Peregrine: help

Sachit Butail

November, 2012

Contents

1 Introduction

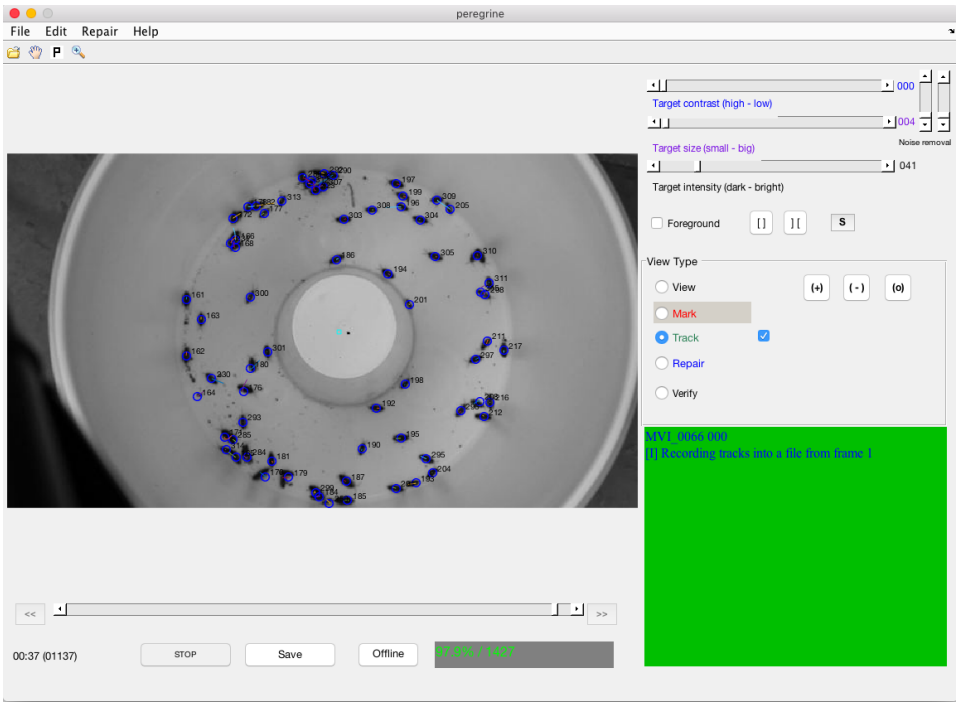

Figure 1: Tracking shape and orientation of three fish in a flow tank.

Peregrine is an automated tracking system developed to analyse and study animal behaviour. Versions of this tracking system have been used/developed as part of [1, 2, 3, 4, 5, 6]. Please refer to those for technical details. This document describes basic steps to get started and use peregrine. The following videos located in the current folder can be used for a quick start:

- initial_setup.mov
- camera_calibration.mov

2 Initial setup

Peregrine is developed in MATLAB and shared using a dropbox folder. For best results the input to peregrine should be in the form of frames with sequential filenames. The length of each filename should be the same, for example, exp01_001.bmp,

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	1	1	55.499	-6.7637	0.25673	0.58371	86.339	1	0.000131	-0.0066521	0.38471	0.92304	
2	1	2	56.684	13.985	1.1122	-0.87997	97.442	1	0.0049763	0.030075	0.98755	0.1573	
3	2	1	55.276	-5.0267	0	0	96.389	1	-0.0015587	0.014326	0.46413	0.88577	
4	2	2	57.229	13.226	0	0	103.73	1	-0.0194	0.16096	-0.99754	0.070146	
5	3	1	55.245	-4.6705	-0.42651	-0.49495	99	1	0.00069473	0.0013312	0.44851	0.89378	
6	3	2	57.119	13.164	-1.7882	1.6944	101	1	-0.023058	0.16719	-1	-1.22E-16	
7	4	1	54.878	-4.3277	0.1081	-0.37019	104	1	0.00050186	-0.0006713	0.41687	0.90897	
8	4	2	57.145	12.968	-2.1517	3.7531	101	1	-0.023633	0.13363	-0.99246	-0.12255	
9	5	1	54.953	-4.1089	2.3791	0.75698	95	1	-0.0005307	0.00089688	0.40085	0.91614	
10	5	2	57.063	13.049	-3.6376	3.66	100	1	-0.025411	0.14411	-0.96083	-0.27714	
11	6	1	54.78	-3.4952	0.45362	6.3239	92	1	-0.0012489	0.0034225	0.40085	0.91614	
12	6	2	56.909	12.652	-2.8358	0.61514	106	1	0.027038	0.10735	0.89768	0.44065	
13	7	1	54.574	-2.7577	1.4282	4.8254	97	1	-0.0014674	0.0059003	0.40085	0.91614	

