Tracking *C. elegans*

Valerie Simonis

June 12, 2014

Abstract

This paper presents and discusses the challenges and primary considerations in developing a single-worm motorized tracking system in the aim of recording and quantifying *C. elegans* locomotory behavior given certain experimental manipulations.

1 Introduction

The nematode *Caenorhabditis elegans* is a popular organism for neural studies due to its simple structure (959 somatic cells) and known connectome (302 neurons and 8000 connections). Traditionally, this nematode is studied to model its behavioral output, observed through motion, given different experimental manipulations. Given that a nematode's behavior is only modulated by internal states (genetics) and external stimuli (such as presence or absence of food or touch-stimulus), manipulating these states and stimuli can lead to discovery about neural circuits' purposes and activity. For example, mutations can be applied to a worm to simulate human diseases like Alzeihmer's, Autism Spectrum Disorders and other neural-disruption diseases. The impact of such genetic manipulations can then be measured to infer linkages between genes, neurons and behavior. *C. elegans* related research often aims to discover which neurons control behavior in the hope that findings can be generalized to aid in the development of a human brain activity map to better understand diseases caused by neural disruption.

A common practice in the *C. elegans* Neuroscience research field is to quantify a worm's locomotory behavior using Computer Vision techniques. This interdisciplinary approach hopes to provide more precise measurement and description tools of *C. elegans'* locomotory behavior to make inferences in the field of Neuroscience.

Specifically, our collaborator, Dr. Kim at Rosalind Franklin University of Medical Sciences, wishes to know which neurons affect the food related behavior of *C. elegans*. In order to discover this, he proposes to quantify a wild-type worm's off-food behavior and find similarly behaving mutant worms when placed in an on-food environment. For our purposes, worm food is an *E. coli* bacterial lawn spread on an agar plate assay. Knowing which worm(s) from a list of possible candidate mutants behave on-food as the wild-type (naturally occurring) off-food, will suggest to Dr. Kim which neurons are part of the food-related behavior neural circuit. Knowing how similarly these worms behave will infer the strength of certain neural paths over others under the food-presence condition. In order to observe and quantify the intricacies of *C. elegans'* food related behavior, Dr. Kim needs a system to record, measure, quantify and model *C. elegans* behavior through video and image processing.

2 Background

Due to the ineffectiveness of humans to observe and record worm activity with precision over either long or short time scales, it has become almost standard to record freely moving *C. elegans* either one at a time or in a group. These videos are then processed after the fact, to segment the worm and provide intermediate representations

of the worm's body (binary image, skeleton and sampled skeleton points). Features are then extracted from the intermediate representations and further mined for patterns to quantify phenotypes as in [1] or to support hypotheses linking genetic manipulation to observed locomotory behavior.

Some recording implementations ([2, 3]) focus on one area of the dish, negating the need to recenter the worm(s) under study. Quantitative methods developed for this type of system are used to track the centroid of multiple worms only in that area of the assay and do not require a motorized tracker. Other implementations ([4]) used either a motorized stage (usually when using a microscope providing this functionality) or a motorized X-Y translation stage upon which a camera is affixed. Moving the camera over the stage (containing the assay and thus animal under study) is preferred to minimize the possible effects of the mechanical motion on the nematode's locomotory behavior.

We wish to record one worm at a time and are interested in comparing his locomotory behavior off-food to candidate mutant worms on food. It has been suggested in [5] that wild-type *C. elegans'* off-food behavior is remarkably different from an on-food environment. Worms under this condition can travel longer distances in more frenetic behavioral patterns (ie: fast with many turns), requiring a larger dish to give the nematode space to search for food without colliding with the edge of the dish. Other research groups use an *E. coli* bacterial lawn (food) to restrict the area in which the animal will visit, and to minimize its maximal speed as explained in [6]. This highlights the importance of developing a system for which a minimal amount of parameters need to be tweaked for appropriate motorized tracking is minimal for any type of worm under any experimental condition. For this system to be useful to Dr. Kim, the motorized tracking parameters need to be generalizable to any mutant- or wild-type of worm regardless of speed and activity.

2.1 Acquiring Video for Analysis

A critical first step in analyzing and quantifying worm behaviors involves acquiring video. In order to capture subtle differences in individual animal behavior, it is of high importance to capture worm movement at appropriate time and imaging resolution under imaging conditions favoring accurate extraction of intermediate representations of the worm's per-frame posture.

A high and accurate frame rate will provide time scales suitable for motion analysis. It will also provide more image data from which to extract features, allowing data scientists to throw out video frames for which segmentation and skeletionizing is ineffective, in which the image has become corrupt, or the nematode has been blurred by the motion of the camera. Traditionally challenging postures to segment or skeletonize are looped or coiled. The practice of throwing out these frames is slightly misguided, however, as often the frames being thrown out are those of high importance for locomotory analysis.

