Quantifying the Accuracy of C. elegans Image Analysis

Jacob Graves and Roger Mailler
Department of Computer Science
University of Tulsa
Tulsa, United States
Email: jacob-graves@utulsa.edu, mailler@utulsa.edu

***Abstract*—The nematode *Caenorhabditis elegans* is an important model organism for many areas of biological research including genetics, development, and neurobiology. A common technique used in studying the locomotion of the worm is to take video of the worm in motion and analyze it to extract relevant data. A number of different software solutions exist to analyze these videos, yet there is no technique to determine the accuracy of the statistics being produced. We have developed a method to quantify the accuracy of a given analysis pipeline by using video of a biologically accurate simulation. Using this process we develop a metric to quantify the accuracy of a given pipeline, and we demonstrate this metric by comparing different implementations of a popular pipeline.**

***Keywords*-Image analysis; Feature extraction; Caenorhabditis elegans; Simulation**

I. Introduction

The nematode *Caenorhabditis elegans* is an important model organism for many areas of biological research including genetics, development, and neurobiology. Extensive research has been performed on video analysis and feature extraction on microscopic video of *C. elegans* locomotion. A number of different software solutions exist to analyze these videos, yet there is no method to determine the accuracy of the statistics being produced.

In this work, a method is developed to validate the accuracy of video analysis pipeline through use of an accurate simulated model of the worm. The ALIVE simulator is a simulation that produces a biologically accurate 3D model of the worm [6]. By using video from this simulator coupled with the data containing the exact underlying position and shape of the worm, we are able to correlate the raw video data with the actual values that describe the worm. We can then take this video and feed it into a given video analysis pipeline and compare its results to the actual values.

To test our method, we took a popular analysis pipeline and varied the thinning algorithm it uses to create skeletons. Then we validated each of these pipelines on three variations of worm movements all produced in the ALIVE simulator. We derived an accepted result for segment angle and segment angle velocities for the video clips, and compared that to the result measured from the pipelines. This exposed a few flaws in the accepted pipelines and allowed us to empirically measure the fitness of an analysis pipeline.

II. Background

A. Imaging

A common form of worm analysis involves tracking a single worm at a higher magnification over time to analyze its motion and behavior. One use for this research is in identifying different strains of worm based on their motion patterns over time. Since a complete neurological mapping of the worm exists, studying different mutants of the worm allows for a further understanding of the working of nervous systems. Many mutants cause disruption in the typical sinusoidal motion in *C. elegans*. These range from obvious changes in locomotion that can be spotted through a microscope, to very subtle changes that show themselves through a thorough statistical analysis of the worm's motion [9].

Tracking a single worm is also useful for researchers who are trying to abstract information about a worm's motion in order to artificially reproduce realistic locomotion in a simulator. For systems such as [8], [1], [5], a neurological representation of the locomotion functions of the worm is reproduced that must be verified against the locomotion of a worm *in vitro*. This requires an accurate and thorough set of statistics on worm motion to verify the images against.

Our system focuses on single worm tracking at a relatively high magnification and feature extraction about its forward locomotion. Imaging of *C. elegans* for this purpose is typically performed by a "pipeline" of effects through which a video is processed to produce a "skeleton" of the worm. Data analysis is performed on this skeleton and the prominent features are extracted. A current pipeline, utilized in [4], performs the skeletonization operation by thinning the binarized image of the worm, then discretizing the skeleton into the desired number of points. This skeleton is usually stored in an intermediate data file to be processed for a number of statistics such as amplitude, velocity, segment angle, and segment velocity. These are the four statistics we will compute and compare in this paper. This pipeline will be discussed in detail in Section III.

