Getting started with the p53Cinema single cell analysis package

José Reyes*, Kyle Karhohs*, Galit Lahav
Deparment of Systems Biology, Harvard Medical School, Boston USA
* These authors contributed equally to this work
Correspondence to: jreyes.lab@gmail.com, Galit@hms.harvard.edu

Introduction

p53Cinema is a software package for semiautomatic tracking of individual cells in live cell imaging datasets. We developed the software while simultaneously using it to learn about the biology of individual cells. Thus, many of its features were tailored to address our interests as biologists. The software is useful to track a wide variety of cell lines, including fast moving cells. Further, we envision that it will prove valuable to generate gold-standard datasets to test and improve the accuracy of automated tracking algorithms. We hope that this is helpful for your work! If you use this for your research, please cite the paper where we first introduced this tracking method:

Reyes, J., Chen, J.Y., Stewart-Ornstein, J., Karhohs, K.W., Mock, C.S. and Lahav, G. Fluctuations in p53 signaling allow escape from Cell-Cycle arrest. *Molecular Cell*, 2018 71(4), pp.581-591.

0. System requirements

- Matlab 2013 or higher.
- Matlab image analysis toolbox.
- Imaging data. The software is designed to process grayscale TIF files.
- The panel package from Matlab exchange (only for plotting cellular genealogies).

Download the latest p53Cinema release from Github: https://github.com/balvahal/p53CinemaManual/releases

Add the p53Cinema folder to your Matlab path (add with subfolders). In Matlab 2013-2018 you can go to the 'HOME' tab and 'Set path' options. In doing so, Matlab will be able to locate the p53CinemaManual scripts regardless of which directory you are currently working on.

1. Suggested file directory structure

The p53Cinema will accommodate a wide variety of directory structures. One strategy that has worked for us is to work within an experiment folder, with subdirectories for the raw image files and tracking files:

```
Experiment folder
       Subdirectory: RAW_DATA
       Subdirectory: Tracking
```

IMPORTANT NOTE: The quantification scripts iterate over all .mat files in a specified directory. Having a directory exclusively dedicated to contain tracking files will prevent potential errors/duplications during data quantification.

When you start Matlab, it is convenient to change directory to the experiment folder. For example:

```
> cd 'O:\\Sysbio\LahavLab\Microscopy\M20181020_p53DynamicsMovie\';
```

Or use the Matlab browser to manually select the directory.

2. Creating a database file

A database file is a tab delimited file linking image filenames with their metadata. The database file requires the following columns (the order is not important): filename, group_label, channel_name, position_number, timepoint.

Normally it is possible to extract all the relevant information from the filename. Use the function createDatabaseFile as a starting point to generate this table.

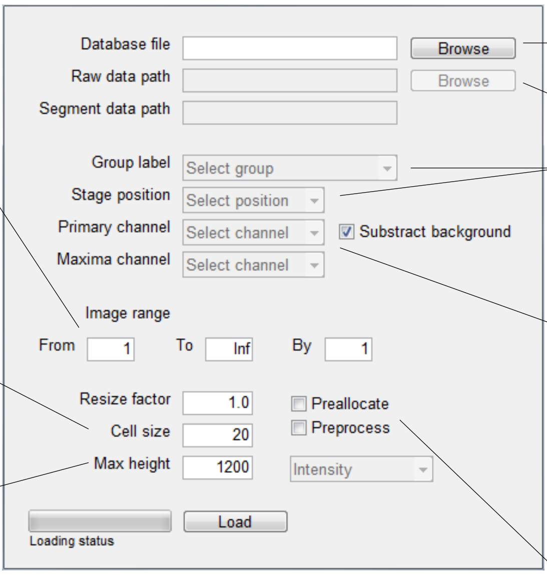
```
> database = createDatabaseFile(<rawdatapath, String>, <outputFile, String>);
% For example:
> database = createDatabaseFile('RAW_DATA', 'M20181020_database.txt');
```

3. Open the tracking software

Use the following command to open the software in Matlab

```
> master = p53CinemaManual_object_master;
```

This will open the *fileManager* window


Database file [ ] Browse
Raw data path [ ] Browse
Segment data path [ ]
Group label: Select group
Stage position: Select position
Primary channel: Select channel
☑ Substract background
Maxima channel: Select channel
Image range
From 1 To Inf By 1
Resize factor 1.0
☐ Preallocate
Cell size 20
☐ Preprocess
Max height 1200
Intensity
Load
Loading status

Give the program your database file.

