Multi-Worm Tracker User Guide
Version 1.3.0

Introduction

The Multi-Worm Tracker is software that tracks, in real time, small objects that move across a static background. It saves summary data as text files and comes with a post-processing analysis and graphic program, Choreography, that can read this summary data.

The tracker program, MWT, runs under Windows XP or later. The analysis program, Choreography, runs on any platform with Java 1.6 (also called Java 6) or later.

Installation

See the Software Installation Guide for instructions on installing MWT and Choreography.

Please note that settings files are not compatible between versions. All settings files must be created and saved with the version of the MWT on which they will be used. Failure to take this in to account will result in the MWT exhibiting unpredictable behavior that will make the software unusable.

Using MWT

MWT can be run by double-clicking on its icon.

It has been discovered that when there is no Measurement Computing PCI-CTR05 card installed, you will be prompted every time you run the MWT executable that "No boards have been configured." in a dismissable dialog box. There is no way to remove this behavior, however the dialog can be safely dismissed by clicking "OK" and does not impact the execution of the MWT.

When MWT starts, it displays the main instrument window and a camera selection window.

Selecting a Camera

The first step is to select the image source. The camera selection window provides three options.

- IMAQ. Use this if you have a CameraLink camera.
  - If you have more than one CameraLink camera, you must enter the interface name associated with your camera in the field "IMAQ interface name."
- IMAQdx. Use this if you have a gigabit ethernet, firewire (IEEE 1394) or USB DirectShow camera.
  - If you have more than one camera using GigE or IEEE 1394, you must enter the IMAQdx session name associated with the desired camera in the drop down box labeled "IMAQdx session."

The most recent version of this document is available at https://sourceforge.net/projects/mwt/files/mwt/1.2.0%20Release/documentation/MWT_User_Guide.pdf/download

- AVI file. Use this if the image source is a movie saved to disk.
  - You must enter the full path for the movie you would like to load. This can be entered in the "AVI path" field. The file icon to the right of the field can be used to access a file system browser.

For the IMAQ interface, it is possible to use external triggering to control image acquisition. This is enabled on the camera selection window by clicking on the button labeled "Use external trigger for camera control (IMAQ only)." The triggering parameters are not modifiable by the user and are as follows; 1000 ms timeout, rising edge triggered, external trigger on line 0 (for NI-1427 cards), with the rising edge of the trigger signal beginning the start of acquisition of a frame. The trigger signal must be active before the start of the MWT, even during the initialization stage. The MWT will not function correctly otherwise.

If you have only a single device that uses a given interface type, you should not need to change the default interface names. If you have multiple devices, you can use the Measurement and Automation explorer to find the interface name of the device that you wish to use with MWT.

Occasionally, MWT will fail to recognize the camera and provide an error message the first time it is started after a reboot. If this occurs, stop and restart MWT. If an error is still generated, close MWT and try again. If the error persists, it is most likely not an issue with the MWT software.

If any error occurs during camera selection, either due to a hardware issue or user error, an error message will be reported to the screen and MWT will stop itself. It can then be restarted after the problem has been addressed.

Interacting with MWT's main instrument window

The main window of MWT is a standard LabView virtual instrument front panel. What this means is that the device works similarly to an electronic instrument; at the top left of the window there are three icons: an arrow, a pair of circling arrows, and a stop sign. When MWT starts, the arrow will be solid black, indicating that it is running. The stop sign will be bright red, meaning that you can turn off MWT by pressing it.

When MWT finishes an experiment, or you press the large red "Stop" button at the bottom right, MWT will stop running. You will need to press the upper left arrow (now white with a black outline) to run it again. You should never need to hit the LabView front panel stop icon at the top left; this stops MWT without allowing it to perform housekeeping, so if you do stop it this way, you should run MWT again and then hit the lower-right "Stop" button before doing a real experiment.

Setting Up Your Experiment

The left side of the MWT instrument window allows you to set parameters that govern tracking. The only items that must be set are the output path and prefix; then, if MWT is running, the red dot next to "prefix" should turn green and the large green "Go" button will change from being faded out to being active. If this does not happen, either MWT is not running, or MWT is not allowed to place files at the output path; try to select the output path again.

Most parameters are best adjusted while MWT is running, as one will get feedback in the main image window. Some changes, such as to binning and some advanced settings, for example, require MWT to be stopped and restarted (with the "Stop" button and white run-arrow) to re-initialize the data properly, however.

The parameter settings work as follows.

Object Tracker Settings

MWT can look for bright objects on a dark background or dark objects on a bright background; this behavior is controlled by the top switch.