Figure 2: Output of the tracker is stored in a csv file where each **row** corresponds to a single target in a single frame. Table 1 lists the column ids.

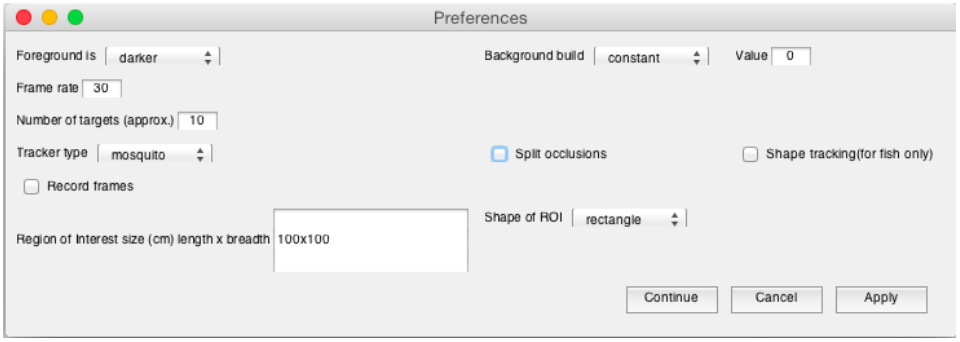

Figure 3: Preferences

exp01_002.bmp, exp01_003.bmp, and so on. If recording with multiple cameras make sure to append the filename with the camera identifier, for example, exp01_cam01_001.bmp, exp01_cam01_002.bmp, and so on. (Also, see video initial_setup.mov)

1. Ensure all video frames are in a single folder.
2. Change MATLAB current folder to where peregrine is located.
3. Type *peregrine* on the command window and select File → open. Select any frame from the list of frames.
4. Press start to check if the video is playing properly.
5. Check the Background checkbox. Now, using the sliders "Target contrast" and "Target intensity" try to have the foreground only visible as white blobs on the screen. Set "Target contrast" to 0 if the targets are moving very slowly or remain still in some frames.

Table 1: Column identifiers in the csv file output

column no.	variable
1	frame no.
2	target-id
3	horizontal-coordinate (cm)
4	vertical-coordinate (cm)
5	horizontal-velocity (cm/s)
6	vertical-velocity (cm/s)
7	size (pixels)
8	1 (homogeneous coordinate to keep things linear)
9	shape parameter 1
10	shape parameter 2
11	heading (x component)
12	heading (y component)

6. Use the crop tool next to the background to Crop a region of interest; you may also remove unwanted areas within the region of interest using the button next to it (not recommended as this will also make the targets invisible if they enter such region, however recommended if there is a persistent noise source in that area).

7. Select the "Mark" radio button in the View type to see if all targets are selected (Fig. 4). Verify the count status on the right of the "Live" button.

8. Select Edit $\rightarrow$ preferences to set preferences (Fig. 3). Most values can remain as default except "Region of Interest (ROI) size (length, breadth in cm)" and "Shape of ROI (0=square, 1=circle)", which should be specified as per the video. Setting "Split occlusions=1" allows occlusions to be resolved but should only be used if the target shapes can be approximated as ellipses (e.g. fish, bugs)

9. If you obtain consistent targets and most of your targets are seen marked in the "mark" view-type, then press save and move to Tracking.