Select the RAW_DATA directory

Choose ***group*** and ***stage position***

You can load only a subset of images if you will, or load all images if you set the 'To' field to Inf. This is useful to test how well the software is identifying cells. You can also partially track datasets, quantify data and complete cell tracks later.

The ***cell size*** parameter is the size of the window used to blur images to identify single local maxima associated with individual cells. A higher number will identify fewer local maxima per cell, but it may fail to distinguish neighboring cells.

Change this number to match the size of your monitor.

Primary channel: This is the channel that you will *visualize* while tracking. Background substraction is recommended unless you are visualizing brightfield images.

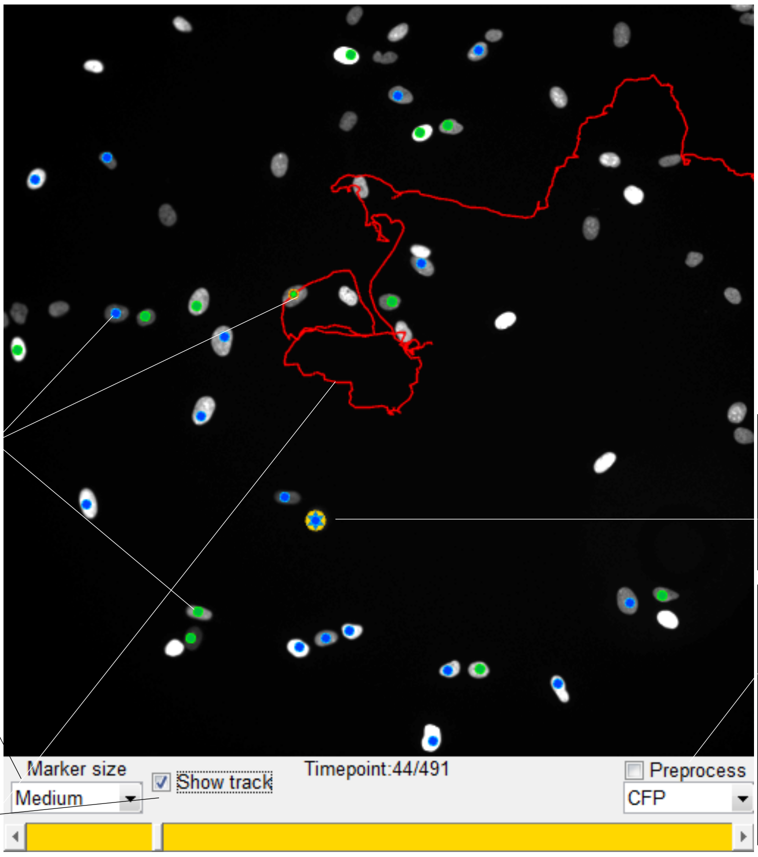
Maxima channel: This is the channel that the software uses to *predict* the location of individual cells. In most cases this will be the same as the primary channel (nuclear marker).

You should check ***preallocate*** and ***preprocess*** to get the most out of the software. In particular, preprocess will use the 'Maxima channel' to identify cells in each images by identifying local maxima in images in based on intensity or shape criteria.

Click ***Load*** to start tracking

This will open four windows:

A. imageViewer



Centroids are annotated as colored circles. The red centroid is the current cell. Green centroids are cells present at the start and end of the image sequence. Blue centroids are absent in either the beginning/end of the image sequence (incomplete tracks)

You can change the size of the circles using this drop menu.

