

[20] Colin-York HE, Javanmardi Y, Barbieri L, Li D, Korobchevskaya K, Guo Y, Hall C, Taylor A, Khuon S, Sheridan G, et al. Spatiotemporally super-resolved volumetric traction force microscopy. Nano Lett 2019.

[21] Chenouard N, Smal I, De Chaumont F, Maška M, Sbalzarini IF, Gong Y, Cardinale J, Carthel C, Coraluppi S, Winter M, Cohen A. Objective comparison of particle tracking methods. Nature Methods 2014;11(3):281.

[22] Meijering E, Dzyubachyk O, Smal I. Methods for cell and particle tracking. In: Methods in enzymology, Vol. 504. Elsevier; 2012, p. 183–200.

[23] Patel M, Leggett SE, Landauer AK, Wong IY, Franck C. Rapid, topology-based particle tracking for high-resolution measurements of large complex 3D motion fields. Sci Rep 2018;8(1):5581.

[24] Liu S-L, Li J, Zhang Z-L, Wang Z-G, Tian Z-Q, Wang G-P, Pang D-W. Fast and high-accuracy localization for three-dimensional single-particle tracking. Sci Rep 2013;3:2462.

[25] Boltyanskiy R, Merrill JW, Dufresne ER. Tracking particles with large displacements using energy minimization. Soft Matter 2017;13(11):2201–6.

[26] Pereira F, Stüer H, Graff EC, Gharib M. Two-frame 3D particle tracking. Meas Sci Technol 2006;17(7):1680.

[27] Polacheck WJ, Chen CS. Measuring cell-generated forces: a guide to the available tools. Nature Methods 2016;13(5):415.

[28] Tinevez J-Y, Perry N, Schindelin J, Hoopes GM, Reynolds GD, Laplantine E, Bednarek SY, Shorte SL, Eliceiri KW. Trackmate: An open and extensible platform for single-particle tracking. Methods 2017;115:80–90.

[29] Rasmussen CE. Gaussian processes in machine learning. In: Summer school on machine learning. Springer; 2003, p. 63–71.

[30] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E. Scikit-learn: Machine learning in Python. J Mach Learn Res 2011;12:2825–30.

[31] Khang A, Rodriguez AG, Schroeder ME, Sansom J, Lejeune E, Anseth KS, Sacks MS. Quantifying heart valve interstitial cell contractile state using highly tunable poly (ethylene glycol) hydrogels. Acta Biomater 2019;96:354–67.

[32] van der Walt S, Schönberger JL, Nunez-Iglesias J, Boulogne F, Warner JD, Yager N, Gouillart E, Yu T, the scikit-image contributors. Scikit-image: image processing in Python. PeerJ 2014;2. e453.

[33] Hunter JD. Matplotlib: A 2D graphics environment. Comput Sci Eng 2007;9(3):90–5.

[34] Lejeune E, Linder C. Quantifying the relationship between cell division angle and morphogenesis through computational modeling. J Theoret Biol 2017;418:1–7.

[35] Lejeune E, Linder C. Understanding the relationship between cell death and tissue shrinkage via a stochastic agent-based model. J Biomech 2018;73:9–17.

[36] Johnson CM, Hanson MN, Helgeson SC. Porcine cardiac valvular subendothelial cells in culture: cell isolation and growth characteristics. J Mol Cell Cardiol 1987;19(12):1185–93.