UNIVERSITY OF CALIFORNIA, SAN DIEGO

Tracking and Analysis of *C. elegans* Behavior Using Machine Vision

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Electrical and Computer Engineering (Communication Theory and Systems)

by

Kuang Man Huang

Committee in charge:

Professor Pamela Cosman, Chair
Professor Serge Belongie
Professor Truong Nguyen
Professor William Schafer
Professor Nuno Vasconcelos

The Dissertation of Kuang Man Haung is approved and it is acceptable in quality and form for publication on microfilm:

Chair

University of California, San Diego

2008

DEDICATION

To my parents and my family: I will be eternally thankful for your support and encouragement. Without your help, this work would not have been possible.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Chapter 2 & 3, in part, is a reprint of the material as it appears in Kuang Man Huang, Pamela Cosman, and William Schafer, "Machine Vision Based Detection of Omega Bends and Reversals in *C. elegans,*" Journal of Neuroscience Methods, Vol. 158, Issue 2, pp. 323-336, December 2006. I was the primary researcher and the co-authors Dr. Pamela Cosman and Dr. William Schafer directed and supervised the research which forms the basis for this chapter. This work was supported by a research grant from the National Institutes of Health (R01 DA018341).

Chapter 4, in full, is a reprint of the material as it appears in Kuang Man Huang, Pamela Cosman, and William Schafer, "Automated Detection and Analysis of Foraging Behavior in *C. elegans*", Journal of euroscience Methods, accepted January 31st 2008. I was the primary researcher and the co-authors Dr. Pamela Cosman and Dr. William Schafer directed and supervised the research which forms the basis for this chapter. This work was supported by a research grants from NIDA (DA18341 and DA12891).

Chapter 5, in part, is a reprint of the material as it appears in Kuang Man Huang, Pamela Cosman, and William Schafer, "Automated Tracking of Touching C. elegans with Articulated Models", submitted to Special Issue on Biomedical Imaging - Journal of Signal Processing Systems for Signal, Image, and Video Technology. I was the primary researcher and the co-authors Dr. Pamela Cosman and Dr. William Schafer directed and supervised the research which forms the basis for this chapter.

VITA

1999 Bachelor of Electro-physics National Chiao Tung University, Taiwan

2003 Master of Science, University of California San Diego

2003-2008 Research assistant, University of California San Diego

2008 Doctor of Philosophy, University of California San Diego

PUBLICATIONS

K. Huang, P. Cosman and W. R. Schafer, "Machine Vision Based Movement Analysis of *C. elegans* Coiler Mutants", *Journal of euroscience Methods*, vol. 158, pp. 323-336, 2006.

K. Huang, P. Cosman and W. R. Schafer, "Automated tracking of multiple *C. elegans* with articulated models", *IEEE International Symposium on Biomedical Imaging*, pp. 1240-1243, 2007.

K. Huang, P. Cosman and W. R. Schafer, "Automated Detection and Analysis of Foraging Behavior in *C. elegans*", *IEEE Southwest Symposium on Image Analysis and Interpretation Preliminary Technical Program*, pp. 1240-1243, 2007. 2008.

K. Huang, P. Cosman and W. R. Schafer, "Automated Detection and Analysis of Foraging Behavior in *C. elegans*", *Journal of euroscience Methods*, accepted January 31st 2008.

K. Huang, P. Cosman and W. Schafer, "Automated Tracking of Touching *C. elegans* with Articulated Models", submitted to *Special Issue on Biomedical Imaging - Journal of Signal Processing Systems for Signal, Image, and Video Technology.*

FIELDS OF STUDY

Major Field: Electrical and Computer Engineering (Communication Theory and Systems)

Studies in Image Processing
Professor Pamela Cosman

Studies in Neurobiology
Professor William Schafer

ABSTRACT OF THE DISSERTATION

Tracking and Analysis of *C. elegans*
Behavior Using Machine Vision

by

Kuang Man Huang

Doctor of Philosophy in Electrical and Computer Engineering Department
(Communication Theory and Systems)

University of California, San Diego, 2008

Professor Pamela Cosman, Chair

The behavior of the nematode *C. elegans* has proven increasingly useful for the genetic dissection of neurobiological signaling pathways and for investigating the neural and molecular basis of nervous system function. Locomotion is among the most complex aspects of *C. elegans* behavior, and involves a number of discrete motor activities such as omega bends (deep bends typically on the ventral side of the body which reorient the direction of forward locomotion), reversals (changes in the direction of the locomotion wave that cause a switch from forward to backward crawling), and foraging (a rapid, side-to-side movement of the nose). Here we use

automated methods to automatically detect these activities, which rely in part on a new method for obtaining a morphological skeleton describing the body posture of coiled worms. These new methods have made it possible to reliably detect events that are time-consuming and laborious to detect by real-time observation or human video analysis. We also present an algorithm for tracking and distinguishing multiple *C. elegans* in a video sequence, including when they are in physical contact with one another. Our method makes it possible to identify two worms correctly before and after they touch each other, and to find the body poses for further feature extraction. The algorithm has many applications in the study of physical interactions between *C. elegans*.

Chapter 1

Introduction

1.1 *C. elegans* and automated analysis system

The nematode *Caenorhabditis elegans* is widely used for studies of nervous system function and development. *C. elegans* is approximately 1 μm in length and feeds on bacteria (Figure 1.1). It has a simple nervous system which is well characterized at the anatomical level: an adult hermaphrodite contains only 302 neurons, each with a precisely determined position, cell lineage and synaptic connectivity. Despite its anatomical simplicity, the *C. elegans* nervous system mediates diverse and intricate patterns of behavior. The sense organs of *C. elegans* are capable of perceiving and responding to a wide range of environmental conditions, including heavy and light touch, temperature, volatile odorants, food and other nematodes. Because a particular neuron can be positively identified based on its position, it is possible to eliminate the function of an individual neuron or a group of neurons by using laser ablation. Moreover, because of their short generation time and completely sequenced genome, *C. elegans* is well suited to analysis of the molecular and cellular basis of nervous system development and function. However, many genes with critical roles in the nervous system have effects on behavior that are difficult to describe precisely, or occur over time scales too long to be compatible with real-time

scoring by a human observer. Analyzing these genes relies on the rigorous description of animal behaviors. Standard methods for classifying the behavioral patterns of mutant *Caenorhabditis elegans* rely on human observation and are therefore subjective. Behavioral assays in this organism, particularly in more complex behaviors such as locomotion, are often highly imprecise. For example, over 100 genes have been described which when mutated lead to abnormal or uncoordinated movement [3]. These uncoordinated ('Unc') mutants are usually classified into a number of descriptive categories, including 'kinker', 'coiler', 'slow', 'sluggish' and 'loopy' animals [1, 3]. Table 1.1 contains descriptions of behaviors of these mutants. Since these categories are somewhat vague, and are always scored subjectively by a human observer, it is not uncommon for the same Unc mutant to be described differently by different researchers, or for two mutants with clearly distinguishable mutant phenotypes to be assigned the same classification.

Table 1.1: Description of uncoordinated mutants [1, 2].

unc-1	Strong coilers.
unc-3	Weak coiler tends to coil tail active.
unc-10	Weak coiler tends to back loopy movement in reverse; fairly active slightly small and thin.
unc-17	Severe coiler at all stages rather small and thin.
unc-26	Severe kinker small scrawny flaccid little movement.
unc-32	Severe coiler little movement in adult; moves well in L1 but coils in response to touch in L2 and later stages; rather small and thin.
unc-37	Weak coiler fairly active.
unc-75	Weak coiler especially in reverse; moves forward well; sluggish; short.
unc-77	Irregular loopy movement both forward and reverse; active; thin.

Automated systems [4-7] consisting of a tracking microscope and image processing software have been developed and used to analyze the movements of *C. elegans* at high magnification. The tracking microscope is capable of following an individual animal for long time periods and saving a time-coded series of digital images representing its motion and body posture over the course of the recording. In [8], quantitative morphological and locomotion features were measured from the acquired image data with this system and used by the

classification and regression tree algorithm (CART) to quantify the locomotion patterns and classify the behavioral phenotypes of *C. elegans* mutants. The system was used to investigate the similarities between different behavioral patterns based on their clustering in multidimensional feature space. From a complex data set consisting of 253 features measured from recordings of 797 individuals representing 8 distinct genotypes, principal component analysis was used to represent each mutant type as a cloud of data points in low-dimensional feature space. The k-means algorithm and Euclidean distance measurements were also used to explore the natural structure of the behavior data and to compare the similarities of mutant phenotypic patterns. The results provided a precise and comprehensive definition of several important *C. elegans* phenotypes. Studies previous to [8] focused primarily on simpler features such as body length, brightness, and speed. In this dissertation, we focus on automating the analysis of more subtle or challenging behaviors: coiling, reversals, omega bends, foraging and multi-worm interactions.

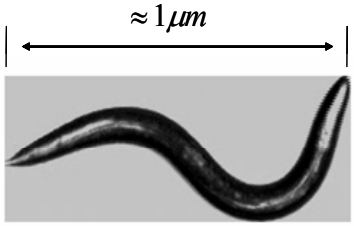


Figure 1.1: A typical image of *C. elegans*.

1.2 Coiling skeletonizing and behavior studies

For measuring some features such as body length, body width, velocity and angle changing rate, a morphological skeleton needs to be obtained by applying a skeletonizing algorithm on a binary worm body image. Skeletonizing is a process for reducing foreground regions in a binary image to a skeletal remnant that largely preserves the spatial extent and connectivity of the original region while throwing away most of the original foreground pixels. We use morphological thinning that successively erodes away pixels from the boundary (while

preserving the end points of line segments) until no more thinning is possible, at which point what is left approximates the skeleton [9]. The skeleton is useful because it provides a simple and compact representation of a shape that preserves many of the topological and size characteristics of the original shape (Figure 1.2). Thus, for instance, we can get a rough idea of the length of a shape by computing the number of pixels on the skeleton.

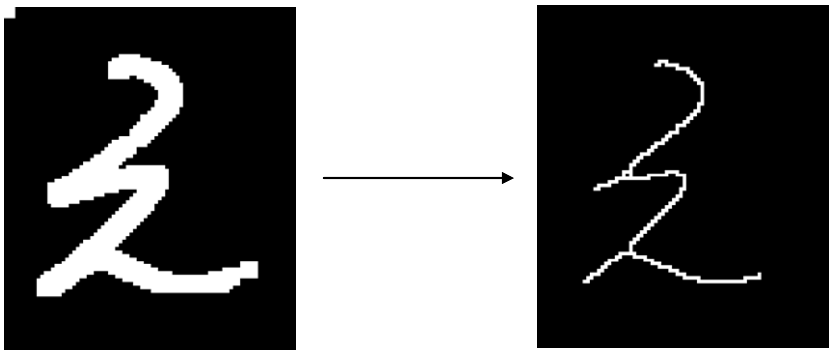


Figure 1.2: An example of the skeleton from a simple shape.

Determining the correct morphological skeleton using a standard algorithm is challenging when the binary worm body shape has an internal hole, which can be caused when the worm bends its body and touches itself. In these cases, the normal skeletonizing algorithm does not correctly identify the true skeleton nor the correct ends of the worm (i.e. head and tail). In this dissertation we describe an algorithm which generates skeletons for these body positions using a parameterized body model and locates the division line between overlapping portions of the worm body. Several specific features which can be measured from these coiled skeletons are particularly useful in the automated classification of *C. elegans* mutant types. Applications of this method for the analysis of specific body postures and behavioral events will be discussed.

In behavioral studies, it is often critical to parameterize a complex behavior by identifying the simpler behavioral events that underlie it. For example, among the most important

behaviors to a nematode are those involving navigation of a sensory gradient to move toward an optimal condition. Nematodes accomplish this navigation primarily using a movement called an omega bend, in which the animal makes a single deep body bend in a shape of the capital Greek letter omega, usually on the ventral side of the body [10-12]. An omega bend serves to reorient the animal's head and allows it to continue to crawl forward, but in a different direction. Studies of omega bends have always relied exclusively on the time-consuming analysis of video recordings by human observers.

When avoiding noxious compounds, nematodes often exhibit a second form of behavior, known as an escape reflex. When a worm is touched or presented with a toxic chemical stimulus, it will switch the direction of the locomotion wave, causing the animal to instantaneously crawl backward instead of forward. After a short period of backward crawling, the animal will execute an omega bend and crawl forward in a different direction, away from the noxious stimulus. A variety of genes affect the frequency of these reversals in direction. Moreover, the switch between forward runs and bouts of reversals has been shown to play an important role in worm touch avoidance behavior [13]. However, abnormalities in reversal frequency, and particularly in reversal distance, are very difficult to detect by manual observation, and have only been verified by careful assays of individual animals [14]. In previous studies, reversals have been detected automatically by following the path of the animal's centroid and identifying a large change in the direction of bearing of the centroid trace [15]. A more recent paper [6] described an algorithm that used skeleton points and the positions of the head and tail relative to the worm body to detect reversals. In this dissertation, we have developed algorithms based on skeleton analysis to detect omega bends and reversals, and characterize parameters relevant to these behaviors. In particular, the spatial polarity and temporal correlation of omega bends following reversals is investigated for wild-type and several of the more active coiler mutants.

Another *C. elegans* behavior that has received comparatively little attention is foraging. Foraging is a term used to describe rapid, side-to-side movements of the nose generated by the worm as it explores its environment. Several neurons, including the OLQ and IL1 sensory neurons and the RMG motorneurons, have been shown to be required for this behavior [16, 17]. Various genes conferring a foraging abnormal ("Fab") phenotype have also been identified; for example, the AMPA-type glutamate receptor gene *glr-1* is required for foraging [18], and the G-protein alpha-subunit gene *goa-1* as well as other genes in the Go/Gq signaling pathway affect the rate of foraging [19, 20]. However, the precise nature of the foraging movements in wild-type and mutant strains has not been characterized. In this dissertation, we present an automated method to detect and analyze foraging behavior of *C. elegans* in a video sequence. We also measure foraging-related parameters which have not previously been studied. The algorithm has applications in classifying and characterizing genetic mutations associated with this behavior.

1.3 Multi-worm tracking algorithm

Another main problem we consider is tracking and distinguishing multiple *C. elegans* in a video sequence, including when they are in physical contact with one another. Single-worm systems can provide a considerable amount of information about each animal that is recorded, but since statistically-significant characterization of any worm type requires the analysis of multiple animals, collecting data one animal at a time is often frustratingly slow. On the other hand, multiple-worm recordings do not typically provide as much information as single worm recordings due to their lower magnification. In addition, in existing multi-worm systems, any time two individuals touch, segmentation of separate animals is difficult and so their individual identities are lost by the system. When the animals separate, the system is unable to determine the correspondence between individuals before and after touching. The inability to separately

segment and track individual animals when they touch seriously limits the ability of a multi-worm system to characterize the behavior of an individual in a population over time.

In this dissertation, we address the problem of combining a part-based articulated model [21, 22] with a dynamic programming algorithm to determine the correct location of the individual worm bodies. Our work accurately resolves the individual body postures of two worms in physical contact with one another and identifies them correctly before and after they touch each other, and can still maintain track of the worms their bodies have non-crawling displacement. Furthermore, we solve the problem that the skeleton-based reversal detection algorithm in [23] fails when two worms touch each other because of the difficulty of obtaining morphological skeletons.

1.4 Strains and culture methods

In all the experiments described in this dissertation, routine culturing of *C. elegans* was performed as described in [3]. All worms analyzed in these experiments were young adults; fourth-stage larvae were picked the evening before the experiment and tracked the following morning. Experimental animals were allowed to acclimate for 5 minutes before their behavior was analyzed. Plates for tracking experiments were prepared fresh the day of the experiment; a single drop of a saturated LB (Luria broth) culture of *E. coli* strain OP50 was spotted onto a fresh NGM (nematode growth medium) agar plate and allowed to dry for 30 minutes before use. The alleles and predicted products of the genes used were as follows: *syd-1(ju82); unc-1(e1598); unc-3(e151); unc-10(e102); unc-17(e245); unc-26(m2); unc-32(e189); unc-37(e262); unc-75(e950); unc-77(e625)* for behavioral studies and *npr-1(ky13)* for multiple-worm tracking experiments. Unlike the laboratory standard N2 strain which is a solitary feeder, tending to disperse on encountering bacterial food, *npr-1(ky13)* mutants are social feeders, strongly aggregating together, thus providing an opportunity to study touching behavior.

1.5 Acquisition of image data

C. elegans locomotion was tracked with a Zeiss Stemi 2000-C microscope mounted with a Cohu High Performance CCD video camera essentially as described in [7]. The microscope was outfitted for brightfield illumination from a 12V 20W halogen bulb reflected from a flat mirror positioned at an angle of approximately 45 degrees. A computer-controlled tracker was used to maintain the worms in the optical field of the microscope during observation (Figure 1.3). The experiments conducted in this dissertation used videos made with the following methods:

1. For the experiments in Chapters 2 and 3, an image frame of the animal was captured every 0.125 second (8Hz) for at least five minutes (8×60×5 = 2400 images per video). Next, we binarized the image using an adaptive threshold and found the connected component with the largest area. The original image was then trimmed to the smallest axis-aligned rectangle that contained this component, and saved as eight-bit grayscale data. The dimensions of each image, and the coordinates of the upper left corner of the rectangle box containing the worm body in the tracker field were also recorded simultaneously. The microscope was fixed to its largest magnification (50 X) during observation. The number of pixels per millimeter was fixed at 312.5 pixel/mm for all worms.

2. For the experiments in Chapter 4, in order to detect foraging events which happen within very short time periods, image frames of the animal were captured at a higher frequency of 30Hz for at least one minute (30×60 = 1800 images per video). The rest of the image acquisition method is the same as it is for the experiments in Chapters 2 and 3.

3. For the experiments on two worms touching in Chapter 5, an image frame was captured every 0.125 second (8Hz) and then saved as AVI video files. We used the smaller magnification of 2.5X to ensure that both of the worms are in the optical field of the microscope

during observation. For each video, the recording is initiated when the two worms are separated. The recording continues until they touch and then move apart. The length of every video is different, ranging from 30 sec to 178 sec, because the time length for each pair of worms to aggregate and separate is different.

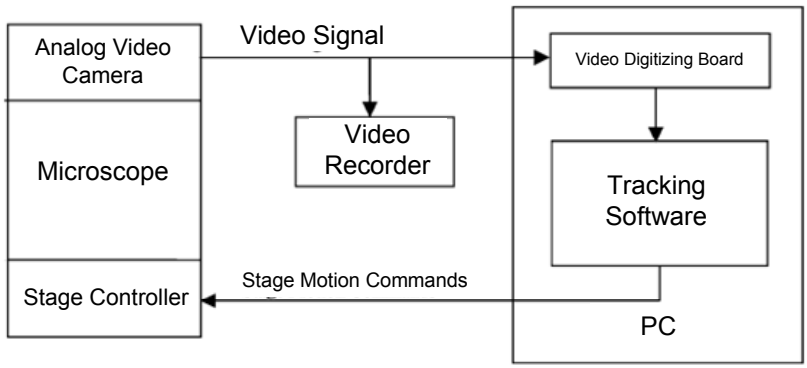


Figure 1.3: Tracking and imaging system. The CCD video camera is fitted to the microscope and outputs analog video images to the digitizing board on the PC. The tracking software computes the centroid of the worm and sends commands to the stage controller to re-center the field of view on the worm.

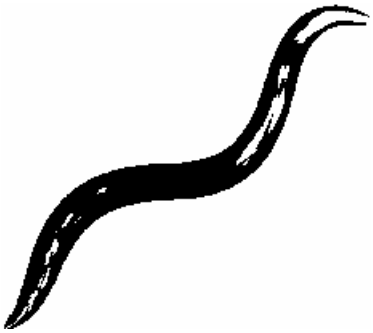
1.6 Image pre-processing

To facilitate analysis, the grayscale images were subjected to preliminary image processing to generate a simplified representation of the body [7]. First a local thresholding was applied on the grayscale images by using a 5 by 5 moving window (Figure 1.4a). The center pixel inside the moving window was assigned to 1 if the mean value of the window was less than 70% of the background pixel value or the standard deviation was larger than 30% of the mean value. Otherwise, the center pixel was assigned to 0 as background (Figure 1.4b). Next, a morphological closing operator (binary dilation followed by erosion) was used (Figure 1.4c). A corresponding reference binary image was also generated by filling holes inside a worm body based on image content information. The difference between these two binary images provided a good indication of which image areas are worm body and which are background. In order to remove unwanted

isolated objects, the connected components were labeled by scanning the image in x and y directions sequentially, and the largest component was selected to be the worm body in the image.

Following binarization, a morphological skeleton is obtained by applying a skeletonizing algorithm [24]. Redundant pixels on the skeleton are eliminated by thinning. To avoid branches on the ends of skeletons, the skeleton is first shrunk from all its end points simultaneously until only two end points are left. These two end points represent the longest end-to-end path on the skeleton. A clean skeleton can then be obtained by growing out these two remaining end points along the unpruned skeleton by repeating a dilation operation (Figure 1.4d).

(a)



(b)

(c) (d)

Figure 1.4: (a) Gray level image acquired from a video sequence. (b) Corresponding binary image after thresholding. (c) Binary image after hole filling and closing operator. (d) Skeleton after skeletonizing and pruning algorithm.

Chapter 2

Skeletonization and classification of coiler mutants

A major motivation for establishing *C. elegans* as an experimental molecular genetic system was to understand how genes control behavior and locomotion. A thorough understanding of the ways in which genes control these aspects of biology relies on the accuracy of phenotypic analysis. Therefore, an enhanced capability to analyze all the complexities of nematode movement will help our understanding of how genes control behavior. In this chapter, we use image feature extraction techniques to quantitatively define and classify the behavioral patterns of *C. elegans* nervous system mutants. An automatic tracking and image processing system is used to measure morphological and behavioral features from videos of *C. elegans.* Measurement of some features such as body length and angle change rate, whether by a human observer or a computer, requires a precise analysis of the animal's body posture.

In most cases, we can determine the correct posture using a standard morphological skeleton algorithm [7, 24]. However, sometimes the binary worm body shape has an internal hole, which can be caused when the body bends into an omega shape (Figure 2.1a) or a spiral (Figure

2.1b). In such cases, the morphological skeleton which is correct from an image processing point of view is not necessarily a useful summary of the worm body shape from a biological point of view. For example, Figure 2.1d shows the binarization of the worm body from Figure 2.1c, and Figure 2.1e shows the morphological skeleton, correct from an image processing point of view, generated from the binary shape of Figure 2.1d. Figure 2.1f on the other hand, shows the desired skeleton from a biological perspective. In previous studies, frames such as Figure 2.1c could be recognized as failing the skeletonization (for example, because the Figure 2.1e skeleton is too short) and so the frame was discarded [4, 8].

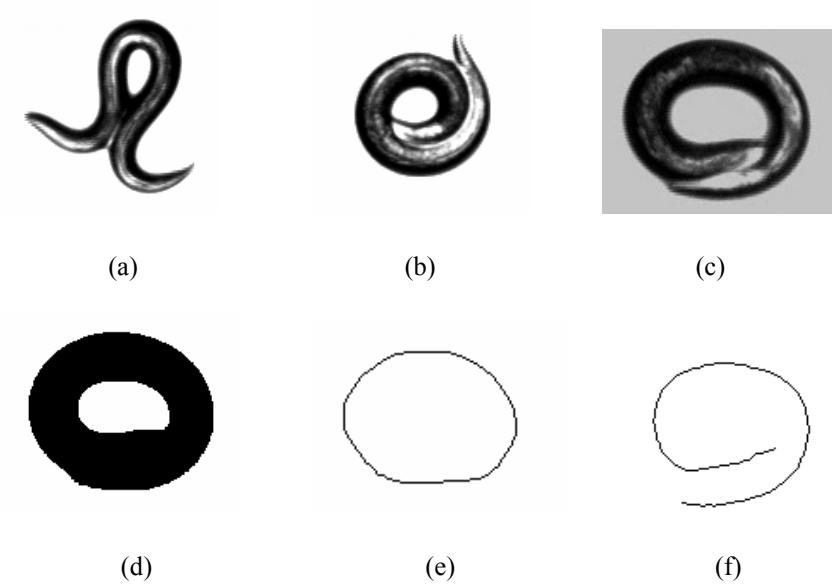


Figure 2.1: (a) Typical omega bend. (b) Coil or spiral. (c) Grayscale image. (d) Binary image. (e) Skeleton from an image processing point of view. (f) Desired skeleton from a biological point of view.

In this project, we develop an algorithm to solve this problem in order to study coiler mutants, which take on these body postures frequently. In particular, we use a parameterized body model (Figure 2.2a) and locate the division line between overlapping portions of the worm body. The length L and width W are the average length and width of the worm body obtained from images without internal holes. Briefly, we use W in the model to find the touching parts of the worm body, because touching parts will have width greater than W (Figure 2.2b). Then we

cut the touching part from its exterior boundary to interior boundary to recover the original posture. The skeletonizing algorithm is then applied to obtain the morphological skeleton. Using this method, we can extract features from the obtained skeletons and robustly define the body postures of wild-type animals as well as of mutants that coil frequently. Furthermore, we can use these features to demonstrate the successful classification and compare the similarities of a data set consisting of wild type and coiler mutants.

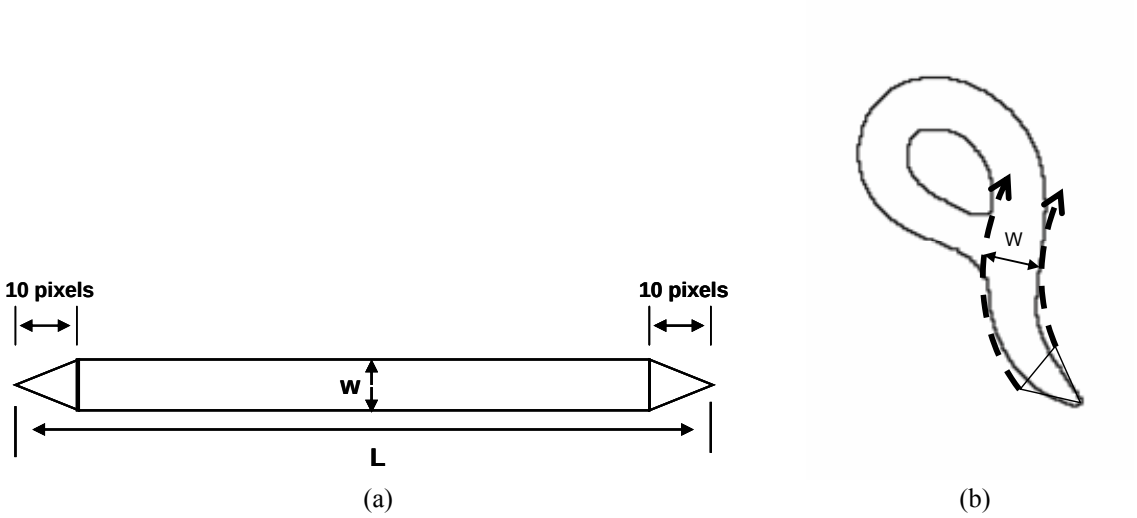


Figure 2.2: (a) The worm body matching model. (b) We find the overlapping part of the worm body using the model, by starting from a protruding end and then moving along the body length, looking for a place where the blob width exceeds W.

In this chapter, we first explain how the morphological and locomotion features are extracted. We then give a detailed description of our coiling skeletonizing algorithm. Finally we present results assessing the robustness of our skeletonizing algorithms by classification and characterization of coiler mutants.

