Laser Tweezer Lab
Analysis Manual for Brownian Motion

The following document describes how to evaluate data on Brownian motion for the laser tweezer lab. The following steps are needed to be made, and will be described in details.

1. Record the motion of beads with the camera.
2. Make sure all necessary programs are installed on the computer you want to perform the analysis
3. Copy the video files to the computer where you would like to analyze them.
4. Convert the video file to frame files readable by the BrownianApplication analysis software.
5. Use BrownianApplication to track particles and save the particle trajectories.
6. Calculate the diffusion coefficient and draw graphs

---

1. Record Brownian motion with the camera.

Pay attention that the liquid in which the beads are is not drifting, therefore you keep the beads moving only due to Brownian motion.

2. Make sure the necessary programs are installed on the computer you want to perform the analysis.

Softwares are installed on the computer in the corner of room 607. You can use this computer for the analysis. If you want to install some of the programs on another computer, you can find the list of necessary softwares below, together with the location where you can get them from.

1.BrownianApplication:
http://www.advancedlab.org/mediawiki/images/3/3d/BMC_Executable.zip
2. Microsoft .NET Framework:
http://www.microsoft.com/Net/Download.aspx
3. FFmpeg:
Windows: http://sf.net/project/showfiles.php?group_id=205275&package_id=248632
Mac: http://ffmpegX.com/download.html
4. Python 3.0:
http://www.python.org/download/
5. Matlab:
http://www.columbia.edu/acis/software/licenses/matlab/index.html

3. Copy the video files to the computer where you would like to analyze them.

The video files recorded by the DVD recorder have the extension ".vob".

4. Convert the video file to frame files readable by BrownianApplication.

The BrownianApplication analysis software requires image files as input, together with a frame file that contains the names of all images files and their time. The conversion from video file to image files can be done using FFmpeg.

A python script ('**Video2FrameFiles.py**') is available that does the conversion, creating the format BrownianApplication requires. The script can be run using the following arguments

```
C:\> python Video2FrameFiles.py video_file_name output_folder_name
image_file_format start_time duration
```

The arguments are the following:
video_file_name: name of the video file that will be converted to images. Make sure NOT to put a '\' at the end of the folder name.
output_folder_name: the program will create a new folder with this name and will write the frame and image files here. DO NOT specify the path to the folder, it will be placed into the C:\BMC Data\ folder.
image_file_format: format of the image files (png, jpg, bmp, etc).
start_time: start time of the converted part of the video. Has to be given in hh:mm:ss format (e.g. 01:12:35).
duration: duration of the converted part of the video. Has to be given in hh:mm:ss format (e.g. 00:00:15).
Example:
C:\> python Video2FrameFiles.py C:\\video01.mpg Video01 png 01:12:35 00:00:15

To facilitate running the script multiple times, we recommend the usage of a batch file that can be rerun again. Typing in the above command into the batch file instead of the command line will make it easier to run similar commands. An example batch file is provided ('**start.bat**'), one can start with editing this.

5. Use Brownian Application to track particles and save the particle trajectories.

Run BrownianApplication.exe. Three windows will appear: 'PassThru Window', 'Images Window' and 'Control Window'. You will only need the latter two, make sure you can see both of them on the screen at the same time.

Take a look at the control window. First you have to load the image files:

1. Specify the directory where you have the image files. '**C:>BMC Data**' is the path to the directory that contains the files (the python script automatically creates the folder here). The folder name is what you specified for the python script (output_folder_name). Write this below 'C:>BMC Data' (the default directory name is '**Onion**').
2. Load the image files: press the '**Load Saved Files**' button. If the program successfully load the files, it will tell you at the bottom of the Control Window, saying 'Loaded Frames from C:\BMC Data\output_folder_name.'

You can start and stop showing the images as a video in the Images Window by pressing the button '**File Reader Go**'. When the program showed all the loaded images, it stops and you have to reload the images to continue (the same way as you loaded them at first – press the button 'Load Saved Files').

Second, you have to set up the program to track the particles. Check the boxes '**Persist Tracks**', '**Track Particles**' and '**Find Particles**'. If you load the images and press 'File Reader Go', you can see that, in the 'PostProcessing Image' frame of 'Images Window', particles that are identified by the program are signed with a little blue or red circle that follows them. In the 'PostProcessing Image' frame of 'Images Window', particle tracks are shown for particles that are identified. Tracking disappears when the program loses the particle, therefore you can see long track only for particles that are seen by the program for long (throughout several images).

To help the program better identify and track particles, you can modify four variables on the Control Window:

1. '**Min Particle Diameter**': the minimum diameter of particles being tracked.
2. '**Max Particle Diameter**': the maximum diameter of particles being tracked.
3. '**Tracking Reach (max metric)**': maximum number of particles being tracked.
4. '**ZScore Threshold**': minimum brightness of particles being tracked.

Since the effect pf the above variables to tracking is not trivial, you should play around with them and see when you seem to track particles the most accurately. As a rule of thumb, 'Max Particle Diameter' and 'Tracking Reach' can usually be set to a large value (~100), while the system is more sensitive to 'Min Particle Diameter' and 'ZScore Threshold'. Their values usually need to be slightly increased, but this will depend on the specific properties of the images.
After you feel like you optimized particle tracking with the above parameters, you can start recording data. But before you do so, the program has to know how large the image is that you have loaded, so it can correctly record the distances. Use the '**Pixels to Meters Conversion**' box that has a default value 1, and type in the correct pixels to meter conversion value. You can easily calculate this value by taking an image of beads with known sizes.

Record data. It will be saved to the 'MyDocuments' folder. You can specify the file name in one of the boxes in Control Window, the default name is '**Particles.txt**'. After you specify the file name, you need to initialize the measurement. Press the button '**Set Start Time**' that will reset the buffer and only data recorded after this point will be saved. Load data again and start recording (particle tracks are being recorded while they are shown on the Images Window, therefore you will need to use the buttons 'Load Saved Files' and 'File Reader Go'). To save the recorded data into file, you need to press the button '**Save Particle Data**'. Upon successful saving, the program will give the message
'Particle Tracks Saved to C:\Documents and Settings\User\My Documents\Particles.txt'
on the bottom of Control Window.

6. Calculate diffusion coefficient and draw graphs

Use the output of BrownianApplication to calculate the diffusion coefficient and draw the necessary graphs for the lab report. You can write your own code to do it, or there are available Matlab scripts to do this. The script called '**BrownianAnalysis.m**' reads in the data from 'Particles.txt', calculates the average diffusion coefficient of beads, and draws two graphs: the mean displacement square of beads as a function of time (draws the average with red and also draws the displacement square of each bead with green) and a bull's eye plot. If needed, you can modify the code or use subroutines that are also available to perform different parts of the analysis separately, such as loading data, calculating the diffusion coefficient, etc.