B. Simulators

Due to the relative simplicity of *C. elegans* and the ready availability of a complete neural mapping, frequent attempts

have been made to develop simulations of the organism. A number of different approaches have been taken in simulating the motion of the worm. Neibur and Erdos produced a simulation in part to show that the muscle activations that cause movement could be reproduced assuming the worm has stretch receptors that allow for the propagation of movement down the worm [7]. Bryden [1] and Karbowski [5] developed simulators that make biologically accurate neural networks and apply them to their simulators to reproduce the range of motion for the organism. These simulations represent the body as a set of uniformly distributed points in two-dimensional space. This prevents them from replicating the proper weight distribution, and more importantly, the non-uniform placement of the muscles that are used to generate locomotive force in the actual worm. They also fail to directly simulate the environment, but instead apply constant frictional forces at these discrete points along the body.

To remedy this, the ALIVE simulator was developed. This simulator uses a biologically accurate three dimensional model of the worm and implements physically accurate interactions of the worm with its environment. The simulator produces photo-realistic images of the worm moving (for the context of image analysis). The simulator will be discussed in more detail in Section III-B.

III. Materials

A. C. elegans *Video Toolkit*

Currently, various software solutions exist to analyze and extract features from videos of the worm *C. elegans*. However, these software packages have not been widely accepted by the community because they are implemented with a single process for converting video into data, documented poorly, tend to be slow, and implemented using MATLAB's outdated, undocumented image processing algorithms.

By developing a standard and easy to use interface for creating and sharing worm analysis pipelines, we hope to foster a convenient and open comparison of existing pipelines. Also, a video analysis pipeline program has the potential to allow researchers without experience in programming to develop customized analysis pipelines. By incorporating multiple different algorithms for performing the same operations, it is easy to compare and contrast the benefits of competing algorithms for image manipulation. A pipeline that provides a way to view the processed image during each effect allows each effect to be more finely tuned, producing better overall results.

The software we have developed rectifies these deficiencies by providing a versatile toolkit of video analysis techniques where the end user can customize the analysis pipeline. Our software, written in Java, not only provides versions of the image analysis algorithms found in current software packages, but also provides alternate and improved techniques to allow the user to mix and match processing steps to accommodate their specific needs. Each effect includes settings and a preview function to allow a researcher to fine tune each effect to their video or microscope, allowing for more accurate and specialized results. Since the software is developed in Java, the program is lightweight and fast, especially compared to other solutions that use MATLAB image processing algorithms.

Our extensive toolkit contains common pre-coded effects used in *C. elegans* feature extraction, such as bend angles, segment length, and segment velocities. The visual interface allows easy connection of effects to create a custom video pipeline, which can then be saved and shared with other CVT users to allow a standard for comparing analysis pipelines. CVT also incorporates modern and legacy image manipulation algorithms. It takes input from video files and folders, allowing for real time or batch processing. The use of color coded "ports" clearly demonstrate the expected inputs and outputs for an effect, allowing easy creation and editing of pipelines.

To facilitate the pipeline creation and management process, our program provides an intuitive drag and drop interface that allows a user to select processing algorithms, set their parameters, and connect them together. Users can then test and refine their pipeline by viewing the video at each stage of processing. The output of the process is a standardized file format that contains the key characteristics of the worm's motion in a format that is compatible with statistical analysis software.

For developer support, we have also incorporated a standard interface for designing and plugging effects into the toolkit. This allows any effect to be inserted into a pipeline, including effects that may output data in a custom format. This flexibility allows any pipeline and data format to be supported with minimal Java coding.

B. *ALIVE Simulator*

Validating a pipeline requires a realistic depiction of the worm in which we know the underlying values of the worms position for each frame. To do this, a biologically accurate model of the worm was required. This was obtained by using the ALIVE simulator developed at the University of Tulsa. This powerful simulator was developed as a biologically accurate simulation of the worm *C. elegans*. This simulator produces realistic locomotion of a three dimensional model of an adult *C. elegans*.