2.1 Image feature extraction

2.1.1 Head and tail recognition

After a clean skeleton is obtained in each frame without coiling as described in section 1.6, we use the approach in [7] to extract the head and tail information of entire video sequences. Basically, we use three clues to address this problem:

1) Even though the distance that the worm body travels between two adjacent image frames could be large, the head and tail positions relative to the body centroid tend to change little.
2) Because of the distribution of fat deposits, the worm's head area is usually brighter than the tail area.
3) The worm's head usually moves more frequently than the tail, which is related to foraging behavior.

First we divide the end points of all uninterrupted video segments into two groups according to the following criterion: Let $pt_1(t)$ and $pt_2(t)$ denote the end points in frame *t* that were assigned to groups 1 and 2 respectively; $pt_A(t+1)$ and $pt_B(t+1)$ denote the two end points in frame *t*+1 that have not yet been assigned to either group. We compute the distances $dist(pt_m(t+1), pt_n(t))$ where $m \subset \{A, B\}$ and $n \subset \{1,2\}$. The end point $pt_M(t+1)$ will be assigned to group *N* if $(M, N) = \arg\min_{(m,n)} dist(pt_m(t+1), pt_n(t))$ and $(\overline{M}, \overline{N}) \neq \arg\max_{(m,n)} dist(pt_m(t+1), pt_n(t))$. If the rule is not satisfied, then the current frame is considered to be "undecided" and the grouping process needs to restart from the next frame to avoid errors propagating. Next for each frame, we use the two end points as references to generate a cutoff line to isolate the two end sections from the rest of the body. The median brightness of the two end sections for each frame and the mean values (over time) of these median values of group 1 and 2 are computed. If the difference between mean values of groups 1 and 2 is greater than 20% of the larger mean value, the group

with the higher brightness is considered to be the head. For some mutant types, the brightness differences between head and tail are smaller because of digestive abnormalities. In this case, we compare the local movement distance for the two end points and the group with larger total movement distance is labeled as the head.

2.1.1 Feature extraction

Feature extraction is applied to obtain 64 basic features. These include body length, width, fatness, brightness, and angle change rate. These 64 features (listed in Table 2.1) are calculated for each of the 2400 frames in each video by using software coded in C or MATLAB. The maximum, minimum, and mean values over time for most of these features were then computed to form 188 features in total for each video. Some features (e.g., the average brightness over the worm body) can be computed in every single frame. Certain features such as movement distance and speed could not be obtained from one single frame. In this case, we took 4, 8 and 40 frames (0.5, 1 and 5 seconds) in a sliding window and computed features within windows. The maximum, minimum and mean values were also calculated from these sets of numbers.

Table 2.1: All feature variables used in CART analysis

CART variable name	Description
AREAMIN, AREAMAX, AREAAVG	Min, max, average worm body area
HGHTMIN, HGHTMAX, HGHTAVG	Min, max, average height of MER (minimum enclosing rectangle) area
WDTHMIN, WDTHMAX, WDTHAVG	Min, max, average width of MER area
LNGTHMIN, LNGTHMAX, LNGTHAVG	Min, max, average worm body length
WHRATMIN, WHRATMAX, WHRATAVG	Min, max, average width/height ratio
MERFLMIN, MERFLMAX, MERFLAVG	Min, max, average ratio of worm area to MER area
MAJORMIN, MAJORMAX, MAJORAVG	Min, max, average length of best-fit ellipse's major axis
MINORMIN, MINORMAX, MINORAVG	Min, max, average length of best-fit ellipse's minor axis
ECCTYMIN, ECCTYMAX, ECCTYAVG	Min, max, average eccentricity of best-fit ellipse
MVHLFMIN, MVHLFMAX, MVHLFAVG	Min, max, average distance moved in 0.5 second
MV1MIN, MV1MAX, MV1AVG	Min, max, average distance moved in 1 second
MV5MIN, MV5MAX, MV5AVG	Min, max, average distance moved in 5 second
HDTHKMIN, HDTHKMAX, HDTHKAVG	Min, max, average head thickness
TLTHKMIN, TLTHKMAX, TLTHKAVG	Min, max, average tail thickness
CNTHKMIN, CNTHKMAX, CNTHKAVG	Min, max, average center thickness
HDTLRMIN, HDTLRMAX, HDTLRAVG	Min, max, average head's thickness/length ratio
TLTLRMIN, TLTLRMAX, TLTLRAVG	Min, max, average tail's thickness/length ratio
CNTLRMIN, CNTLRMAX, CNTLRAVG	Min, max, average center's thickness/length ratio
HTTHRMIN, HTTHRMAX, HTTHRAVG	Min, max, average head/tail thickness ratio
HCTHRMIN, HCTHRMAX, HCTHRAVG	Min, max, average head/center thickness ratio
TCTHRMIN, TCTHRMAX, TCTHRAVG	Min, max, average tail/center thickness ratio
AMPMIN, AMPMAX, AMPAVG	Min, max, average amplitude of skeleton wave

Table 2.1 continued

CART variable name	Description
AMPRMIN, AMPRMAX, AMPRAVG	Min, max, average amplitude ratio of skeleton wave
ANCHRMIN, ANCHRMAX, ANCHRAVG	Min, max, average angle changing rate of skeleton wave
ANCHSMIN, ANCHSMAX, ANCHSAVG	Min, max, average angle changing rate (S.D.) of skeleton wave
LNMFRMIN, LNMFRMAX, LNMFRAVG	Min, max, average ratio of worm length to MER fill
LNECRMIN, LNECRMAX, LNECRAVG	Min, max, average ratio of length to eccentricity of best-fit eliipse
FATMIN, FATMAX, FATAVG	Min, max, average fatness of worm (ratio worm area to length)
LNWDRMIN, LNWDRMAX, LNWDRAVG	Min, max, average ratio of length to width
CNTMVMIN, CNTMVMAX, CNTMVAVG	Min, max, average moving distance of centroid
HDBRIMIN, HDBRIMAX, HDBRIAVG	Min, max, average brightness of head
TLBRIMIN, TLBRIMAX, TLBRIAVG	Min, max, average brightness of tail
CNTBRMIN, CNTBRMAX, CNTBRAVG	Min, max, average brightness of center
AVEBRMIN, AVEBRMAX, AVEBRAVG	Min, max, average average brightness
HTBRRMIN, HTBRRMAX, HTBRRAVG	Min, max, average head/tail brightness ratio
HDWDMIN, HDWDMAX, HDWDAVG	Min, max, average width of head
TLWDMIN, TLWDMAX, TLWDAVG	Min, max, average width of tail
CNTWDMIN, CNTWDMAX, CNTWDAVG	Min, max, average width of center
AVEWDMIN, AVEWDMAX, AVEWDAVG	Min, max, average average width
HTWRMIN, HTWRMAX, HTWRAVG	Min, max, average head/tail width ratio
HDANGMIN, HDANGMAX, HDANGAVG	Min, max, average head's angle changing rate
TLANGMIN, TLANGMAX, TLANGAVG	Min, max, average tail's angle changing rate
CNTANMIN, CNTANMAX, CNTANAVG	Min, max, average center's angle changing rate
AVEANMIN, AVEANMAX, AVEANAVG	Min, max, average average angle changing rate

Table 2.1 continued

CART variable name	Description
HAREAMIN, HAREAMAX, HAREAAVG	Min, max, average area of head
TAREAMIN, TAREAMAX, TAREAAVG	Min, max, average area of tail
CAREAMIN, CAREAMAX, CAREAAVG	Min, max, average area of center
HDAMPMIN, HDAMPMAX, HDAMPAVG	Min, max, average amplitude of head
TLAMPMIN, TLAMPMAX, TLAMPAVG	Min, max, average amplitude of tail
CTAMPMIN, CTAMPMAX, CTAMPAVG	Min, max, average amplitude of center
AVAMPMIN, AVAMPMAX, AVAMPAVG	Min, max, average average amplitude
HDCTDMIN, HDCTDMAX, HDCTDAVG	Min, max, average distance of head to center
TLCTDMIN, TLCTDMAX, TLCTDAVG	Min, max, average distance of tail to center
HTANGMIN, HTANGMAX, HTANGAVG	Min, max, average angle between head-center and tail-center
HCANGMIN, HCANGMAX, HCANGAVG	Min, max, average anlge between head-center and horizontal line
TCANGMIN, TCANGMAX, TCANGAVG	Min, max, average anlge between tail-center and horizontal line
RADIUMIN, RADIUMAX, RADIUAVG	Min, max, average looping radius
RADSDMIN, RADSDMAX, RADSDAVG	Min, max, average looping radius (S.D.)
OVLENMIN, OVLENMAX, OVLENAVG	Min, max, average looping length
OVLNRMIN, OVLNRMAX, OVLNRAVG	Min, max, average looping radius/body length ratio
TIGHRMIN, TIGHRMAX, TIGHRAVG	Min, max, average tightness (hole area/body area)
LPLTMIN, LPLTMAX, LPLTAVG	Min, max, average time length of looping lasting
LOOPNUM,	Number of loops
BEND	Bending direction of the worm body (dorsal/ventral)

Some values are easily affected by noise or errors during image processing or capture. Throughout this chapter, we used the 90^{th} and 10^{th} percentile values as our maximum and minimum values for each feature in order to avoid extreme values caused by noise or errors during image capture and processing. Some of these features are described in detail:

1) *Worm length and area:* The worm length is the number of pixels on the skeleton. The head, tail and center area are obtained by counting the pixels in these areas.
2) *Width/thickness/fatness:* The widths of the head, tail and center are defined to be the average widths of the head, tail and center sections. Worm thickness is defined in [4, 7] to be the ratio width/length and this is measured at the head, tail and center positions of the skeleton, where the head and tail positions are defined to be 7 pixels away from the head and tail end points on the skeleton. We define the fatness to be the ratio of worm area to length.
3) *Angle change rate:* The angle change rate is defined in [4, 7] as the ratio of the average angle difference between every pair of consecutive segments connected by skeleton points which are 5 pixels apart along the skeleton. The angle change rate measures how sharply a worm body bends.
4) *Brightness:* The brightness can be measured by the median pixel value of the head, tail, center and whole worm body areas.

2.2 Coiling skeletonizing

In this section, we describe details of our coiling skeletonizing algorithm to obtain the morphological skeleton in order to study coiler mutants. Images with holes can be classified into three groups: *A)* images with the worm body touching only in the horizontal (*xy*-plane) direction and having a protruding head/tail (Figure 2.1b), *B)* images with the worm body touching only in the horizontal direction but not having any protruding head/tail (Figure 2.1c), and *C)* images with

the worm body overlapping vertically that is, in the *z*-direction, going out of the plane of the agar plate. For images in group *C*, because the worm covers part of its body with another part, its body area is smaller than its usual size. For every binarized image, we compare the size of the worm body area to a threshold. Among the mutants studied in this paper, *unc-17*, *unc-26* and *unc-32*, which have smaller body areas (5500 pixels) than the other strains, were strong coilers/kinkers. Only these strains have the possibility of coiling so tightly that the skeletonization should be directly abandoned (images in group *C*). So we chose 5000 pixels as the threshold in our experiment, but this threshold could be chosen as 90% of the average worm body area if either a different magnification were used, or if strong coilers with different body sizes were studied. If the body area is smaller than the threshold, we decide the image is in group *C* and abandon it because the correct body posture will not be available when a worm has vertical overlapping.

For images in groups *A* and *B*, after the binarization of the original grayscale image, we obtain the exterior boundary and the interior boundary of the worm body by first eroding it with a 3×3 square structuring element and then performing the set difference between the binary image and its erosion. Figures 2.3a and 2.4a show the grayscale original images of two examples that will ultimately be classified as belonging to class *A*. Figures 2.3b and 2.4b depict the corresponding exterior and interior boundaries. Figure 2.5a shows the grayscale original image of an example that will be classified as class *B*. Figure 2.5b shows the exterior and interior boundaries. In each case, these two boundaries are sampled at an interval of 5 pixels (the sampling interval should be adjusted according to the magnification used for data acquisition) to get $N+1$ sampled points p_i (i = 0, 1, 2, 3…N, where $N+1$ is the total number of sampled points). To decide if this image has a protruding head/tail, the inner angle θ between each pair of segments $p_i - p_{i-2}$ and $p_i - p_{i+2}$ in the exterior boundary will be measured to find a point O which has the furthest distance from the interior boundary among all points with $\theta < 90°$. If no such point exists, then this image is classified as group *B*. Otherwise, this image belongs to group *A*. In

Figures 2.3c and 2.4c, the point *O* is found and the images are classified as class *A*. In Figure 2.5c, there is no point where the interior angle is less than 90°, so the image is class *B* (no protruding head/tail).

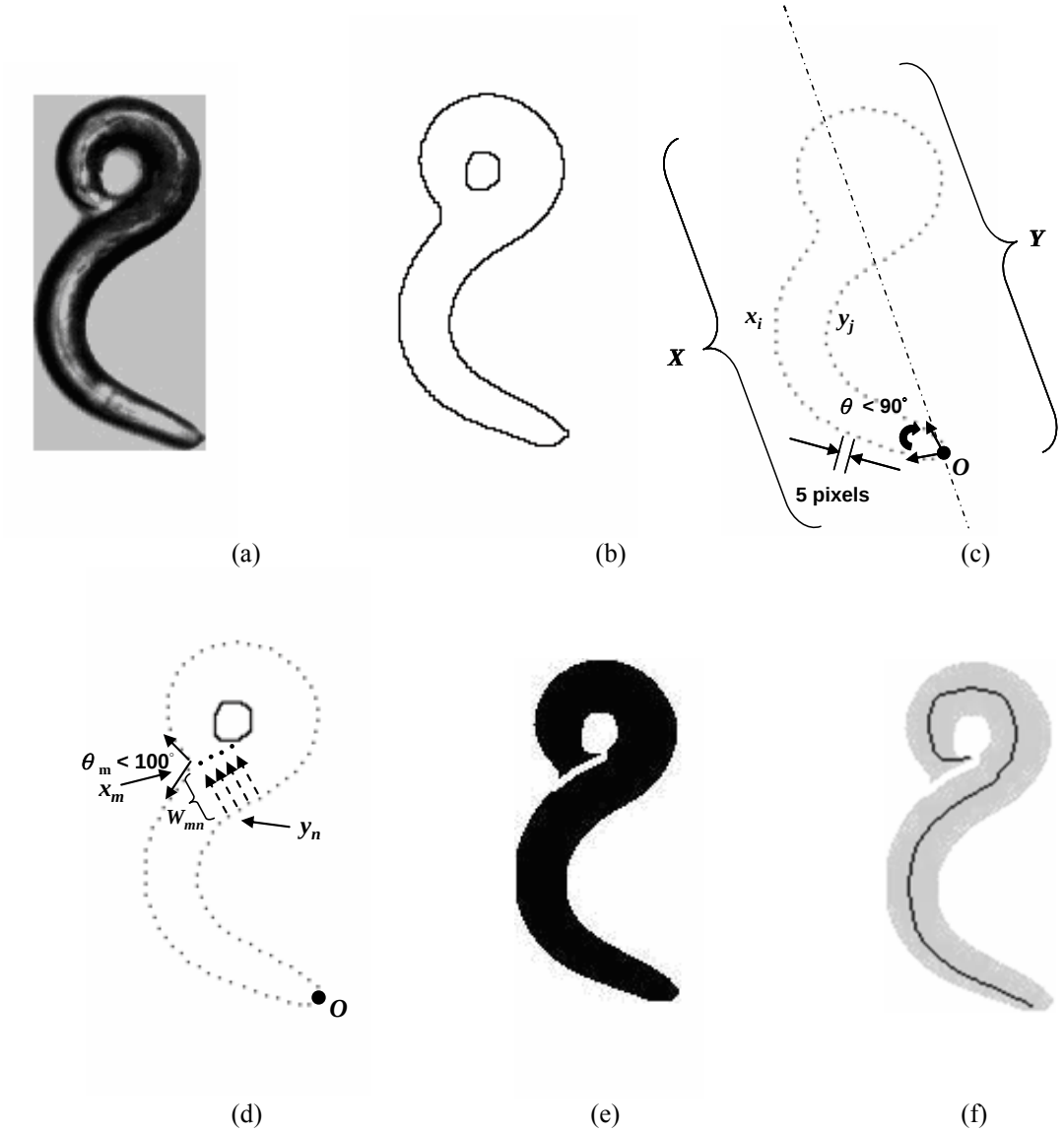


Figure 2.3: An example of coiling skeletonizing for group *A* (example 1). (a) The original grayscale image. (b) The exterior and interior boundaries. (c) Sampled boundary and two sets *X* and *Y*. (d) Finding the starting point and generating the division line. (e) The cut image. (f) The morphological skeleton (solid line) obtained from the cut image.

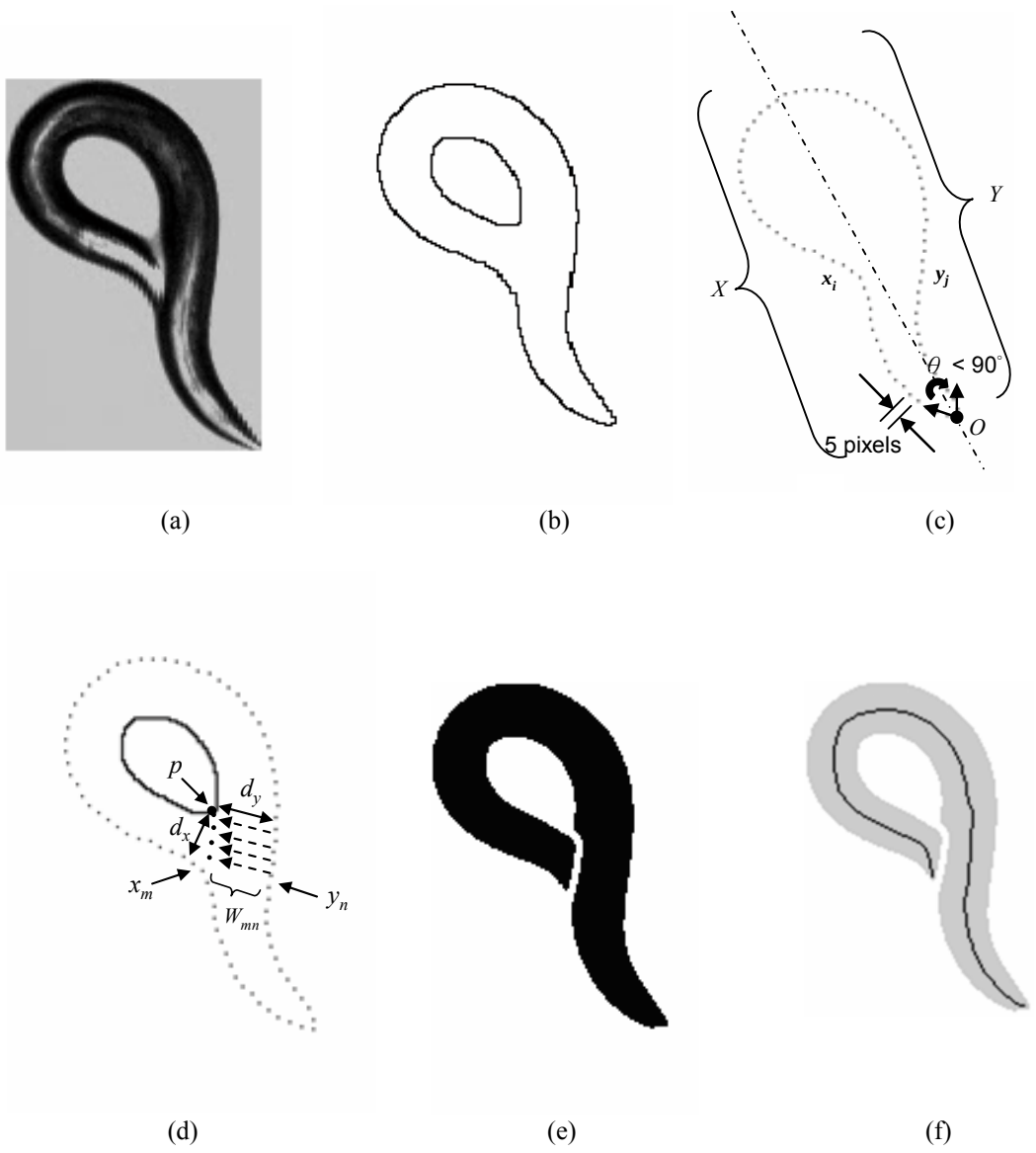


Figure 2.4: An example of coiling skeletonizing for group *A* (example 2). (a) The original grayscale image. (b) The exterior and interior boundaries. (c) Sampled boundary and two sets *X* and *Y*. (d) Finding the starting point and generating the division line. (e) The cut image. (f) The morphological skeleton (solid line) obtained from the cut image.

If this image is in group A, all sampled points are divided into two sets: set X contains $N/2$ points clockwise next to point O and set Y contains $N/2$ points counterclockwise next to point O. For each point x_i in X, we search every point y_j in Y to find the one closest to x_i. We calculate the distance W_{ij} between x_i and y_j as well as the outer angle θ_i (the angle between each pair of segments $x_i - x_{i-2}$ and $x_i - x_{i+2}$). First we check for all points if $\theta_i < 100°$, if the answer is *yes*, then x_i is considered a possible starting point of the worm body-touching. If the answer is *no* for all points, we compare W_{ij} to the average width W of the worm body. If $W_{ij} > W$, then x_i will still be considered a possible starting point. The search continues until the first possible starting point of the body-touching is found. We repeat the same process for the set Y. We have found that at most one set will have a possible starting point with θ smaller than 100° (Figure 2.3d). Sometimes neither set X nor Y has an outer angle θ smaller than 100°. In this case, both sets X and Y will have a possible starting point where $W_{ij} > W$. We find a point p in the interior boundary which is the closest to the two possible starting points found, then we compare d_x and d_y which are the shortest distances from the point p to sets X and Y. Whichever side has the shorter distance will be considered the head/tail part (because the head and tail are less thick) and the possible starting point in that side will be considered the real starting point of the worm body-touching (Figure 2.4d).

Assume x_m is found to be the real starting point and y_n is its closest sampled point in the other set, we can keep locating division points which are W_{mn} (the distance between x_m and y_n) pixels away from the next points y_k ($k > n$) until the interior boundary is reached. These division points are connected to form a division line and then a skeletonizing algorithm is applied on this cut image (Figures 2.3e, 2.4e) to get the correct skeleton (Figures 2.3f, 2.4f).

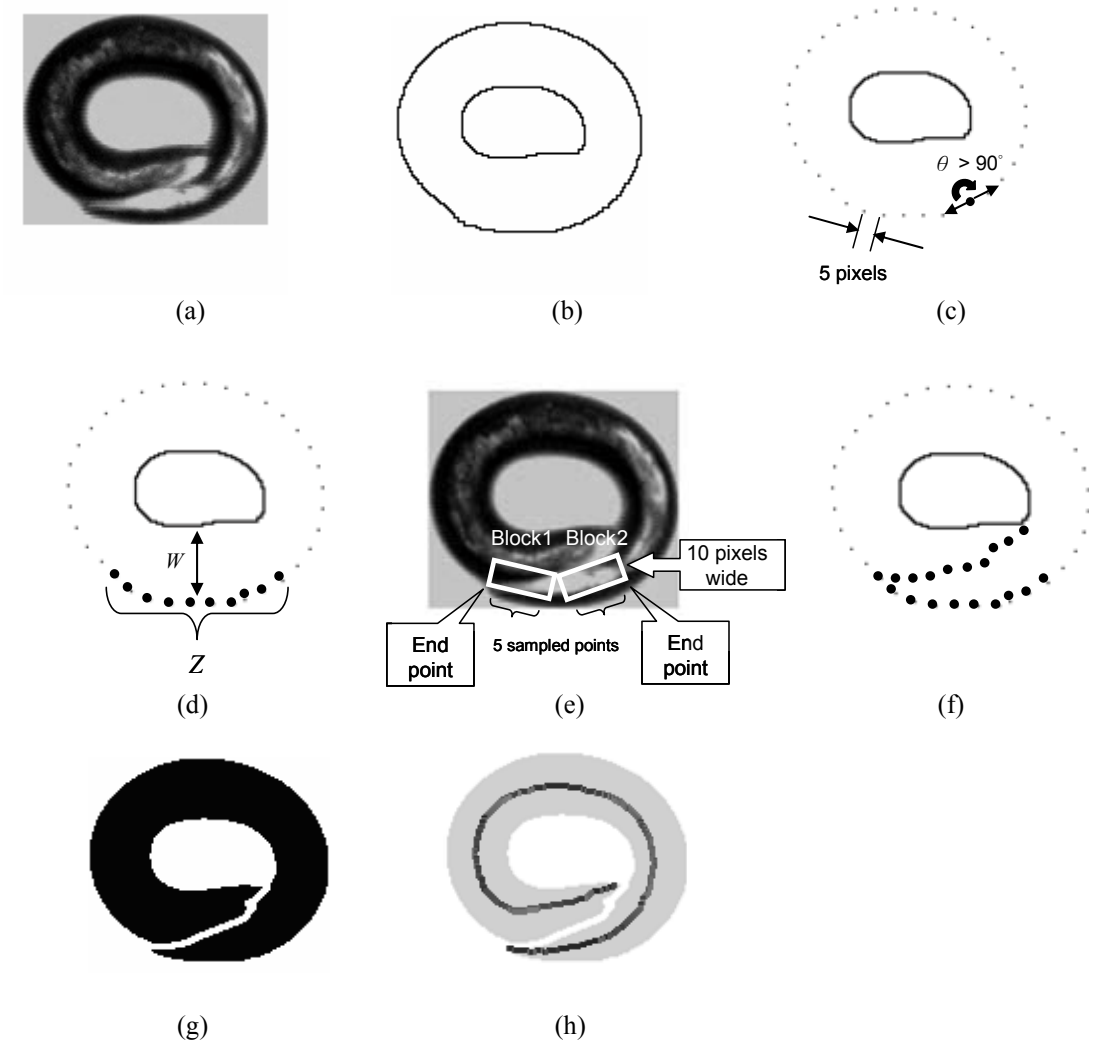


Figure 2.5: An example of coiling skeletonizing for group *B*. (a) The original grayscale image. (b) The exterior and interior boundaries. (c) Sampled boundary. There is no point where the angle is less than 90°.(d) Finding the overlapping part. (e) Comparing the variance of the pixel values in the two areas. (f) Locate division points from the starting point to the interior boundary. (g) The cut image with the division line. (h) The morphological skeleton (solid line) obtained from the cut image.

If the image is in group *B*, we calculate the distance *d* between the interior boundary and every sampled point on the exterior boundary. If $d > W$, we put this point into a set *Z*. The set *Z* covers the overlapping part of the worm body (Figure 2.5d). The two end-points of this set are the two possible starting points of the overlapping. We measure the variance of the pixel values in two rectangular blocks around these two points and compare them (Figure 2.5e). Each block has 10-pixel width with one of the longer edges formed by one end-point and the fifth point from the end-point. The size of the rectangular blocks should be adjusted according to the magnification used. The side with larger variance is considered the starting point because the place where the head or tail tapers to a point usually has very different pixel values compared to the body part it is touching. The histograms of the two rectangular blocks are shown in Figure 2.6. In Figure 2.5f, we can see that a number of points are located gradually from the starting point with increasing distance (0, 2, 4, 6……*W*) from points in the set *Z* until the interior boundary is reached and connected together to generate a division line (Figure 2.5g). Then we apply a standard skeletonizing algorithm on this cut image to obtain the correct skeleton of the worm body (Figure 2.5h). The block diagram of the whole coiling skeletonizing process is shown in Figure 2.7.

Because *C. elegans* can contract and extend its body, small variations in body length can be accepted as correct skeletons. With empirical observation, 20% of L (the average length of the worm body calculated from images without internal holes) was felt to be an upper limit for this variation in body length. Therefore every skeleton obtained with this algorithm was also compared to L. If the difference between them is greater than 20% of L, which is nearly the difference between the maximal and the minimal worm body length in frames without body touching, then this skeleton is assumed to be incorrect and is not used for further feature extraction.