The "object contrast" is the fraction of the total dynamic range of the camera by which an object must differ from the background in order to be counted. For example, a 10-bit camera has a dynamic range of 0 to 1023. If the tracker is looking for dark objects with 10% contrast on a background of 850, the darkest parts of the object must be at least 10% of 1023 below 850—that is, 102 counts below, or 748.

The "fill hysteresis" controls how objects are filled by the image processing algorithm. Starting from pixels that surely have enough contrast, the algorithm fills outwards until pixels are found with the percentage less contrast given by fill hysteresis. To continue the previous example, if the fill hysteresis is 40%, the boundary of the object will be not at 102 counts below background but 102 – 40% or 61 counts below background. This dual-threshold behavior—one for the dark part of an object and one for the border—helps avoid finding many spurious single-pixel "objects" that are just camera noise.

The "maximum object size" and "minimum object size" set how many pixels one expects in a valid object; MWT may detect objects outside of this range but will not track them or provide any data about them.

The "object size hysteresis" is a fudge factor that applies to objects already being tracked; a value of 20% means that an object that was originally within the maximum/minimum size bounds is allowed to go 20% below the minimum or 20% above the maximum before MWT stops tracking it. This helps prevent objects that are close to the size limits from flickering in and out of tracking.

Advanced Settings

"Image binning" takes the camera image and groups pixels in N by N blocks where N is the bin size. If the binning is changed, you must stop and restart MWT for image processing to continue normally.

The "Skeletonize" switch instructs MWT to save an eleven-point spine along the center of every tracked object at every time-point; this can be useful for shape analysis, but may slow the frame rate somewhat due to the extra processing. Skeletons increase the size of the output files by about 50%.

The "Outline" button saves a compressed representation of the outline of every tracked object at every time-point; this can also be useful for shape analysis and may also slow down processing slightly. Outlines increase the size of the output files by about 50%. Note that skeletons can be computed from outlines but not vice versa.

The Advanced Settings button contains several other options that rarely need to be altered

- "Aggregate Output" controls whether or not the raw data from objects is output to individual files for each one, or collected in a single file in groups of 1,000. There is a known issue with NTFS file systems where, when accessing directories with over 10,000 files, there will be random, extreme performance degradation, which will appear in the data stream as a pause of 1 second or more, with tracking resuming normally afterwards. The MWT will only generate that many raw data files when tracking populations of 70 individuals or more for experiment durations of an hour or more. If 70 or more is the expected population size, Aggregate Output should be set to reduce the number of files generated. Otherwise, it can be unset. A word of warning, however; If aggregate output is turned on and Choreography is used to analyze the data, analysis cannot be performed on a user-defined subset of the data stream, the entire data stream must be read in each time.
- "Number of Image Adaptation Bands" controls what fraction of the full image is scanned for new objects (and adapted according to the adaptation rate) each frame. Scanning a full image is very expensive, so for maximum performance this should be as small as possible while still allowing the largest expected object to fit comfortably within one band. It is not necessary to use powers of two, but we typically do so.
- "Adaptation rate" controls the how long of a moving average is used to form the background. The background adapts 2^-(adaptation rate setting) per frame.
- "Frames of adaptation" specifies how many images MWT should use to adapt to the background before starting to track objects when one hits "Go". This creates an initial memory image for the experiment.
- "Object border" specifies how far around an existing object the next scan will look. For example, an object border of 10 would look in a 35x30 square for an object that was bounded by a 15x10 square (by looking in the same spot but adding 10 pixels on each side). Fast-moving objects may require larger borders, but this will slow down the processing.
- "AFG resource name" is used to identify a Tektronics Arbitrary Function generator, if such a device is attached to the system. If one is, a descriptive name will appear in the drop-down box. Even though it appears in the box, this name must be selected or you will not be able to use the generator with the MWT.

- "Image bit depth" tells MWT the range of values to expect from the camera (how many bits per pixel). If this is not set properly, MWT may have trouble detecting images, or may bleed data from one pixel to the next, or otherwise improperly interpret the image data
- "Bit depth from camera" is normally set, so MWT reads the expected number of bits per pixel from the camera directly. If this is unchecked you can enter image bit depth manually.
- "Use Background Division Image Correction" controls which type of image correction is used within the MWT, background division or background subtraction. The default behavior is for this control to be turned off, thus using background subtraction. Background division image correction is turned on by setting this control, which will cause a significant degradation of performance due to the increased processing time needed for division over subtraction, but can provide cleaner image correction in special cases. Background subtraction image correction is the preferred algorithm.

Data Output Settings

The path specifies where MWT output data should be saved. At that path, MWT will create a directory with the format YYYYMMDD_HHMMSS where the time and date are the time at which the experiment was started.

The prefix specifies a set of characters prefixed to every file in the output directory. These characters must therefore be valid ones for filenames. A prefix must be at least one character long. Starting the prefix with spaces is a bad idea; the software will probably not get confused, but you probably will.