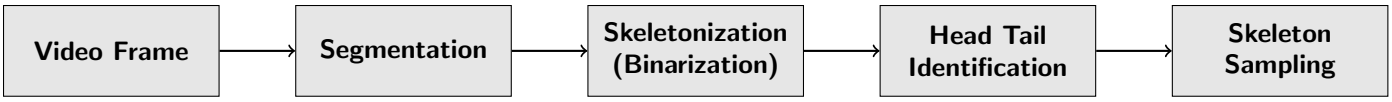

Figure 1: Standard image processing sequence for intermediate representations

In order to provide an advantage to the segmentation process, optimizing the image quality is important. This can be accomplished by providing even and uniform illumination at time of capture, facilitating the process of finding intensity thresholds that best separate a worm from its agar background. In order to minimize unwanted image artifacts effecting the selection of only worm pixels during segmentation, it is recommended to ensure proper care in the pouring of the agar (substrate upon which the worm is crawling), and imaging the plate with few fingerprints on the dish (trivial, but important!) Also, the dish should be imaged with the agar portion of the dish facing up, allowing possible condensation observed over long recording periods to travel downwards, minimizing its impact on the image quality.

Currently, most trackers available are written in MATLAB/C++ and require the Image Analysis Toolbox in order to acquire frames from a camera. Many require expensive hardware, making *C. elegans* motorized tracking an expensive and exclusive endeavour.

2.2 Quantifying Worm Behavior through Video Analysis

Though slightly outside the scope of our current investigation, it is important to note some of the standard methods for extracting features. Given our position to control the quality of the images to be provided to the feature extraction and analysis routines, some design decisions at our current stage may impact the effectiveness of further study.

A popular approach in the field is to generate decompositional features from the videos acquired [4]. In general, four "intermediate" representations are calculated using the image processing sequence from Figure 1.

The original gray scale image is only used to provide information about the animal's *translucency* (for head/tail identication, mainly). From the gray scale image, a binary image is generated by applying a threshold segmentation. From this image *size* and *shape* features are calculation (as **morphology features**). From the binary image, the centroid is found as a representation of the worm's *position* extrapolated to **trajectory features** across multiple frames (such as *direction, displacement, velocity and acceleration*). From the segmentation, a medial axis is determined using skeletonization techniques. This is a complex problem when applied to looped or curled postures in which occlusion might occur [7]. This skeleton is then downsampled to a number of representative points (anywhere between 13 [8] and 49 [1]) ordered from head to tail. The head and tail distinction can be automatically attempted using the presence of intensity, curvature or width profile differences between a nematode's head and tail, though this task is not inherently trivial.

Features extracted from these intermediate representations have been widely used by the Schafer research lab for phenotype analysis as inputs to: phenotype classification using decision trees [9] or random forests [6] or phenotypic clustering [10]. Also, these features can be used to annotate video frames with some commonly described behaviors such as crawling and turns (reversals, shallow turns, omega loops, and gamma loops.)

Other approaches to extracting features involve tracking only the worm's centroid frame-to-frame to get information about the worm's complete path (or trail). This path's curvature can be modeled as piece-wise harmonic functions [11] or using differential tangent angles and arc lengths [12].

Finally, [13] discovered that any worm *posture* can be represented using four basic shapes (or eigenworms). Both eigenworm and trail analyses model posture over time. An advantage to using eigenworms over trail analysis is these basic shapes are learned from all frames (so each frame contributes), but any individual frame can be modelled using this approach (frame-centric approach). Trail analysis models postures across multiple frames without frame-to-frame distinction. In [**stephens2010**], the eigenworm model is coupled with center-of-mass trajectory analysis.

Using unsupervised learning techniques, [14] uses eigenworm representations of the worm **postures** developed in [13] to identify and discover behavioral motifs (or patterns) to use in clustering phenotypes. I project we will possibly build on this approach, focusing on a smaller, more focused subset of candidate mutants of interest to Dr. Kim, instead of a wide variety of mutants as profiled in [1] in order to find model posture-similarity between two individual worms.

In terms of observing worms off-food, it has been attempted by [12] (details to follow) and [15] for 70 min for 3 mutant types in a 15cm assay at 3 fps. (We can do better).

Ultimately, given our goal of determining similarity between two worms based on their locomotory behavior, once we complete the image acquisition software, we will investigate the utility of many of the features described, including: **morphology**, **posture**, and **trajectory** features.

3 Results and Discussion

This section reports some of the more challenging aspects of this project as well as important design and implementation considerations.