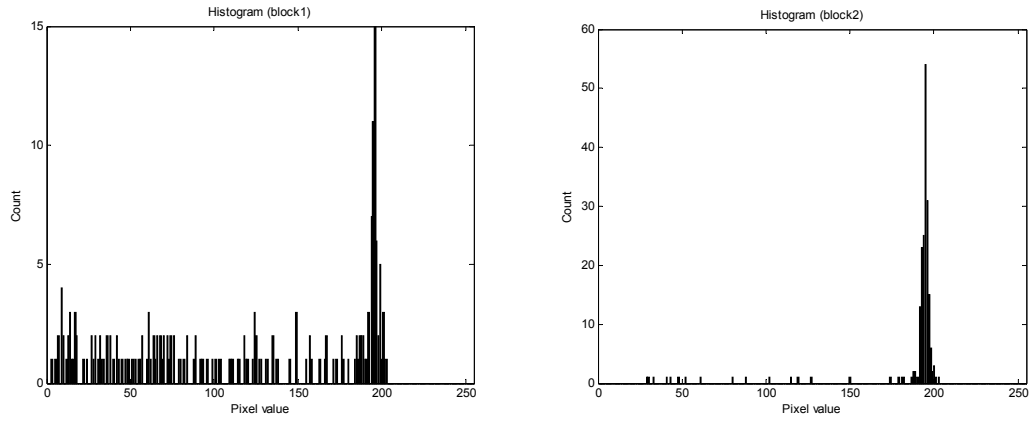


Figure 2.6: The histograms of block1 and block2 in Figure 2.5e.

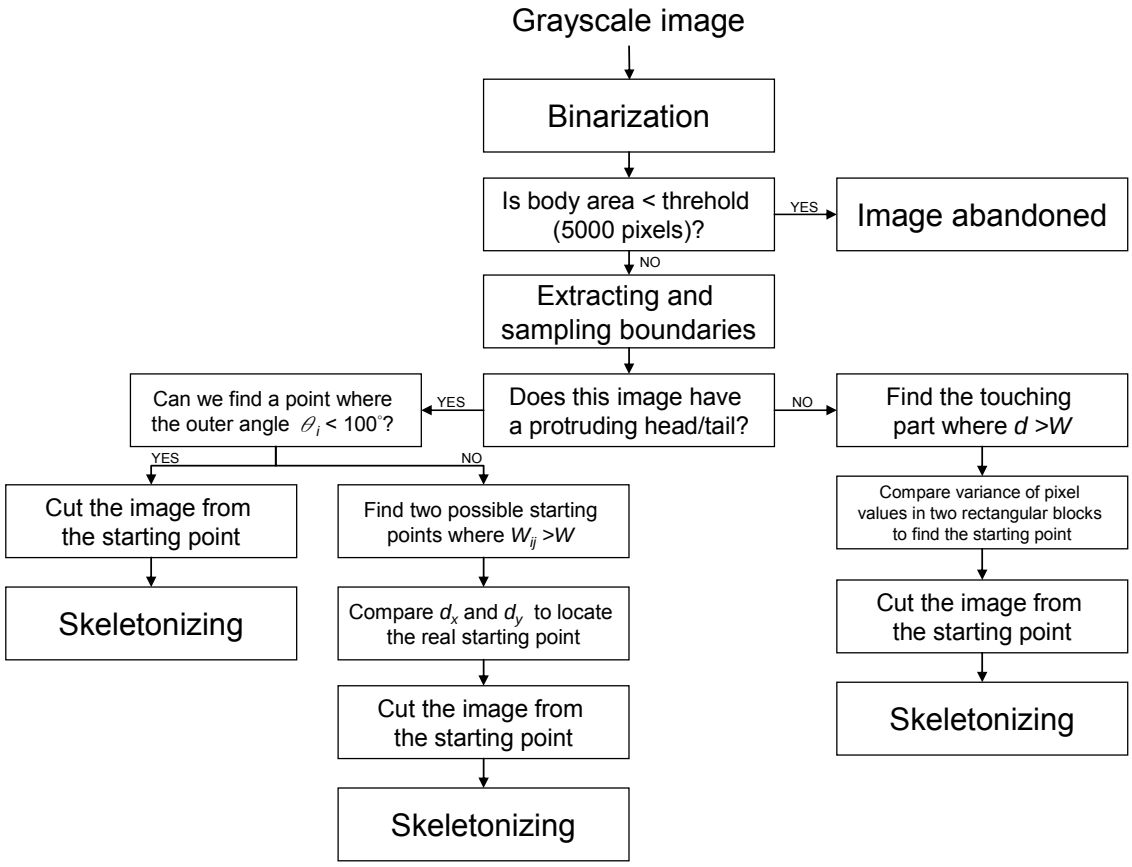


Figure 2.7: The block diagram of the coiling skeletonizing algorithm.

2.3 Classification

The Classification and Regression Trees (CART) algorithm is used to analyze features and classify mutant types [25]. It is a tree structured statistical analysis and data mining tool. It uses a method known as binary recursive partitioning [26]. The term “binary” implies that each group, represented by a node in a decision tree, can only be split into two groups. Thus, each node can be split into two child nodes, in which case the original node is called a parent node. The term “recursive” refers to the fact that the binary partitioning process can be applied over and over again. Thus, each parent node can give rise to two child nodes and, in turn, each of these child nodes may themselves be split, forming additional children. The term “partitioning” refers to the fact that the dataset is split into sections or partitioned. CART analysis consists of four basic steps [25]. The first step consists of tree building, during which a tree is built using recursive splitting of nodes. Each resulting node is assigned a predicted class. The rule is to assign the largest percentage of cases for each terminal node, which is called plurality. The assignment of a predicted class to each node occurs whether or not that node is subsequently split into child nodes. The second step consists of stopping the tree building process. At this point a “maximal” tree has been produced, which probably greatly overfits the information contained within the learning dataset. The third step consists of tree “pruning”, which results in the creation of a sequence of simpler and simpler trees, through the cutting off of increasingly important nodes. The fourth step consists of optimal tree selection, during which the tree which fits the information in the learning dataset, but does not overfit the information, is selected from among the sequence of pruned trees. The goal of CART is to successively divide the training set in a way that the data associated with the terminal nodes of the tree do not have a mix of classes; rather each node should be as pure as possible. In this dissertation, by using extracted features, we were able to distinguish different coiler mutants using this binary decision tree algorithm.

2.4 Results

2.4.1 Verification of the skeleton algorithm by human observers

The coiling skeletonizing algorithm was tested on 55 5-minute videos (8Hz) from 11 mutant types. Of the 132000 image frames in these videos, more than 26000 of them involved body touching or overlapping. For all of these, cut images and skeletons were generated by the algorithm for a human observer to verify. The positions of the start and end points of body touching in every cut image were examined. The skeletons were also compared to the grayscale images to decide if the obtained body posture was correct. Experimental results are shown in Table 2.2. The rate of obtaining a biologically correct skeleton is over 93%.

Table 2.2: Verification results for the coiling skeletonizing algorithm. Data were collected from 55 5-minute videos (8Hz) from 11 mutant types. The first column shows the mutant type. The second column shows the number of correct skeletons obtained with our algorithm. The number of wrong skeletons not due to vertical body overlapping is listed in column 3. The number of wrong skeletons due to vertical body overlapping is listed in column 4. The average correct rate is over 93%.

Strain name	Correct skeletons	Wrong skeletons	Wrong skeletons due to vertical body overlapping	Frames rejected by the comparison to the threshold (5000 pixels)
syd-1(ju82)	649 (94.5%)	16 (2.3%)	22 (3.2%)	0
unc-1(e1598)	5178 (99.2%)	41 (0.8%)	0	0
unc-3(e151)	1705 (93.0%)	125 (6.8%)	3 (0.2%)	0
unc-10(e102)	1690 (90.8%)	20 (1.1%)	152 (8.2%)	0
unc-17(e245)	3245 (82.7%)	156 (4.0%)	525 (13.3%)	3895
unc-26(m2)	3358 (91.4%)	79 (2.2%)	235 (6.4%)	3326
unc-32(e189)	5321 (97.4%)	40 (0.7%)	104 (1.9%)	0
unc-37(e262)	2310 (91.6%)	35 (1.4%)	176 (7.0%)	0
unc-75(e950)	946 (93.6%)	37 (3.7%)	28 (2.7%)	160
unc-77(e625)	1647 (96.1%)	20 (1.2%)	47 (2.7%)	0
Wild type (N2)	189 (99.5%)	1 (0.5%)	0	0
Total	26238 (93.4%)	570 (2.0%)	1292 (4.6%)	7381

2.4.2 Assessment of skeleton algorithm by classification of coiler mutants

To evaluate the effectiveness of our system in characterizing the postures of coiled animals, we tested the ability of the automated binary classifier CART to correctly identify different mutant strains that frequently adopt coiled postures. We compared our data from 10 different uncoordinated mutants with wild-type worms. The test group included five mutants categorized by [1] as "weak coilers" (*unc-3, unc-10, unc-37, unc-75*, and *unc-77*) as well as two "strong coilers" (*unc-17* and *unc-32*), one "forward uncoordinated" mutant (*unc-1*), one "strong kinker" (*unc-26*) and one mutant with superficially normal locomotion (*syd-1*). The data set contained 40 5-min videos at 8Hz of each mutant type. Shown in Figure 2.8, a classification tree with 11 terminal nodes (depicted as ellipses) was generated by CART using 10-fold cross validation to determine the tree size. Only 9 different features, appearing in diamond-shaped boxes in Figure 2.8, were chosen by CART and used as splitting nodes in this classification tree. The names and descriptions of these 9 features were as follows: ANCHSMAX - Max value of angle change rate; AVEBRMIN - Min value of worm average brightness; CNTMVAVG - Ave value of centroid movement distance; HAREAMIN - Min value of head area; HDTLRMAX - Max value of head thickness-length ratio; HTTHRMAX - Max value of head-tail thickness ratio; LNGTHMIN - Min value of worm body length; TAREAMIN - Min value of worm tail area; TLCTDAVG - Ave value of tail-centroid distance. This subset of features selected from among the 188 features to be used as the basis for splitting, as well as the thresholds used for splitting, were chosen automatically by the CART algorithm based on the data, with the cross-validation used to guard against overfitting. At each step, a splitting test separates a parent node into exactly two child nodes based on a criterion related to a single feature. For example, suppose we have a worm of unknown type with **TLCTDAVG** = 113.82, **LNGTHAVG** = 193.43 and **ANCHSMAX** = 11.72. Starting at the top, the first question is:

Is **TLCTDAVG** $\leq$ 116.09?

In this case, the answer is *yes* and the case goes to the left. This classification tree will do the same for **LNGTHAVG** and then **ANCHSMAX** and the worm winds up in the ellipse on the left and is classified as *unc-75*. The classification result shows the significance of the previously generated skeletons. Several of the features such as angle change rate, body length as well as head-tail recognition rely on measurements of the skeleton.

The cross validated classification probabilities for the system with the new algorithm compared to the standard morphological skeleton algorithm are given in Tables 2.3A and B respectively. The mutant names of all video data are in the first column and the first row lists the groups to which each video was classified. The success rates are listed along the diagonal while the off-diagonal entries represent the misclassification rates. We see that the CART tree using the data from the new skeletonizing algorithm has significant improvement for several mutants. For example, the correct classification rate for *syd-1* improves by 12.5%, *unc-10* by 20%, *unc-37* by 10%, and *unc-77* by 12.5%. There are a few minor changes as well: *unc-3* gets worse by 5%, and *unc-17* and *unc-32* each improve by 2.5%. Overall, the new skeletonizing algorithm allows us to do better classification for the video data. Given that the old system already showed better success at correctly classifying difficult-to-distinguish mutant types than a human expert [7], we consider this to be a good result.

Table 2.3: The cross validated classification probability tables from two systems. In each table, the first column lists the actual mutant types for the video data. The first row lists the results of the classification procedure. The success rates are listed along the main diagonal while the off-diagonal entries represent the misclassification rates.

A. Results with new skeletonizing algorithm:

	syd-1	*unc-1*	*unc-3*	*unc-10*	*unc-17*	*unc-26*	*unc-32*	*unc-37*	*unc-75*	*unc-77*	Wild type (N2)
syd-1	**80.0%**	0	0	2.5%	0	0	0	2.5%	0	0	15.0%
unc-1	5.0%	**85.0%**	0	0	0	2.5%	2.5%	0	0	0	5.0%
unc-3	5.0%	7.5%	**67.5%**	0	0	0	10.0%	7.5%	2.5%	0	0
unc-10	0	2.5%	0	**92.5%**	0	0	0	0	0	2.5%	2.5%
unc-17	0	0	0	0	**77.5%**	10.0%	5.0%	0	0	5.0%	2.5%
unc-26	0	0	0	0	22.5%	**72.5%**	0	0	5.0%	0	0
unc-32	0	12.5%	0	0	7.5%	2.5%	**57.5%**	10.0%	5.0%	5.0%	0
unc-37	5.0%	0	7.5%	0	0	2.5%	10.0%	**67.5%**	0	2.5%	5.0%
unc-75	0	2.5%	0	0	7.5%	0	2.5%	0	**87.5%**	0	0
unc-77	0	7.5%	5.0%	12.5%	10.0%	0	0	2.5%	0	**62.5%**	0
Wild type (N2)	25.0%	5.0%	0	0	0	0	0	0	0	2.5%	**67.5%**

B. Results with previous system [7]:

	syd-1	*unc-1*	*unc-3*	*unc-10*	*unc-17*	*unc-26*	*unc-32*	*unc-37*	*unc-75*	*unc-77*	Wild type (N2)
syd-1	**67.5%**	0	0	5.0%	0	0	2.5%	2.5%	0	0	22.5%
unc-1	5.0%	**85.0%**	0	0	0	2.5%	2.5%	0	0	0	5.0%
unc-3	5.0%	7.5%	**72.5%**	0	0	0	7.5%	0	2.5%	2.5%	2.5%
unc-10	0	0	5.0%	**72.5%**	0	0	15.0%	0	0	5.0%	2.5%
unc-17	2.5%	0	0	0	**75.0%**	12.5%	5.0%	2.5%	0	2.5%	0
unc-26	0	0	0	0	20.0%	**72.5%**	2.5%	0	5.0%	0	0
unc-32	0	10.0%	0	12.5%	5.0%	2.5%	**55.0%**	7.5%	5.0%	2.5%	0
unc-37	0	0	10.0%	5.0%	0	2.5%	12.5%	**57.5%**	0	7.5%	5.0%
unc-75	0	2.5%	0	0	7.5%	0	2.5%	0	**87.5%**	0	0
unc-77	0	7.5%	5.0%	10.0%	10.0%	0	0	17.5%	0	**50.0%**	0
Wild type (N2)	22.5%	5.0%	0	2.5%	0	0	0	0	0	2.5%	**67.5%**

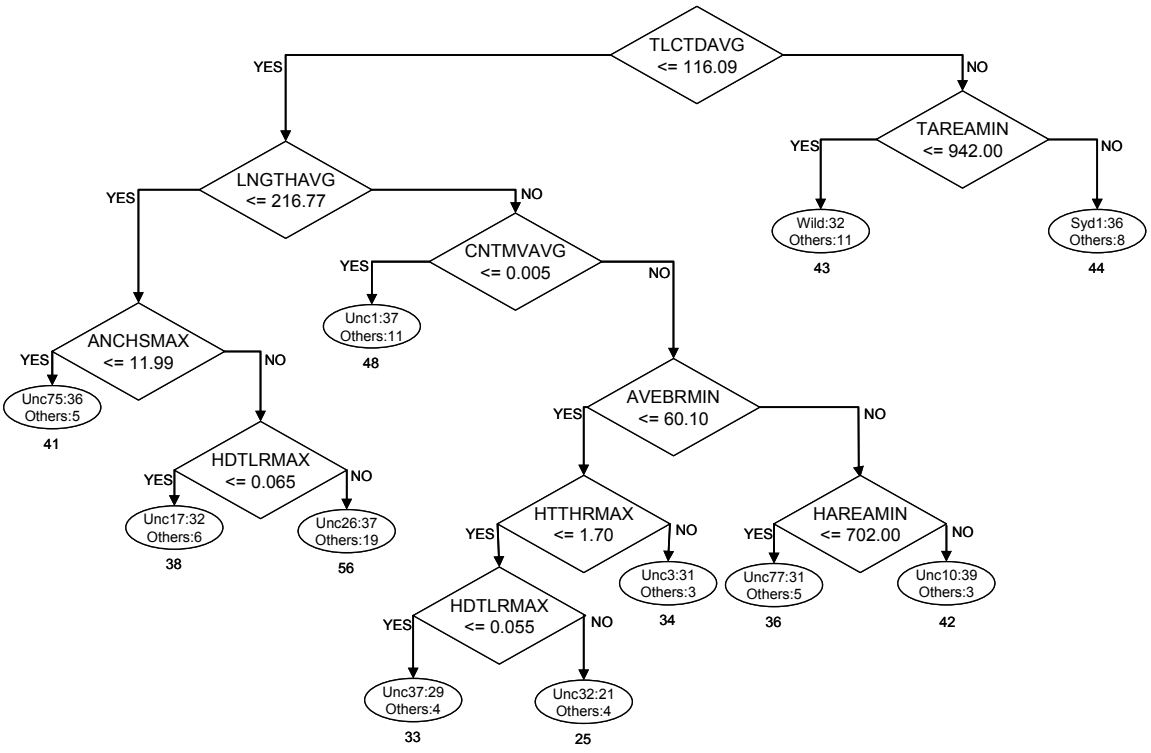


Figure 2.8: The classification tree reliably identifies the type of a given worm using only 9 features. The tree was constructed using the CART algorithm as described.

2.4.3 Characterization of coiler mutants

In order to study the similarities among different mutant types, we analyzed the clustering of these 440 worms. In particular, we sought to determine how the feature data clustered in multidimensional space and to then correlate the clustering pattern of the feature data with the known biology of the mutant types. For each worm, 188 features describing aspects of the animal's movement, body texture, or body posture were measured and designated as a single data point with multiple dimensions. Because we measure those features in different units, all features were normalized to avoid one feature dominating others. We used the Sigmoidal method

[27] for this normalization. It is defined as $y = \frac{1 - e^{-x}}{1 + e^{-x}}$, where $x = \frac{f - mean(f)}{stdev(f)}$ and f is the original input feature.

Principal Component Analysis (PCA) [28] was then used to project our 188-dimensional data set to two dimensions (Figure 2.9). Data points belonging to the same mutant type occupy one local area of feature space. We found that the clouds for *unc-17*, *unc-26* and *unc-32*, which have been described in the literature as having strong kinker (*unc-26*) and strong coiler (*unc-17* and *unc-32*) phenotypes [1] were located together on the left of the feature space, far from the cloud of wild type. The clouds of *unc-3*, *unc-10*, *unc-37*, *unc-75* and *unc-77*, which were described as weak coilers, all locate near the middle of the space (and are all shown with 'x' markers in Figure 2.9). The centroid of each data cloud is depicted by a square mark. The Euclidean distances between cluster centroids are given in Table 2.4. In the weak coiler group, centroids of mutants with similar sizes are closer to each other (*unc-3*, *unc-10* and *unc-37* are bigger worms; *unc-75* and *unc-77* are smaller). The cloud for *syd-1*, which behaves superficially like wild-type, was the closest among all mutant types to the centroid of wild type.

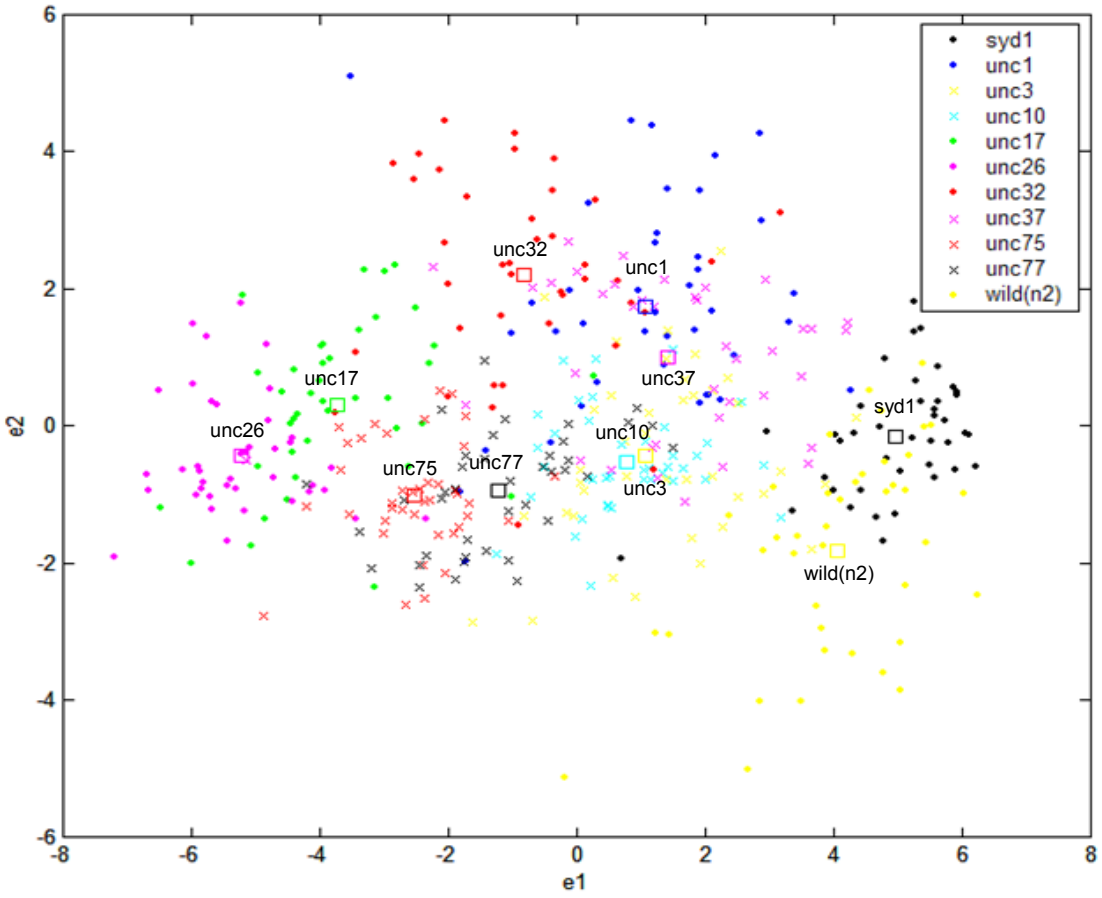


Figure 2.9: Distribution of behavior data points in feature space.

Table 2.4: Euclidean distances between cloud centroids

	unc1	*unc3*	*unc10*	*unc17*	*unc26*	*unc32*	*unc37*	*unc75*	*unc77*	Wild (n2)
syd1	4.33	3.63	4.21	8.71	10.20	6.24	3.73	7.53	6.24	1.89
unc1		2.19	2.29	5.02	6.66	1.94	0.82	4.52	3.52	4.65
unc3			0.58	5.13	6.57	3.40	1.43	3.90	2.62	3.05
unc10				4.58	5.99	3.15	1.65	3.32	2.03	3.54
unc17					1.67	3.48	5.20	1.79	2.80	8.08
unc26						5.14	6.79	2.77	4.03	9.39
unc32							2.53	3.63	3.16	6.32
unc37								4.41	3.27	3.86
unc75									1.30	6.63
unc77										5.36

Table 2.5: Classification using cloud centroids

	syd1	*unc1*	*unc3*	*unc10*	*unc17*	*unc26*	*unc32*	*unc37*	*unc75*	*unc77*	Wild (n2)
syd1	**77.5%**	0	2.5%	2.5%	0	0	0	0	0	0	17.5%
unc1	2.5%	**40.0%**	0	2.5%	0	0	15.0%	30.0%	0	10.0%	0
unc3	2.5%	5.0%	**22.5%**	25.0%	0	0	2.5%	22.5%	0	10.0%	10.0%
unc10	0	2.5%	32.5%	**42.5%**	0	0	2.5%	5.0%	0	12.5%	2.5%
unc17	0	0	0	0	**57.5%**	22.5%	7.5%	2.5%	7.5%	2.5%	0
unc26	0	0	0	0	15.0%	**80.0%**	0	0	5.0%	0	0
unc32	0	15.0%	2.5%	0	5.0%	0	**65.0%**	0	2.5%	10.0%	0
unc37	15.0%	25.0%	15.0%	5.0%	0	2.5%	12.5%	**20.0%**	0	2.5%	2.5%
unc75	0	0	0	0	15.0%	5.0%	0	0	**62.5%**	17.5%	0
unc77	0	0	5.0%	17.5%	0	2.5%	2.5%	0	27.5%	**45.0%**	0
Wild(n2)	35.0%	0	2.5%	5.0%	0	0	0	0	0	2.5%	**55.0%**

These centroids can also be used to classify these coilers (that is, a worm is classified based on which cluster centroid it is closest to). But some mutants are very close to each other and many data points are incorrectly classified. In Table 2.5, we see that the success rate is lower than 30% for *unc-3* and *unc-37* (marked in red). To study further similarities between these close worms, we use k-means clustering to investigate the natural structure of the data, and the Gap Statistic algorithm to determine the optimal number of clusters [8, 29]. For this analysis, each data point was treated individually without regard to mutant type. The k-means algorithm is an elementary but very popular clustering method. Suppose we have k clusters with centers $C = \{c_1,...c_k\}$ and their corresponding non-overlapping divisions of feature space are defined as $D = \{D_1,...D_k\}$. Let $\|.\|^2$ denote "squared Euclidean distance" and our data set is x_i: i = 1, 2,….440. Lloyd (1957) developed an algorithm for k-means clustering which will always converge. It starts with an initial set of k representative points. All points in our data set are assigned to whichever of the k points is closest based on squared Euclidean distance. Next, each of the k representative points will be relocated to be the centroid of the data points which were assigned to it. We choose $C = \{c_1,...c_k\}$ so that $C = \arg\min_C \sum_{j=1}^{k} \sum_{x_i \in D_j} \|x_i - c_j\|^2$. Then we have a new set of k representative points and repeat the same process from the assignment step. The algorithm iterates between steps of data point assignment and cluster centroid calculation until

convergence is reached. The algorithm will not necessarily find the global optimal cluster centroids because the final convergence relies on the initial choice of k representative points.

Another key issue in k-means clustering is to find the optimal number of clusters in the data set. In this dissertation, we used the Gap Statistic [29] to determine the optimal cluster number for our behavioral data. Briefly, the Gap Statistic standardizes the graph of $\log(W_k)$ by comparing it to its expectation under an appropriate null reference distribution of the data, where W_k is the total within-cluster sum of squares around the cluster centers and k is the number of clusters.

We generate 440 samples from a reference distribution which is uniform along each feature dimension. We do this B times (we use B = 10 in this dissertation), then we compute the Gap statistic as follows:

$$Gap(k) = \frac{1}{10}\sum_{i=1}^{10}\log(W_{ik}^{*}) - \log(W_k)$$

where W_{ik}^{*} is the within-cluster sum of squares of the i[th] reference dataset and k = 1, 2, 3, ……K, where K is the predetermined maximum number of clusters. Eleven strains were investigated in this experiment, so we use K = 11 as our maximum number. The Gap plot is shown in Figure 2.10. Because k = 7 is the first k that satisfies $Gap(k) \geq Gap(k+1)$, 7 is identified as the optimal number of clusters.

In the new cluster plot (Figure 2.11 and Table 2.6) with 7 centers, we can see that *syd-1* mostly comprises a group by itself (group 1) as do *unc-32* and wild type (groups 5 and 7). The *unc-1* and *unc-37* are grouped into group 2, *unc-3* and *unc-10* into group 3, *unc-17* and *unc-26* into group 4 and *unc-75* and *unc-77* into group 6. Strains belonging to the same group tend to have similar behavioral patterns. Table 2.6 shows us the new classification result using these 7 centers.