As discussed earlier, other simulations have done a number of significant simplifications in order to model the worm's motion, including depicting the worm as a uniform two-dimensional model. These simplifying assumptions limit the ability of these simulations to accurately depict the non-uniform friction that results from the worm's contact with the world around it and subsequently the complex neural control that is needed to generate the worm's characteristic sinusoidal pattern of locomotion.

Leveraging the tremendous increases in computational power and advances in numeric methods, the ALIVE simulator rectifies these deficiencies by representing a biologically accurate 3D model of the body of *C. elegans* in a virtual environment that mirrors the physical properties of its natural world. This simulator, which has been under development for nearly two years, is built using an open-source 3D game and physics engine. The model accurately depicts the physical properties of the real organism including its nonuniform weight, size, shape, and musculature. In addition, the simulator models the interaction between the worm and its environment to include surface tension, friction, inertia, and gravity. The simulator faithfully reproduces forward and reverse locomotion of *C. elegans* on an agar surface, and the model is cross validated using video recordings of worms that were converted to quantitative data by image analysis software [6].

The worm model itself contains 25 discrete segments of biologically correct length. This is then powered by a sine wave propagated down the worm to simulate forward locomotion. This has been tuned so that the worm moves in a realistically validated way. The frequency and update rate can be easily adjusted to create unusual movement patterns that can make for theoretically realistic movement variants [6].

To further test the various pipelines, we decided to use the power of the ALIVE simulator to create three distinct variants of worm motion to test the versatility of the pipelines. The first worm variant is the typical forward locomotion of the *C. elegans*. The ALIVE simulator by default creates a worm and moves it in the form of a typical adult worm. The simulator does this by calculating muscle activations by propagating a sin wave down the different segments of the body. This creates a typical movement that has been cross validated with video of real worms in forward locomotion. This is the typical configuration of the worm that is analyzed in most video analysis pipelines, and the data that is extracted represents such statistics as amplitude, velocity, bend angles, and bend angle velocities. The second variant moves with very high bend angles and a low bend angle velocity. This makes for a very slow moving worm that appears to make many pirouettes. This covers the pipeline's ability to handle very high bend angle situations. The last variant moves with very low bend angles, but at a very high speed. This worm quickly moves its segment back and forth which allows the worm to quickly move across the plate. This motion is not necessarily represented in the motion of the real worm, but it does put the worm in low angle situations that tests the sensitivity of the pipeline.

These three variants demonstrate a range of the worm's potential motion and test the sensitivity of the pipeline to changes in the worm's style of motion.

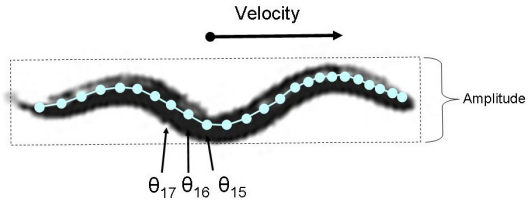

Figure 1. Features of *C. elegans* locomotion

C. WormAnalyzer

To produce the statistical analysis of the skeleton files, we used the WormAnalyzer software that was also developed at The University of Tulsa. This software package takes in skeleton files and batch processes them to produce the relevant statistics. The software extracts 49 features from the worm's motion, four of which are relevant here: amplitude, velocity, segment angle, and segment angle velocity. Amplitude is the measure of the maximum amplitude at any given time. After a file is processed the average amplitude for the worm over the duration of the file is given. Velocity measures the worm's magnitude of velocity (speed) from one frame to the next by tracking the worm's centroid. The average velocity over the file is reported. For the segment angles, one measurement is produced for each joint in the worm's skeleton where two segments meet. This segment angle is a measurement of a single joint's angle at each frame. The average angle over the course of the video is reported. The segment angle velocity is the rate of change of segment angle over time. The average angular velocity over the course of the file is produced. The software outputs these statistics, which can then be further analyzed using statistical packages [6].