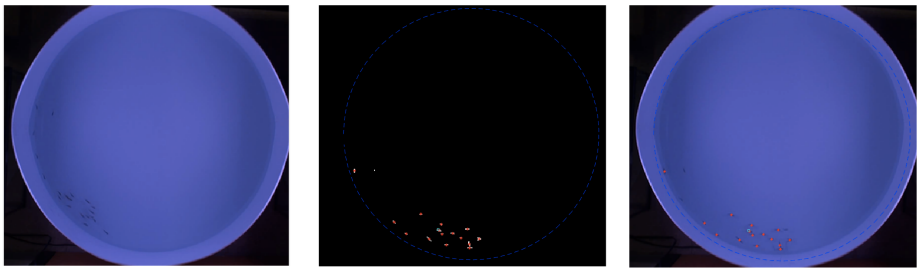

Figure 4: Setting thresholds to achieve the best possible tracking performance. (a) raw video; (b) with background checked and thresholds set; and (c) after selecting "mark" in the view type. Note the circular region of interest (ROI) as a blue dotted line. Video data from [7].

3 Tracking

Peregrine allows two types of tracking: position and shape. Position tracking outputs the target position and velocity for multiple targets and shape tracking outputs shape information in the form of coefficients of a parabola fit on the target midline.

3.1 Position and velocity tracking

Position and velocity tracking uses a Kalman filter [8], and global data-association [9] to track multiple targets in a video. To start tracking (also, see video initial_setup.mov):

1. Check that all preferences are set properly including ROI size and shape and most targets are marked in most of the frames. Ensuring this will reduce the time spent in repairing the tracks.

2. Now bring the slider to zero position and view-type to track. If you want to run the tracker without recording to see how the tracker performs press start. If everything looks okay, then stop, bring the slider back again and check the checkbox next to the track view to record the tracks.

3. Now press start again. To speed up the tracker, you can toggle the "live" button. **Make sure to press save once the tracking is done**.

4. If three-dimensional reconstruction is to be obtained, then it is critical that both tracks start and end at the same time

3.2 Shape tracking

The shape tracking follows the approach described in [6]. Briefly, the measurements consist of a centroid, orientation, and two shape parameters that represent a parabola in the body frame. The parabola and corresponding orientation is found by brute-force search. The final measurements are fed into a particle filter with 200 particles. In the case of Shape tracking the csv file will have four extra parameters corresponding to shape in the output file.

3.3 Three-dimensional tracking

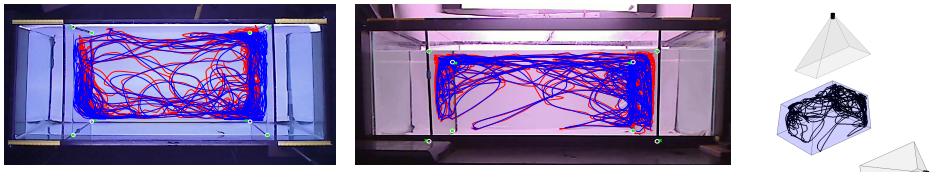

Figure 5: Three-dimensional reconstruction.

Trajectory data can be used from two different camera views to reconstruct full three-dimensional position of targets (Fig. 7). In order to reconstruct 3D data, the cameras must be calibrated. This section explains camera calibration steps and the procedure to obtain three-dimensional data.

3.3.1 Camera calibration

Each camera has intrinsic parameters (focal length, location of centroid, distortion coefficients) and extrinsic parameters (orientation, position) that must be determined to estimate the position of an object within the camera frame. For a single overhead view, we use an orthographic projection model. In an orthographic projection, assuming that the target moves primarily in a two-dimensional plane that is parallel to the camera plane, the pixel positions $\mathbf{u} \in \mathbb{R}^2$ is related to real world coordinates $\mathbf{r} \in \mathbb{R}^3$, with $r_3 = d$ constant depth, through a linear relationship [10]

$$\mathbf{u} = K\mathbf{r}, \tag{1}$$

where

$$K = \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

f_u, f_v is the focal length along the horizontal and vertical camera axis, $[u_0, v_0]^T$ is the principal point denoting the center of the camera sensor, and d is the depth of the two-dimensional plane on which the targets are being filmed.