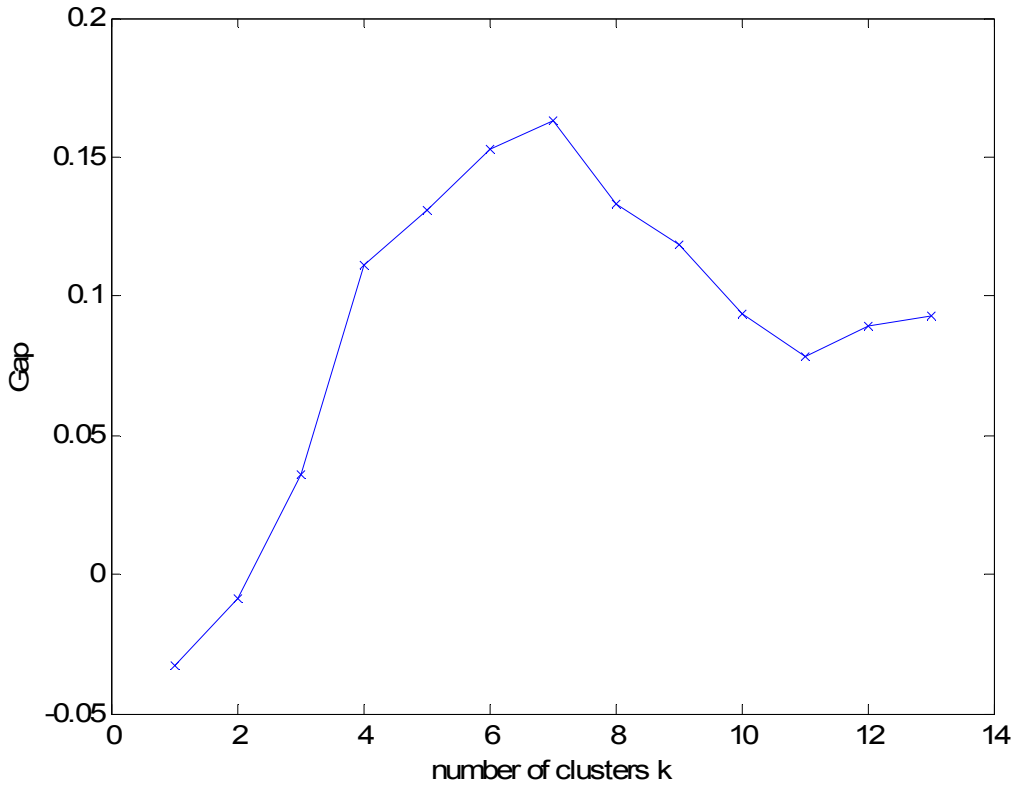


Figure 2.10: Gap plot obtained by the Gap Statistic algorithm. We identified the optimal number of clusters as the first point k where Gap(k) $\geqq$ Gap(k+1).

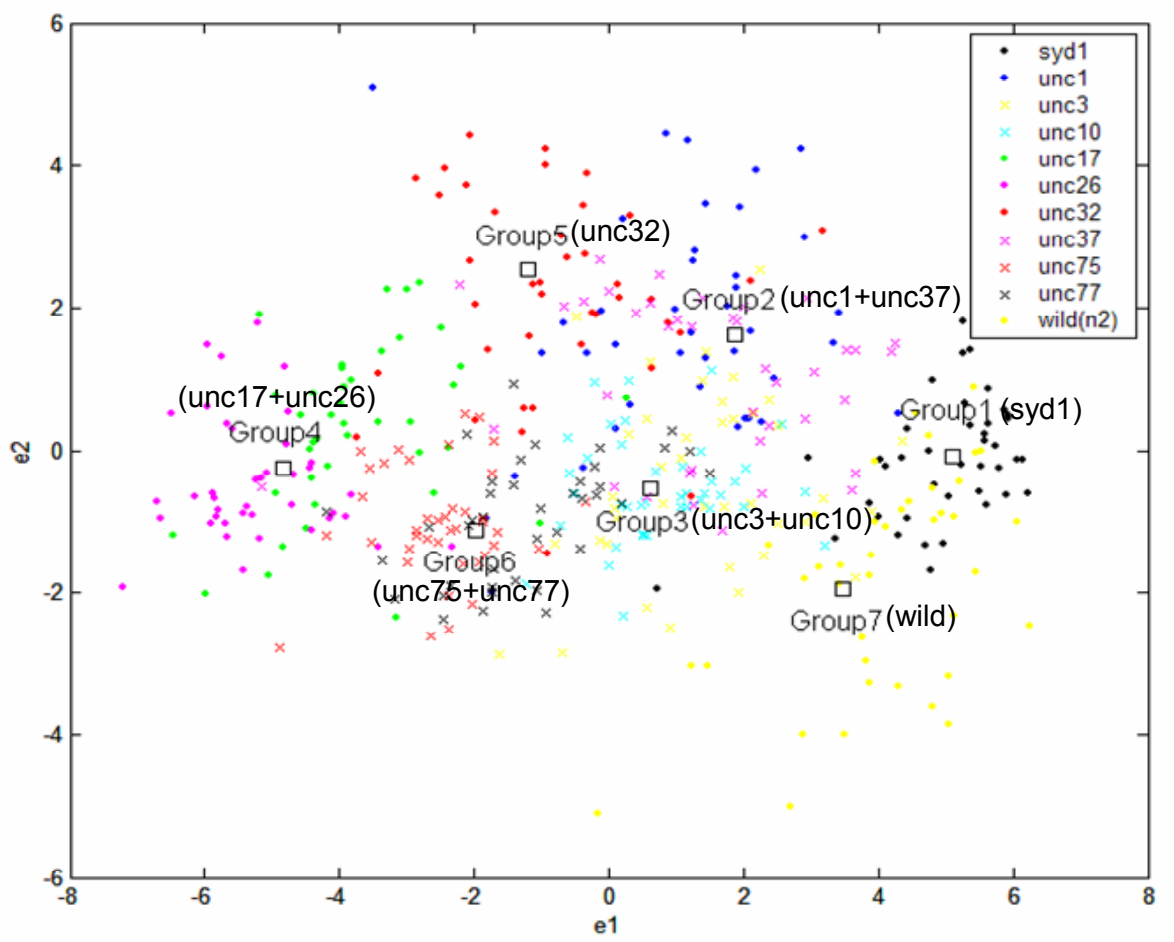


Figure 2.11: Cluster plot with 7 centers marked as square.

Table 2.6: Classification using new cloud centroids

	Group1	Group2	Group3	Group4	Group5	Group6	Group7
syd1	**82.50%**	0	2.50%	0	0	0	15.00%
unc1	2.50%	**62.50%**	7.50%	0	20.00%	7.50%	0
unc3	2.50%	27.50%	**42.50%**	0	2.50%	7.50%	17.50%
unc10	0	5.00%	**85.00%**	0	0	5.00%	5.00%
unc17	0	0	2.50%	**65.00%**	17.50%	15.00%	0
unc26	0	0	0	**95.00%**	0	5.00%	0
unc32	0	15.00%	2.50%	5.00%	**65.00%**	12.50%	0
unc37	10.00%	**50.00%**	20.00%	2.50%	12.50%	2.50%	2.50%
unc75	0	0	2.50%	15.00%	0	**82.50%**	0
unc77	0	0	32.50%	2.50%	2.50%	**62.50%**	0
Wild(n2)	40.00%	0	0	0	0	2.50%	**57.50%**

2.5 Conclusion

The contributions of this chapter are as follows. This is the first automated study of *C. elegans* coiler mutants. Our new algorithm extracts biologically meaningful skeletons from coiled body postures having internal holes. This represents an important advance in the automated analysis of nematode behavior, particularly with respect to the study of certain classes of locomotion-abnormal mutants. Whereas wild-type nematodes, and animals with locomotion patterns close to wild-type, coil infrequently, for coiler mutants a significant proportion of frames in a given video sequence will contain internal-hole images.

Moreover, we have demonstrated that locomotion features extracted from obtained skeletons can be used to classify coilers with highly similar phenotypes. Among the coilers we studied, *unc-3, unc-10, unc-37, unc-75* and *unc-77* are all described as weak coilers; *unc-1, unc-17, unc-26* and *unc-32* are described as strong coilers or kinkers. Yet each of these types could be distinguished from others in its described class by a classifier using the features extracted by our system. For example, from our classification tree (Figure 2.8), we can see that *unc-75* is the shortest within weak coilers, and has smaller values of length-related features (body length and tail-centroid distance). *unc-10* is the largest weak coiler and it has longer length and larger body area. For strong coilers, *unc-32* is the longest and *unc-1* showed particularly low centroid movement. *unc-26*, which has larger head width-length ratio, tends to have a fatter head when compared to *unc-17* (where fatness is defined as the ratio of worm area to length). *syd-1* does not coil very often and moves very smoothly, similar to wild type in previous human observation. We found that it also has length very similar to wild type. The most significant difference between *syd-1* and wild-type is that *syd-1* has larger tail area than wild-type.

This study also provides insight into the quantitative basis for the previously described categories of uncoordinated mutants. To investigate the relative similarities between the different mutant behavior patterns, we analyzed their clustering in multidimensional feature space. Using

the Gap Statistic, we determined that the optimal number of clusters for these coilers is 7. This indicated that even among a single descriptive class of Unc mutants (e.g. weak coilers), several distinct phenotypic patterns could be observed. Among the weak coilers, three natural clusters were identified, consisting of *unc-3*/*unc-10*, *unc-75*/*unc-77* and *unc-1*/*unc-37*. Interestingly, in at least some cases, animals with different descriptive categorization exhibit highly similar phenotypes when evaluated by quantitative criteria. For example *unc-1* was not previously described as a weak coiler, but rather as "forward uncoordinated", yet both natural clustering and Euclidean centroid distance indicated a very similar phenotype to the weak coiler *unc-37*. Likewise, *unc-17*, a "strong coiler" and *unc-26*, a "strong kinker" comprised a single natural cluster and were mutually closest to one another in Euclidean centroid distance. These results illustrate the limitations of subjective definitions of mutant types with respect to both precision and accurancy.

Part of this chapter is a reprint of the material as it appears in Kuang-Man Huang, Pamela Cosman, and William Schafer, "Machine Vision Based Detection of Omega Bends and Reversals in *C. elegans,*" Journal of Neuroscience Methods, Vol. 158, Issue 2, pp. 323-336, December 2006. I was the primary researcher and the co-authors Dr. Pamela Cosman and Dr. William Schafer directed and supervised the research which forms the basis for this chapter. This work was supported by a research grant from the National Institutes of Health (R01 DA018341).

Chapter 3

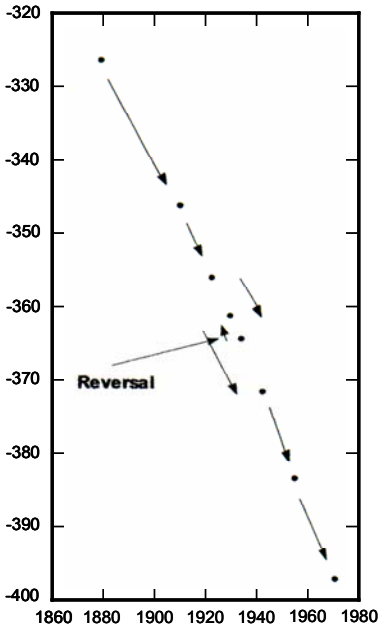
Detection and analysis of omega bends and reversals

In this chapter, we describe a novel algorithm to automatically detect omega bends and reversals. It relies in part on the new method described in chapter 2 for obtaining a morphological skeleton from a coiled worm. Omega bends occur when the worm takes on the shape of a capital omega; the worm curves its head around to touch the middle part of its body, then sharply bends away from its body (Figure 2.2a). This posture commonly occurs when the worm makes a large turn with a reorientation of its movement direction greater than 135° [11]. Turns of this sort have been shown to be critical for navigation (or taxis) behaviors used to seek food and other chemoattractants and to avoid toxins and other chemorepellents [12]. Different kinds of omega bends also exist for certain worm mutants such as *syd-1(ju82), unc-10(e102)* and *unc-37(e262)*. For *syd-1(ju82)* and *unc-10(e102)*, instead of resembling the Greek letter Ω, the worm's head sometimes touches the middle of the body, then crosses underneath it and goes in another direction. Identifying omega bends is more difficult for *unc-37(e262)* because the movement of this mutant type is jerky or even interrupted by other movements. Studies of omega bends have

always required training of human observers because the movement is difficult to detect. The studies are also time-consuming and require observation by humans. In this chapter, we describe the details of our automated skeleton-based algorithm. This new algorithm has made it possible to reliably detect events and characterize relevant parameters.

We also propose a skeleton-based algorithm to detect reversals. *C. elegans* usually moves in a sinusoidal wave. When a worm is touched or presented with a toxic chemical stimulus, it will switch the direction of the wave, causing the animal to instantaneously crawl backward instead of forward. This backward movement is defined as a reversal and is characterized by the distance and frequency of the worm moving back into the recent previous path. Several methods to detect reversal movement using the animal's centroid were developed in previous studies. In [4], the trajectory of the centroid was sampled at intervals of constant distance. The turning angle at every vertex was computed; if the angle was greater than 120°, then the position was considered to be a reversal (Figure 3.1a). In [7], a moving window was used to record the previous 20 centroid locations. A reversal was detected when the new centroid was closer to any of the 19 previous centroid locations than to the most recent past (Figure 3.1b). However, sometimes the worm head's bending movement can affect the centroid's position. Using centroids alone may cause some non-reversal movements to be classified as reversals. [6] used relative distances between the positions of head/tail in adjacent frames and skewer fits to detect reversals. A skewer fit is the line segment connecting the head and the tail. Instead of using the distance between the head and the tail as in [6], we found it to be more robust to use two points near the two ends as our reference points to decide if the worm was moving forward or backward. In [6], the angle changing of skewer fits was also used to eliminate certain false positives, but is discarded in our algorithm.

(a)



(b)

Figure 3.1: Reversal detection based on centroid: (a) directional change detection method, (b) centroid location tracking method.

Together, these new algorithms make it possible to reliably detect events that are time-consuming and laborious to detect by real-time observation or human video analysis. They make it possible to identify mutants with subtle behavioral abnormalities, such as those in which omega bends are dorsoventrally unbiased or uncorrelated with reversals. These methods should therefore facilitate quantitative analysis of a wide range of locomotion-related behaviors in this important neurobiological model organism. In chapters 1 and 2, we described the skeleton algorithm, including image acquisition, pre-processing, and its use in feature extraction and classification. In this chapter, first we present skeleton-based algorithms to detect omega bends and reversals. Then we compare omega bend and reversal behavior of mutant and wild type animals and investigate the temporal correlation between these body movements.

3.1 Omega bend detection

A typical omega bend starts with the worm making a big turn by approaching its body with its head, then its head will pass its tail and go to a new direction. As defined in [30], the reorientation has to be greater than 135° or the worm has to touch its body in order for the movement to be considered an omega bend. For coiler mutant types, sometimes an omega bend is interrupted by other movements before it is finished. Our goal is to construct an algorithm which can automatically detect complete omega bend events that meet the definition as given in [30]. In our algorithm, we divide an omega bend into three parts:

1) Start of an omega bend:

   For each frame, we compare the distance d_{hm} (the distance between the worm head and the middle skeleton point which is defined as the 15th skeleton point) and the distance d_{tm} (the distance between the worm tail and the middle skeleton point) and calculate the angle θ between these two segments. We use L to denote the worm's body length. If d_{hm}

$< d_{tm} - 0.05L$ and $\theta < 45°$, then this frame will be considered a starting frame of an omega bend (Figure 3.2a).

2) Middle of an omega bend:

Because the reorientation of an omega bend has to be greater than 135°, the angle θ (same definition as in Start of an omega bend) has to be smaller than 45° in all frames of an omega bend (Figure 3.2b). Sometimes a worm touches itself during an omega bend. In all frames with touching, the bending angle θ is obviously smaller than 45° (close to 0), so these frames also satisfy the criterion of the middle of an omega bend.

3) End of an omega bend:

When a worm comes out of an omega bend, its head moves away from its body to a new direction. At this moment, its tail should be closer to its middle skeleton point then its head. So $d_{tm} < d_{hm} - 0.05L$ in the end of the omega bend (Figure 3.2c).

If the first frame, the last frame and all frames in the middle of a sequence satisfy these criteria of start, end and middle of an omega bend, we declare an omega bend.

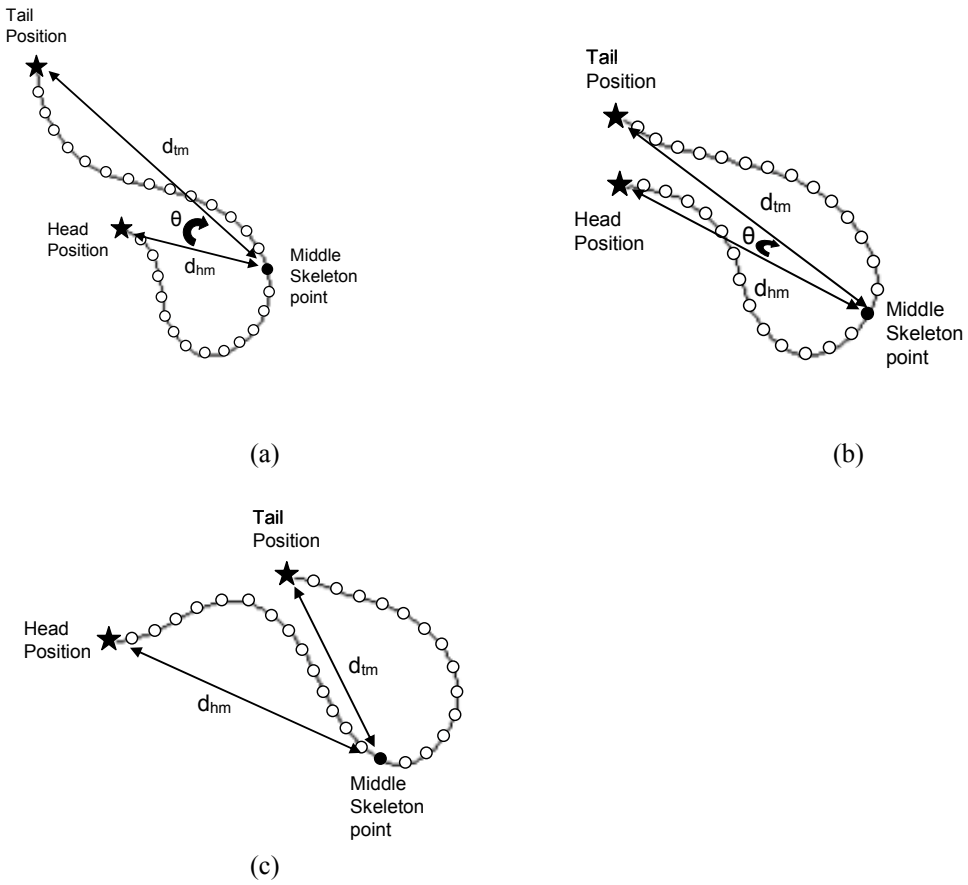


Figure 3.2: (a) Start frame of an omega bend. Segment d_{tm} must be greater than d_{hm} plus 5% of body length and θ must be less than 45°. (b) A frame during an omega bend. θ must be less than 45°. (c) End frame of an omega bend. Segment d_{hm} must be greater than d_{tm} plus 5% of body length.

3.2 Reversal detection

After a morphological skeleton is obtained, 30 evenly spaced skeleton points are extracted. The two end points on the skeleton represent the head and tail positions. Two reference points (R_h, R_t) are located at 20% of the body length from the head and tail end points (the sixth skeleton point) (Figure 3.3a). These mark where the tail segment and head segment begin.

Any frame without body looping (that is, without an internal hole) will be considered a reversal frame if the following criteria are satisfied:

1) The distance between the head position four frames earlier (t = n - 4) and the current reference point R_h (t = n) has to be greater than the distance between the current head position (t = n) and the current reference point R_h (t = n). When this criterion is satisfied, it means that the worm's head is moving toward its previous body position.
2) The distance between the current tail position (t = n) and the previous reference point R_t (t = n - 4) has to be greater than the distance between the previous tail position (t = n - 4) and the previous reference point R_t (t = n - 4) by at least 2.0% of the body length (Figure 3.3b).

When the second criterion is satisfied, it means that the worm's tail is moving away from its previous body position. It will exclude those frames in which the worm only waves its tail instead of reversing. The "2.0% of the body length" in the second criterion makes the reversal length showing on the computer long enough (≥ 1mm) to be verified by human observation. The reason we chose to compare the frames at n and n – 4 (instead of n – 1) is to make our result less sensitive to other movements besides reversals such as tiny tail waving or head foraging movements.

After the reversal frames are found, we combine them to locate every reversal sequence happening during the video. The reversal distance and length are obtained by calculating the

moving distance of the centroid and the time interval between the start frame and the end frame of a reversal sequence.

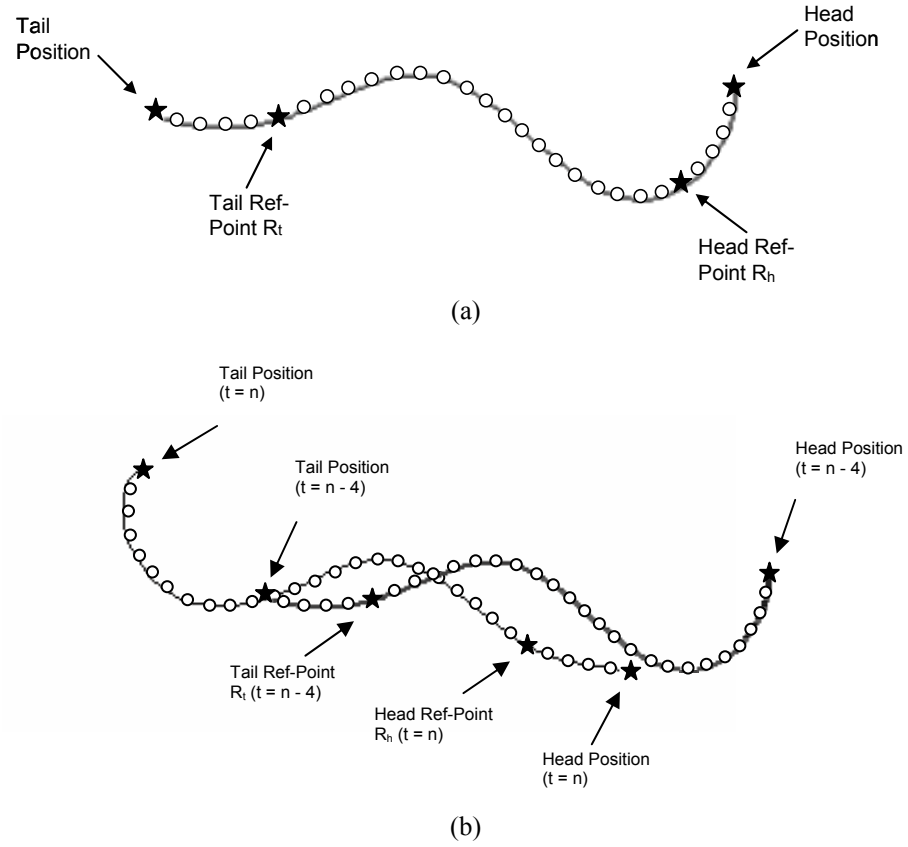


Figure 3.3: (a) Skeleton with 30 sampled skeleton points. Two reference points are defined as the sixth skeleton points from two end points (head position and tail position). (b) Reversal detection method. The frame at $t = n$ is compared to the earlier frame at $t = n - 4$.

3.3 Results

3.3.1 Verification of omega bend detection by human observers

Our algorithm for the detection of omega bends was tested on 100 5-minute videos (8Hz) in which 303 omega bends were detected by a human observer. The experimental results showed that our algorithm correctly detected over 93% of these 303 omega bends. Also, over 95% of detected events are actually omega bends, so the false positive rate is low even while the true positive rate is high (Table 3.1).

Table 3.1: Verification results for the omega bend detection algorithm. Data were collected from 100 5-minute videos (8Hz) from 5 mutant types. The first column shows the mutant type. The second column shows both the number of correctly detected omega bends with our algorithm and the associated ratio of correct detections to total detections. The number of wrong detections is listed in column 3. The number of missed omega bends is listed in column 4.

Strain name	# of detected omega bends	# of wrong detections	# of omega bends missed
Wild type (N2)	57 (100%)	0	7
syd-1(ju82)	43 (100%)	0	0
unc-10(e102)	41 (97.6%)	1	3
unc-37(e262)	83 (91.2%)	8	0
unc-75(e950)	60 (92.3%)	5	9
Total	284 (95.3%)	14	19

3.3.2 Verification of reversal detection algorithm by human observers

Our reversal detection algorithm was tested on the same data set of 100 5-minute videos (8Hz) in which 1621 reversal events were detected by a human observer. The experimental results showed that our algorithm can correctly detect reversals at a high rate (96.9%) that compared favorably with that of the previously-described [6] skeleton-based algorithm (86.3%). This higher correct detection rate was furthermore accompanied by a lower rate of false alarms (only 10 wrong detections compared to 14 for the previous algorithm). We applied the reversal detection algorithm described in [6] to our data set of 100 5-minute videos with 1621 reversal events so the results are directly comparable (Table 3.2).

Table 3.2: Verification results for the reversal detection algorithm. Data were collected from 100 5-minute videos (8Hz) from 5 mutant types. The first column shows the mutant type. The first column shows the mutant type. The second column shows both the number of correctly detected reversals with our algorithm and the associated positive predictive value (PPV). The PPV is the ratio of correct detections to total detections. The number of wrong detections is listed in column 3. The number of missed reversals is listed in column 4.

Strain name	Our algorithm			Algorithm from [6]		
	# of reversals detected	# of wrong detections	# of reversals missed	# of reversals detected	# of wrong detections	# of reversals missed
Wild type (N2)	494 (100%)	0	0	473 (100%)	0	21
syd-1(ju82)	342 (99.4%)	2	12	334 (99.4%)	2	20
unc-10(e102)	262 (99.6%)	1	3	252 (97.7%)	6	13
unc-37(e262)	105 (96.3%)	4	13	87 (94.6%)	5	31
unc-75(e950)	368 (99.2%)	3	22	303 (99.7%)	1	87
Total	1571 (99.4%)	10	50	1399 (99.0%)	14	172

3.3.3 Time correlation of omega bends and reversals

In another experiment, we used our system to compare omega bend and reversal behavior of mutant and wild type animals. In particular, we investigated the temporal correlation between these body movements, since omega bends have recently been observed to frequently follow reversals [30]. We focused on those mutants in our data set, *syd-1*, *unc-10*, *unc-37*, *unc-75* and *unc-77*, which are described as "fairly active" in the literature. Our reversal and omega bend detection algorithms were used on all video data of these 5 mutants and wild type. For each omega bend event in each video, we searched for the closest reversal happening within 5 seconds before the omega bend event. If no reversal occurred within 5 seconds, the omega bend event was considered to have not followed a reversal. We also calculated the average time interval between pairs of reversals and omega bends in each mutant type, as well as whether the omega bend involved contraction of the dorsal or ventral body muscles. The results are shown in Table 3.3.

Table 3.3: Relationship between omega bends and reversals. Average time intervals between pairs of reversals and omega bends are listed in brackets.

