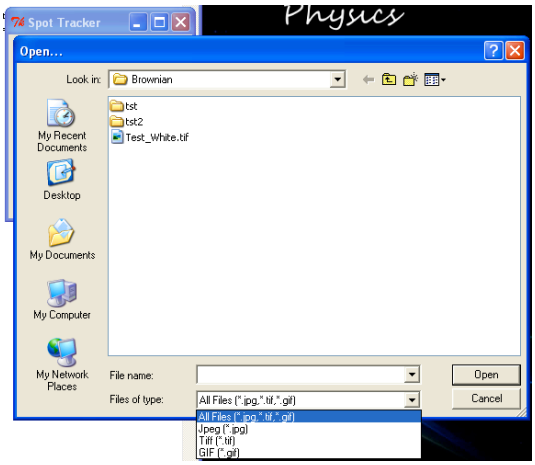
Spot Tracker
Author: Donald J Woodbury

Distribution Information

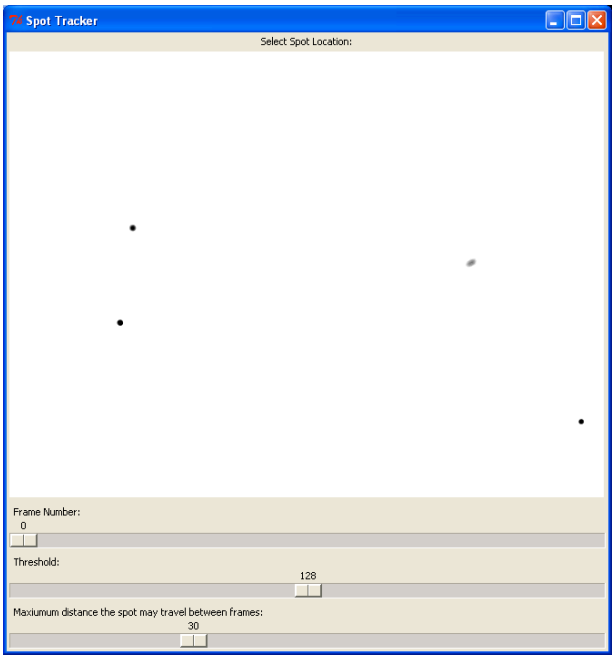
This distribution contains two main files: SpotTracker.py and the supporting Tools file. This program is written for python 2.6 or higher, with support from the numpy, scypy, Python Imaging library and Tkinter modules. The remainder of this documentation was written for the Brownian.py program, which is nearly identical to that which is included here. Only two modifications were made to adapt the program for the optical Tweezers experiment. First, the sign was flipped in the "points_below_threshold" function to make it track the bright points (points above the threshold) instead of dark ones. Finally, the function allowing the user to see all tracks within the image was removed since it requires additional supporting files and does not seem applicable to the Optical Tweezers experiment.

Instructions for running the Program

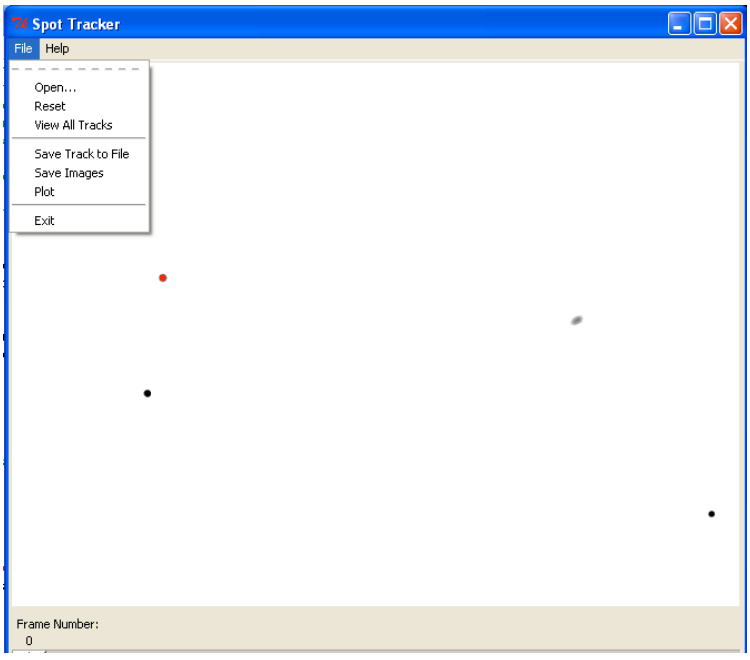
To launch the program, the user must open and run the file Brownian.py. When this is done, a window should appear, prompting the user to select the image sequence to be analyzed.