IV. Methods

A. Pipeline Creation

The analysis pipeline we constructed is similar to that described in Geng et. al [4] with some minor differences. To create the pipeline, we used the CVT software described in Section III-A. The first element in the pipeline is an input cell that allows a batch process of videos to be read in and analyzed. Upon each new video a message is sent to all the nodes in the pipeline so that they can initialize in whatever means appropriate. Each frame of the videos are read in and then passed along to the next pipeline processor until the output nodes are reached.

First, each frame is converted from color to grayscale using a simple combination of the 24-bit Red/Green/Blue values for each pixel. The next processor in the pipeline performs a binarization of the video using a local thresholding algorithm. This algorithm uses a sliding 3 X 3 window to determine whether a given pixel represented by a single grayscale value should be converted to black (foreground) or white (background).

Table I
SEGMENT ANGLE ERROR OF DIFFERENT THINNING METHODS

	OPATA-8		SPTA		TPTA	
(in degrees)	PEV	MaxErr	PEV	MaxErr	PEV	MaxErr
Default	1.90	4.36	2.07	7.44	2.03	5.55
Large Ang	1.87	4.62	2.48	9.63	2.14	6.40
Small Ang	1.76	4.01	1.88	6.17	2.07	4.52

Table II
SEGMENT VELOCITY ERROR OF DIFFERENT THINNING METHODS

	OPATA-8		SPTA		TPTA	
(in degrees)	PEV	MaxErr	PEV	MaxErr	PEV	MaxErr
Default	11.28	38.76	15.87	49.99	13.37	46.75
Large Ang	27.82	92.66	32.01	109.48	27.92	95.00
Small Ang	7.79	23.57	13.01	42.18	10.77	33.80

This processor colors the pixel black if the standard deviation of the intensity of the pixel and its surrounding pixels is greater than the mean of the entire image, or if the mean intensity of the pixel and its surrounding pixels was greater than the background pixel intensity.

Next, we remove small objects from the image using a region labeling algorithm that indexes each pixel in the image according to the region to which it belongs. Once all of the black regions are labeled, we remove all of the regions except for the largest. This isolates the worm (which is black) onto a white background. We then used the same method to fill holes in the worm's image by inverting the colors such that only the background region is maintained.

With a binary image of just the background and worm in hand, we apply a thinning algorithm to reduce the worm's image down to a core body that represents the worm's basic shape. A number of options are available to thin the worm. For this paper, we chose to vary the thinning algorithms in order to determine which is the most accurate to use in a *C. elegans* imaging pipeline. This decision allows us to use our newly developed metrics to track a very small difference in a pipeline, demonstrating how this validation method can be used to incrementally choose and validate individual elements in a pipeline in order to create the most robust and accurate pipeline possible. Traditional means of choosing a thinning algorithm might include thinning an image or video of a worm and then judging by hand how accurate it seems to be. There are no real metrics to guarantee that this is actually computing relevant values, so the choice of which thinning algorithm to use would be made almost arbitrarily.

To demonstrate the power of this metric, we chose three published thinning algorithms: a single pass thinning algorithm (SPTA) [12], a triple pass thinning algorithm (TPTA) [11], and a one-pass parallel asymmetric thinning algorithm (OPATA-8) [3]. SPTA is a single pass sequential thinning algorithm that uses both flag map and bitmap simultaneously to decide if a boundary pixel can be deleted. TPTA is an older and simpler thinning algorithm that tends to take longer than SPTA and in theory yield less desirable results. OPATA-8 is a more complicated thinning algorithm based on pattern matching each 3x3 set of pixels in the image to one of 18 pre-established patterns. This is an expensive operation when it must be done for every pixel in every frame of a video, but theoretically it produces very good results.

This shape produced by these thinning algorithms is often not a single line, but has multiple endpoints. We reduce it to a single line by selecting the endpoints that are furthest from one another and removing all others.

The output node produces a set of text files, which we refer to as body files. Each body file contains one line per frame of video with each line giving a timestamp and the pixel locations of the body of the worm. The number of pixels (or length of the body) is dependent on the size of the worm and also exhibits some variability due to the binarization of the image and subsequent thinning.