	Wild type (N2)	*syd-1 (ju82)*	*unc-10 (e102)*	*unc-37 (e262)*	*unc-75 (e950)*	*unc-77 (e625)*
# of videos watched	56	81	40	68	40	40
# of omega bends following reversal (total)	75 (0.43sec)	111 (1.01sec)	74 (1.26sec)	45 (1.83sec)	37 (2.88sec)	71 (0.75sec)
# of omega bends following reversal (ventral)	75 (0.43sec)	98 (0.86sec)	58 (1.10sec)	22 (1.90sec)	16 (3.05sec)	47 (0.67sec)
# of omega bends following reversal (dorsal)	0	13 (2.16sec)	16 (1.84sec)	23 (1.76sec)	21 (2.74sec)	24 (0.92sec)
# of omega bends NOT following reversal (total)	3	9	20	147	58	48
# of omega bends NOT following reversal (ventral)	2	5	11	103	16	28
# of omega bends NOT following reversal (dorsal)	1	4	9	44	42	20

We observed that in wild-type as well as *syd-1* mutant animals, omega bends showed a distinct ventral bias. In fact, for omega bends that immediately followed a reversal, nearly all involved a bend on the ventral side of the body. *unc-10* and *unc-77* mutants also have a statistically significant ventral bias (p-value 0.0000057 and 0.0045 for *unc-10* and *unc-77* respectively using a Chi-Square test) [31]. However, the ventral bias is less strong for these mutants than for wild type (p-value ≈ 0). However, for *unc-37* and *unc-75* mutants, dorsal bends were as common as ventral bends, indicating that the ventral bias was lost in these animals. Furthermore, these mutants also showed a much higher frequency of omega bends that were uncorrelated with reversals, and the average time interval between omega bends and reversals was significantly increased. Thus, the *unc-37* and *unc-75* mutants appeared to have abnormalities in both the ventral bias of omega bends as well as their temporal correlation with reversals. Since we looked at single alleles of these mutants, it is not possible to conclude definitively that the

ventral bias phenotype is due to the *unc-37/unc-75* mutation rather than another mutation in the genetic background. However, these results demonstrate that it is possible to detect such phenotypes (which would be extremely tedious and time-consuming for a human observer) using the automated system.

We also studied the variations in angle change rate before and after all reversals detected by our algorithm. The angle change rate R and its standard deviation can be obtained from the thirty skeleton points of the worm [4]. In Chapter 2, it was defined as the ratio of the average angle difference between every pair of consecutive segments along the skeleton to the worm length. Omega bends are tightly related to high angle change rate, because the worm body is very curved while undergoing an omega bend. We examined the angle change rate from frames within 10 seconds before and 20 seconds after every reversal event. The results are shown in Figure 3.4. Blue curves represent mean angle change rate averaged over randomly chosen non-reversal moments and red curves show angle change rate before and after reversal moments. For wild type (Figure 3.4a), which has a strong reversal-omega bend relationship, the angle change rate increases significantly after reversals. This increasing of angle change rate due to omega bends is also much higher than when the worm is moving forward. Among the five coilers, only *syd-1* and *unc-10* have behavior similar to wild type (Figure 3.4b and c). In Figure 3.4d, e, and f, we can see that even though the angle change rate after reversals is still higher than before reversals, it is not overwhelming compared to moments when the worm is moving forward, as it is for wild type, *syd-1* and *unc-10*. We also notice that in Figure 3.4f, the red curve (angle change rate before and after reversal moments) is more similar to *syd-1* and *unc-10*, which matches the result in Table 3.3 that more than 60% of omega bends follow reversals for *unc-77*. However, its blue curve in Figure 3.4f (angle change rate averaged over randomly chosen non-reversal moments) is closer to the blue curve in Figure 3.4e. This is also shown in the cluster plot where *unc-75* and *unc-77* have similar behavior patterns (the cloud of *unc-77* is close to the cloud of *unc-75*).

3.4 Conclusion

In this chapter, we developed and tested an algorithm for automatic detection of omega bends and reversals (correctly detected 93% of the omega bend events and correctly found 96.9% of reversal events). The relationships between omega bends and reversals for 6 strains (wild type, *syd-1*, *unc-10*, *unc-37*, *unc-75* and *unc-77*) were also examined and compared to each other. Using our automated omega bend and reversal detection algorithms, we found that the ventral bias of omega bends normally observed in wild-type was largely absent in some mutants such as *unc-37* and *unc-75*. Likewise, the temporal correlation between omega bends and reversals was defective for these two strains. Wild type, *syd-1*, *unc-10* and *unc-77*, whose centroids of their clouds are close to each other in the cluster plot in Chapter 2, have very similar omega bend and reversal behaviors. The ventral bias is very strong particularly in wild type and *syd-1*, and most detected omega bends were tightly coupled temporally to reversals for these 4 strains. *Unc-77*, although it has omega bends and reversal behavior similar to the other three strains (wild type, *syd-1* and *unc-10*) is very close to *unc-75* in the cluster plot. Complex behavioral features such these have been shown in [30] to reveal important aspects of sensory perception and motor control in the *C. elegans* nervous system; however, manual scoring of these features is tedious and labor-intensive. The development of automated methods for the study of complex behavioral patterns and the identification of mutants with abnormalities in them promises significantly to enhance the understanding of behavioral mechanisms in this important neurobiological model.

Part of this chapter is a reprint of the material as it appears in Kuang-Man Huang, Pamela Cosman, and William Schafer, "Machine Vision Based Detection of Omega Bends and Reversals in *C. elegans,*" Journal of Neuroscience Methods, Vol. 158, Issue 2, pp. 323-336, December 2006. I was the primary researcher and the co-authors Dr. Pamela Cosman and Dr. William Schafer directed and supervised the research which forms the basis for this chapter. This work was supported by a research grant from the National Institutes of Health (R01 DA018341).

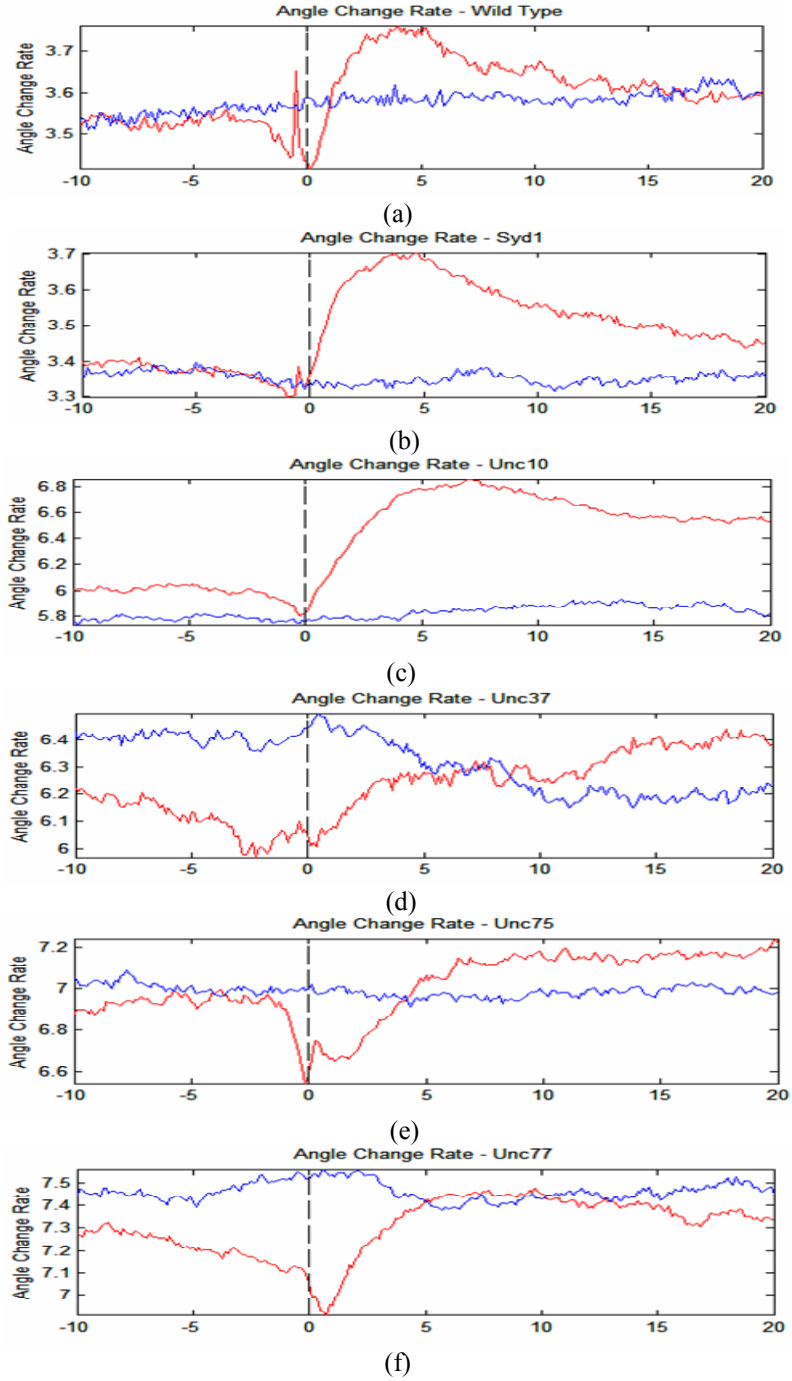


Figure 3.4: The angle change rate before and after reversals, compared with angle change rate for non-reversal moments: (a) wild type (n2) (b) *syd-1*(*ju82*) (c) *unc-10*(*e102*) (d) *unc-37*(*e262*) (e) *unc-75*(*e950*) (f) *unc-77*(*e625*).

Chapter 4

Detection and analysis of foraging behavior

Despite its anatomically simple nervous system, *C. elegans* is capable of surprisingly diverse patterns of behaviors. While some of these, such as feeding, egg-laying, and defecation, are mechanically simple [32, 33], other behaviors involve complex motor programs involving intricate coordination of muscle groups. These include locomotor behaviors such as backward and forward crawling, swimming, and copulation [34]. Recently, there has been increasing interest in quantitatively characterizing and modeling these more complex motor programs [35, 36]; however, significant questions remain regarding the nature of these behaviors and how they are generated by the nervous system. Among these behaviors, one that has received comparatively little attention is foraging. Foraging is a rapid, side-to-side movement of the nose generated by *C. elegans* as it explores its environment. In [37], a foraging movement is defined as a complete cycle of movements by the tip of the nose from the ventral side through the dorsal

extreme or vice versa during time intervals when the animal was moving forward. In previous studies, foraging events were scored by human observers which is tedious and labor-intensive.

In this chapter we provide the first quantitative description of foraging movements in *C. elegans*. Using video data collected using an automated tracking system, we have been able to reliably detect foraging events and measure the depth and frequency of the nose bends. We also measure foraging-related parameters which have not previously been studied and use Fourier analysis of these data to identify characteristic frequencies that can be used to parameterize foraging patterns. These analyses provide more precise methods for defining the effects of specific genes and neurons on *C. elegans* behavior. This chapter is organized as follows. We first give a detailed description of the foraging detection algorithm, including image acquisition and pre-processing. We then evaluate the algorithm by testing it on a variety of videos of mutant worms, and verifying the results with manual observations. We also describe how to extract foraging-related parameters, and finally we combine these parameters with Fourier analysis to analyze foraging behavior.

4.1 Locating the worm nose

To facilitate analysis, the grayscale images were subjected to preliminary image processing to generate a simplified representation of the body [7]. First any images which were snapped when the stage was moving (the current coordinate of the stage was different from the previous coordinate) will be discarded because these images were usually blurry. Then for each good image frame, an adaptive local thresholding algorithm followed by a morphological closing operator (binary dilation followed by erosion) was used. As described in [7], a corresponding reference binary image was also generated by filling holes inside the worm body based on image content information. The difference between these two binary images provided a good indication

of which image areas are worm body and which are background. Following binarization, a morphological skeleton was obtained [7, 24].

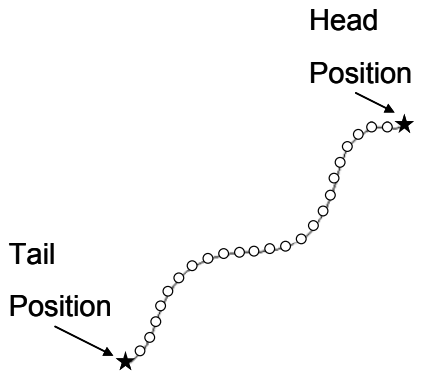


Figure 4.1: Skeleton with 25 sampled skeleton points.

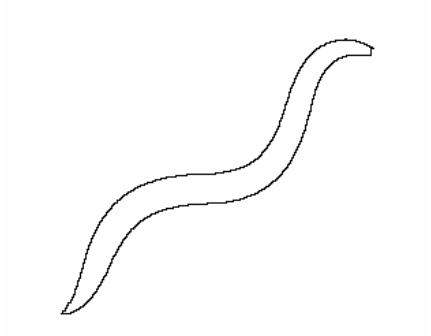


Figure 4.2: The exterior contour of the worm body.

After a morphological skeleton is obtained, 25 evenly spaced skeleton points are extracted. The two end points on the skeleton represent the head and tail positions (Figure 4.1). Using the approach in [7], the head is recognized for entire video sequences using the brightness (the head is usually brighter than the tail) and the distance moved between the current frame and the previous frame (the head usually moves more than the tail) for the two end points. However, the head position on the skeleton as calculated in [7] is not precisely the same as the nose position recognized in the original grayscale image. To detect foraging events which are related to subtle nose movement, we need to locate the nose position more precisely. First we obtain the exterior boundary of the worm body by eroding it with a 3×3 square structuring element and then

performing the set difference between the binary image and its erosion (Figure 4.2). A cutoff line is then placed which passes through the first skeleton point p_1 and is perpendicular to the skeleton tangent line at p_1. The cutoff line cuts the exterior contour into two parts (Figure 4.3a). The smaller part which contains the head point is the nose section of the contour and will be isolated from the rest of the body (Figure 4.3b). We compute the distances between the point p_1 and each pixel on the nose section of the contour. The 10 points having the longest distances from the point p_1 are selected and used to compute the spatial average point p_n, which we define to be the position of the nose (Figure 4.3c).

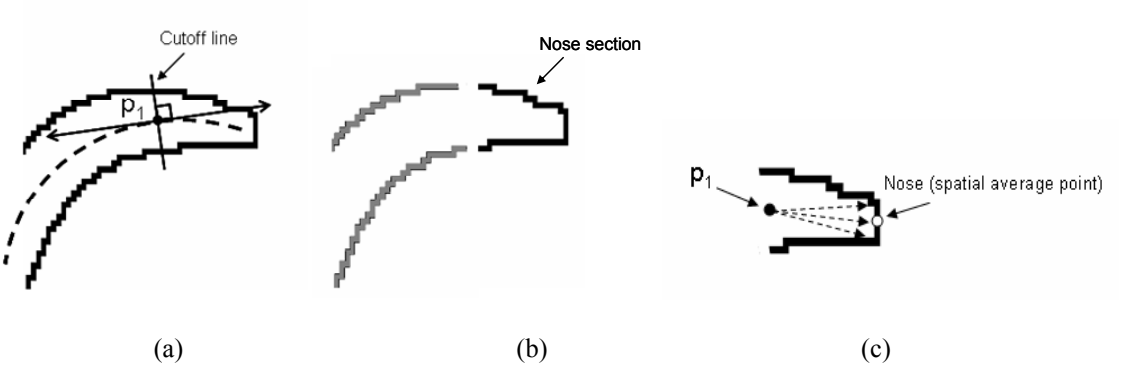


Figure 4.3: (a) Placing a cutoff line at the first skeleton point p_1 perpendicular to the tangent line at p_1. (b) Isolating the nose section from the rest of the body. (c) Computing the spatial average of the 10 points furthest from p_1.

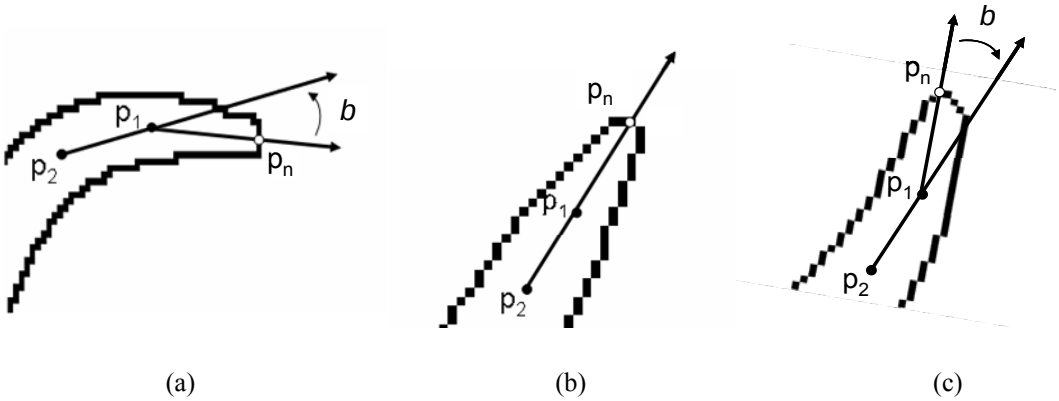


Figure 4.4: Computing the nose bending angle b in every frame. (a) Nose bends to the right of the midline. (b) Nose points straight ahead. (c) Nose bends to the left of the midline.

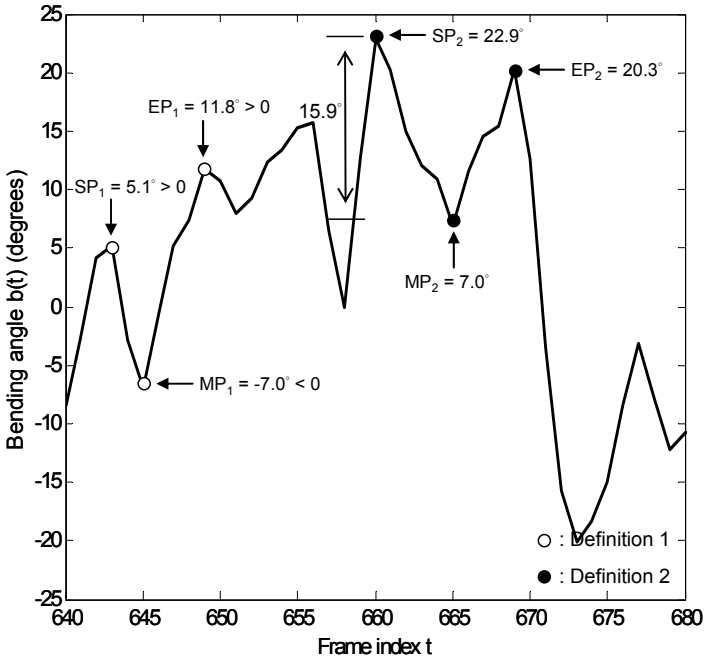
4.2 Foraging event detection

In [37], a foraging movement is defined as a complete cycle of movements by the tip of the nose from the ventral side through the dorsal extreme or vice versa during time intervals when the animal was moving forward (Figure 4.4). Based on the definition of foraging in [37], any detected foraging events during reversals or omega bends will not be counted as foraging events. We used the method described in Chapter 3 to detect reversals and omega bends automatically.

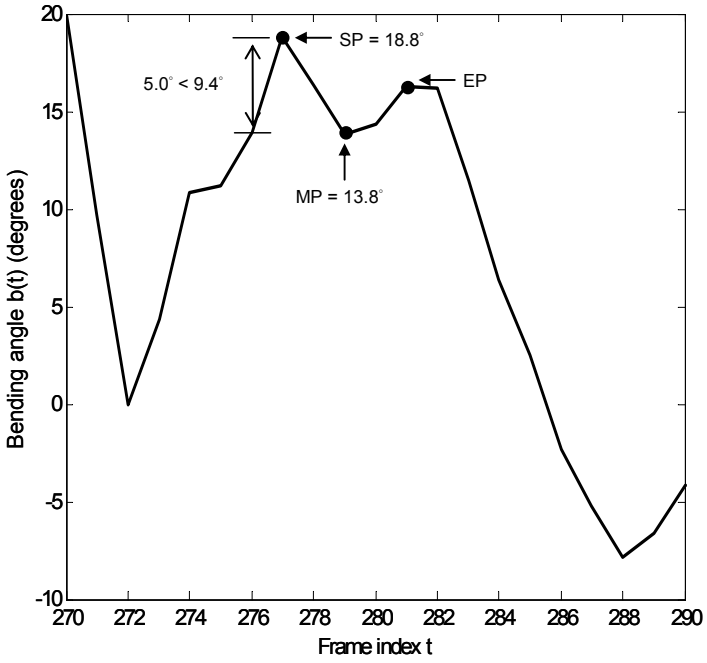
To investigate this kind of side to side nose movement in a video, we measure the angle b between the segments (p_n,p_1) and (p_1,p_2) where p_1, p_2 are the first and second skeleton points from the head and p_n is the nose position (Figure 4.4). Figure 4.5 shows plots of nose bending angle *b(n)* over time from a video. Here *n* is the frame index. The overall curve is in a rough sinusoidal shape because the worm generally moves in a sinusoidal wave. We can also notice that there are some dips and peaks (extrema) marked on the curve in Figure 4.5a. Each set of three consecutive extrema (consisting either of two local maxima and the local minimum between them, or else of two local minima and the local maximum between them) represents a side-to-side motion of the nose and is therefore considered to be a candidate foraging event. From the beginning of each video, we search for and examine each set of three consecutive extrema. We denote the first, second, and third local extreme values (in a time-wise order) to be its start point SP, middle point MP, and end point EP respectively. A set of three consecutive extrema is considered to be a foraging event if that segment of the video does not contain any bad frames (frames discarded due to stage movement) and if either of the following two criteria is satisfied:

1) sign[SP] = sign[EP] = -sign[MP]: when this criterion is satisfied, it means that the worm's nose is waving across the middle line (b = 0) to reach the other side then waving back to accomplish a complete foraging movement. An example of a foraging event of this kind is shown by white dots (SP_1, MP_1 and EP_1) in Figure 4.5a.

2) sign[SP] = sign[EP] = sign[MP] and abs[SP-MP] > α abs[SP]: when this criterion is satisfied, it means that even though the worm's nose does not cross the middle line, the moving angle of the nose is still larger than some fraction α of its starting bending angle which, depending on α, is noticeable enough to be considered to be a foraging event. An example of a foraging event of this kind (e.g. α = 0.5) is shown by black dots (SP_2, MP_2 and EP_2) in Figure 4.5a. The angle difference between SP_2 and MP_2 is 15.9° which is larger than 0.5 x SP_2 =11.45°. Figure 4.5b shows an event that does not satisfy either of the above two criteria (when α = 0.5).



(a)



(b)

Figure 4.5: (a) An example of nose bending angle over time from a video. A detected foraging event of definition 1 is shown in white dots and the other event of definition 2 (α = 0.5) is shown in black dots. (b) An example of a non-foraging event.

Once a set of three consecutive extrema is decided to be a foraging event, the search for additional foraging events will start from the end point EP of the previous event to avoid having foraging events overlapping.

4.3 Results

In this section, first we present verification results assessing the robustness of our foraging detection algorithm and decide the best value of α for our algorithm. Then we analyze foraging behavior in the following steps: 1) We start by investigating the previously obtained nose bending signals by estimating their power spectra. 2) Some parameters such as waving amplitude and frequency (which will be described in detail later) are extracted from detected foraging events. 3) Then we combine the obtained parameters with power spectrum analysis and use t-tests to study the similarities of foraging behavior between wild type and each of the four mutant types. The experiments were implemented in Matlab on a 2.33 GHz Pentium-IV desktop computer.

4.3.1 Verification of the foraging detection algorithm by human observers

Our algorithm for the detection of foraging events was tested on 25 (5 videos for each strain) 1-minute videos (30Hz). First, a trained human observer examined all the videos to locate all the foraging events. By applying the above algorithm with α varying from 0 to 1, the performance result is shown as a receiver operating characteristic (ROC) curve [38] in Figure 4.6 and Table 4.1. There is a sharp bend in the ROC curve when α = 0.5, (at which point the True Positive fraction is over 90% while the False Positive fraction is less than 10%), indicating α = 0.5 is the best for this algorithm. For α = 0.5, the foraging event detection results for individual strains are given in Table 4.2.

Table 4.1: The True Positive, True Negative, False Positive, and False Negative values for the ROC curve. The highlighted row is the final α used in the foraging detection.

Rate of non-foraging events detected as foraging (False Positive)	Rate of foraging events detected as foraging (True Positive)	Rate of foraging events detected as non-foraging (False Negative)	Rate of non-foraging events detected as non-foraging (True Negative)	α
1.0000	0.9637	0.0363	0	0
0.7116	0.9609	0.0391	0.2884	0.1
0.4890	0.9554	0.0446	0.5110	0.2
0.3176	0.9491	0.0509	0.6824	0.3
0.1819	0.9442	0.0558	0.8181	0.4
0.0597	**0.9351**	**0.0649**	**0.9403**	**0.5**
0.0435	0.7901	0.2099	0.9565	0.6
0.0287	0.6820	0.3180	0.9713	0.7
0.0205	0.5941	0.4059	0.9795	0.8
0.0153	0.4986	0.5014	0.9847	0.9
0.0119	0.4010	0.5990	0.9881	1.0

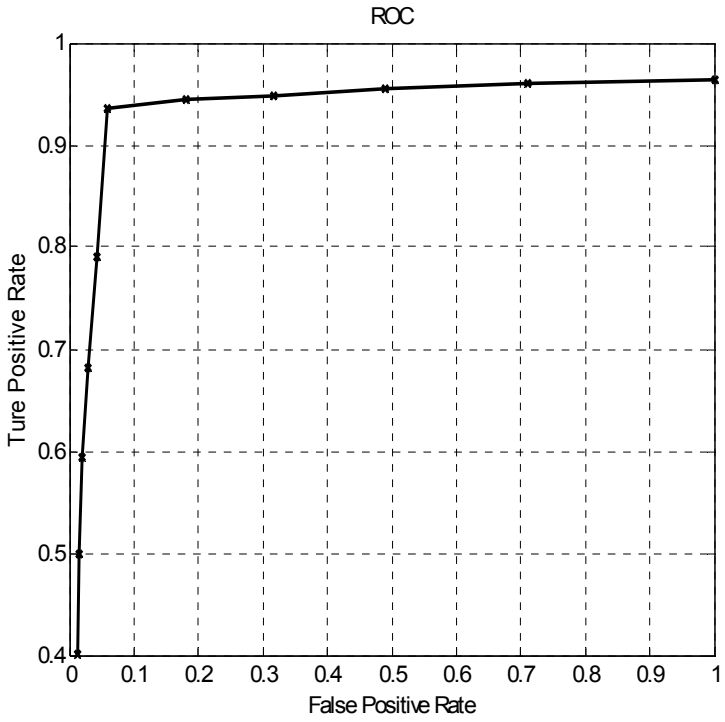


Figure 4.6: A plot of the receiver operating characteristic (ROC) curve with α varying from 0 to 1.

Table 4.2: Verification results of each strain for the foraging detection algorithm (α = 0.5). The first row shows the mutant type. The second row shows the number of frames where the worms were moving forward. The number of detected events is listed in row 3. The number of foraging events not detected is listed in row 4. The number of detected events which are real foraging events is listed in row 5.