Normally, MWT saves one image at the beginning of its run. Saving images is a very time-consuming operation compared to the other image processing MWT performs. However, if you want MWT to save images as it goes, enter a non-zero value in the "Raw Image Save Interval (s)". Images are saved in PNG format. Make sure you will have adequate space to store all the images generated during a run, as the MWT does not check for necessary space, and will therefor fail if it attempts to save an image for which there is no space. Saving images does negatively impact the performance of the tracker and is limited by the write speed of the computer's hard drive. This means that one can, in practice, only save an image every 2 seconds. While shorter intervals can be entered, they will not be honored. Higher rates of image saving can only be achieved with specialized hardware and software not included with the Multi-Worm Tracker.

"Experiment duration" sets how long MWT should run after "Go" is pressed before the experiment is automatically terminated. If the value is set to 0, the MWT will run indefinitely until "Stop" is pressed.

Stimulus Settings

Tap, Puff, Custom 1, and Custom 2 are four distinct output channels that MWT can trigger. Each can be independently selected with on/off switches.

The "onset delay" for each specifies how long MWT should wait (in seconds) before delivering the first stimulus.

The “interval” specifies how long in seconds MWT should wait between successive stimuli after the first (if there are any after the first). Due to timing constraints within LabView, the minimum interval is 2 seconds. Any value entered below that will be coerced to two seconds. However, through clever use of multiple stimuli triggers (such as Puff and Tap triggers on a single stimuli), any permutation of fine stimuli trigger control can be achieved.

The “event #” sets how many stimuli to deliver of that type.

The drop-down menus for interval type let you select:

- Fixed; the interval is the value specified each time.
- Exponential; the interval is treated as a Poisson process with the listed mean.
- Custom; this brings up the Advanced window and uses manually-entered stimulus times from that list.

The “Advanced” buttons bring up windows for rarely-used settings. Every stimulus type has the same user interface for entering custom timing intervals. This is a vertical column of numbers with up and down arrows and an text field in the upper left corner. Enter the times at which you want the stimuli to appear in order in the column of numbers. If you run out of room, use the arrows (or text field) to scroll the column so you can see more entries. Times entered are absolute times in the experimental run.

All advanced stimuli settings can be changed at user-defined points in an experimental run. All of the settings described below are input as a 2D array. The first column is the time, in milliseconds, since the beginning of the experiment at which the value in column two is the active state of that setting. The first time entry in any setting must be 0.

The MWT does correct for illegal values, but not nonsense stimulus configurations. It also does not guarantee a specific ordering at which settings will be updated. Thus, if you have a stimulus set to trigger 1000 milliseconds after the start of the experiment but also have stimulus settings changing 1000 milliseconds after the start of the experiment there is no way of predicting if you will get the pre-1000ms settings or the post-1000ms settings when the stimulus is triggered at 1000ms.

In addition to being able to specify exact stimulus times explicitly, there are additional options for each stimulus:

- Tap stimuli can be bursts of taps instead of a single tap. “Duty cycle” is the fraction of time spent high instead of low. “Time between tap onset (ms)” is the time (in milliseconds) between the start of one tap and the start of the next during the burst. “Tap number” is the number of taps in the burst. The maximum total duration of a burst is 65000 milliseconds. The time between tap onset must be in the range of 3 to 6500 milliseconds. Each tap must have at least 2.7 ms of time low and 0.3 ms of time high; the duty cycle will adjust itself automatically if this restriction is not met.
- Puff, Custom 1 and Custom 2 are all single pulse stimuli. They only allow the duration to be set; the valid range is 3 to 65000 milliseconds.

- Custom 1 provides the ability to set a Tektronics AFG signal generator to generate waveforms with the parameters specified by a list. Every row in the array labeled "AFG settings" is a time point in the run of an experiment. The times are specified as absolute times. The frequency and amplitude fields specify the parameters of the waveform the AFG is to generate. The type of waveform is restricted to "pulsed sine wave."

Through the Advanced interface, it is possible to set "nonsense" stimuli configurations, such as configuring a puff to last for 6 seconds, but with a 2 second stimulus interval. The MWT does not perform any sanity checking on stimulus settings. The result of such insane settings are undefined, which could lead to anything from destroying the stimuli delivery mechanism to continuous, random stimulus trigger signals. Any outcome on this spectrum is undesired.

Saving Configurations

There are Save Configuration and Load Configuration buttons on the bottom of the window. MWT must be running for these to work. They work as you would expect. MWT configurations are saved in an XML format, which is described elsewhere.