For multiple cameras we use a perspective projection model [10, 3]. In this case, a three-dimensional point by $\mathbf{r} \in \mathbb{R}^3$, is mapped to a two-dimensional pixel position $\mathbf{u} \in \mathbb{R}^2$ through a nonlinear function. In a distortion-free form, a perspective projection model has four intrinsic parameters, namely, focal length (f_u, f_v) and principal point $([u_0, v_0]^T)$, and six extrinsic parameters, namely position $\boldsymbol{t} \in \mathbb{R}^3$ and orientation $R \in SO3$, where $SO3$ is the special orthogonal group of rotations. A model that includes distortion and skew in camera axes will include three more intrinsic parameters, namely the distortion coefficients $\{k_1, k_2\}$, and skew coefficient α_c. The distortion-free pixel image is related to real world coordinates by a nonlinear mapping [11]

$$s\tilde{\mathbf{u}} = K \begin{bmatrix} R & \mathbf{t} \end{bmatrix} \tilde{\mathbf{r}}, \tag{3}$$

where K as before is the 3×3 camera matrix, s is a scaling factor, and $\tilde{\mathbf{r}} = \begin{bmatrix} \mathbf{r}^{\mathrm{T}} & 1 \end{bmatrix}^{\mathrm{T}}$ and $\tilde{\boldsymbol{u}} = \begin{bmatrix} \boldsymbol{u}^{\mathrm{T}} & 1 \end{bmatrix}^{\mathrm{T}}$ denotes the homogeneous representation. The 3×4 matrix $P = K \begin{bmatrix} R & \boldsymbol{t} \end{bmatrix}$ is called the projection matrix [10]. Solving the camera calibration problem requires estimating the values of $R, \boldsymbol{t}$ and $\boldsymbol{r}$ given a set of points $\boldsymbol{u}_i$. This is a under-determined nonlinear optimization problem that requires additional constraints in order to be solved. In addition, for a multi-view setup the values of $R, \boldsymbol{t}$ must be transformed to a common inertial frame.

Calibration involves recording videos of a checkerboard after the cameras have been set as per the experimental setup. It is important that the setup stays the same throughout the full experimental study. This includes camera position, orientation, video resolution, frame rate, and lighting. Any change in the experimental setup will require recalibrating the cameras.

For calibration we will use the MATLAB calibration toolbox [12]. The input to this toolbox are a set of images with checkerboard in varying position/orientations; the output (after passing through a custom-made script) is a calibration file that stores camera properties in the form of calibration matrices[1]. (The toolbox is available in the *TOOLBOX_calib* folder within peregrine). Use the toolbox to perform calibration as described in the documentation for each camera `http://www.vision.caltech.edu/bouguetj/calib_doc/index.html`.

[1]Intrinsic calibration usually needs to be done less frequently than extrinsic calibration. We do intrinsic calibration for underwater experiments in water to reduce effects due to refraction. Do intrinsic calibration again if there is marked change in the environment such as temperature and humidity

For intrinsic calibration for each camera:

1. Connect the camera and start recording video
2. Move the checkerboard slowly in air at different positions, orientations in the camera field of view.
3. Start-stop record (small video). Export frames into a single folder with the camera name. Typically there should be at least 30 frames with high variability in the positions for each camera.
4. Follow the calibration steps as described in the MATLAB calibration toolbox [12].

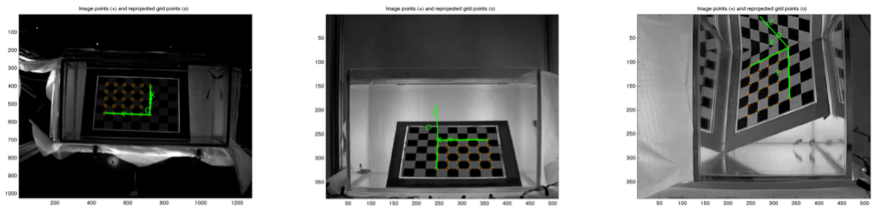

Figure 6: Calibration

For extrinsic calibration:

1. Mark the calibration board to help you outline the same section of the checkerboard in multiple cameras (see arrow in fourth vertical column the figure 6. I used it to select the same section in each frame)
2. Put the checkerboard in tank (with water in it) and visible in at least two cameras.
3. Do a quick start and stop recording.
4. Repeat steps 2 and 3 between combinations of two cameras until you cover all cameras. e.g. if you have three cameras one possible combination would be camera 1 and camera 2, then camera 2 and camera 3. If you have two cameras just do camera 1 and camera 2.
5. Put combination of frames representing the same time instant from each camera in a separate folder.

Run the script *get_camera_calib*. (Also, see camera_calibration.mov) To ensure that the orientation is also correct, follow the same order of corners in each view (note how the frame orients the same way in all views in Fig. 6. The script will dump a calibration file (camera_calibration.mat) in the location where extrinsic images are stored.

3.3.2 Three-dimensional reconstruction

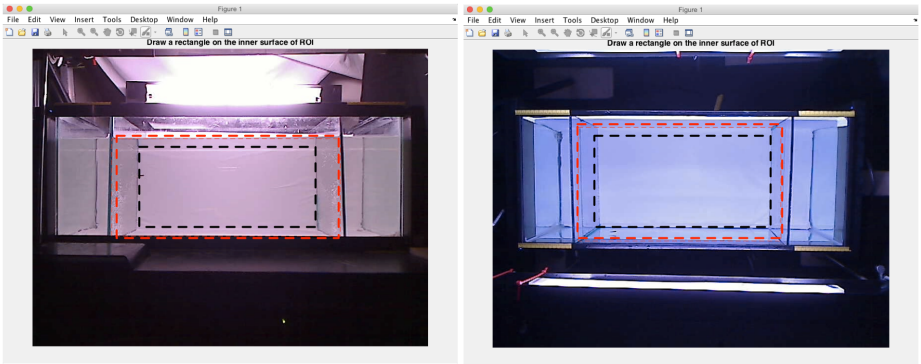

Figure 7: Region selection (outer=red and inner=black surface of ROI) during 3D reconstruction.

The current version only supports single target reconstruction for target id 1.

Select File → Reconstruct (3D). Note that camera 1 should be the same as in the calibration. The script assumes that camera calibration, and tracking on individual views has been performed. Each view has a separate folder containing the frames, X00 data file, and the configuration file. To further optimize the calibration, the user is asked to draw rectangles corresponding to the outer and inner surface of the region of interest. Note that the data output from 3D reconstruction is in mm, not cm.

3.4 Command line mode

pg_cmd

4 Repair mode

The repair mode allows the user to fix the tracks (See also section 5.) Tracks may be repaired by:

Auto repair: This function will remove spurious tracks and clean up the output file. Follow the instructions. This function assumes that the calibration was performed in cm. Run this prior to any manual repair.

Stitching broken tracks: Broken tracks result when a target is not tracked for a certain amount of time. In this case the id of the target will change with some missed frames in between. To fix this, select Repair $\rightarrow$ Select target, and then follow it until the track stops. At this point the instruction panel will become red, asking user to input. Select Repair $\rightarrow$ Add point, until the target is again retracked. Switch to this track when available (see Switch tracks below).

Switching tracks: Switching tracks is needed before and after an occlusion that was not resolved properly. To switch a track, simply select Repair $\rightarrow$ Switch, and the ids will be switched.

Adding a new track: In cases where the target is not tracked in the first frame, a new track may be created by simply selecting Repair $\rightarrow$ Add point. However, no prior target must be selected. To ensure that no target is already selected, click Repair $\rightarrow$ Clear id, and then click on the untracked target.

5 Keyboard shortcuts

Keyboard shortcuts allow faster use (navigation and repair) once you have used peregrine for a few videos.

6 Global observations

Once trajectory data is available, the command *get_obs* computes the various measures of collective behavior. We assume that both position and velocity of each target (for example fish) is available at all times during the experiment. We also assume that the each fish identified by a unique number that can be used to quantify the behavior during the trial. Unless specified, we make the assumption that the heading is the same as the direction of motion. (This assumption is dropped, for example, when the fish are in a water tunnel.)