Mutant type	dgk1(nu62)	glr1(n2361)	goa1(n1143)	trpa1(ok999)	wild type
Total frames	5513	8105	5387	6709	6273
Detected foraging	243	354	302	268	299
Foraging Not detected	8.8%(21)	7.9%(27)	3.7%(11)	4.0%(10)	7.9%(24)
Real Foraging %	90.2%(219)	89.0%(315)	94.0%(284)	90.3%(242)	94.0%(281)

4.3.2 Fourier analysis of foraging events

In another experiment, we used spectral analysis to investigate the foraging behavior of each strain. In particular, we estimated the power spectrum for each strain from 150 videos (30 videos for each strain) by averaging multiple periodograms (Welch-Bartlett method, [39]). The periodogram is an estimator of the power spectrum, introduced by Schuster [40]. To calculate the periodogram for each strain, first we subdivided each bending angle signal $b_p(n)$ ($1 \le p \le 30$ videos for each strain) into small segments in a sliding window and overlapping fashion. The window used in this experiment is a Hamming window of duration L = 30 with values $w(m)$ = $0.54 - 0.46\cos(2\pi m / 29)$, for $0 \le m \le 29$. We set D = 20 ($D < L$) to be an offset distance to make the segments overlap and create more segments, and let K_p be the number of segments in the pth video, which changed with different videos (for example, a video glr-1_001 with 1745 frames was divided into $\frac{1745}{D} \approx 87$ segments). Then the qth segment of the pth video consists of the following L values:

$$b_{pq}(m) = b_p(qD + m)w(m) \qquad 0 \le m \le L-1 \quad , \quad 0 \le q \le K_p - 1$$

The estimated power spectrum (periodogram) $\hat{B}_{pq}(e^{j\omega})$ of the qth segment of the pth video can be calculated using the discrete Fourier transform as follows:

$$\hat{B}_{pq}(e^{j\omega}) \equiv \frac{1}{L}\left|B_{pq}(e^{j\omega})\right|^2 = \frac{1}{L}\left|\sum_{m=0}^{L-1} b_{pq}(m)e^{-j\omega m}\right|^2$$

The spectrum estimate of each strain was obtained by averaging the periodograms from all of its $\sum_{p=1}^{30} K_p$ curves as follows:

$$\hat{B}(e^{j\omega}) = \frac{1}{\sum_{p=1}^{30} K_p}\sum_{p=1}^{30}\sum_{q=1}^{K_p}\hat{B}_{pq}(e^{j\omega}) = \frac{1}{\sum_{p=1}^{30} K_p}\sum_{p=1}^{30}\sum_{q=1}^{K_p}\frac{1}{L}\left|B_{pq}(e^{j\omega})\right|^2 = \frac{1}{\sum_{p=1}^{30} K_p}\sum_{p=1}^{30}\sum_{q=1}^{K_p}\frac{1}{L}\left|\sum_{m=0}^{L-1} b_{pq}(m)e^{-j\omega m}\right|^2$$

We also obtained a different averaged spectrum only from segments where foraging events happened. In this case, after foraging events were detected, a Hamming window was placed at each foraging event (the center of the window points to the center of the foraging event) and the periodogram of each event was computed. For each strain, the foraging spectrum is averaged from all *N* periodograms where *N* is the number of detected foraging events for the strain. The results of this Fourier analysis are in Figures 4.7 and 4.8. Figure 4.7 includes spectra of complete videos and Figure 4.8 shows spectra from foraging events only. In Figure 4.7, we can see that there is a main frequency for each strain which corresponds to the frequency of the approximately sinusoidal worm body wave. These main components also exist in Figure 4.8. But we can also find subtle bumps on the outer sides of the main component which are caused by the foraging. To observe these bumps more clearly, we take the ratio of the spectra in Figure 4.8 to those in Figure 4.7. The results are shown in solid curves in Figure 4.9. We also computed the ratio of the spectrum generated from randomly chosen segments to the spectrum generated from the overall video and the results are shown in long-dashed curves in Figure 4.9. We can see that the curves from randomly chosen segments are close to 1 and do not have prominent peaks.

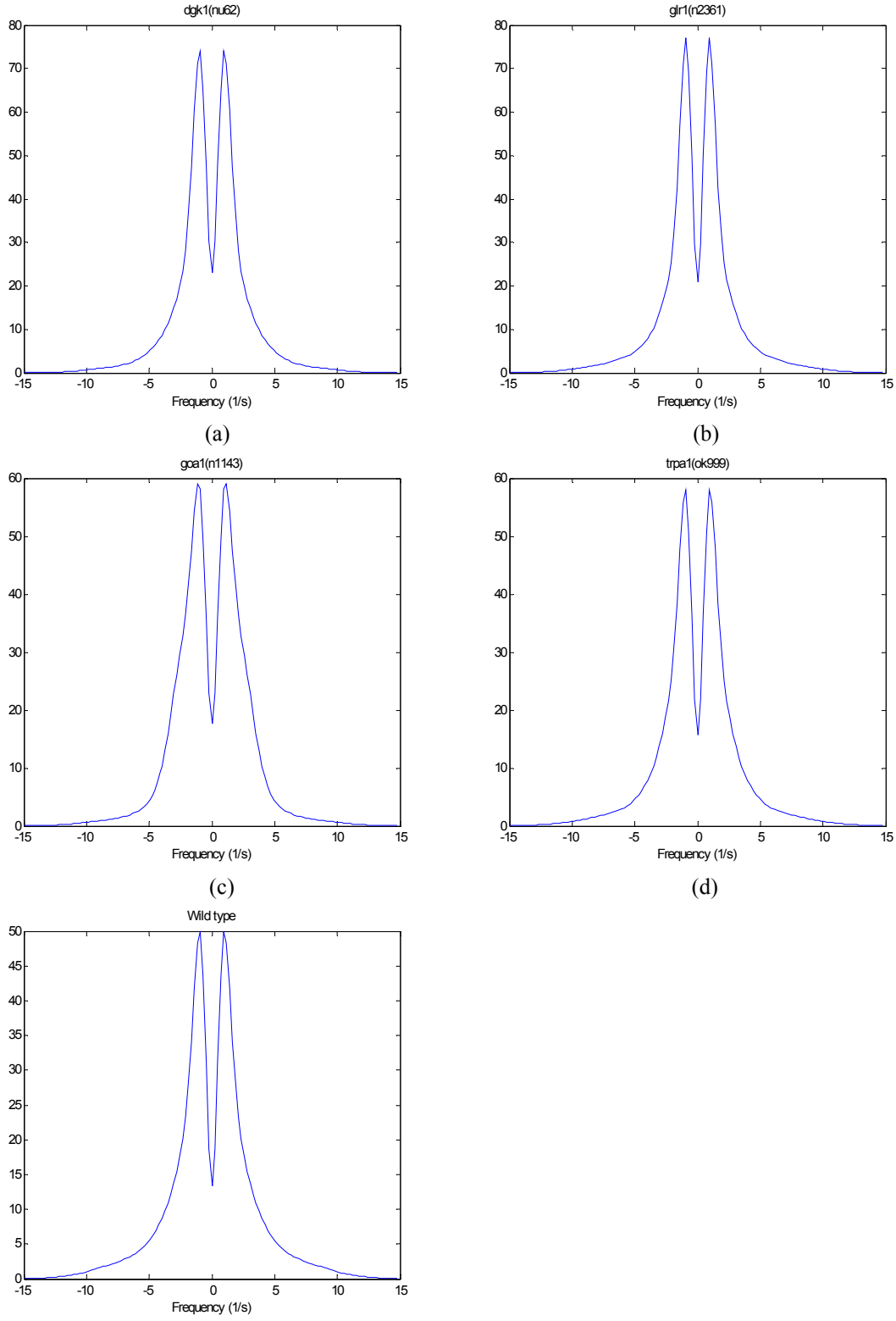


Figure 4.7: The estimated overall power spectra for all strains: (a) *dgk-1*, (b) *glr-1*, (c) *goa-1*, (d) *trpa-1* and (e) wild type.

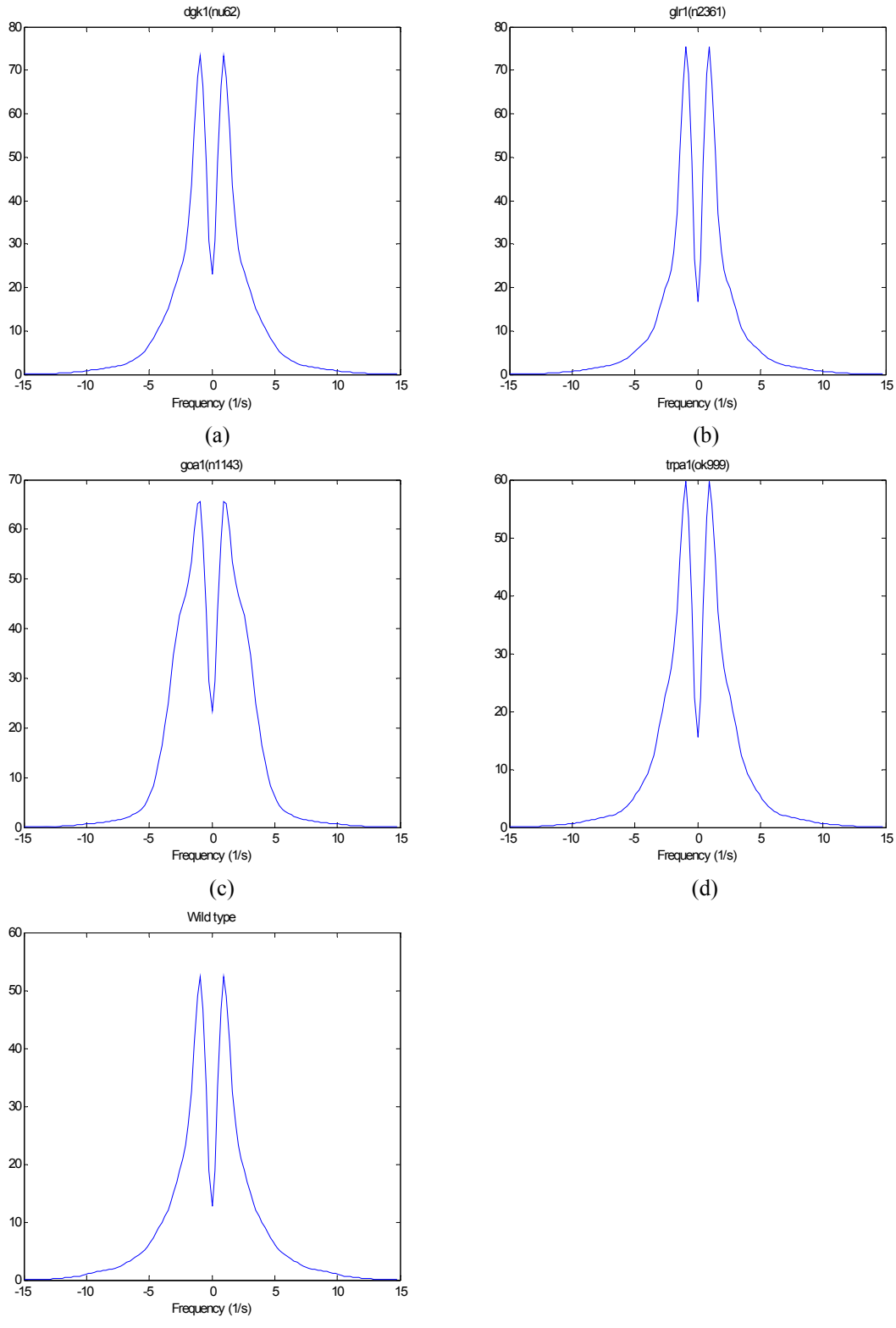


Figure 4.8: The estimated power spectra from detected foraging events for all strains: (a) *dgk-1*, (b) *glr-1*, (c) *goa-1*, (d) *trpa-1* and (e) wild type.

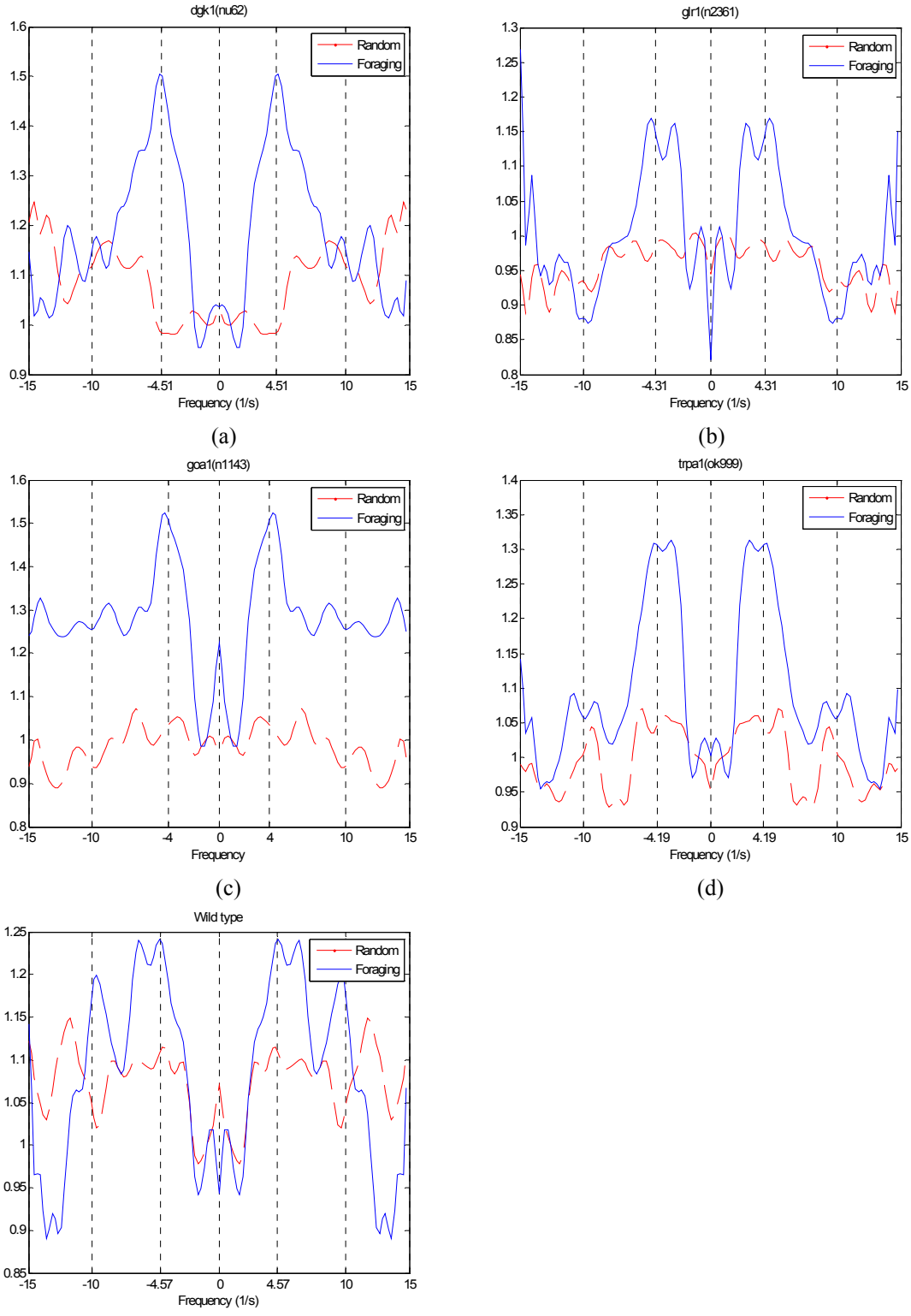


Figure 4.9: The ratio of Figure 4.8 to Figure 4.7 for all mutant types (solid line) and the ratio of the spectrum generated from randomly chosen segments to the spectrum generated from the overall video (long-dashed line).

4.3.3 Statistical analysis of foraging events

After foraging events were detected for each strain, we extracted several basic features from all detected events from bending curves (as shown in Figure 4.5). These include: waving amplitude, initial waving direction, time interval between adjacent foraging events, and the frequency of nose waving during an individual foraging event. These features are described in detail as follows:

1) Wave amplitude: it is defined to be the depth of nose bending (in degrees) during a foraging event. Once a foraging event is found, we compute {abs[SP-MP]+abs[EP-MP]}/2 to be its amplitude.

2) Initial waving direction: is the side from which the worm starts a foraging event (SP > 0 is defined to be left and SP < 0 is defined to be right).

3) Time interval between adjacent foraging events: for adjacent events, this parameter is defined to be the time interval between the end point EP of the first event and the start point SP of the second event.

4) Frequency of individual foraging event: this parameter represents the inverse of the period of an individual foraging event, which can be expressed as 1/T where T is the time length between the start point SP and the end point EP of the event.

The mean values and standard deviation values of each feature were computed for each strain and the results are listed in Table 4.3. The first row shows the mutant type. The second row shows the number of foraging events detected by the algorithm. The third and fourth rows show the number of foraging events which start respectively from the left and right side of the body. The mean values and standard deviation values of features are listed in rows 5 and 6 (waving amplitude), rows 7 and 8 (time interval between adjacent events) and rows 9 and 10 (the frequency of nose waving during foraging events).

We compared the results in Table 4.3 to the previously obtained power spectrum ratios for each strain. Among these 5 mutant types, *goa-1* has the largest average amplitude and its curve in Figure 4.9 generally has higher values than the other mutant types. The average foraging frequencies in Table 4.3 computed from all detected events for *dgk-1* and *goa-l* are also very close to their peak values in Figures 4.9a and 4.9c. *glr-1*, *trpa-1* and wild type have two nearly equal maximal peak values in Figures 4.9b, 4.9d and 4.9e, but their average foraging frequencies still locate within the main lobes of the spectra in Figure 4.9. Their standard deviation values of foraging frequency are also larger than those for *dgk-1* and *goa-1*.

Table 4.3: Foraging related features extracted from all events detected by the algorithm. The first row shows the mutant type. The second row shows the total number of foraging events for each strain detected by the algorithm. The third and fourth rows show the number of foraging events which start respectively from the left and right side of the body. The mean values and standard deviation values of features are listed in rows 5 and 6 (waving amplitude), rows 7 and 8 (time interval between adjacent events) and rows 9 and 10 (the frequency of nose waving during foraging events).

Mutant type	*dgk-1(nu62)*	*glr-1(n2361)*	*goa-1(n1143)*	*trpa-1(ok999)*	wild type
Total foragings	1669	1604	1820	1480	1789
Left	813	841	911	735	927
Right	856	763	909	745	862
Amplitude AVE	15.13	14.91	19.26	15.71	14.15
Amplitude STD	9.30	8.27	11.14	9.00	8.34
Interval AVE	0.199	0.386	0.171	0.249	0.216
Interval STD	0.245	0.465	0.206	0.325	0.293
Frequency AVE	4.51	4.31	4.00	4.19	4.57
Frequency STD	1.58	1.64	1.57	1.66	1.61

In order to study the similarities of foraging behavior between mutants, we generate scatter plots of each parameter (amplitude, interval and frequency as in Table 4.3) for wild type and each of the four mutants. The mean values of three parameters of each video are computed. The scatter plot of each parameter is generated by using each video as a data point (Figure 4.10). From these plots, we make three observations: *1)* for mutant types *dgk-1* and *goa-1,* which are

hyperactive for locomotion and foraging, their time intervals between adjacent events are almost all within a range from 0.1 to 0.3 seconds. Mutant types *glr-1* and *trpa-1* which forage more slowly have much wider distribution with many data points having larger values than other mutant types. *2) goa-1* has generally higher values in the plot of amplitude, while most of the data points of other mutant types are within a range of 10~20 degrees. *3) dgk-1, glr-1* and wild type have similar ranges of distribution for frequency of individual foraging event (4~5 1/second) and *goa-1* has lower distribution in the plot which matches the result in Table 4.3.

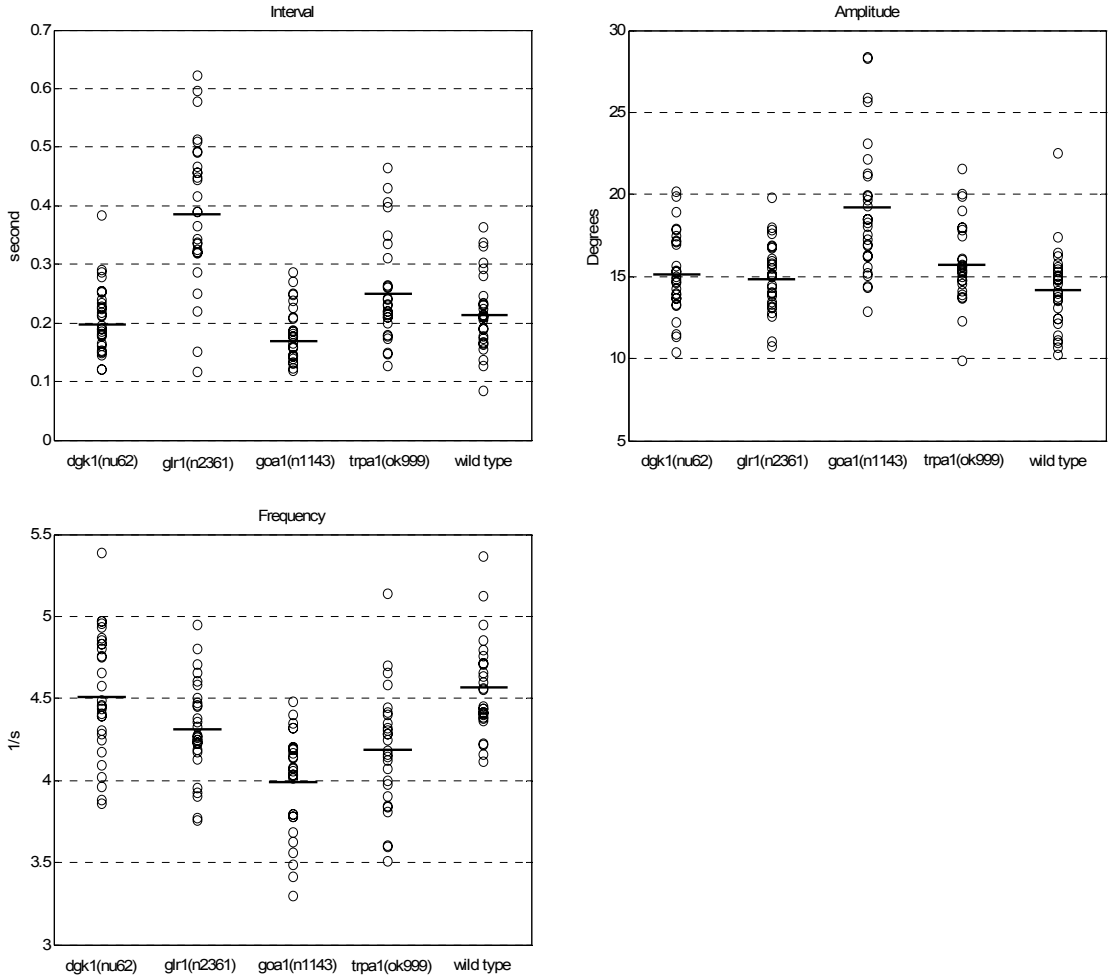


Figure 4.10: The scatter plot of three parameters for all strains: (a) time interval between adjacent events, (b) amplitude and (c) frequency of individual foraging event. The solid lines show the overall mean values.

We also computed the foraging rate, which is defined in [37] to be the number of foraging events within 10 seconds, for each strain and the results are in listed in Table 4.4. The first row lists the mutant type and the second row lists the total number of frames in which the worms were moving forward in the 30 videos of that strain. The third row lists the number of detected events declared by the algorithm to be foraging events. The fourth row shows the number of foraging events in 10 second (300 frames). In Table 4.4, we can see that mutant types *dgk-1* and *goa-1* which are hyperactive for locomotion and foraging have higher foraging rate in 10 seconds whereas *glr-1* and *trpa-1* forage more slowly than wild type. Furthermore, wild type and *dgk-1* have very close foraging rate in 10 seconds.

For each strain, we also want to determine whether the foraging behavior is a periodic and continuous oscillation or a process in which foraging movements occur sporadically. Our hypothesis is that the foraging behavior of each strain is an almost periodic process with a certain time T between adjacent foraging events. If this is true, we expect all time intervals between adjacent events for each strain to be close to each other and at least locate within a certain range (in this experiment, we use the Interval AVE in Table 4.3 ± 0.1 second to be our range). We use a Chi-Square test [31] to compare the expected and observed results from all videos of each strain and the results are listed in Table 4.4. We can see that all *p*-values are less than 0.05 (≈ 0) which means the difference between the expected and observed result is significant and the foraging behavior is more or less a sporadic process rather than periodic.

Table 4.4: The average number of foraging events within 10 seconds. The first row lists the mutant type and the second row lists the total number of frames in which the worms were moving forward. The third row lists the number of foraging events detected by the algorithm. The fourth row shows the average number of foraging events in 10 seconds. The bottom row shows the Chi-Square and *p*-values of sporadic test for each strain.

Mutant type	*dgk-1(nu62)*	*glr-1(n2361)*	*goa-1(n1143)*	*trpa-1(ok999)*	wild type
Total frames	34136	43728	32379	35010	36835
Detect foraging	1669	1604	1820	1480	1789
# of foraging in 10 seconds	14.67	11.00	16.86	12.68	14.57
Chi-Square (*p*-value)	1067 (≈ 0)	1016 (≈ 0)	1203 (≈ 0)	928 (≈ 0)	1190 (≈ 0)

4.4 Conclusion

We have described a new algorithm for the automated detection and quantitative analysis of foraging in *C. elegans*. This algorithm makes it possible to use videos of crawling nematodes collected from an automated tracking system to detect foraging events with a reliability comparable to what is achieved by a human observer. The ability to automatically count the number of foraging events in a unit time facilitates the rapid quantification of the foraging rate, a key behavioral parameter in a number of previous *C. elegans* studies [19, 20, 37]. In the past, this rate has been calculated by observer analysis of video recordings, a laborious and time-consuming process.

In addition to making it possible to count foraging events, our algorithm also makes it possible to measure parameters of foraging behavior that have not previously been measured quantitatively. For example, we measured foraging amplitude. Previous studies noted anecdotally that abnormalities in specific neurons and genes can affect the depth of foraging bends. For example, it was reported that ablation of two classes of mechanosensory neurons, OLQ and IL1, leads to deeper foraging bends [18], as do mutations in the *trpa-1* gene which encodes a mechanosensory channel that functions in these neurons [37]. However, neither of these studies

was able to present quantitative data to verify or measure the differences between normal worms and lesioned or mutant animals. In the current study, we quantified the foraging amplitude phenotypes of *trpa-1* mutants as well as several other foraging-abnormal mutants. For example, the average amplitude of *trpa-1* was found to be 15.71 compared to 14.15 for wild type, and this difference was statistically significant at the 5% significance level ($p = 0.0011$). This parameter should prove useful for future analysis of *C. elegans* behavioral mutants affecting the neurons controlling foraging bends.

We also derived two other novel foraging-related parameters in this study. One is foraging frequency. We verified the derived frequencies by using Fourier analysis and the power spectrum of the foraging traces of individual worms. The second parameter is the time interval between adjacent foraging events. Combined with the significance test (*t*-test) [41], we showed that *dgk*-1 has the closest relationship to wild type among the four mutants. It has two categories (interval and frequency) with *p*-values greater than 0.05 which is considered to be insignificant while other mutants have none (Table 4.5). We also used a Chi-Square test to examine whether the observed data are periodic. This result suggests that during periods of active foraging, the foraging behavior is more or less sporadic rather than a process in which foraging movements occur periodically and continuously.

Table 4.5: The results of the significance test (p-value < 0.05 is considered to be significant). The second to fourth rows show the p-values for comparisons of amplitude, time interval and frequency respectively.

	dgk-1(nu62) against wild type	*glr-1(n2361) against wild type*	*goa-1(n1143) against wild type*	*trpa-1(ok999) against wild type*
p-value (amplitude)	0.0087	0.0187	0.0000	0.0011
p-value (interval)	0.1129	0.0000	0.0018	0.0234
p-value (frequency)	0.1556	0.0018	0.0000	0.0003

This chapter is a reprint of the material as it appears in Kuang-Man Huang, Pamela Cosman, and William Schafer, " Automated Detection and Analysis of Foraging Behavior in *C. elegans* " Journal of Neuroscience Methods, accepted January 31st 2008. I was the primary researcher and the co-authors Dr. Pamela Cosman and Dr. William Schafer directed and supervised the research which forms the basis for this chapter. This work was supported by a research grants from NIDA (DA18341 and DA12891).

