**[program name]**+ note: do more literature review, try to add examples of papers using each program to the main table

http://link.springer.com/book/10.1007/978-94-011-0699-3

**Background**

There are many automated motion tracking solutions for behavior or ecology studies availaable, but most of them have limitations. Some may be too specific to certain organisms, becoming unusable for studies of other organisms. Some are limited to very specific environments, such as high-contrast, two-dimensional fields. Some have significant hardware requirements, such as sensors, markers or extensive camera setups. Many are not particularly user-friendly, and several are no longer in development. The programs which solve these problems tend to be expensive, some costing thousands of dollars for a license. Other solutions cited or mentioned in the literature are not available for download or purchase. Some are listed in the table below:

|  |  |  |  |
| --- | --- | --- | --- |
| Program | Cost | Notes | Cited In Or Used By |
| [CBVision](http://cwee.ucdavis.edu/projects/CBVision) | Free | Requires Linux knowledge  Risk of erasing hard drive due to carelessness | N/A |
| [Ctrax](http://ctrax.sourceforge.net/) ([pub](http://dx.doi.org/10.1038/nmeth.1328)) | Free | Too specialized (flies, experimental conditions) | 118 |
| [iSpy Connect](http://www.ispyconnect.com/) | Free | Not specialized for biological research | N/A |
| [Motr](http://motr.janelia.org/) ([pub](http://dx.doi.org/10.1016/j.jneumeth.2013.05.013)) | Free | Too specialized (mice, experimental conditions) | 7 |
| [Multi-Worm Tracker](http://sourceforge.net/projects/mwt/) | Free | Complicated installation  Requires LabView | N/A |
| [SOS-track](http://sourceforge.net/projects/sos-track/) ([pub](http://dx.doi.org/10.1371/journal.pone.0041642)) | Free | Too specialized (experimental conditions)  Requires Matlab | 6 |
| [ZebraZoom](http://sourceforge.net/projects/zebrazoom/) ([pub](http://dx.doi.org/10.3389/fncir.2013.00107)) | Free | Too specialized (zebrafish larvae) | 5 |
| [ImageJ](http://rsbweb.nih.gov/ij/) | Free | Library use requires Java programming work | N/A |
| [Flydra](http://rsif.royalsocietypublishing.org/content/early/2010/07/13/rsif.2010.0230.full) | ? | Requires significant hardware investment per site  Software not available | ? |
| [EthoVision XT](http://www.noldus.com/animal-behavior-research/products/ethovision-xt) | >5k | The majority of these programs are very well-suited for the function of the proposed program, but are either very expensive or have their price only available by quote (and as such, are likely very expensive). | [a lot](http://www.noldus.com/EthoVision-XT/Selected-publications) |
| [GroupScan](http://cleversysinc.com/csi_products/groupscan) | ? | N/A |
| [LabTrack](http://www.bioras.com/products/content/labtrack) | ? | N/A |
| [LoliTrack](http://www.loligosystems.com/?action=shop_show&varenr=AB10190) | >4k | [[1](http://www.loligosystems.com/?action=references_show_all&menu=14&keyword=PIT)][[2](http://www.loligosystems.com/?action=references_show_all&menu=14&keyword=VIDEO%20TRACKING)] |
| [PhenoTracker](http://www.animalab.pl/en/phenotracker/) | ? | N/A |
| [Viewpoint](http://www.viewpoint.fr/en/p/software/videotrack) | ? | [some](http://www.viewpoint.fr/en/p/software/videotrack/publications) |
| [WINanalyze](http://www.mikromak.com/en/products/en_winanalyze.htm) | ? | N/A |

Most of these programs were found through these sources:

http://www.sciencedirect.com/science/article/pii/S0169534714001074

http://www.samuelpean.com/how-survive-video-tracking-world-ethovision-imagej/

Citation/usage numbers were found by searching Scopus for the title of the publication introducing the software. If this publication could not easily be found, lists of publications provided by the company were used instead, if available.

**Program Function and Goal**

This program would use video files and yield a list of the points during the video where motion was detected. This list could then be used to shorten the time required to manually analyze behavior recorded in the video, since a researcher could skip to the intervals of interest, instead of needing to watch the entire video to find those intervals. This would be ideal for when one is working with many long videos, especially ones with long and frequent uneventful intervals.