1. *Group cohesion:* The degree of cohesion in groups is described in terms of the average nearest-neighbor distance (ANND) [13, 14]. Given the two-dimensional position $\boldsymbol{r}_i[k]$ of the i-th fish at frame k, the ANND during that

Table 2: Keyboard shortcuts (the window should be active)

Action	Shortcut key
Navigation	–
Start/Stop	spacebar
Next/Previous frame	left arrow / right arrow
Save	ctrl+s
Repair mode	–
Select target	t
Switch/update target	s
Add a point	a
Delete next few frames	d
Clear id	c

frame is [14]

$$\mathrm{ANND}[k] = \frac{1}{N}\sum_{i=1}^{N} \min_{j\in\{1,\ldots,N\}, j\neq i} (\|\boldsymbol{r}_i[k] - \boldsymbol{r}_j[k]\|), \tag{4}$$

where N is the total number of fish and $\|\cdot\|$ denotes the standard Euclidean norm.

2. *Group coordination:* is captured through the polarization (Pol) that measures the degree of alignment between the directions of motion of the fish [15]. Given the two-dimensional velocity $\boldsymbol{v}_i[k]$ of the i-th fish at frame k, the polarization at that frame is

$$\mathrm{Pol}[k] = \frac{1}{N}\Big\| \sum_{i=1}^{N} \hat{\boldsymbol{v}}_i[k] \Big\|, \tag{5}$$

where $\hat{\boldsymbol{v}}_i[k] = \frac{\boldsymbol{v}_i[k]}{\|\boldsymbol{v}_i[k]\|}$ is the direction of motion.

3. *Group speed:* is computed to describe average speed of the subjects, and is given by

$$S[k] = \frac{1}{N}\sum_{i=1}^{N} \|\boldsymbol{v}_i[k]\| \tag{6}$$

4. *Group angular momentum*: the normalized group angular momentum $\hat{L}$ is indicative of milling patterns [16, 17], with $\hat{L} = 1$ for single milling formations and $\hat{L} = 0$ for a coherent one-directional formation [16]. Depending on how $\boldsymbol{r}_i$ is defined, the angular momentum can be computed about the group center of mass [18], or about the center of the observation region.

$$\hat{L}[k] = \frac{\| \sum_{i=1}^{N} \boldsymbol{r}_i[k] \times \boldsymbol{v}_i[k] \|}{\sum_{i=1}^{N} \|\boldsymbol{r}_i[k]\| \|\boldsymbol{v}_i[k]\|} \tag{7}$$

5. *Group interaction:* with a focal fish (or an external stimulus such as a robot) is quantified in terms of values such as distance, alignment and relative speed. We denote the focal fish or target index with a subscript $\boldsymbol{r}_f$. Specifically the observables are

   (a) *Average distance to the focal fish:* Average distance of all other fish to the focal fish

   $$\hat{d}[k] = \frac{1}{N-1} \sum_{i \in \{1,\ldots,N\}, i \neq f} \|\boldsymbol{r}_i[k] - \boldsymbol{r}_f[k]\| \tag{8}$$

   (b) *Minimum distance to the focal fish:* Minimum distance to the focal fish between all other fish

   $$\tilde{d}[k] = \underset{i \in \{1,\ldots,N\}, i \neq f}{\operatorname{argmin}} \|\boldsymbol{r}_i[k] - \boldsymbol{r}_f[k]\| \tag{9}$$

   (c) *Group relative speed with respect to the focal fish:*

   $$\tilde{S}[k] = \Big(\frac{1}{N-1} \sum_{i \in \{1,\ldots,N\}, i \neq f} \|\boldsymbol{v}_i[k]\| \Big) - \|\boldsymbol{v}_f[k]\| \tag{10}$$

   (d) *Alignment to the focal fish:*

6. *Group behavior:* quantifies on an average the behavior of subjects such as percentage time spent freezing, thrashing, or swimming. These values allow us to determine if the subjects were in general scared or not. The specific observables are:

   (a) *Time spent freezing:* To elucidate the behavior of the subjects, we score the time spent freezing during each trial. This value is computed automatically based on the condition that a fish is considered freezing during a frame if it spends two continuous seconds within a ball of radius of 2 cm [19]. Freezing is recorded if at least one fish satisfied this condition.