Setting Region of Interest

A region of interest (ROI) is not required to be set. However, it is useful for restricting the amount of data the MWT needs to process by sectioning off parts of the input image that can and should be ignored. The buttons for accomplishing this are just above the image display area of the front panel. ROIs are entered one at a time through the interface that will be described below.

The two buttons associated with ROIs are called Clear Region of Interest and Set Region of Interest.

The Clear Region of Interest button will cause the last set ROI to be deleted. ROIs are deleted in the reverse order they were set. The button must be pressed for every ROI the user wishes to delete.

Set Region of Interest creates a new window. This window contains an image display area, a toggle-able switch labeled "ROI type," with one side labeled Rectangle and the other Ellipse, five buttons on the right labeled Set ROI from Image, Set ROI from coordinates, Reset Input, Apply, and Cancel, and four numerical input fields, which specify the upper left and lower right corners of a bounding box containing the desired region of interest.

When the ROI window initially opens, there will be a display of the current image input and the Set ROI from Image, Set ROI from coordinates, and Apply ROI buttons will be grayed out. Only Cancel and Reset Input will be active. At any time, Cancel will cause the ROI window to close without adding a new ROI. Reset Input will remove any drawn ROIs and set the coordinate input fields to 0, and disable and grey out the Set ROI from Image, Set ROI from coordinates, and Apply ROI buttons.

ROIs can be specified in one of two ways; either by left-clicking with the mouse and dragging it to draw a green outline of the corresponding ROI type, as specified by the ROI type toggle, on the image itself or by typing specific coordinates in to the numerical input fields. Both rectangular elliptical ROIs are specified as two points. For rectangular ROIs, the points define the upper left and lower right corners of the rectangle. For elliptical ROIs, the points define the upper left and lower right corners of a rectangle that encloses an ellipse whose major and minor axis equal the length and width of the rectangle.

In either case, until you press either the Set ROI from image or Set ROI from coordinates, an ROI is not created. Neither of the Set ROI buttons will become active until you have either clicked and dragged a shape on the image, or entered coordinates in to the numerical input fields. If there is no green ROI shape on the image, Set ROI from image will be disabled. Once either, or both, of the Set ROI buttons are no longer greyed out and thus are active, you must press one of those two buttons for an ROI to be set in the MWT. If you do not press one of those two buttons, the Apply ROI button will not become active and you will not be able to create an ROI.

You are free to change the ROI in any manner you see fit, and can press either of the Set ROI buttons as many times as you like to update the existing ROI. Pressing Set ROI from image will cause the numerical inputs to update to reflect the coordinates of the drawn ROI. Conversely, pressing Set ROI from coordinates will cause the drawn ROI to update to reflect the entered coordinates.

The drawn ROI, if it exists, or the coordinates, and the ROI type represent are the ROI that will be created when Set ROI from image or Set ROI from coordinates is pressed next. They do not necessarily reflect a previously set ROI.

Once all the required points have been entered for the desired ROI type, the Apply ROI button will become enabled and the specified ROI should be drawn on the screen. The green ROI shape is a User Interface convenience. The ROI that will be set is also drawn in white by the MWT library, thus the green ROI shape may overlap it.

Reset Input removes this most recently specified ROI, allowing the user to define a new ROI. If the user is satisfied with the ROI, pressing the Apply ROI button will cause that ROI to be saved to the MWT. If the user does not wish to set an ROI, they can press the Cancel button to close this window without making any changes.

Multiple ROIs of different types can be set. All ROIs will be processed during an experiment. Any overlapping ROIs will be treated as a single ROI for processing purposes. There is a known issue with how ROIs are handled which will cause the MWT to become unresponsive and have to be shutdown. The issue is only with multiple ROIs and occurs when two disconnected ROIs are then connected by a third ROI. This is to be avoided if at all possible. There is no issue with any two ROIs overlapping, only when formerly disconnected ROIs are later connected.

All ROIs will be displayed in the main MWT window, under the Fixed display type.

Setting Reference Object(s)

A reference object is not required to be set. However, it is a useful tool which causes all measurements to be reported relative to the location of the specified object. The buttons for accomplishing this are just above the image display area of the front panel.

There are two buttons associated with reference objects; Clear Reference Object(s) and Set New Reference Object.

The “Clear Reference Object(s)” button will cause the last set reference object to be deleted. In order to delete all reference objects, the button must be pressed for each object set. The reference objects will be deleted from most recent to oldest.

Set New Reference Object causes a new window to be created. This window contains an image display area which shows an unprocessed live feed from the image input source with any and all previously set reference objects painted in white and Regions of Interest painted in white, as well as four buttons; Set Reference Object from Image, Reset Input, Apply Reference Object, and Cancel. Set Reference Object from Image, Reset Input, and Apply Reference Object are disabled until a location on the image is selected.