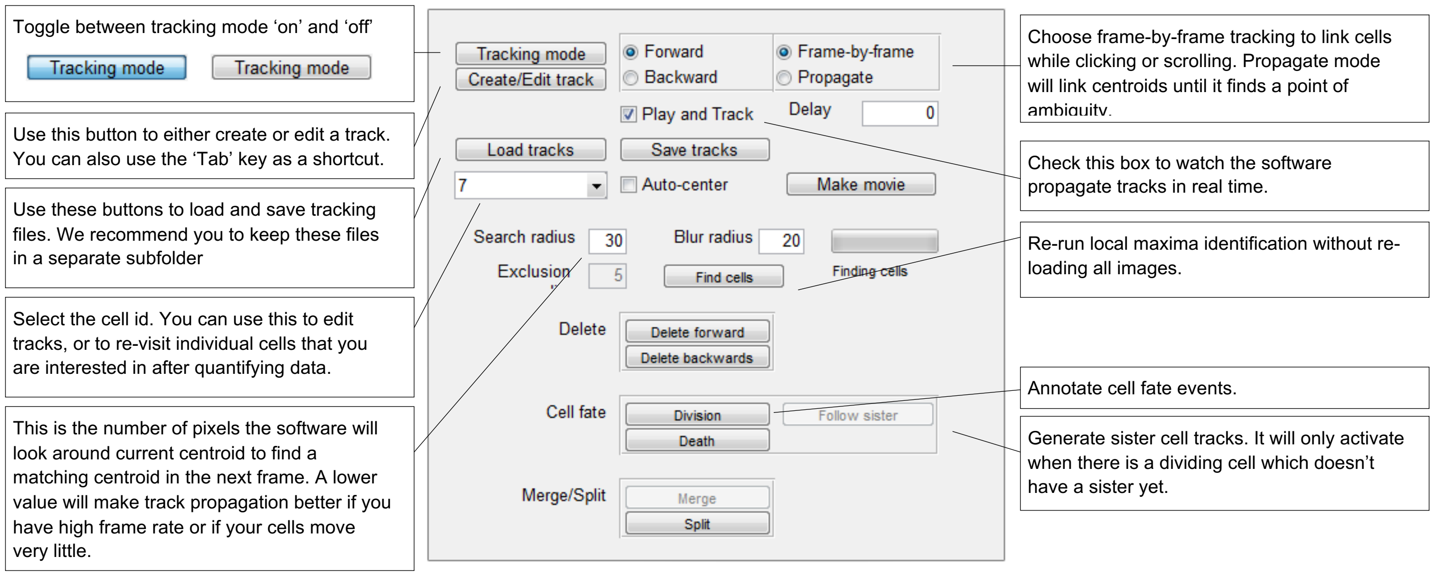
You can choose to show the full track of the current cell (red) or not.

Cell fate events are annotated as rings surrounding centroids. Yellow denotes division events. If there is a blue star, it represents that two cells divide at the same centroid (linked sisters).

Change the current channel to visualize. Note that because it is not preallocated, the software will read images one by one, and thus it is rather slow. Sometimes looking at other channels is helpful to solve ambiguities in tracking.

Check 'preprocess' to substract background in this alternate channel.

B. cellTracker



Toggle between tracking mode 'on' and 'off'

Use this button to either create or edit a track. You can also use the 'Tab' key as a shortcut.

Use these buttons to load and save tracking files. We recommend you to keep these files in a separate subfolder

Select the cell id. You can use this to edit tracks, or to re-visit individual cells that you are interested in after quantifying data.

This is the number of pixels the software will look around current centroid to find a matching centroid in the next frame. A lower value will make track propagation better if you have high frame rate or if your cells move very little.

Choose frame-by-frame tracking to link cells while clicking or scrolling. Propagate mode will link centroids until it finds a point of ambiguity.