Chapter 5

Multi-worm tracking

Several automated tracking and analysis systems have been previously developed for *C. elegans*. These tracking systems can be divided into two categories. The first class of systems track single worms at a high magnification and use a motorized stage to keep the worm in the middle of the camera field of view [4-7]. These systems can quantify the detailed posture of the individual worms and perform phenotype classification. Systems in the second category track multiple worms at a low magnification [42, 43]. They are able to capture the worm's gross motion characteristics, such as velocity.

Some behaviors of significant interest to researchers, such as mating and social feeding [43, 44], by their very nature involve physical interaction between animals. For any automated system to be useful in characterizing these behaviors, it is essential that the position and body posture of a worm can be followed during and after physical contact with another animal. In this chapter we describe a new method for tracking multiple *C. elegans* that makes it possible to accurately resolve the individual body postures of two worms in physical contact with one another by using a modeling algorithm.

Model matching algorithms can be roughly divided into two categories [45, 46]. The first class is “contour based” which represents objects in terms of their boundaries. One popular approach is called active contours or “snakes”, which deform elastically to fit the contour of the target structure [47]. Another method uses combinations of trigonometric functions with variable coefficients to represent objects [48, 49].

The second class consists of “appearance-based” approaches. In this case a model is used to simulate the complete appearance (shape, color, texture, etc.) of the target object in the image. Bajcsy and Kovacic describe a volume model that deforms elastically to cover the object region [50]. In [51], a model composed of a collection of parts is used to represent objects in terms of a constellation of local features. All parts in this model are constrained with respect to a central coordinate system. There have been other part-based modeling algorithms. Fischler and Elschlager introduce an articulated model with all parts arranged in a deformable configuration which is represented by spring-like connections between pairs of parts [52]. This articulated model has recently been used for tracking an individual person in videos [53, 54]. In [21], this method is further improved with an efficient algorithm for finding the best match of a person and a car in images. A number of methods to track a three-dimensional human figure using articulated models also have been proposed. [55] uses a 3D articulated model combined with a modified particle filter to recover full articulated human body motion. [56] tracks a 3D human body with 2D contours using information from multiple cameras with different viewpoints. [57] presents an algorithm which projects 3D motion of a figure onto an image plane instead of using multiple cameras.

Multiple object tracking has been an intensive area of research due to its many applications [58-60]. However, traditional methods fail when the objects are in close proximity or present occlusions. A snake, for example, can be used to track individual deformable moving binary objects. But when two binary objects touch each other, the boundary between them is not

represented in the image data, and so a snake algorithm would likely fail without usable image information at the boundary. Tracking multiple objects separately in videos is achievable by using appearance-based models. Some methods track and separate people with occlusions by using cues such as appearance (color and texture of clothing, etc.) from frame to frame [61-64]. Some approaches are able to track multiple rigid objects in simple interactions. In [65], Khan uses particle filtering combined with a pairwise penalty function that only depends on the number of pixels which overlap between the appearance templates associated with the two targets to track multiple ants. In [66], Qu uses the "magnetic potential" and "inertia potential" to solve the "error merge" and "false object labeling" problems after severe occlusion of targets.

Although there have been numerous algorithms for multiple-object tracking, additional issues arise when tracking *C. elegans* worms. The worm tracking problem differs from previous tracking experiments in several respects: *1)* The worms are not rigid bodies; they are highly deformable, *2)* the actual worm body is transparent, and *3)* the worm moves almost entirely in a 2-D plane. The fact that the worm body is transparent means that color information is completely unavailable and even grayscale can be unreliable. Therefore we cannot use the color or texture to easily distinguish the animals from the background of transparent bacteria layers. To solve this technical problem, the illumination of the microscope is chosen to make the worm body look dark and the background look bright in grayscale images. This means the grayscale images are very nearly binary, and in fact we convert them to binary as a pre-processing step. While the binary nature of the images makes it relatively easy to compute body posture features for single-worm videos, it is a difficult problem to track and distinguish touching deformable binary blobs when there is more than one object in the scene. The fact that the worm moves almost entirely in a 2-D plane also makes this tracking problem somewhat different from many prior studies. Although portions of a worm's body can cross over/under its own body or the body of another worm, the worm's body is sufficiently flat that such crossings do not involve appreciable curvature up and

out of the plane of the agar plate. Thus the projection of the worm's body onto the plane of the plate has a roughly constant length.

These three significant differences make the worm-tracking problem quite unique compared to other multiple object tracking problems, such as those involving people and cars. Prior research on tracking multiple *C. elegans* is limited to [67] and [68], of which the latter is a preliminary version of the research presented in this chapter. In [67], a deformable worm model and several motion patterns are used to define an overall locomotory space of *C. elegans* and to describe its general dynamic movements. Using a combination of "predicted energy barrier" and multiple-hypothesis tracking, multiple touching worms can be identified and separated with a high success rate. The algorithm presented in [67] focuses on the crawling mechanism, which means sudden position-shift of the worm bodies caused by the swimming mechanism of the worm or even the movement of the stage may lead to loss of a track. Our work differs from [67] in that the algorithm presented here can accommodate both crawling and swimming mechanisms or other causes of sudden shifts such as stage movement. Our work differs in that we also introduce various pixel-based, feature-based, and human observation based methods for evaluating the accuracy of the body matching algorithm, and we evaluate some features (angle change rate, reversals) extracted from the matched body positions.

In this chapter we present a method for tracking and distinguishing multiple *C. elegans* in a video sequence, including when they are in physical contact with one another. The worms are modeled with an articulated model composed of rectangular blocks, arranged in a deformable configuration represented by a spring-like connection between adjacent parts. Dynamic programming is applied to reduce the computational complexity of the matching process. Our method makes it possible to identify two worms correctly before and after they touch each other, and to find the body poses for further feature extraction. All joint points in our model can be also considered to be the pseudo skeleton points of the worm body. It solves the problem that a

previously presented morphological skeleton-based reversal detection algorithm fails when two worms touch each other.

In the next section, we first review the basic concepts of dynamic programming which have been previously used in many applications. We then give a detailed description of our articulated model and multi-worm tracking algorithm. Finally we present experimental results and evaluate how well the algorithm performs on three different goals: to find the best body poses for both worms, to identify two worms correctly before and after they touch each other, and to detect reversals even when morphological skeletons are not available due to the two worms touching.

5.1 Dynamic programming

Dynamic programming, introduced by Richard Bellman in the 1940s, was first used to describe the process of solving problems where one needs to find the best decisions one after another. It is a method to solve problems with properties of optimal substructure [69]. Optimal substructure means that we can use optimal solutions of subproblems to find the optimal solutions of the overall problem. Basically, we can solve a problem with optimal substructure using a three-step process:

1. Break the problem into smaller subproblems.
2. Solve these problems optimally using this three-step process recursively.
3. Use these optimal solutions to construct an optimal solution for the original problem.

The following is an example of a particular type of shortest path problem [69]. Suppose we wish to get from A to J in the road network of Figure 5.1:

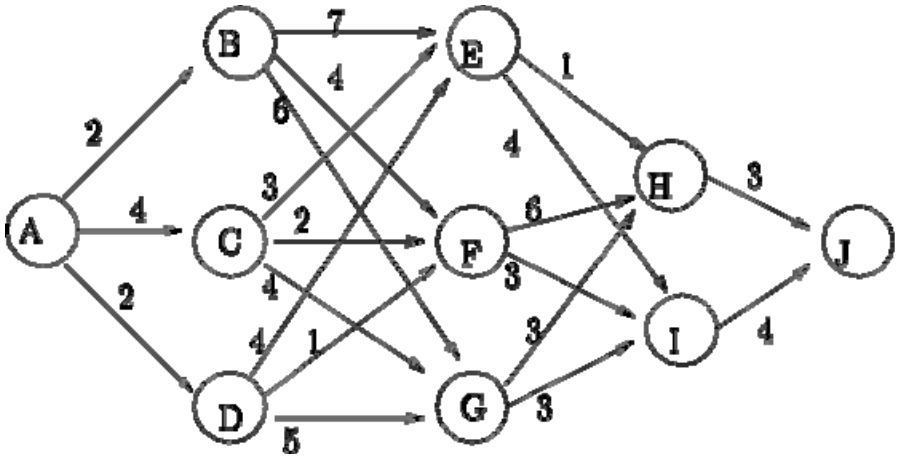


Figure 5.1: The road network.

All the numbers represent distances between pairs of nodes. To solve this problem, first we break it in several stages. Stage 1 contains node A, stage 2 contains nodes B, C, and D, stage 3 contains node E, F, and G, stage 4 contains H and I, and stage 5 contains J. The states in each stage correspond to the node names. For example, stage 3 contains states E, F, and G.

Let's do our calculation step by step:

1) *Stage 4:* Because there is only one path from either of H and I to J, we do not really have to make any decision in this stage.

2) *Stage 3:* In this stage we have 3 nodes E, F and G. From F we can either go to H or I. If we choose H, the total distance from F to the destination J will be *dist*(F,H) + *dist*(H,J) = 9. On the other hand, if we choose I, the total distance from F to J becomes *dist*(F,I) + *dist*(I,J) = 7. Therefore we know that the shortest path from F to J is F→I→J. The rest of the calculation for this stage is in Table 5.1.

Table 5.1: Calculation for stage 3

	Total distance to J		The shortest distance	Decision go to
	via H	via I		
E	4	8	4	H
F	9	7	7	I
G	6	7	6	H

3) *Stage 2*: We continue working back through the stages one by one, each time completely computing a stage before continuing to the preceding one. The results are:

Table 5.2: Calculation for stage 2

	Total distance to J			The shortest distance	Decision go to
	via E	via F	via G		
B	11	11	12	11	E or F
C	7	9	10	7	E
D	8	8	11	8	E or F

4) *Stage 1*: In this stage, we can see that the final decision goes to C or D.

Table 5.3: Calculation for stage 1

	Total distance to J			The shortest distance	Decision go to
	via B	via C	via D		
A	13	11	10	10	D

If we trace back from stage 1, our final decision is A→D→E→H→J or A→D→F→I→J.

To summarize, there are a number of characteristics to this problem and to other dynamic programming problems. These are:

1. The problem can be divided into stages with a decision required at each stage. In the example, they were defined by the structure of the graph. The decision was where to go next.
2. Each stage has a number of states associated with it. In the example, the states were the node reached.

3. The decision at one stage transforms one state into a state in the next stage. The decision of where to go next defined where you arrived in the next stage.
4. Given the current state, the optimal decision for each of the remaining states does not depend on the previous states or decisions. In the example, it was not necessary to know how you got to a node.
5. There exists a recursive relationship that identifies the optimal decision for stage *j*, given that stage *j*+1 has already been solved.
6. The final stage must be solvable by itself.

The key point in dynamic programming is to determine stages and states so that all of the above hold. The recursive relationship makes finding the values relatively easy.

5.2 Worm model

Because the *C. elegans* worm body is more or less cylindrical, in this paper, we model its projection as being composed of *N* rectangular parts with length L and width W in the ratio 2:1. To generate a more accurate and robust worm body model, parameters *N*, L and W are learned from the image data. They can differ for different input videos. For a given video, let l and w be the average length and width of the worm body calculated from all non-touching frames. These values are calculated automatically using the method described in [7]. Typical values of l and w were approximately 95 and 6 pixels respectively. Then we set $W = 0.9w$ and $L = 2W$ and $N = round(\frac{l}{L})$. In this experiment, W ranged from 4 to 8 pixels (L ranged from 8 to 16 pixels) and N ranged from 6 to 9 parts. The position of each part in the image can be defined by the triple (x, y, θ), which specifies the coordinate of the center and the orientation of the part (Figure 5.2a). Adjacent parts are connected by two joint points (Figure 5.2b), which may coincide (Figure

5.2c) but also might be chosen to not be coincident. When (x, y, θ) is determined for each of the N rectangular parts composing the worm body model, we refer to this as a worm body pose.

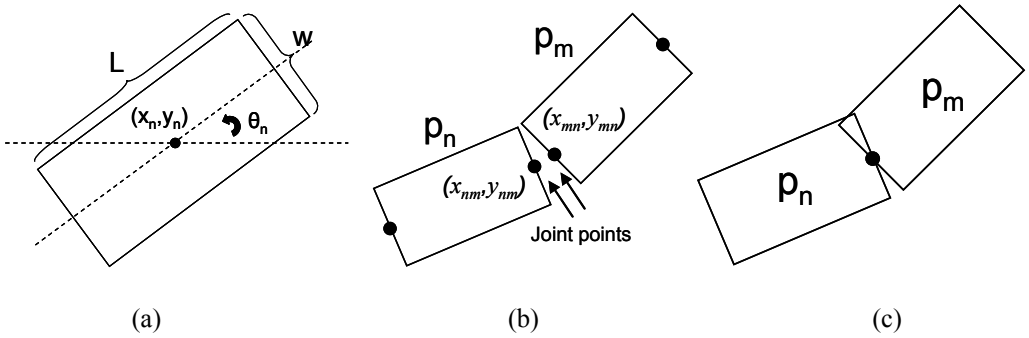


Figure 5.2: (a) One rectangular part and its parameters, L and W are the length and width of the rectangle and (x, y, θ) specifies the coordinates of the center and the orientation of the part, (b) two parts of the worm model and their joint points, (c) two parts of the worm model with the joint points coinciding.

We seek to find the best match of the worm model to the actual binary worm image data. The concept of best match incorporates both how well the rectangular parts fit the image pixel data, and also how well the rectangular parts fit worm body anatomy. By "worm body anatomy" we mean how well the parts fit with each other into a smooth worm body (for example, adjacent parts should not have large gaps between their joint points) [21, 22, 52], as will be discussed in the next paragraphs.

We begin by considering how well a rectangular part fits the image pixel data and deterministically examining the set of K most plausible pixel positions from the object (K can vary based on different image resolutions used in different experiments). These are the positions which have the lowest match cost to place our rectangular part. The match cost $m(I,p_i)$ of a part p_i with 12 different possible orientation angles (15°, 30°, 45°,…., 180°) at every possible integer pixel position (x,y) can be computed by convolving the binary worm image I with a convolution kernel composed of a "match" rectangle with different orientation angles (with the same size as

our rectangular part) embedded in a larger "no match" rectangle (Figure 5.3) [21, 22]. The entries of this convolution kernel are defined by the following equation:

$$k(x,y)=\begin{cases} -1 & , if \quad (x,y)\in No \quad Match \\ \dfrac{WL}{S}e^{-(\frac{2x^2}{W^2}+\frac{2y^2}{L^2})} & , \ if \quad (x,y)\in Match \end{cases}$$

where $S=\sum_{(x,y)\in Match} e^{-(\frac{2x^2}{W^2}+\frac{2y^2}{L^2})}$, W and L are the width and length of a rectangular part and (x,y) are the coordinates relative to the center of the kernel $(\frac{-W}{2}\le x\le\frac{W}{2},\frac{-L}{2}\le y\le\frac{L}{2})$. The convolution sum $c(i,j)$ at each pixel position (i,j) is the sum of the two terms $c_{foreground}(i,j)$ and $c_{background}(i,j)$:

$$c(i,j)=c_{foreground}(i,j)+0.2\times c_{background}(i,j)$$

where $c_{foreground}(i,j)=\sum_{\substack{(x,y)\in Match \\ I(i+x,j+y)=1}} k(x,y)$ and $c_{background}(i,j)=\sum_{\substack{(x,y)\in NoMatch \\ I(i+x,j+y)=0}} 1$, and the match cost m at pixel position (i,j) in this dissertation can be expressed by the following equation:

$$m=e^{\frac{-c(i,j)}{WL}}$$

In this convolution kernel, points close to the y axis ($x = 0$) have larger weights. Despite the fact that we set $W=0.9w$, in some images, it is possible that the original worm body width is slightly smaller than the width W of the rectangular part. By using this kernel, pixel positions close to the middle of the body will have relatively smaller match cost than positions close to the edge of the body, which means the likelihood for a part to be placed along the middle line is larger than other possible positions.

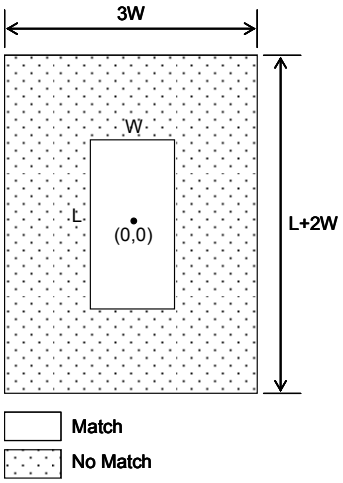


Figure 5.3: Convolution kernel used to calculate match cost.

For each image frame, as will be discussed, we generate a list of M plausible body poses, from which we would like to choose the best ones for each of the two worms. These M poses are the ones which have the lowest values of the match cost plus the deformation cost. The deformation cost measures how each worm body pose agrees with worm body anatomy. The pairwise deformation cost is defined as the following:

$$d(p_m, p_n) = W_x \times \left| x_{m,n} - x_{n,m} \right| + W_y \times \left| y_{m,n} - y_{n,m} \right| + W_\theta \times \left| \theta_m - \theta_n \right| \qquad (1)$$

where $|.|$ denotes absolute value, and $x_{m,n}$, $x_{n,m}$, $y_{m,n}$ *and* $y_{n,m}$ are the xy coordinates of joint points between adjacent parts p_m and p_n (Figure 5.2a). The angle θ_m is the orientation angle in degrees of the part p_m. W_x, W_y and W_θ are weights for the cost associated with a horizontal (x-direction) offset between joint points of adjacent parts, a vertical (y-direction) offset between joint points of adjacent parts, and a difference in the orientation angle between the two parts. The deformation cost attains its minimum value of 0 when two parts have the same orientation angle and the joint points between them coincide. In the next two sections, we show how to generate a list of M plausible body poses, which is decided by how well the model matches the object in the image

and how well it lowers the deformation cost. To make finding the best match computationally efficient, we use a dynamic programming algorithm.

5.3 Multi-worm tracking algorithm

In this dissertation, we use dynamic programming to solve sequential decision problems with a compositional cost structure in which the decisions at one stage become the conditions governing the succeeding stages [70, 71]. We try to minimize the energy defined by both the match quality of the model and the restrictions between pairs of parts [52, 72]. The model contains N rectangular parts. We suppose the general cost function of each part p_i can be expressed by the following equation:

$$E_i(p_i) = d(p_{i-1}, p_i) + m(I, p_i) + E_{i-1}(p_{i-1}) \qquad \text{for } i = 1 \text{ to } N\text{-}1 \qquad (2)$$

where p_i is defined by (x_i, y_i, θ_i) as previously described, $d(p_{i-1},p_i)$ defined in equation (1) measures how much the adjacent parts p_{i-1} and p_i contribute to the deformation cost, and $m(I,p_i)$ measures how well the part p_i matches the image I with its current position.

5.3.1 Worm Body Poses Sampling

To generate a list of body poses, we begin by taking the $\frac{K}{3}$ (x_0, y_0, θ_0) triples with the lowest match cost from the previously generated list of K positions, to place our first part p_0, which is one of the two end parts (it could be either the head or the tail). The reason why the lowest match cost for placement of one single part will correspond almost surely to one of the two ends is because background pixels on 3 sides of the central "match" region will be correctly included in the "no match" region of the kernel in Figure 5.3, whereas for a typical part in the middle of the worm, background pixels only on 2 sides of the central "match" region will appropriately correspond with the "no match" region of the kernel. Using only the one-third best

positions of all K positions only for part p_0 slightly reduces complexity without significantly increasing the chance of generating bad worm body poses. After the part p_0, all K positions will be possible candidates to place parts from p_1 to $p_{N\text{-}1}$.

The energy $E_0(p_0)$ of each part p_0 is just the match cost at the position because p_0 is the first part of the model. If $E_0(p_0)$ of each p_0 is known and we define $p_{i-1}(p_i)$ to be the best position for the part $p_{i\text{-}1}$ as a function of the position of the next part p_i, then the best position of the part p_0 in the input image I is:

$$p_0(p_1) = \arg\min_{p_0}(d(p_0, p_1) + m(I, p_1) + E_0(p_0)) \tag{3}$$

That is, the best position of the part p_0 can be decided given the position of its next part p_1. The minimum energy $E_I^*(p_1)$ of the given part p_1 (minimized over all possible values of p_0) becomes:

$$E_1^*(p_1) = \min_{p_0}(d(p_0, p_1) + m(I, p_1) + E_0(p_0)) \tag{4}$$

Based on the same concept, the best location of the part $p_{i\text{-}1}$ ($1 < i < N$) as a function of the position of the next part p_i is:

$$p_{i-1}(p_i) = \arg\min_{p_{i-1}}(d(p_{i-1}, p_i) + m(I, p_i) + E_{i-1}^*(p_{i-1})) \tag{5}$$

and then $E_i^*(p_i)$ will be:

$$E_i^*(p_i) = \min_{p_{i-1}}(d(p_{i-1}, p_i) + m(I, p_i) + E_{i-1}^*(p_{i-1})) \tag{6}$$

We continue this forward process until the other end of the model $p_{N\text{-}1}$ is reached. Finally for the part $p_{N\text{-}1}$, the best configuration is the one that has the minimum cost $E_{N\text{-}1}$.

$$p_{N-1opt} = \arg\min_{p_{N-1}}(E_{N-1}^*(p_{N-1}))$$

and the optimum positions of all parts can now be traced in the backward step from the part p_{N-1} to p_0 by using equations (3), (4), (5) and (6). In our experiment, we chose the M configurations with minimum cost to be the plausible poses for each frame.

5.3.2 Multi-worm Match

In all of our multi-worm videos, the two worms start out separated. Each video can therefore be divided automatically into subsections which are of two types: *A)* where the two worms are clearly separated (the distance between the centroids of the two worms is longer than the length of the worm body) and *B)* where the worms are close to or touch each other.

Type A: For any subsection of the video in which the two worms are clearly separated, after M possible worm poses are composed in each frame, we apply a dynamic programming algorithm again *over the time domain* to find the best *temporal sequence of poses* that move smoothly for the first worm within that first separated section of the video. This time we try to minimize the cost function L that combines the match cost of the whole worm body pose m_{total}, which is equal to E_{N-1} in equation (2), and the Euclidean distance d_{total} between the current and the previous worm body poses:

$$L = W_d \times d_{total} + W_m \times m_{total} \tag{7}$$

In this paper, W_d and W_m are also set by limited experimentation and chosen to be in the ratio $5e^{-2}$: 1. We let W_d be smaller than W_m, which gives those body poses with low match cost higher priority to be chosen, and allows the worm body to move from frame to frame without increasing the cost too much. The Euclidean distance between two body poses is calculated as the following:

$$d_{total} = \sum_{n=0}^{N-1} d(p_n^{i-1}, p_n^i)$$

where p_n^i is the *n*th part of the body pose in the *i*th image frame. Then we remove the first worm from the images in the sequence and repeat the dynamic programming algorithm to find the best sequence for the second worm.

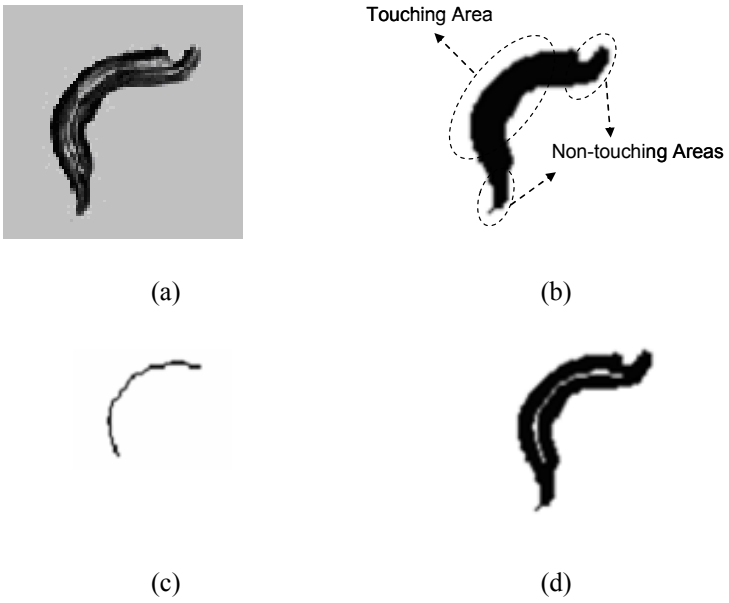


Figure 5.4: (a) Two worms in the original grayscale image, (b) two worms in the binary image, (c) the binary image after eroding, (d) image b minus image c shows the two worms partially separated.

Type B: For the close/touching portion of the video, we begin by using the fact that the area where two worms touch each other is thicker than other areas to divide touching worms (Figure 5.4a, 5.4b). For any close/touching frame, in order to remove non-touching worm body parts, we first erode the touching worm body object $\frac{W}{2}$ times (where W is the width of a rectangular part in the body model) with a 3 by 3 structuring element (Figure 5.4c), then we subtract the eroded image from the original binary image to get a new image. The two worms may be only partially separated in the new image (Figure 5.4d). But this method will increase the chance of finding good worm body poses by heightening the match cost for those body poses overlapping with the non-filled area. If the two worms are only close to each other without

touching or crossing each other, the new image will be just equal to the original binary image because there will be no object in the eroded image.

The dynamic programming approach can make the process of finding the best sequence more computationally efficient, and therefore we use it for the portion of the video where the worms do not touch, which is the majority of the video and also the portion of the video where the body-fitting is easier. However, for touching/close frames, it is a more difficult problem to simulate the two worm body poses correctly. When worms touch, the binary foreground blob can be much larger than a typical single worm body, and therefore choosing good body poses in a frame requires not only (as in Section 5.3.1) the match cost between image blob pixels and model pixels, and the deformation cost of deforming the model, but also must rely on the overlapping cost of the two worm bodies (a cost which doesn't enter into the case where the two worms are far apart), as well as the distance between the current possible pose and the previous poses (a cost which only entered into the dynamic programming in the second step, as we try to find the sequence of poses).

For this reason, we modify our approach. For each touching/close frame, one of the two worms will be chosen as the primary worm. As discussed in section 5.3.1, we have a set of M plausible body poses for the frame, obtained using the dynamic programming approach to minimize the match cost and deformation cost. From this set, first the best H body poses are chosen to be candidates for the primary worm based on both the match cost of the whole body pose and the distance between the pose in the current frame and the pose in the previous frame. For each of these H candidates, we find the best body pose for the secondary worm from those M poses to fill the remainder of the object based on the overlapping cost. The overlapping cost is defined to be the sum of two terms: 1) the number of object pixels covered by both the primary worm body and the secondary worm body, 2) the number of object pixels not covered by any worm body. Then we choose the best single set of body poses for the two worms, which is the set

that achieves minimum value of the overlapping cost plus the Euclidean distance between the current pose and the previous pose for both worms from those *H* sets. To avoid the whole result being dominated by only one worm, the assignment of the primary worm and the secondary worm alternates from one frame to the next.

As a final step for all frames in both the separated portion and the close/touching portion of the video, we apply a two dimensional Gaussian filter to smooth the final results of every frame. Figure 5.5a shows the integer valued convolution mask which approximates a Gaussian with σ of 1.4 in this experiment and Figure 5.5b and 5.5c show the worm bodies before and after Gaussian smoothing. The block diagram of the whole multi-worm match process is shown in Figure 5.6.