The Cancel button will close this window without setting a reference object. Reset Input will cause the last selected location to be cleared, allowing a different location to be set, and will also cause the Set Reference Object and Apply Reference Object to become disabled. Set Reference Object will load the specified point in to the MWT, cause any resulting Reference Object that is located to be painted, and enable the Reset Input and Apply Reference Object buttons.

A reference object is any high-contrast area of the image that is expected be static throughout an experiment. They are selected within this window by clicking with the left mouse button on the object as it is displayed in the image display area. The location specified is painted with a green '+'. This green '+' represents the location of a potential reference object that will be loaded when the Set Reference Object from Image button is pressed. Until that button is pressed, no update is made to the MWT's reference objects. At this point, assuming the location specified contains a high contrast object, the object indicated should be painted white, which is a sign that it has been accepted by the MWT, and the Apply Reference Object and Reset Input buttons will be enabled. If the object is not desired, or was not correctly selected, the Reset button can be used to clear it. Alternatively, a new point can be selected by left clicking on the desired location of the image and Set Reference Object from Image can be pressed again, causing this new location to be searched. If the object is as desired, then the Apply button will save it in the MWT and close the Set Reference Object window.

Viewing the Experiment

The central part of the MWT instrument window provides visual feedback about the experiment. During initialization, the image in the window is typically constantly updated (unless the switch is set to “do not display”). The image is not resized, so typically only a subset is displayed and can be relocated with scroll bars. What is shown depends on the drop-down menu at the top.

- Fixed Image is the image with the static background subtracted (gray). Whiter objects are brighter than background; blacker objects are darker. Objects being tracked will be filled (typically white) and their centers will be marked (typically black). Skeletons and outlines, if selected, may also be marked (typically in black).
- Memory Image is what MWT believes the background to be
- Raw Image is the uncorrected input from the camera

At the bottom of the window, MWT reports how many objects it's tracking along with updates on any stimuli it will or has delivered, and an estimate of the time per frame (in seconds).

Starting the Experiment

Press “Go”.

The button should gray out, the display will switch to “do not display” (put it back at “display” to see what is going on, though this will slow down the tracking somewhat), and the frame rate and stimulus indicators will report on the current status.

If you have not set the experiment duration, at some point you should press “Stop”. MWT will write out data for individual objects when it loses track of them, but it will not write out summary data until Stop is pressed (or the experiment completes its duration). The MWT consumes system memory as it runs, which is by design. Consequently, it will run until there is no more memory available on the system, or the experiment duration elapses or the stop button is pressed, which ever occurs first. If the MWT runs out of memory, it will crash and no summary data will be output.

Summary Graphs

The right side of MWT shows summary graphs of the average data, plotted with the best information MWT has at the time. Later analysis helps considerably to clean up artifacts, but these graphs can give an idea of what is happening.

Each of the four graphs can be independently switched to display a different type of data. If two graphs are given the same data type, the graphs appear identical to each other.

The graph types are the same as the output data found in the .summary file (see below), except for image number and time.

Understanding MWT Output Data

MWT produces as output three sets of files.

MWT saves a single file called prefix.summary (where prefix is whatever you entered in the prefix box).

MWT saves one or more image files. There is always a prefix.png file that shows the initial image; if images were saved along the way, they appear as prefix_NNNNN.png files, where NNNNN is the elapsed time of the experiment in milliseconds at which the image was taken.

MWT usually saves one file for each tracked object. These appear as prefix_NNNNN.blob. If the option is set to group blob files, up to 1000 objects will be grouped per file, and the extension will be "blobs" instead of "blob". The format is exactly the same, except the objects are separated by lines that start with % and, after a space, are followed by the object ID number.

The output format of the .summary and .blob files follows.

Summary File Format

The summary file is a text file that has one row for each image that was captured and processed. The columns are as follows.

1. Image number (counts up from 1)
2. Image time (in seconds from the beginning of the experiment)
3. Number of objects tracked
4. Number of objects that have lasted long enough to be averaged into summary data
5. Average duration for an object to have been tracked
6. Average speed in pixels/second
7. Average angular speed in radians/second
8. Average length of object in pixels
9. Average relative length of object, where relative length = current length / mean length
10. Average width of object in pixels
11. Average relative width of object
12. Average aspect ratio of object (width / length)
13. Average relative aspect ratio (current W/L) / (mean W/L)
14. Average "end wiggle": angle in radians between the last 20% of the body and the rest of the body (using whichever end shows a greater angle).
15. Average number of pixels filled as part of object detection
16. There may be additional data starting at the 16th column, including one or more of the following.
    - A % followed by one or more numbers describing whether or not an event occurred and which events they were. Each number is output in hexadecimal format, printed as "0x" followed by the number. This number is a bit field, where the first four bits of this number are flags specifying if an event took place. If the event occurred, the value of the bit is 1, and 0 otherwise. The first, or lowest-order, bit corresponds to Tap stimulus, the second to Puff, the third to Custom 1 and the fourth to Custom 2. For example, a value of 0xB in binary is 1011 which indicates that Custom 2, Puff and Tap stimuli were delivered, but no Custom 1 stimuli, that frame.