Check this box to watch the software propagate tracks in real time.

Re-run local maxima identification without re-loading all images.

Annotate cell fate events.

Generate sister cell tracks. It will only activate when there is a dividing cell which doesn't have a sister yet.

C. contrast

Move sliders to adjust the range of pixel intensities in your image.

D. zoomMap

Use the '+' and '-' keys to zoom in and out. You can also pan by clicking and dragging the highlighted region in the zoomMap.

3. Start tracking

How do you create and edit tracks?

Use tracking mode toggle button to open and close individual tracks.

1. Activate 'tracking mode' and click a cell that doesn't have a centroid to open a new track.
2. Track this cell (described in detail below).
3. While on 'tracking mode', you can track a new cell or edit a pre-existing track by clicking the 'Create/Edit track' button. If you click a cell without a centroid, you will create a new track. If you click a cell with a centroid, you will select its track for editing. You can use the 'Tab' key as a shortcut for create/editing tracks.
4. Deactivate the 'tracking mode' to finish this tracking session. The next time you activate it, it will trigger a 'create/edit' event by default (depending on which cell you click).

How does tracking work?

If you checked the 'preprocess' box in the fileManager, p53Cinema will define a set of candidate centroids in each image, which you can use to track. As you hover around cells, you will notice that a small green circle will get attracted to local maxima (as if cells were magnets). Once you 'left click', the centroid will be recorded at the location of such green dot.

When p53Cinema fails to identify a centroid, you can manually impute it using the mouse right click. This is common when there are differences in intensities in nearby cells, and it can be alleviated by choosing a smaller 'cell size' in the fileManager, at the expense of potentially more centroids within individual nuclei.

Every time you click the software predicts where the cell is in the next image using a nearest neighbor criterion. The software also automatically links centroids when you scroll through the image sequence or when you use the keyboard shortcuts '>' and '<'.

If the centroid linkage is working well, you can propagate tracks without the need of manual scrolling. The software will stop when there is a point of ambiguity in the tracking, in which case you can indicate which the correct centroid is and continue tracking. If you uncheck 'Play and Track', the software will automatically construct the track and jump to the point of ambiguity. In some cases, this can save a lot of time.

How do you delete centroids and correct mistakes?

Click the ‘backspace’ button to delete a centroid. The track should switch from red to green color. This means that the track exists, but there is no longer a centroid for such track in the current frame. If you click ‘backspace’ while a track is green, you will delete the whole track (think of a centroid in a frame as a shield for the track). You can also use the ‘delete forward’ and ‘delete backwards’ buttons to delete all centroids from a certain frame all the way to the end/beginning of the image sequence.

How do you annotate cell fate events?

To annotate division events, click the ‘division’ button or the ‘space’ bar. To annotate death events, click the ‘death’ button or the ‘Ctrl’ key. You can use the ‘up’ and ‘down’ keys to jump between division events. Use the ‘track sister’ button to start a new track from a division event and link sister cells. Linked sisters will show a yellow circle with a blue star.

How do you annotate cell fate events?

Use the ‘save tracks’ button to save current tracks. The software does NOT save tracks by default, so be sure to save them before you close the GUI. Use the ‘load tracks’ button to load them into a new tracking session.

Note about keyboard shortcuts

The use of keyboard shortcuts requires that the imageViewer is the active figure. Clicking a button/checkbox/slidebar will make keyboard shortcuts irresponsive. To use them again, just click a centroid in the imageViewer and continue tracking.

4. Quantify data

*Obtain single cell tracks by sampling pixels around centroids (**recommended to get started**)*

This method will:

1. Read images corresponding to a particular channel and substract background, for each tracked position and timepoints.
2. Sample an area around centroids as defined by the blurRadius parameter to quantify the intensity of individual cells.
3. Aggregate information from all tracking files in single cell matrices of dimensions NxM. N is the number of single cells in the dataset and M is the number of timepoints in the movie.
4. Generate single cell matrices for division events and death events.
5. Consolidate information from linked sisters so that both sisters contain intensity and cell fate information from mother tracks.