The user may select either an animation file (Tiff or Gif) or an image sequence. If an image sequence is selected, the user should first ensure that the images are named in such a way that the frame number is indicated at the end of the file name (e.g. Image1.jpg) and then select the first frame in the sequence they wish to analyze. The program will then search that same folder for images that follow the same naming structure and order them based on the frame number. Once the image sequence is selected, the Tkinter window should populate with the first image, labels and scrollbars.



The user must then specify the three parameters that govern the way the algorithm processes images. The first scrollbar allows the user to scroll through all of the images in the sequence and stop on the first frame they wish to have analyzed. The second scrollbar specifies the maximum brightness a pixel may have for it to be considered as part of the spot being tracked. Since the program converts all images to 8-Bit greyscale, pixel brightness ranges from 0 to 255. Finally, the last scollbar specifies the area around the spot that the program will search for the spot in the new frame. Once these parameters are set, the user may click on the spot they wish to be tracked and tracking will begin immediately.



Once tracking is finished, the window will be refreshed to display the all frames in the image sequence where the spot was detected. The two scrollbars for the threshold and maximum distance should disappear, and a new menu at the top of the window should replace the label instructing the user to select the spot in the image. At this point the user may either scroll through the images to see where the spot was detected or they may use the functions in the file menu to save the track data to a text file, save the images or plot the results.

When saving the track to a file the user will first be prompted to select the location and name of the file (which should end in .txt) and then to specify the frame rate. The result will be a tab delimited file, each row containing the time, x position and y position, in that order. Next, the user may choose to save the image sequence, with the spot indicated on each frame, to a folder. In this case, when the Save Images option is selected, the user will again be prompted to choose the name and location of the images to be saved. When this is done, all of the images in the sequence will be saved to the selected folder in the format "Basename_framenumber.jpg". Finally, a plotting option is offered. When this is selected, the user will be prompted to write titles and labels for the graph before a Pylab plot of the spot's position is launched. This plot is very limited in that it does not display error bars or any analysis, but it is meant to provide an instant visual of the spot's motion.

One option that has been included in this program is the ability to display the tracks from all spots in the image sequence. This feature is included only for the purpose of being an informative visual and includes no quantitative analysis methods. A warning for selecting the "View All Tracks" option: this is a very computationally intensive algorithm and may require several seconds to load. The speed at which it is able to process the images is highly dependant on how many points have been detected, so if the analysis is not complete after 30 seconds or so, it is possible that either the images are unsuitable for this kind of analysis or that the threshold is set too high. Selecting view all tracks again will revert the window back to displaying the single spot detected.

Finally, if the user wishes to either change parameters and perform the analysis again, analyze a different spot or open a new image sequence altogether, the two methods "Reset" and "Open…" have been included. Selecting Reset, will simply reset the window to the state it was in before the spot was tracked, allowing the user to adjust settings or select a new spot to track. Selecting "Open…" will effectively restart the program from the beginning allowing the user to select an entirely different image sequence.

About the spot tracking algorithm

In brief, when the user selects the location of the spot in the first frame the program will search within a square (with side length of twice the maximum distance the spot may travel between frames) around that spot for pixels of intensity below the given threshold. The locations of all of these pixels are then averaged and their centroid is then taken to be location of the spot in that frame. To locate the spot in the next frame, the same process is repeated, searching a box around the previous spot location for dark pixels. This process has the effect of being very computationally efficient and not reliant on there being only one spot in the image sequence. If two spots come close enough to each other to both fall in the box being searched however, it is possible that the program will give false readings or even loose the spot track all together. If this limitation proves to be a major hindrance for real world spot detection applications, the algorithm may be modified without too much complication to better suit the specific application.