- A %% followed by pairs of numbers that specify how objects were found and lost. The first number indicates the object number before this image was processed; the second, what it turned into afterwards. 0 means that no object was found. Thus, 0 24 means a new object was found and was given number 24; 24 0 means that the existing object 24 was lost; 24 40 25 40 means that 24 and 25 collided and the resulting object was labeled 40; and 40 67 68 means that compound object 40 fell apart into two objects which were labeled 67 and 68.

- A %%% followed by pairs of numbers if and only if objects were grouped into blobs files. The numbers are in the form: N X.Y, where N is the object number, X is the file number, and Y is the byte offset within the file.

Blob File Format

The blob files are text files that have one row for each image that the object was present. The file name itself lists the number of the object. The columns are as follows.

1. The image number (where 1 is the first image in the experiment)
2. The time (in seconds from the beginning of the experiment)
3. The x-coordinate of the center of mass of the object (to a fraction of a pixel)
4. The y-coordinate of the center of mass
5. The number of pixels in this object
6. The x-coordinate of a vector pointing along the long axis of the object and with length equal to the standard deviation of the pixels along that coordinate axis
7. The y-coordinate of that vector
8. The standard deviation of pixels measured along the orthogonal direction
9. The length, in pixels, of the worm measured along its long axis (from furthest forward to furthest back)
10. The width, in pixels of the worm orthogonal to the long axis (from leftmost to rightmost)
11. There may be additional data if the “skeleton” or “outline” options were selected
    - A % followed by 11 pairs of numbers; these give x and y offsets for points along the center of the object that form the spine. These numbers are relative to the center of mass of the object (rounded onto the closest pixel).
    - A %% followed by 3 numbers followed by a text string. The first two numbers are the x,y coordinates of one pixel on the outline of the object. The next number is the number of pixels in the entire outline. The string contains a compressed form of the outline; since the outline is 4-connected (cannot be connected only on the diagonal), each additional outline point is encoded with two bits: the first bit specifies whether the outline moves in x (0) or y (1) and whether it goes in the negative direction (0) or the positive direction (1). These bit pairs are packed three at a time to give a number between 0 and 63, and then added to the ASCII character 33 ('!') to give an ASCII character between 33 and 96. If there are not enough pairs to fill the last character, the end is padded with 00 if one more pair is needed, or 0001 if two pairs are needed.

Blobs File Format

Blobs files are output optionally and replace Blob file output. The file name describes which group of one thousand objects are stored in that file. However, blobs are not stored in numeric ID order, but in temporal order of when they were destroyed. The Blobs files are text files that have headers which separate individual object histories. Each object stored in the Blobs file is preceded by a % followed by ID number of that object on a single line. The object history that follows is exactly the same as that described above in the section on the Blob File Format.

Using Choreography

Choreography is a mostly-command-line Java program; it takes as input the dated-and-timed directories produced by MWT, or zipped copies thereof, and produces refined summary data based on the directives given on the command line.

To start Choreography, enter

```
java -jar Chore.jar --help
```

which will produce a synopsis of how to use Choreography.

It typically is not convenient to run Choreography from where the file is located. In this case, you can enter the full path or drag the Chore.jar file onto the command line (in most operating systems). You can also do the same with the target directory or .zip file for analysis.

Thus, a typical analysis on a Windows machine might proceed like like

```
java -jar
```

(drag Chore.jar from C:\MWT onto the command line)

```
 -t 5 -p 0.025 -M 2 -o s
```

(drag 20090329_152706.zip from C:\data onto the command line)

to create a command-line argument of

```
java -jar "C:\MWT\Chore.jar" -t 5 -p 0.025 -M 2 -o s "C:\data\20090329_152706.zip"
```

that will run Choreography and produce a summary output file containing the average speed.

With a little experimentation and reading the help messages produced by Choreography, an individual familiar with command-line programs should be able to decipher how to use the program. Ideally, once the desired processing has been determined, Choreography will be run automatically by scripts.

Using the Graph View

Choreography can graph summary data when one uses the --graph option. This brings up a window with a horizontal time axis and one vertical axis for each type of output data. To turn individual graphs on and off, click on the check box under the axis for that data type. To rescale or move a y-axis, hover over the axis and use the scroll wheel or drag the axis to the desired position. To rescale the x-axis, do the same over the time axis.