Because we wanted to gather statistics based on a non-uniform segmentation of the worm's body, it was necessary to identify the location of the worm's head. To do this a simple heuristic was applied to capturing video. We made sure that the first frame of every video captured featured the head of the worm to the right of the tail of the worm. This was possible since rotating and placing the camera in the simulator is quick and easy. The first frame of each file was then marked with a head tag on the end that who had the largest x coordinate. With the head tag in place, subsequent frames of the video were properly rearranged such that the end point closest to the last head location was identified as the head. This turns out to be a very robust and reliable mechanism if the video being processed was taken at high enough frame rates.

These head-tagged body files are then post-processed in batches. The WormAnalyzer software takes a directory of head-tagged body files and produces skeleton files that provide a description of the location and position of each segment of the worm. Like the simulator, the size of these segments are not uniform, but are based on the muscle placement as reported in Varshney et al [10].

For convenience, the simulator also creates skeleton files in the same format as the WormAnalyzer. This allows us to directly compare the output from both processes using the same metrics calculated in exactly the same way. The skeleton files are then processed to extract features of the worm's movement using a technique similar to the one reported in Cronin et al [2]. Currently, this software extracts 49 features from the worm's motion including the velocity of the centroid of the worm, the amplitude of the worm's body, the average angle at each joint location, and the angular velocity of each joint (see Figure 1). The software outputs data files that give these features on a frame by frame basis and a set of summary statistics that can be further analyzed using statistical packages.

Table III
AMPLITUDE AND VELOCITY AVERAGES

	Default		Large Angle		Small Angle	
	Amp	Vel	Amp	Vel	Amp	Vel
Sim	168.37	204.08	180.10	40.79	62.64	374.70
OPATA-8	186.39	259.30	176.11	113.02	59.98	391.56
SPTA	182.34	304.71	170.10	131.56	59.03	404.95
TPTA	177.73	409.15	165.13	201.29	58.21	506.10

B. Testing

To test the pipelines, first video was captured for each of the worm variants. To do this, the simulation was run numerous times for each variant. The zoom level was held constant, and the ”camera” remained in one place while the worm performed forward locomotion. Screen capture software was used to record video of the worm moving. No video was taken while the camera was repositioning. The simulator produces a file which details the underlying values of the worm body in the simulation. Each set of values was timestamped, and the timestamp was also captured in the video. The data file from the simulator was then split into smaller files corresponding with the video clips such that each video file would have a corresponding data file that contained a record of the underlying values during that clip. Over 20,000 frames of worm motion were collected over the three variants. Three distinct sets of video was produced: default, large angle, and small angle movement.

Each set of video was batch processed through the CVT three times. Each time a different thinning algorithm was used. The pipeline was constructed such that data files were outputted that contained the thinned skeleton and a timestamp for each frame of the video. These files were then fed into the WormAnalyzer software for statistical analysis. All pipelines used the same segmentation code to produce the 24 anatomically correct skeleton points and analysis code to produce the statistics. Statistics were recorded for amplitude, velocity, segment angles, and segment angle velocities.

Then, the values produced from the simulator were fed into the WormAnalyzer software to produce the same statistics as those produced from the video analysis. These values produced from the simulator files were used as the ”accepted value” since the underlying value was known. The results from the other pipelines are used as the ”experimental value.” Using these values the root mean square (RMS) error for each pipeline/statistic combination was computed. This value represents how accurate the pipeline was in creating accurate skeletons and therefore accurate statistics. The testing was focused on isolating only the thinning algorithm every other operation in the pipeline is exactly identical, including the statistical analysis. This allowed for the differences in accuracy to be attributed to the accuracy of the thinning algorithm. This demonstrates how a broad measure of accuracy of a pipeline can be focused to help validate each video effect in a pipeline.