(b) *Excursions of focal from shoal:* To measure the number of excursions from the shoal of a particular fish, we use the definition of shoal membership as described in [20]. In particular, we (i) compute the time series of average nearest neighbor distance of the focal fish,(ii) distribution of ANND for all fish over all frames is computed. Use the mode of this distribution as the threshold, divide the time series into segments where the individual's ANND goes above the mode, and (iii) find the distribution of the maximum ANND of each segment. The upper p=0.05 quantile of the max ANND distribution determines if a segment is an excursion or not.

7. *Tail-beat frequency:* Tail-beat frequency is computed using the tail-tip position estimated from the body frame location. For a signal x, the *fft* routine implements the following DFT

$$X(k) = \sum_{j=0}^{N-1} x(j)\omega^{jk}, \tag{11}$$

where $\omega = \exp(-2\pi i/N)$ and $k \in \mathbb{Z}$, and $x(j)$ is the coefficient for the sinusoidal frequency ω^{jk}, which is equal to jk/N. Since matlab returns a double sided spectrum of magnitudes, the amplitude is obtained by doubling the magnitude of $X(k)$. Also refer to the example in `http://www.mathworks.in/help/matlab/ref/fft.html`.

7 Acknowledgement

The authors would like to acknowledge assistance (development and testing) from Dynamical Systems Laboratory, New York University, and Collective Dynamics and Control Laboratory, University of Maryland.

References

[1] S Butail and D A Paley. Three-dimensional reconstruction of the fast-start swimming kinematics of densely schooling fish. *Journal of the Royal Society Interface*, 9(66):77–88, 2011.

[2] S Butail, N C Manoukis, M Diallo, J M C Ribeiro, T Lehmann, and D A Paley. Reconstructing the flight kinematics of swarming and mating in wild mosquitoes. *Journal of the Royal Society Interface*, 9(75):2624–2638, 2012.

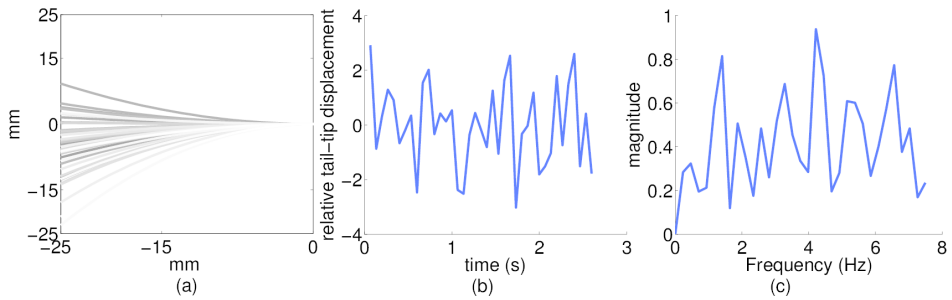

Figure 8: Tail-beat frequency computation: (a) tail end of the fish in successive frames (increasing in time as the color changes from grey to dark, with 0 marking the fish body center) when the fish was not occluded and away from the tank wall; (b) detrended relative tail-tip position during that sequence; and (c) Fast Fourier transform of the tail-tip position for analysis in the frequency domain. The frequency with the maximum magnitude (here 4.2 Hz) is selected as the tail-beat frequency