Most data sets are plotted as a line. Standard deviation and quartiles are instead plotted as a range with a semi-transparent region between the top and bottom of the range.

Using the Data Map View

Choreography can plot a map of the tracks of objects when one uses the --map option. This brings up a single window with x and y scale bars in microns, a double-slider, and a variety of options.

The main map window can be panned by dragging and can be zoomed in and out with the scroll wheel.

Minimap windows can be opened from the main map window menu; these will show the neighborhood around the main viewing area if the "sync" button is depressed, or will stay wherever they were left if not synced. Only one minimap is synced at a time. Switching which minimap is synced allows one to quickly flip between different views of the same data.

The double-slider at at the bottom of the main map window controls what fraction of the total experiment time is displayed, and at what time the objects are drawn. The portion of time drawn is in blue, and is between "start" and "end"; "now" refers to the time at which the object is drawn (the white arrow on the slider). The start, end, and "now" points can be dragged on the slider (the cursor will change when over each) or set manually or with arrows on the numeric entry fields. The entire selection can also be dragged (the cursor will appear as a fat double-headed arrow). The bottom slider is a zoomed-in view of the top slider for finer control. Normally, it zooms into the "now" position; by right-clicking on the top slider at the appropriate point, one can set it to zoom into the start or end parts of the selected range.

There are a series of drop-down menus near the bottom of the window. These are:
Grayscale, Sunset, …: different color schemes to color the tracks
Black, White, Green, Image, Dimmed: the background color; the last two will show the image taken when the MWT started. If additional images have been saved, they will be displayed at the appropriate time.
Pixel, Spot, …, : how to draw each data point. "Value" will, when zoomed in, show numeric values corresponding to that timepoint (as determined by the last drop-down)
Time, ID, Value: how to color each data point. "Value" will color the points according to the data type selected in the last dropdown.
time average, …: the quantity used to color and/or label data points. These are the same quantities sent to the output file (so you must output at least something). Since this shows only single tracks, some quantities are unhelpful (e.g. standard deviations are always zero).

At the bottom of the window is a set of play controls that will move "now" forward or backward in time. The rate relative to the real duration of the experiment can be set as a percentage (1-100000%). The screen will update at a maximum of 33 Hz, slower if the processing takes longer than that.

The left-most drop-down menu by the play controls allows one to select an object number. If an object number is selected, the main view will follow that object as it is played. If an object disappears, it will disappear from the list; if you type in the number instead of selecting it from the list, the value will not disappear and the view will follow the object at those times for which it exists. This feature is particularly helpful for hand-validation of data.

Understanding the Output of Choreography

Choreography provides a wide range of output types and statistics.

Output Types

time: is always the first column, unless it is specified somewhere else. It is given in seconds from the beginning of the experiment.

frame: the image number (starting at 1 at the beginning of the experiment)

number: the number of objects tracked

goodnumber: the number of objects that pass all criteria given to Choreography for counting a worm as valid

persistence: the lifetime of present objects; this differs from the MWT persistence score because the MWT only counts persistence-up-to-the-instant, while this measure considers the full lifetime of the object (both before and after the current timepoint)

area: the area in square millimeters of objects

speed: the speed in mm/second of objects. This speed is averaged over the duration given in the -s parameter; it is equal to the distance between the most widely spaced pair of points over that period of time (half ahead of and half behind the current timepoint) divided by that duration.

angular: the angular speed in radians/second of objects; this is calculated over the same interval as speed, but reports the greatest difference in angle between primary axes over that time

length: the distance spanned by objects along their major axis (defined to be the axis of a least squares fit), in mm.

rellength: (length at current timepoint) / (mean length)

width: the distance spanned perpendicular to the major axis, in mm.

relwidth: (width at current timepoint) / (mean width)

aspect: the ratio of the width to length

relaspect: (aspect at current timepoint) / (mean aspect)

midline: the length of the object, in mm, measured along the skeleton; this is only available if skeletons were collected (or outlines were collected, and skeletons were generated with a plugin).

morphwidth: the mean width of the central 60% of the object, measured perpendicular to the skeleton. This is only available if the object has a skeleton.

kink: the angle in radians between the line from the first to third point of the skeleton and the fourth through last points; or the angle between first and fourth to last and third to last to last, whichever is greater. This is similar to the MWT endwiggle measure.

bias: the fraction of objects moving in the dominant movement direction as opposed to the other direction. The dominant direction is labeled 1, the opposite -1, and stationary objects are 0. This is usually inaccurate unless the --segment (-S) option is used.

pathlen: the distance that the animal has traveled. This is both signed (forwards is positive, backwards is negative) based on the results of the "bias" output and cumulative. If movement direction cannot be computed, then the length starts again from zero.