V. RESULTS

Approximately 20,000 frames of video were processed to produce their relevant statistics. Each statistic was calculated for each combination of worm variant and thinning algorithm. These statistics were then treated as an experimental value, and the statistics produced for the corresponding videos from the simulator were treated as the accepted value. From this a simple root mean square error term was created measuring the average error over each segment of the worm. This error was calculated for all the worm variant/thinning algorithm pairs. In addition, the maximum error for a segment was recorded. This metric will be referred to here as ”pipeline error value” or PEV. The PEV for a given pipeline is a measurement of its accuracy. A perfect pipeline would produce a PEV of 0 (since it perfectly matched the underlying data values). Table I and II shows the PEV values for each pipeline on the various worms used in the simulation. Figure 2 shows how the pipelines performed over the three different worm variants.

In addition to this, the amplitude and velocity for the worm was measured. Each pipeline produced a single amplitude and velocity value for a given worm variant. This is compared against the given value in Table III.

By all metrics, OPATA-8 is the most accurate thinning algorithm we tested when used on *C. elegans*. All of our metrics point towards its accuracy in analyzing various the movements of all the variants of the worm. Thus, this indicates that any future pipelines that require a thinning algorithm should chose OPATA-8 if their main goal is accuracy of statistics.

In measuring the velocities of the worm as a whole, the two older thinning algorithms (SPTA and TPTA) performed very poorly. After examining the data, we found that these algorithms sometimes greatly alter the length of the worm by making poor choices in the thinning algorithm. The newer and more advanced OPATA-8 goes much further in preserving the length of the worm. Preserving as much of a thinned object as possible was actually a goal in development of OPATA-8 [3]. Velocity of the worm is calculated by first obtaining its centroid, and tracking changes to the centroid over time. When the length of the worm changes wildly from frame to frame, the centroid also changes position. This inaccurate change of position is expressed in the poor performance of these thinning algorithms when calculating velocity of the worm.

It should be noted that our metrics do not take into account the speed of processing of these algorithms. If a pipeline is developed whose main goal is speed and not accuracy, one may chose to go with a different algorithm. This metric still has a place in that it can allow researchers to make an informed decision about the trade off in accuracy and speed, as now both are quantifiable results that are easily obtained.

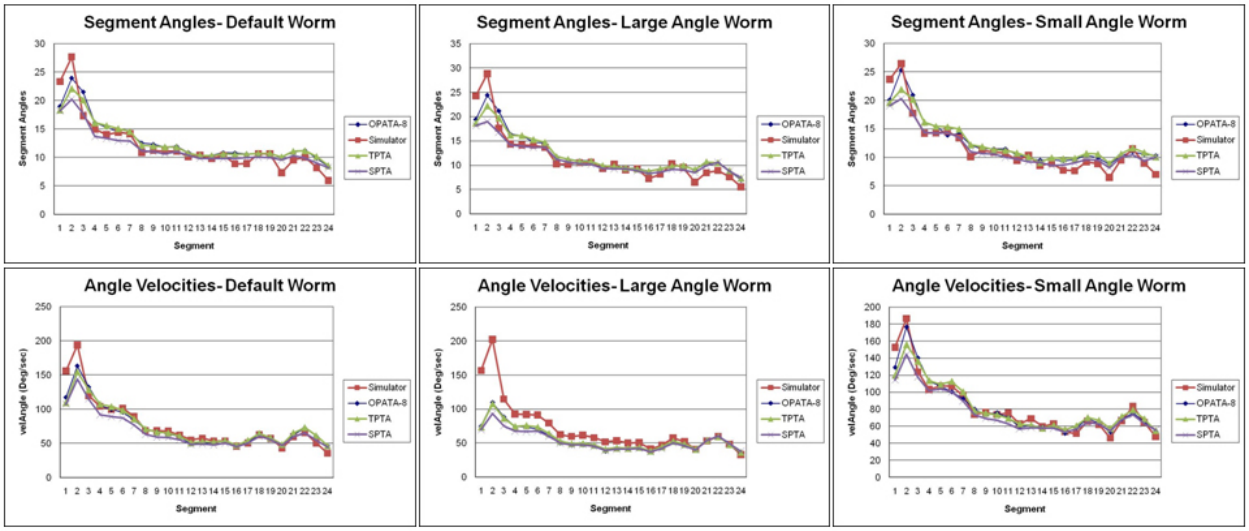

Figure 2. Results of comparison of segment angles and velocities for three locomotion types.