Missing data is has a value of -1. This method measures only one channel at a time. You have to run it independently to measure every channel in your dataset.

```
> database = readDatabaseFile(<database_filepath, String>);
> measurements = getDatasetTraces_fillLineageInformation(<database, MatlabTable>, ...
      <rawdata_path, String>, <tracking_path, String>, <flatfield_path, String>, ...
      <channel_name, String>, <blurRadius, Integer>);
% For example:
> database = readDatabaseFile(‘M20181020_database.txt’);
> measurements = getDatasetTraces_fillLineageInformation(database, ...
      ‘RAW_DATA’, ‘tracking’, ‘’, ‘YFP’, 7);
```

You can access single cell matrices using:

```
> traces_YFP = measurements.filledSingleCellTraces;
> divisions = measurements.filledDivisionMatrixDataset;
> annotation = measurements.cellAnnotation;
```

Obtain single cell tracks by locally segmenting cells

This method conducts a local segmentation in a segmentationChannel and quantifies multiple parameters for each individual cell in a collection of measurementChannels.

```
> database = readDatabaseFile(<database_filepath, String>);
> measurements = getDatasetTraces_localSegmentation(<database, MatlabTable>, ...
      <rawdata_path, String>, <tracking_path, String>, <flatfield_path, String>, ...
      <measurementChannels, cell of strings>, <segmentationChannel, String>);
% For example:
> database = readDatabaseFile('M20181020_database.txt');
> measurements = getDatasetTraces_fillLineageInformation(database, ...
      'RAW_DATA', 'tracking', '', {'YFP', 'CFP', 'Cy5'}, 'CFP');
```

You can access single cell matrices using:

```
> traces_YFP_brightestPixels = measurements.singleCellTracks_foci{1};
> traces_YFP_mean = measurements.singleCellTracks_mean{1};
> traces_Cy5_mean = measurements.singleCellTracks_mean{3};
> traces_area = measurements.singleCellTracks_area;
> traces_solidity = measurements.singleCellTracks_solidity;
```

Obtain only cell fate event tracks (much faster)

If you are only interested in division/death events, you can ignore intensity quantification using:

```
> database = readDatabaseFile(<database_filepath, String>);
> measurements = getDatasetTraces_ignoreTraces(<database, MatlabTable>, ...
      <rawdata_path, String>, <tracking_path, String>, <flatfield_path, String>, ...
      <channel_name, String>);
% For example:
> database = readDatabaseFile('M20181020_database.txt');
> measurements = getDatasetTraces_fillLineageInformation(database, ...
      'RAW_DATA', 'tracking', '', 'YFP');
```

You can access single cell matrices using:

```
> divisions = measurements.filledDivisionMatrixDataset;
> annotation = measurements.cellAnnotation;
> centroid_col = measurements.centroid_col;
```

Note that since you don't have intensity quantification, you can rely on the centroids matrix to know whether a cell was tracked in a particular timepoint (centroid_col(i,t) > 0) or not (centroid_col(i,t)= 0).

5. Data analysis

Extract single cell matrices

```
> traces_YFP = measurements.filledSingleCellTraces;
> divisions = measurements.filledDivisionMatrixDataset;
> annotation = measurements.cellAnnotation;
> lineageTree = measurements.lineageTree;
```

Get number of individual cell tracks and timepoints

```
> numberOfCells = size(traces_YFP,1);
> numberOfTimepoints = size(traces_YFP,2);
```

Filter out incomplete single cell tracks

```
> validCells = sum(traces_YFP == -1,2) == 0;
> traces_YFP = traces_YFP(validCells,:);
> divisions = divisions(validCells,:);
> annotation = annotation(validCells,:);
> lineageTree = lineageTree(validCells,:);
```