curve: the average angle, in radians, of points along the object's skeleton taken two apart. This measure is only available if skeletons exist.

dir: change of direction. Forward motion is 1; motion that backtracks is -1 at the point of turning. This metric is somewhat unreliable unless --segment (-S) is used.

loc_x: the x coordinate of the object, in mm. Usually only useful with object-by-object output (-N).

loc_y: the y coordinate of the object, in mm

vel_x: the x velocity of the object, in mm / sec, averaged over the duration given for -s (default 0.5s)

vel_y: the y velocity of the object, in mm / sec.

orient: the angle of the body, in degrees. Head and tail are not, in general, determined, so this value is only correct modulo pi.

crab: the speed perpendicular to the length of the body, in mm/sec, averaged as specified by -s. Note that the speed along the length of the body is sqrt(speed*speed – crab*crab).

custom: data provided by a plugin.

tap: 0 when a tap did not occur, 1 when it did.

puff: 0 when a puff did not occur, 1 when it did.

stim3: 0 when the first custom stimulus did not occur, 1 when it did.

stim4: 0 when the second custom stimulus did not occur, 1 when it did.

Output Statistics

mean: this is the default, and averages over all good objects for that parameter. Note that even if the object is good in general, it may not be useful for a given statistic due to it not being good throughout the entire speed averaging duration or various other problems.

number: this is the number of objects that would go into any mean or other population statistically

exists: this is 0 if the output type cannot be calculated at this timepoint, 1 if it can. This is typically useful for weighting temporal averages.

max: the highest value of the parameter found in the present set of objects

min: the lowest value of the parameter found in the present set of objects

median: the value of the 50^{th} percentile of the distribution of the parameter found in the present set of objects

p25: the 25^{th} percentile

p75: the 75^{th} percentile

var: the variance

std: the sample standard deviation

sem: the standard error of the mean

Trigger Output

Trigger output is of the same type as the base output form, but averaged over the duration specified in the trigger command. Trigger averages over time first, and then over animals; thus, it is not the same as averaging over the output without using the trigger command.

Be aware that the statistics are averaged in a “stupid” way—they are calculated for each animal exactly as written, and then the appropriate method is applied across those averages.

Object-by-Object Output

The --each-id (-N) option instructs Choreography to run its analysis routines on a list of specified objects, and produce separate output for each one. The files extensions are the same, but after the prefix they contain the corresponding object number. "-N all" will produce output for every valid object.

Plugins

Several plugins are packaged along with Choreography. The plugin architecture assumes that the writer knows a great deal about the internals of the Choreography code. (When changing the code, the best approach is to look at where the methods in the CustomComputation interface are called in Choreography to get a feeling for program flow, and then to look at the existing plugins to use as templates.)

Plugins are invoked with

--plugin PluginName::option1::option2::option3:: . . .

If a plugin provides custom data output, the first plugin's output will go into the first output of type custom (C). If that plugin provides multiple outputs, they will go into columns with successive C's; otherwise, additional C's will deliver the output of later plugins. No existing plugins use this feature at the current time.

To get details on how to use a plugin, the best way is to run Choreography and load the plugin with --plugin PluginName::help (e.g. --plugin Respine::help).

The existing plugins are:

Eigenspine

This plugin computes principal components of the (angles between adjacent segments of the) spine. It provides those eigenvalues, plus error and phase, as custom outputs.

Extract

This plugin saves skeleton and/or outline data.

Flux

This plugin measures crossings into/out of regions of interest, or counts objects inside the regions. It can also be used to blank out other measurements that are not in the region of interest.

MeasureReversal

This plugin detects and provides statistics on reversals. It will either supply data on every reversal (with --plugin MeasureReversal::all), or will look for reversals associated with an event (e.g. --plugin MeasureReversal::tap).

Reoutline

This plugin recomputes the outline of the object by smoothing it and allowing it to take on floating-point values.

Respine

This plugin recomputes the spine (skeleton) of the object given its outline. It takes two parameters: the number of points in the spine, and the fraction of the body to consider when finding the endpoints for the spine.

SpinesForward

This plugin aligns all the spines so that the head is the first element and the tail the last (as judged by movement bias, as well as can be determined). Otherwise, tail may be first some or all of the time, as preprocessing steps don't have the information necessary to make an accurate determination.

(Partial) Version History

1.3.0: The MWT configuration files were changed from binary to XML. This breaks all backwards compatibility but helps ensure it in the future. The MWT UI has been simplified and compacted. Many plugins for Choreography were added, and the Choreography source code was released.

1.2.0: The triggering system for the MWT was improved. Improved algorithms for detection of movement direction and detection of reversals were added to Choreography.

This Document was written by Rex A. Kerr and Nicholas A. Swierczek