VI. Conclusions

We have demonstrated a new technique for analyzing and quantifying the accuracy of a pipeline, and have shown how this tool can be used to make a simple decision such as deciding which thinning algorithm to choose in the implementation of a pipeline. This is a powerful metric that has not existed before in the *C. elegans* community, and should be used as a tool to measure and compare future pipelines for accuracy. This can also be used in the tuning of pipelines for more reliable results in the future. If research is published with this metric, it would provide further confidence in the results obtained with a pipeline, and allow for easier comparison of data across different institutions by having an independent measurement of the accuracy and confidence in the data obtained on other locations.

References

[1] J. Byden and N. Cohen, "Neural control of caenorhabditis elegans forward locomotion: the role of sensory feedback," *Biological Cybernetics*, vol. 98, pp. 339–351, 2008.

[2] C. J. Cronin, J. E. Mendel, S. Mukhtar, Y.-M. Kim, R. C. Stirbl, J. Bruck, and P. W. Sternberg, "An automated system for measuring parameters of nematode sinusoidal movement," *BMC Genetics*, vol. 6, no. 5, February 2005.

[3] W. Deng, S. S. Iyengar, and N. E. Brener, "A fast parallel thinning algorithm for the binary image skeletonization," *International Journal of High Performance Computing Applications*, vol. 14, no. 1, pp. 65–81, 200.

[4] W. Geng, P. Cosman, C. C. Berry, Z. Feng, and W. R. Schafer, "Automatic tracking, feature extraction and classification of c. elegans phenotypes," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 10, pp. 1811–1820, October 2004.

[5] J. Karbowski, G. Schindelman, C. Cronin, A. Seah, and P. Sternberg, "Systems level circuit model of c. elegans undulatory locomotion: mathematical modeling and molecular genetics," *Journal of Computational Neuroscience*, vol. 24, pp. 253–276, 2008.

[6] R. Mailler, J. Avery, J. Graves, and N. Willy, "A biologically accurate 3d model of the locomotion of caenorhabditis elegans," in *Proceedings of The First International Conference on Computational and Systems Biology and Microbiology (BIOSYSCOM 2010)*, March 2010.

[7] E. Neibur and P. Erdös, "Theory of the locomotion of nematodes: Dynamics of undulatory progression on a surface," *Biophysics Journal*, vol. 60, pp. 1132–1146, November 1991.

[8] ——, "Theory of locomotion of nematodes: Control of the somatic motor neurons by interneurons," *Mathematical Biosciences*, vol. 118, no. 1, pp. 51–82, 1993.

[9] G. D. Tsibidis and N. Tavernarakis, "Nemo: a computational tool for analyzing nematode locomotion," *BMC Neuroscience*, vol. 8, no. 86, 2007.

[10] L. R. Varshney, B. L. Chen, E. Paniagua, D. H. Hall, and D. B. Chklovskii, "Structural properties of the caenorhabditis elegans neuronal network," 2009. [Online]. Available: http://www.citebase.org/abstract?id=oai:arXiv.org:0907.2373

[11] T. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Communications of the ACM*, vol. 27, no. 3, pp. 236–239, March 1984.

[12] R. Zhou, C. Quek, and G. Ng, "A novel single-pass thinning algorithm and an effective set of performance criteria," *Pattern Recognition Letters*, vol. 16, pp. 1267–1275, 1995.