Count number of divisions per cell

```
> numDivisions = sum(divisions,2);
```

Extract group_label and position_number for each cell

```
> group_label = annotation(:,1);
> position_number = [annotation{:,2}];
> cell_id = [annotation{:,3}];
> [~, group_number] = ismember(group_label, unique(group_label))
```

Get timing of cell divisions and cell cycle durations (assuming cells divided more than once)

```
> divisionTiming = getDivisionTiming(divisions)
> cellCycleDuration = divisionTiming(:,2) - divisionTiming(:,1)
```

Sort cells based on average YFP signal

```
[~, ordering] = sort(traces_YFP,2)
> traces_YFP = traces_YFP(ordering,:);
> divisions = divisions(ordering,:);
> annotation = annotation(ordering,:);
> lineageTree = lineageTree(validCells,:);

> numDivisions = annotation(ordering,:);
> group_label = annotation(ordering,:);
> cell_id = annotation(ordering,:);
> group_number = annotation(ordering,:);
> divisionTiming = annotation(ordering,:);
> cellCycleDuration = annotation(ordering,:);
```

NOTE: every time you sort or filter a matrix, you should apply that ordering/filter to every matrix/array containing information from single cells in order to maintain the relationship between indexes of different matrices.

Manage lineage information

```
% First column of lineageTree has the progenitor ids
> progenitors = unique(lineageTree(:,1))
> progenitors = progenitors(progenitors > 0)

% Find all cells associated with the first progenitor
> family1 = find(progenitors == progenitors(1))

% Find sublineages of first progenitor after first division
% After division, one sublineage inherits the id of the mother,
% the second sublineage gets a new id
firstDivisionTiming = divisionTiming(family1,1);
firstDivisionTiming = firstDivisionTiming(1);
subfamilies = lineageTree(family1, firstDivisionTiming + 1);
[~,subLineage_id] = ismember(subfamilies, unique(subfamilies));
subLineage1 = family1(subLineage_id == 1);
subLineage2 = family1(subLineage_id == 2);
```

6. Advanced tips

The database file allows you to annotate and organize your files without modifying filenames:

Tip 1. Use the group_label field to define different treatments.

```
> database = readDatabaseFile('M20181020_database.txt');
> database.group_label(ismember(database.position_number, 1:10)) = {'non_treated'};
> database.group_label(ismember(database.position_number, 11:20)) = {'irradiated'};
> writetable(database, 'M20181020_database.txt', 'Delimiter', '\t');
```

Tip 2. Merge two parts of an interrupted experiment.

```
> database = readDatabaseFile('M20181020_database.txt');
> part1 = strcmp(database.group_label, 'M20181020_part1');
> part2 = strcmp(database.group_label, 'M20181020_part2');
> database.timepoint(part2) = database.timepoint(part2) + ...
      max(database.timepoint(part1));
```

Tip 2. Link movie with immunofluorescence data, by merging timepoints from two datasets and consolidating nuclear marker channel names.

```
> database = readDatabaseFile('M20181020_database.txt');
> part1 = strcmp(database.group_label, 'Movie_20181020');
> part2 = strcmp(database.group_label, 'immunofluorescence_20181021');
> database.timepoint(part2) = database.timepoint(part2) + ...
      max(database.timepoint(part1));
> database.channel_name(strcmp(database.channel_name, 'DAPI')) = {'CFP'};
```

Bridge p53Cinema with other software
You can use the centroidsBackup script to convert .mat tracking files into tab-delimited files. This is particularly useful to interface with other packages or to integrate tracking with your custom scripts:

```
> centroidsBackup(<rawdata_path, String>);
% For example:
> centroidsBackup('tracking');
```

p53Cinema will also load tab-delimited centroids with the following columns:

```
cell_id
centroid_col
centroid_row
timepoint
division (0 or 1)
death (0 or 1)
value (it is ok if this is a column of 1)
```