This would fill a niche avoiding most of the problems faced by currently available programs. It would be available for free, due to its simple function. It would be applicable to studies of any species, and it would be extremely simple to use and not require installation.

The tradeoff is that this program would not have the ability to track individual objects (at least not anytime soon), leaving this task for manual analysis, nor would it be able to detect specific behaviours. However, the issues of manual analysis are already mitigated by the availability of a permanent record, and some types of experiments, such as nestbox studies, do not require tracking of individual objects, only identification. This program would be quite useful in such studies.

**Program Details** (technical but broad, also somewhat tentative)

The program would be developed in Python 2.5.4, using the Tkinter and PyOpenCV libraries to provide a graphical user interface and file interface and analysis, respectively.

The program, with the relevant Python modules, would be compiled into a single file using [PyInstaller](http://www.pyinstaller.org/) before distribution, so that users would not need to install anything to use it.

Input would be any video file. Initial support would be for major formats such as AVI and MP4. A long-term goal would be to extend the list of supported video formats, depending on how readily PyOpenCV supports video formats. Non-video or unrecognized files would be skipped.

Input would be done in either of two methods: 1) drag video files onto either the program’s executable or into the program’s main window (likely limited to Windows), or 2) using a standard Open File dialog accessed from the main window (cross-platform).

The user could adjust several output and processing settings before starting an analysis. These would include the sensitivity of motion detection (minimum difference between frames), the time threshold of a motion interval (motionless time required before ending a detection interval), and the format of the output (either a .tsv or .xls spreadsheet, or even a plain .txt). There would also be the option to trim each video of uneventful intervals, yielding either a series of short trimmed videos, or a longer composite video with frames to mark the boundaries of each interval. (The original video would be retained, of course). These settings would be saved in a text file kept in the same directory as the program; if the file did not exist, it would be created when the program is opened, with default settings. Once ready, the user would press a “Start” button, name the output spreadsheet, and the analysis would commence.

Sample default settings file: settings.ini

dir=/ # Output in the same directory as the program

maxload=200 # Maximum data concurrently loaded, in MB

motsens=0.05 # Detect motion if 2 frames are >5% different

motthres=10 # Wait 10 seconds before ending a motion interval

listout=xls # List format; mainly txt, tsv, xls (Excel)

vidout=1 # 0 = list output only, 1 = trim video as well

vidout\_comp=1 # 0 = split trim, 1 = composite trim

vidout\_div=1 # Gap between trims in s for composite trims

The analysis would be done using PyOpenCV methods to read and then compare each frame, limiting the amount of data concurrently loaded to avoid crashes. Motion intervals would be converted from frame index to elapsed time based on the frame rate of the video, which should be located in its file header. It would likely not be possible to automatically convert these elapsed times to actual time, however.

Output files would generally share the name of the input video file, either changing the extension or appending a tag to the name. The list file would only change the extension, defaulting to .txt in unrecognized or invalid cases. Trimmed videos would append “ trimmed” if composite, or append the interval and be placed in a new folder if split.

Each video would generate one list of motion intervals, in the format [H1:M1:S1-H2:M2:S2]. These would be placed in a spreadsheet with the locations (including names) of each video file, the same spreadsheet that was named after the user pressed Start.

Rough notes on program GUI

- Main window should have an area for the list of files, another for adjusting settings, a start button

- Settings labels should have mouseover description

- Could probably figure out an accurate progress bar for each video file based on the header data

- Unreadable videos should be flagged red or something and skipped

- ‘About’ button with link to Packer lab website

- ‘Help’ menu with popups on each of the settings and more; put the list of supported formats here

- Drag-and-drop video files to either the .py? file or the window. If you drop onto the file, it will start the program with those files loaded, then just have to press Start

- Drag-and-drop functionality will probably restrict it to Windows (http://www.java2s.com/Tutorial/Python/0380\_\_wxPython/Filedroptarget.htm)

- Open File dialog more general, so it should be included anyway; Tkinter has it (http://stackoverflow.com/questions/6969110/drag-drop-file-onto-python-script)