3.1 Real-time Processing Requirement

Since the worm under study is only 1 mm long, it's impossible to watch its locomotory behavior without the intermediary of a microscope or zoom-lens mounted camera. This forces a requirement that the recording should occur with a known frame rate in order to ensure fidelity of time scales. This will ensure that a given behavior is represented

by the recording at the same speed at which it is occuring. A known, consistent frame rate will be important to quantify motion features dependent on time such as velocity and acceleration. For the tracking portion of this video acquisition system, the `wormFinder()` has to work fast enough to recenter the worm before it leaves the field of view.

Frame rate Given that this system needs to work real-time, the main development concern has been to acquire, process and record images from the camera in real-time at the frame rate delivered by the camera. However, 45 fps may supply many sequential frames in which the worm has barely moved a pixel. A consideration for whether or not to lower the frame rate lies in the length of videos we plan on acquiring. For our given problem, we will be recording video for a longer timescale than is traditionally done, between 45 min and 1h30 min. An important consideration for our system (both for acquisition and feature extraction) will be its robustness across a large data set. In the event we down-sample our recording (due to space or inability to capture 45 fps), we will need to weigh the computational and space benefit over the potential loss of image data (that will not be able to be later reconstructed).

Speed	Frame Rate (fps)								
	5	10	15	20	25	30	35	40	45
500 μm/sec	12.80	6.40	4.27	3.20	2.56	2.13	1.83	1.60	1.42
200 μm/sec	5.12	2.56	1.71	1.28	1.02	0.85	0.73	0.64	0.57

Table 1: Possible distance travelled between frames (in pixels) for different frame rates and maximal worm speeds (assuming 1280 by 960 resolution and 10mm f.o.v.

Guidance in determining an appropriate frame rate is supplied by Table 1. For Dr. Kim's suggested maximal worm speed of 200 μm/sec, a frame rate of 45 fps means that between frames, the worm would maximally have moved 0.57 pixels. The significance of recording such small intermediate motion has not been tested, but intuition suggests that capturing motion at that precise of a time scale, may be irrelevant to quantifying a nematode's locomotory behavior.

Our camera's maximal frame rate of 45 fps may be more appropriate for a published [4] maximal worm speed of 500 μ/sec. A worm traveling at that speed would have moved 1.42 pixels between frames, which may be an important motion to capture and record.

Image processing efficiency Early development of the system targeted using a Raspberry Pi as the controlling computer for both image processing and camera capture and recording. The memory available on the Raspberry Pi and the Python classes available to interface with the camera were very limiting in terms of processing. The experience, however, did inspire using computationally simple methods for finding the worm, over more common methods (and initially more robust methods) of finding the centroid of the worm from a binarized image.

The most expensive step in the image processing sequence is applying a Gaussian filter to the difference image. When initially developing the system, I was using a matrix data structure from the `numpy` Python package to hold the image. A Gaussian filter applied to a `numpy` image took much longer to process than using OpenCV's `CvMat` object. This would be attributed to OpenCV's efficient C++ implementation. The Gaussian step is vital in ensuring robustness to worm localization.

Another main component of processing images to keep up with real-time capture is to subsample the image stream. A bonus consequence is that this technique allows the worm to move a little bit before the next image is processed, limiting situations in which the image difference image subtracts a current worm image from its reference.

Recent tests on previously captured video have shown that when the search space is the entire image, worm finding can occur at 33 fps. This is improved when the search space is cropped to a minimum observed at 200 fps.

Limitations In the current version of the tracking software, the estimated frame rate is about 22 fps, well slower than the 45 fps promised by the camera manufacturer. This could be due to structure of the program and its inefficient handling of passing around states that need to be shared by different procedures. I am investigating using either a leaner Shared State design pattern or Observer design pattern to circumvent this challenge.

Another influence on frame rate is the interpreted nature of the Python language. Investigation produced evidence that the EBB `centerWorm()` procedure required a second to return after being called in `decideMove()`. During this time, the program waits for the `centerWorm()` method to return, delaying the entire capture process. To circumvent this problem, the Callback design pattern was implememted, sending `centerWorm()` to its own process without waiting for it to return. This allows for a more timely execution of the subsequent methods.

Also, in the event that frame rate is never metronomic, a log file specifying time stamps of each written frame as well as motor instructions is generated, allowing the Feature Extractor to have true time and distance information to accurately generate time and distance features such as velocity and acceleration. These log files are generated using Python's logging library which is well worth learning. The timestamp for writing images is important given the non-metronomic nature of the image processing procedure. As fast as the image processing is and can be, the frame rate estimate changes over time given the different time it takes to find a worm in the whole image vs. a cropped search space.

A more optimal solution to securing the actual frame rate supplied by the camera would incorporate total separation of the capture/recording and finder threads, combined into a main thread. Furthermore, design patterns under consideration will ease the process adding a GUI with controls (instead of just image display) would allow for ease in testing of some of the decision parameters discussed in **??**.

