Software

The software described in this publication was written in Python. It consists of four files:

* CardioMotion.py
* CardioMotion\_image\_loader.py
* CardioMotion\_signal\_extractor.py
* CardioMotion\_peakstats.py

The file CardioMotion.py can be run as a command-line tool. It accepts as input a directory containing microscope output images, and outputs both the signals generated, and the peak statistics calculated.

Environment

This section describes how to set up an environment in which the software can be run. Any python 3 distribution with the following modules installed is required:

* numpy
* pandas
* cv2 (OpenCV)
* lxml
* scipy
* matplotlib (Optional, only used if --graph option is selected)

Detailed set-up procedure

1. Install Anaconda for Python 3

This can be found at https://www.anaconda.com/distribution/#download-section. Choose the correct

operating system and then download and install the Python 3 version.

2. Set up conda environment

Open the Anaconda Prompt (should be in the start menu). Enter the command

conda --version

to check that conda has been installed. This code was tested with conda versions 4.7.5, 4.7.8 and 4.7.10.

Enter the command

conda env list

to list the existing conda environments. If this is a fresh install then there should be exactly one, named base. We create a clone of this environment to install packages into:

conda create --name scvt --clone base

where scvt can be replaced by any desired environment name. Activate the environment with

conda activate scvt

The software needs to be run from within this environment - the above command can be run in the future

from the Anaconda Prompt to re-enter this environment.

3. Install packages in environment

List the existing packages with

conda list

and make sure that the list contains lxml, numpy, pandas and scipy (these should all be installed by default).

Install OpenCV with

conda install -c conda-forge opencv

If this doesn't work because of an UnsatisfiableError then use

pip install opencv-python

instead.

Required script modifications

The script was developed for image files acquired on the Yokogawa CV8000, with the file naming format platename\_Well\_T0298F001L01A01Z01C01.tif

and metadata file (.mes) as produced by the CV8000.

As the script reads both file names and the metadata file, portions of the script must be modified for other imagers / filename conventions.

The sections of the CardioMotion and image\_loader scripts which require modification are annotated by comments within the scripts. These required modifications are as follows:

* Modification of the regular expression defining file names in the image\_loader script. Where this is edited, ensure the find\_images function is updated.
* Either:
  + Re-writing of the image\_loader script for non-CV8000 metadata files

OR

* + Adjusting the default values in the CardioMotion script for fps and scale (these will be used when no .mes file is found).
* Adjust default values in the CardioMotion script for min\_prominence, min\_width and cutoff. For definition of these parameters, use

python CardioMotion.py –help

or alternatively definitions can be read within the CardioMotion script. These values can also be defined within the arguments when running the script – see USAGE section below.

Usage

Assuming that the Python environment is set up correctly, the scripts have been modified as necessary and the source code files are in the working directory, the software can now be run. To see the available command-line options, execute

python CardioMotion.py --help

The only necessary argument is the path to a directory containing a sequence of microscope output imagefiles that match the regular expression used, and metadata file, if the script has been modified to read from this.

python CardioMotion.py /path/to/microscope/output/

e.g:

python CardioMotion.py "C:\Users\Name\Path\To\Microscope\Output\"

The program will first produce signals, and then extract median peak statistics. By default, the results are saved to a subdirectory named CardioMotion\_Data of the source directory, but a different location can be specifed by passing a second directory path, e.g.

python CardioMotion.py /path/to/microscope/output/ /path/to/data/destination/

The output is in .csv format, and the program will log its progress to both the command line and a log file.

The program calculates the length scale (nanometres per pixel) using information in the .mes file, and this can

be overridden manually by including the --scale argument. The fps is also read from the .mes file, and it can be

overridden by including the --fps argument.

python CardioMotion.py --scale 650 /path/to/microscope/output/

python CardioMotion.py --fps 40 /path/to/microscope/output/

The parameters used to determine the peak statistics can also be manually adjusted. These are

**--prominence** Minimum acceptable peak height in nanometres - default is 65

**--width** Minimum acceptable peak width in seconds - default is 0.2

**--cutoff** Proportion of the peak height at which to measure the width - default is 0.25

The --graph arg can also be included to display a plot of each signal with the located peaks marked on - this is

useful for checking the effectiveness of the peak finding algorithm and possibly motivating changing the parameters. Note that this will halt the script at each graph produced.

If the signal data is already present in the output directory then it will not be overwritten (the program will log

a warning). This means that the program can be interrupted and then restarted without having to reprocess every video. It also means that after completion, if the peak detection parameters need changing then the program can be run again and it will only perform the peak finding stage. The program will log exactly which part of the process it is performing at each point in time.