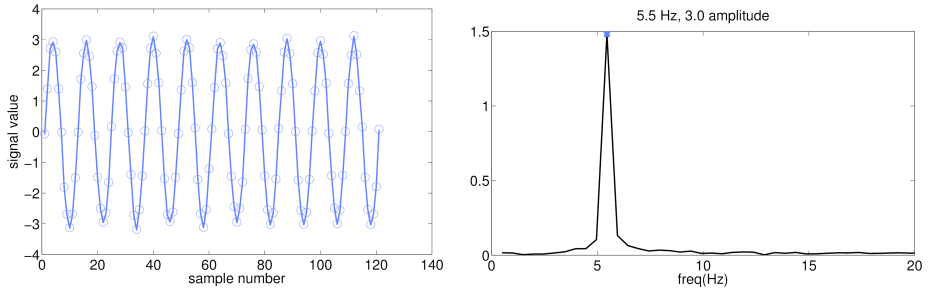

Figure 9: The toy example above shows how FFT is used to compute frequency and amplitude from a given signal. A signal with known frequency (5 Hz) and and amplitude (3 units) is sampled at 60 Hz after corrupting with zero-mean Gaussian noise of 0.1 units standard deviation. The signal is then detrended to remove linear trends. The circles on the left mark the detrended points on the noisy signal. The right plot shows the magnitude after applying the fast fourier transform along the length of the signal. The signal frequency is selected as the one where the magnitude is maximum (5.5 Hz) and the amplitude is twice the magnitude.

[3] S Butail. *Motion Reconstruction of Animal Groups: From Schooling Fish to Swarming Mosquitoes*. PhD thesis, University of Maryland, College Park, 2012.

[4] S Butail, T Bartolini, and M Porfiri. Collective response of zebrafish shoals to a free-swimming robotic fish. *PLoS One*, 8(10):e76123, 2013.

[5] F Ladu, S Butail, S Macrì, and M Porfiri. Sociality modulates the effects of ethanol in zebrafish. *Alcoholism, Clinical and Experimental Research*, 38(7):2096–2104, 2014.

[6] T Bartolini, S Butail, and M Porfiri. Temperature influences sociality and activity of freshwater fish. *Environmental Biology of Fishes*, 98(3):825–832, 2015.

[7] N Miller and R Gerlai. From schooling to shoaling: patterns of collective motion in zebrafish (\emph{Danio rerio}). *PLoS One*, 7(11):e48865, 2012.

[8] Y Bar-Shalom. *Tracking and data association*. Academic Press Professional, Inc., San Diego, CA, USA, 1987.

[9] H W Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, 2:83–97, 1955.

[10] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[11] Z Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 666—-673 vol.1, Kerkyra, Greece, sep 1999. IEEE.

[12] Jean-Yves Bouguet. Camera Calibration Toolbox for Matlab, 2010.

[13] M M Webster, J Goldsmith, A J W Ward, and P J B Hart. Habitat-specific chemical cues influence association preferences and shoal cohesion in fish. *Behavioral Ecology and Sociobiology*, 62(2):273–280, 2007.

[14] A Kolpas, J Moehlis, T A Frewen, and I G Kevrekidis. Coarse analysis of collective motion with different communication mechanisms. *Mathematical Biosciences*, 214(1-2):49–57, 2008.

[15] T Vicsek, A Czirók, E Ben-Jacob, I Cohen, and O Shochet. Novel type of phase transition in a system of self-driven particles. *Physical Review Letters*, 75(6):1226–1229, 1995.

[16] Y Chuang, M R D'Orsogna, D Marthaler, A L Bertozzi, and L S Chayes. State transitions and the continuum limit for a {2D} interacting, self-propelled particle system. *Physica D: Nonlinear Phenomena*, 232(1):33–47, 2007.

[17] I D Couzin and N R Franks. Self-organized lane formation and optimized traffic flow in army ants. *Proceedings of the Royal Society B: Biological Sciences*, 270:139–146, 2003.

[18] M Aureli, F Fiorilli, and M Porfiri. Portraits of self-organization in fish schools interacting with robots. *Physica D: Nonlinear Phenomena*, 241(9):908–920, 2012.

[19] V Kopman, J Laut, G Polverino, and M Porfiri. Closed-loop control of zebrafish response using a bioinspired robotic-fish in a preference test. *Journal of the Royal Society Interface*, 10(78):20120540, 2013.

[20] N Miller and R Gerlai. Redefining membership in animal groups. *Behavior Research Methods*, 43(4):964–970, 2011.