3.2 Hardware Integration

Testing the tracker is made difficult by the usual absence of one or more hardware elements (motors and camera) at time of testing (the complete tracking system lives at RFUMS.) Algorithms are often tested on previously recorded video. Certain conditional boolean controls such as color format of the input or the presence or absence of motors were necessarily built into the programmatic structure to allow for testing on multiple inputs (video or live camera).

3.3 Motor Considerations

An important design decision is to have a very large field of view compared to other experiments. We posit that this will result in fewer (albeit longer) motor movements. This is an important point to consider, given that frames in which which the camera is moving are often dropped for analysis as the camera motion results in a blurred worm. It might still be more beneficial to the analysis step to have more frequent, shorter movements.

3.4 Availability

It is important to note that the software has been developed and tested on a Unix system (Xubuntu 14.04) and may not be compatible to a Windows environment. In general, this may be a deterrent to researchers, as few are comfortable in that environment. One of our major selling points is going to be price point relative to the current systems available from domain juggernauts. However, our program will be free and open source while most others are open source, but dependent upon having MATLAB toolboxes that are costly for the image acquisition portion.

I would love to have this system be compatible with any PC so that any available existing computer in a laboratory could be used. The limiting factor, is going to be the camera. It may not be compatible in a Windows environment (though this hypothesis has never been specifically tested).

4 Conclusions

There may be quite a few programmatic changes to make to ensure good video (data) acquisition given many of the considerations discussed. I am hopeful that once this step is complete, more techniques of feature extraction independent of segmentation and skeletonization can be explored such as optical flow or using local invariant features such as SURF, SIFT and MSER which are more generalizable to other Computer Vision problems.

References

[1] Eviatar Yemini et al. "A database of Caenorhabditis elegans behavioral phenotypes". In: *Nat Meth* 10.9 (Sept. 2013), pp. 877–879.

[2] Daniel Ramot et al. "The Parallel Worm Tracker: a platform for measuring average speed and drug-induced paralysis in nematodes". In: *PLoS One* 3.5 (2008), e2208.

[3] George D Tsibidis and Nektarios Tavernarakis. "Nemo: a computational tool for analyzing nematode locomotion". In: *BMC neuroscience* 8.1 (2007), p. 86.

[4] Zhaoyang Feng et al. "An imaging system for standardized quantitative analysis of C. elegans behavior". In: *BMC bioinformatics* 5.1 (2004), p. 115.

[5] Jesse M Gray, Joseph J Hill, and Cornelia I Bargmann. "A circuit for navigation in Caenorhabditis elegans". In: *Proceedings of the National Academy of Sciences of the United States of America* 102.9 (2005), pp. 3184–3191.

[6] Wei Geng et al. "Automatic tracking, feature extraction and classification of C. elegans phenotypes". In: *Biomedical Engineering, IEEE Transactions on* 51.10 (2004), pp. 1811–1820.

[7] Kuang-Man Huang, Pamela Cosman, and William R Schafer. "Machine vision based detection of omega bends and reversals in C. elegans". In: *Journal of neuroscience methods* 158.2 (2006), pp. 323–336.

[8] Christopher J Cronin et al. "An automated system for measuring parameters of nematode sinusoidal movement". In: *BMC genetics* 6.1 (2005), p. 5.

[9] Joong-Hwan Baek et al. "Using machine vision to analyze and classify Caenorhabditis elegans behavioral phenotypes quantitatively". In: *Journal of neuroscience methods* 118.1 (2002), pp. 9–21.

[10] Wei Geng et al. "Quantitative classification and natural clustering of Caenorhabditis elegans behavioral phenotypes". In: *Genetics* 165.3 (2003), pp. 1117–1126.

[11] Venkat Padmanabhan et al. "Locomotion of C. elegans: a piecewise-harmonic curvature representation of nematode behavior". In: *PloS one* 7.7 (2012), e40121.

[12] Daeyeon Kim et al. "The shallow turn of a worm". In: *The Journal of experimental biology* 214.9 (2011), pp. 1554–1559.

[13] Greg J Stephens et al. "Dimensionality and dynamics in the behavior of C. elegans". In: *PLoS computational biology* 4.4 (2008), e1000028.

[14] André EX Brown et al. "A dictionary of behavioral motifs reveals clusters of genes affecting Caenorhabditis elegans locomotion". In: *Proceedings of the National Academy of Sciences* 110.2 (2013), pp. 791–796.

[15] Katsunori Hoshi and Ryuzo Shingai. “Computer-driven automatic identification of locomotion states in Caenorhabditis elegans”. In: *Journal of neuroscience methods* 157.2 (2006), pp. 355–363.