For the reversal detection discussed later, after the best match configurations of both worms are decided, we can manually assign one of the two end parts to be the head/tail part by using the original images as references. The manual assignment does not need to be done on each frame. It is done only once per video.

$$\frac{1}{115}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

(a) (b) (c)

Figure 5.5: (a) 2-D discrete approximation to Gaussian function with σ = 1.4, (b) the worm body before smoothing, (c) the worm body after smoothing.

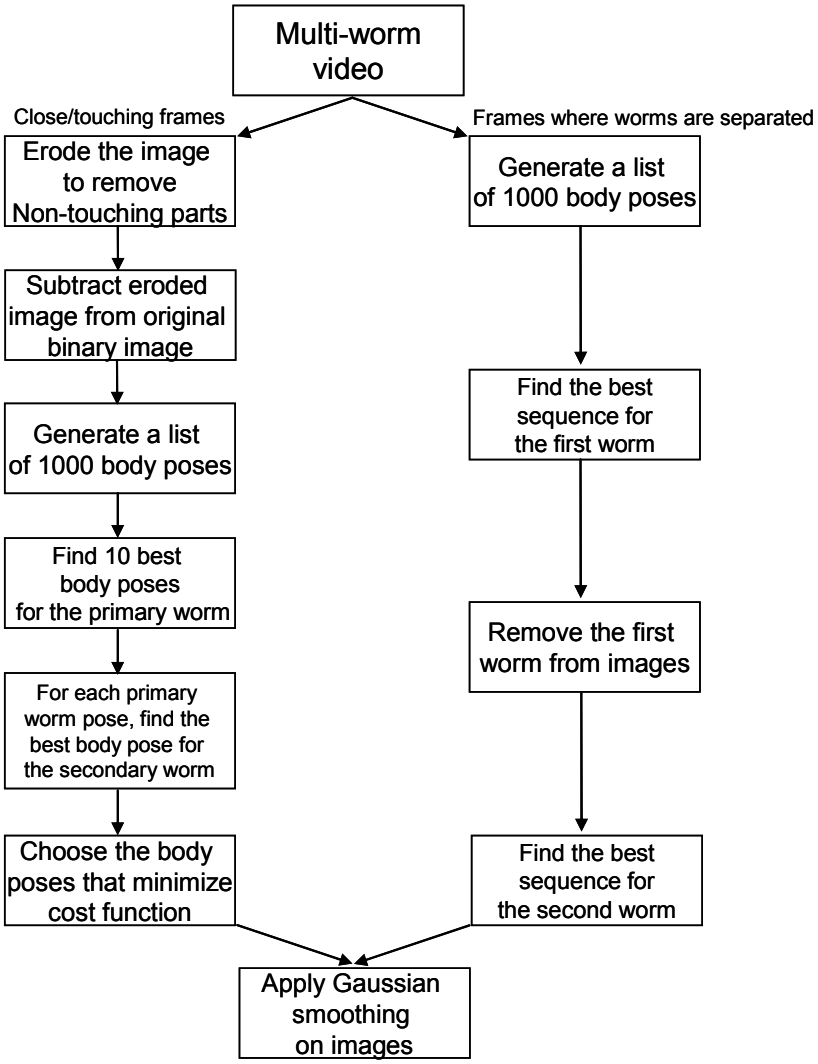


Figure 5.6: Block diagram of the multi-worm match algorithm.

5.4 Results

5.4.1 Experimental results

The experiments were performed using Matlab on a 2.33 GHz Pentium-IV desktop computer. The algorithm was run on 29 different videos which contained 10579 frames in total and each sequence varies from 131 to 498 frames. *K, M,* and *H* were set to be 3000, 1000, and 10 respectively, where *K* is the number of sampled pixel positions, *M* is the number of sampled worm body poses for each image, and *H* is the number of the candidate body poses for each worm in each frame as previously described. The weights W_x, W_y, and W_θ in equation (1) were chosen in the ratio 4:4:3 based on limited subjective evaluation over a set of values. Binary images are obtained from the original images by using the thresholding algorithm from [7]. Some pictorial results are shown in Figure 5.7. Images a, b and c are frames 13, 111, 134 extracted from the first video, images d, e and f are frames 86, 143, 206 from the second video, and images g, h, and i are frames 16, 63, 91 from the third video (with two worms crossing each other). In each image, the left side shows the original grayscale image and the right side shows the matching result. The two worms are represented in red and green colors. These examples illustrate the ability to identify two worms correctly before and after they touch each other. Their body poses are simulated and clearly distinguished during the time the two animals are touching.

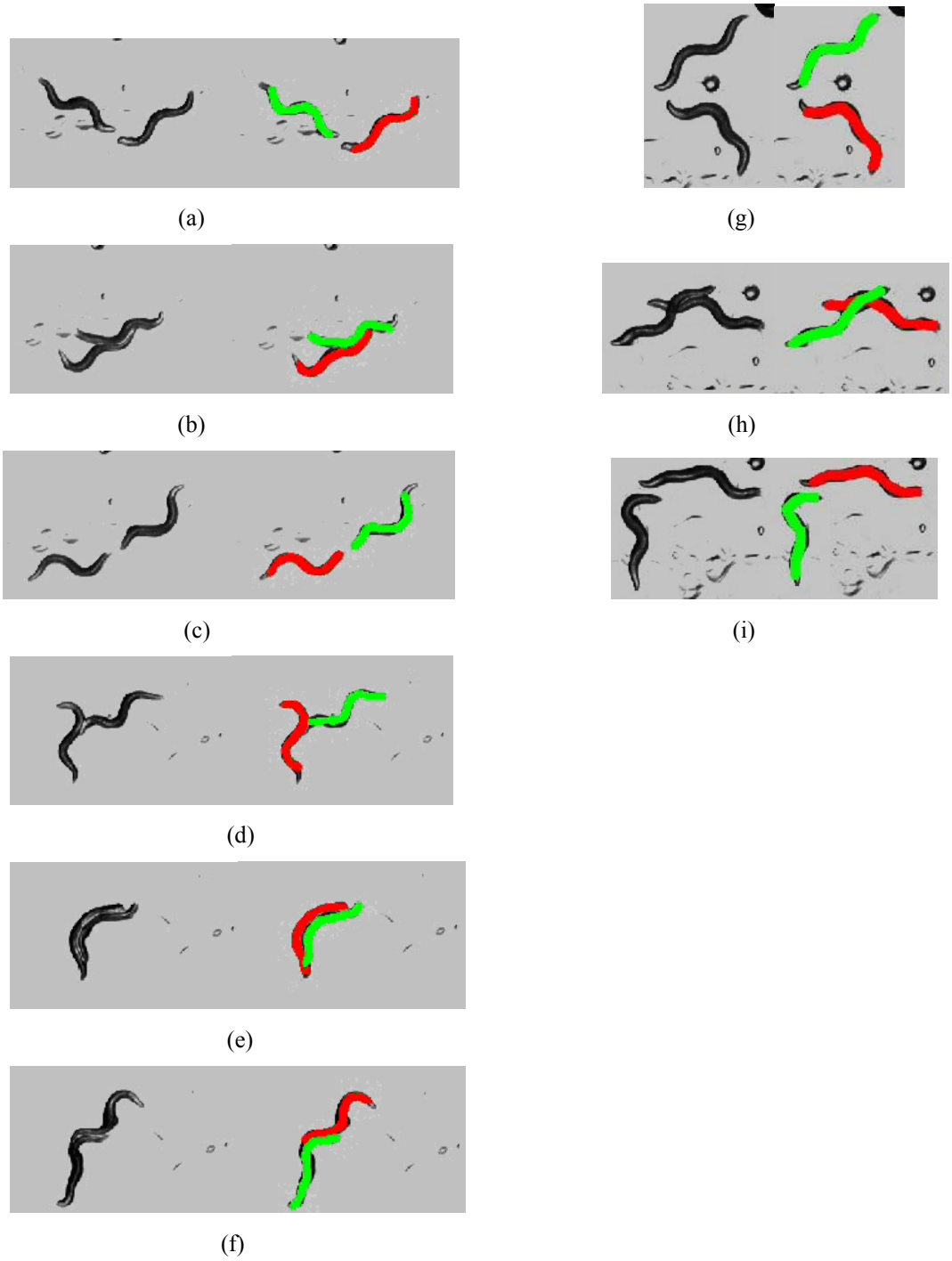


Figure 5.7: Nine images from three videos show the best matching configuration. Images a, b and c are frames 13, 111, 134 extracted from the first video, images d, e and f are frames 86, 143, 206 from the second video, and images g, h and I are frames 16, 63, 91 from the third video (with two worms crossing each other). In each image pair, the left side shows the original grayscale image and the right side shows the matching result. The two worms are represented in red and green colors.

5.4.2 Verification results

Next, we evaluate how well the algorithm performs on the three major goals: *A)* to find the best body poses for both worms for further extraction of motion related features, *B)* to identify two worms correctly before and after they touch each other in every video and *C)* to detect reversals even when morphological skeletons are not available due to the two worms touching.

A) Good estimated pose and Motion-related features

In [67], tracking accuracy is evaluated using the "editing extent," which is defined as the distance between the automatically and manually detected head locations, normalized by the worm length for those worms that are considered incorrectly segmented by the human observer. In order to emphasize that our algorithm can simulate the highly deformable nature of the worm body, we use pixel-based, feature-based and human observer based methods to evaluate the match quality of our algorithm. We begin the evaluation of the algorithm by comparing how well it does against a manual fit of the body model frame by frame. The algorithm was tested on 1913 images (including 793 non-touching frames and 1120 touching frames) randomly chosen from 11 videos with a different pair of worms in each video. Given an original image and the number of parts N in it, we first chose $N+1$ joint points (including two end points) manually in every frame by clicking with the mouse on the image (Figure 5.8a). Then these points are used to compute x, y and θ of each part and to build worm body poses (Figure 5.8b). This is followed by Gaussian smoothing.

(a) (b)

Figure 5.8: (a) 8 joint points chosen manually for the 7 parts in this frame, (b) body pose built from manually selected joint points.

We compare the results from our algorithm to these manually built body poses and evaluate the accuracy by computing the correct percentages defined by the following equation:

$$\text{correct percentage} = \frac{N_{AM}}{N_A}$$

where N_{AM} is defined as the number of object pixels covered by both the automatically generated model and the manually generated model, and N_A is defined as the number of pixels covered by the automatically generated model. A higher value for this percentage indicated that the body poses decided by the algorithm agree more with the manually generated body poses (Figure 5.9).

Figure 5.9: An example of the comparison between the automatically generated model and the manually generated model. The black area is covered by both models; the gray area is the difference between these two models.

For frames without touching, because the two worms are separated and each worm can be easily extracted to calculate its area, we also compare our matching results against both worm bodies in the original images. We compute two different scores:

$$\text{i) predicted positive value} = \frac{N_{MO}}{N_M} \quad \text{and} \quad \text{ii) true positive rate} = \frac{N_{MO}}{N_O}$$

where N_{MO} is the number of pixels covered by both the model (either manually or automatically generated) and the worm body in the original image, N_M is the number of pixels covered by the model, and N_O is the number of pixels covered by the worm body in the original binary image. Predicted positive value is an indication of the probability that a given pixel which our model says is part of the worm body actually is part of the worm body. True positive rate tells us the percentage of the actual worm body covered by our model. The predicted positive value will decrease and the true positive rate will increase if we use models with larger sizes (for example, if we choose $W = w$ instead of $W = 0.9w$).

All results are listed in Table 5.4. We notice that the correct percentages between automatically generated model and manually generated model all range from 72% to 83% except in two videos (015 and 020). In these two videos, the result is better for touching frames than non-touching frames. That is because the background is not very clear due to the un-evenness in the bacteria layer or crawl track left by the worm in these videos, which may cause the size of the binary worm body in some images to be abnormally larger than its usual size and predicted positive values to be very low. Predicted positive values are also over 85% and true positive rates are higher than 70% for almost all frames. In order to reduce the possibility of two worms overlapping in our results for touching frames, the width W of the model is always chosen to be 90% of the actual body width. For this reason, the model tends to be covered by the whole worm body which will cause the predicted positive value to be generally larger than the true positive rate. The results of the comparison between manually generated models and original images are also listed in the last two rows in this table, which clearly shows that our automatic matching algorithm outperforms human observers.

Table 5.4: Comparison results between automatically and manually generated models. The correct percentages between automatically generated model and manually generated model for the two worms are listed in rows 1 and 2. Predicted positive values and true positive rates for the two worms are listed in rows 3 and 4 (automatically generated model) and rows 5 and 6 (manually generated model).

File name		005	006	007	008	011	012	015	016	017	018	020
Automatically generated model against manually generated model (%)	Non-touching	82.7 77.5 (78)	77.8 76.8 (25)	81.1 81.3 (45)	74.7 80.1 (5)	77.8 80.7 (104)	79.6 81.1 (38)	74.1 69.2 (113)	75.3 79.2 (46)	79.8 81.1 (139)	72.1 74.9 (144)	82.4 67.7 (56)
	touching	80.2 73.8 (122)	73.3 75.3 (176)	76.6 78.4 (155)	76.0 77.2 (144)	78.2 78.7 (55)	76.0 79.2 (94)	76.6 78.4 (64)	77.8 80.7 (51)	79.4 78.6 (61)	80.2 81.9 (55)	79.4 77.7 (143)
Automatically generated model (predicted positive %)		89.5 86.2	92.4 89.3	87.9 88.5	91.2 91.5	92.0 87.5	90.5 86.5	94.7 80.5	85.0 90.9	90.9 90.2	84.2 85.4	92.1 86.4
Automatically generated model (true positive %)		83.3 77.9	82.8 78.3	89.8 88.2	75.3 82.4	68.3 83.5	76.9 83.0	70.6 74.5	77.7 80.4	82.3 81.7	83.3 80.1	83.1 73.7
Manually generated model (predicted positive %)		87.1 86.2	86.2 88.8	80.9 85.2	82.4 85.9	92.4 85.0	88.4 84.9	88.2 83.8	86.6 89.3	84.9 87.3	80.7 89.1	87.5 87.1
Manually generated model (true positive %)		81.8 76.6	75.5 76.7	83.9 85.1	68.8 77.2	67.6 80.0	73.9 80.4	68.0 72.0	78.5 75.9	76.8 78.3	78.6 76.9	80.1 71.3

In order to verify that our algorithm can simulate the highly deformable nature of the worm body, the angle change rate is computed from both the manually generated model and the model generated by our algorithm. The results are shown in Table 5.5. The angle change rate is defined in [4, 7] as the ratio of the average angle difference between every pair of consecutive segments connected by skeleton points. The angle change rate is an important feature characterizing body postures of mutants, and was shown in [4], for example, to be significant in distinguishing between *goa-1(n1134)* mutants and wild-type. In this paper, we use the joint points of the parts to be our skeleton points to calculate angle change rate. From Table 5.5, we see that the difference percentages of average angle change rate between our algorithm and manually

generated models are lower than 10% for non-touching (separated) frames and 20% for touching (close) frames in most of the videos.

Table 5.5: Angle change rate verification results. Numbers in each cell are angle change rates for the two worms.

File name		005	006	007	008	011	012	015	016	017	018	020
Automatically generated model (degrees)	Separated	32.6 33.0	33.7 30.6	37.9 31.0	37.0 39.0	29.8 30.9	29.2 35.1	31.7 34.9	33.1 29.3	32.3 32.4	36.1 34.3	36.6 41.3
	Close	30.0 31.4	33.7 30.3	36.2 31.8	32.4 33.6	27.6 33.6	30.6 32.1	25.0 34.3	30.0 24.8	30.2 29.3	28.8 32.3	38.0 35.4
Manually generated model (degrees)	Separated	33.5 36.2	34.5 32.7	40.7 32.5	34.6 40.5	33.3 29.8	30.7 34.3	31.9 38.9	33.5 24.7	34.0 34.6	37.5 34.2	36.1 43.4
	Close	30.4 30.6	33.6 29.4	41.0 32.3	34.2 34.4	28.6 30.8	31.0 34.3	29.9 31.2	30.9 24.5	32.5 33.3	29.4 31.7	42.2 38.6
Difference Percentage (%)	Separated	2.6 8.6	2.2 6.5	6.9 4.6	6.8 3.7	10.4 3.8	4.9 2.5	0.7 10.4	1.0 18.5	5.0 6.3	3.8 0.2	1.2 4.8
	Close	10.4 18.2	2.6 11.4	0.6 0.6	1.2 17.8	16.4 3.4	0.9 0.2	6.9 24.7	8.3 0.9	4.4 3.8	27.6 7.9	14.4 12.3

B) Correct identification

Both worms in all videos are also tracked by a human observer to see if our method can correctly identify worms separately after they touch. From Table 5.6, we see that our program can identify both worms correctly in 26 of 29 videos.

C) Reversals

C. elegans usually moves in a sinusoidal wave. When a worm is touched or presented with a toxic chemical stimulus, it will switch the direction of the wave, causing the animal to instantaneously crawl backward instead of forward. This backward movement is defined as a reversal. In [23], we used two skeleton points near the two ends as our reference points to decide if the worm was moving forward or backward. However, this reversal detection algorithm requires the skeleton points from each worm body which can not be obtained using the method in Chapter 3 when two worms touch each other. By using our modeling algorithm, after all parameters of all parts are obtained with our algorithm, all joint points can be considered to be pseudo skeleton points. We can use the two joint points nearest the two ends to be our reference points to detect reversals. Table 5.6 shows that there were 86 reversals correctly detected by our automatic algorithm. Of these, 57 occurred during the close/touching portion of the video. In addition, the automatic algorithm incorrectly declared 7 events to be reversals (6 of them were in the close/touching portion of the video). Only 3 actual reversals were missed (of which 1 was in the close/touching portion of the video). So our algorithm has a high rate of correct detections while maintaining a low rate of false alarms and false dismissals.

Table 5.6: Identification results and reversal verification results

Number of tested videos	29	
Number of correct identifications	26	
Number of wrong identifications	3	
Number of reversals correctly detected with results from our algorithm and the method in [22]	Touching 57 (98.3%)	Overall 86 (96.6%)
Number of wrong detections	6 (9.5%)	7 (7.5%)
Number of reversals missed	1 (1.7%)	3 (3.4%)

5.5 Conclusion

This chapter presents a method that combines articulated models and dynamic programming for simulating the body poses of *C. elegans* in multi-worm videos. The models are composed of a number of rectangular parts arranged in a deformable configuration. For each video, we begin by using a dynamic programming algorithm to generate many worm body poses in every frame. For those portions of the video where the two worms are clearly separated, a dynamic programming algorithm is used again to find the best sequences over time for both worms. For those portions of the video where the two worms are close or touching each other, we find the best match configuration for the two worms based on the Euclidean distances between pairs of body poses in adjacent image frames. There are several contributions in this chapter. First, the presented method allows us to identify two worms correctly before and after they touch each other in 90% of our videos. Second, we can use these models to accurately resolve the individual body postures of two worms in physical contact with one another. We note that this tracking algorithm for two worms is fully automated and requires no human annotation at any point (including no need for human annotation of the first frame). When this algorithm was combined

with a previously described algorithm for detection of reversals, human annotation was required to identify head and tail once per video, which was needed to identify reversals. However the tracking algorithm itself involves no human intervention, so it is suitable for analyzing large numbers of videos where human annotation would be tedious. We also showed that reversal behaviors of multiple worms can be accurately detected by using our model even when their bodies are in physical contact with one another.

This algorithm will provide many applications towards characterizing physical interactions between animals. Previous research on automated analysis of *C. elegans* videos has shown that large numbers of biologically relevant features can be automatically extracted. These features include, for example, body length and width, average speed, curvature, depth of body bends, and frequency and duration of reversals. These features and others have been shown to be important in classifying and characterizing many different mutant strains [7]. Using the algorithm described in this capter, the body poses of two worms can be identified, so the various features extracted in prior research for single-worm videos can now be extended to videos with two worms. In this chapter, we examined two of these features (angle change rate and reversals) in these two-worm videos. In addition, new features characterizing the interactions themselves (e.g., average duration of bodily contact) can be extracted across different mutant strains and different environmental conditions.

Chapter 6

Summary

This dissertation started with presenting skeletonization for coiler mutants by using a parameterized body model and locating the division line between overlapping portions of the worm body. Using this method, we extracted features from the obtained skeletons and used these features to classify and characterize wild type and coiler mutants. Then the omega bend and reversal detection algorithms were described in detail as we investigated the temporal correlation between these two behaviors. We also presented an automated method to detect and analyze foraging behavior of *C. elegans* in a video sequence using periodograms. After establishing some applications of the single worm tracking system, this dissertation addressed the tracking and distinguishing of multiple *C. elegans* in a video sequence. Our contributions in these aspects are summarized next.

Contributions

Chapter 2 is the first automated study of *C. elegans* coiler mutants. In this chapter, we proposed a new algorithm to extract biologically meaningful skeletons from coiled body postures having internal holes. This represents an important advance in the study of certain classes of locomotion-abnormal mutants. Experiments showed that locomotion features extracted from

obtained skeletons can be used to classify coilers with highly similar phenotypes. We also investigated the relative similarities between the different mutant behavior patterns. Using the Gap Statistic, we determined that the optimal number of clusters for these coilers is 7. This indicated that even among a single descriptive class of Unc mutants, several distinct phenotypic patterns could be observed, which illustrated the limitations of subjective definitions of mutant types with respect to both precision and accurancy.

In chapter 3, we developed and tested an algorithm for automatic detection of omega bends and reversals. This algorithm correctly detected 93% of omega bend events and correctly found 96.9% of reversal events. The relationships between omega bends and reversals for 6 strains (wild type, *syd-1*, *unc-10*, *unc-37*, *unc-75* and *unc-77*) were also examined and compared to each other. We found that the ventral bias of omega bends normally observed in wild-type was largely absent in some mutants such as *unc-37* and *unc-75*. Wild type, *syd-1*, *unc-10* and *unc-77* have very similar omega bend and reversal behaviors. Most detected omega bends were tightly coupled temporally to reversals for these 4 strains. The ventral bias is very strong particularly in wild type and *syd-1*.

The main contributions of chapter 4 can be summarized as follows: 1) we developed and tested a new algorithm for automatic detection of foraging events. 2) We provided quantitative analysis of foraging behavior, which has not previously been achieved, for several mutant types. 3) We verified the existence and quantified the amount of differences in the depth of nose bending between normal worms and lesioned or mutant animals, which was reported only anecdotally in previous studies. In previous studies, foraging events were scored by human observers which is tedious and labor-intensive. The development of automated methods for the study of foraging behaviors makes it possible to reliably detect foraging events and quantitatively parameterize foraging patterns. The algorithms we developed therefore should have significant utility in future studies of foraging behavior.

Chapter 5 presented a method that combines articulated models and dynamic programming for simulating the body poses of *C. elegans* in multi-worm videos. The models are composed of a number of rectangular parts arranged in a deformable configuration. Dynamic programming is applied to reduce the computational complexity of the matching process. There are two main contributions in this chapter. First, the presented method allows us to identify two worms correctly before and after they touch each other in 90% of our videos. Second, we can use these models to accurately resolve the individual body postures of two worms in physical contact with one another. All joint points in our model can be also considered to be the pseudo skeleton points of the worm body. It solved the problem that a previously presented morphological skeleton-based reversal detection algorithm fails when two worms touch each other.

Bibliography

[1] J. Hodgkin, Male phenotypes and mating efficiency in *Caenorhabditis elegans*, *Genetics*, Vol. 103, pp. 43-64, 1983.

[2] Wormbase, (online material http://www.wormbase.org),

[3] S. Brenner, The genetics of *Caenorhabditis elegans*, *Genetics*, Vol. 77, pp. 71-94, 1974.

[4] J. Baek, P. Cosman, Z. Feng, J. Silver, and W.R. Schafer, Using machine vision to analyze and classify *Caenorhabditis elegans* behavioral phenotypes quantitatively, *Journal of Neuroscience Methods*, Vol. 118, pp. 9-21, 2002.

[5] 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, pp. 5, 2005.

[6] Z. Feng, C.J. Cronin, J.H.W. Jr, P.W. Sternberg, and W.R. Schafer, An imaging system for standardized quantitative analysis of *C. elegans* behavior, *BMC Bioinformatics*, Vol. 5, pp. 115, 2004.

[7] W. Geng, P. Cosman, 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, pp. 1811-1820, 2004.

[8] W. Geng, P. Cosman, J. Baek, C. Berry, and W.R. Schafer, Quantitative classification and natural clustering of *Caenorhabditis elegans* behavioral phenotypes, *Genetics*, Vol. 165, pp. 1117-1126, 2003.

[9] A. Jain, Fundamentals of Digital Image Processing, Prentice-Hall, NJ, 1989.

[10] N.A. Croll, Indolealkylamines in the coordination of nematode behavioral activities, *Canadian Journal of Zoology*, Vol. 53, pp. 894-903, 1975.

[11] N.A. Croll and J.M. Smith, Integrated behaviour in the feeding phase of *Caenorhabditis elegans* (Nematoda),, *Journal of Zoology* Vol. 184, pp. 507-517, 1978.

[12] J. Pierce-Shimomura, T.M. Morse, and S.R. Lockery, The fundamental role of pirouettes in *Caenorhabditis elegans* chemotaxis, *Journal of Neurobiology*, Vol. 19, pp. 9557-9569, 1999.

[13] S.R. Wicks and C.H. Rankin, Integration of mechanosensory stimuli in *Caenorhabditis elegans*, *Journal of Neurobiology*, Vol. 15, pp. 2434-2444, 1995.

[14] C.H. Rankin, C. Chiba, and C. Beck, *Caenorhabditis elegans*: a new model system for the study of learning and memory, *Behavioural Brain Research* Vol. 37, pp. 89-92, 1990.

[15] L.A. Hardaker, E. Singer, R. Kerr, G.T. Zhou, and W.R. Schafer, Serotonin modulates locomotory behavior and coordinates egg-laying and movement in *Caenorhabditis elegans*, *Journal of Neurobiology*, Vol. 49, pp. 304-311, 2001.

[16] M. Driscoll and J. Kaplan, Mechanotransduction, *C. elegans* II, Cold Spring Harbor, NY, 1997.

[17] J.M. Kaplan and H.R. Horvitz, A dual mechanosensory and chemosensory neuron in *Caenorhabditis elegans*, *Proceedings of the National Academy of Sciences USA*, Vol. 90, pp. 2227-2231, 1993.

[18] A.C. Hart, S. Sims, and J.M. Kaplan, Synaptic code for sensory modalities revealed by *C. elegans* GLR-1 glutamate receptor, *Nature*, Vol. 378, pp. 82-85, 1995.

[19] M.J. Alkema, M. Hunter-Ensor, N. Ringstad, and H.R. Horvitz, Tyramine functions independently of octopamine in the Caenorhabditis elegans nervous system, *Neuron*, Vol. 46, pp. 247-260, 2005.

[20] L. Segalat, D.A. Elkes, and J.M. Kaplan, Modulation of serotonin-controlled behaviors by Go in *Caenorhabditis elegans*, *Science*, Vol. 267, pp. 1648-1651, 1995.

[21] P.F. Felzenszwalb and D.P. Huttenlocher, Efficient matching of pictorial structures, *IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head, USA*, pp. 66-73, June, 2000.

[22] P.F. Felzenszwalb and D.P. Huttenlocher, Pictorial structures for object recognition, *International Journal of Computer Vision*, Vol. 61, No. 1, pp. 55-79, 2005.

[23] K. Huang, P. Cosman, and W.R. Schafer, Machine vision based detection of omega bends and reversals in *C. elegans*, *Journal of Neuroscience Methods*, Vol. 158, pp. 323-336, 2006.

[24] R. Gonzalez and R. Woods, Digital Image Processing, 2nd ed, Prentice-Hall, NJ, 2002.

[25] L. Breiman, J. Friedman, R. Olshen, and C. Stone, Classification and Regression Trees, Wadsworth, Belmont, CA, 1984.

[26] R.J. Lewis, An introduction to classification and regression tree (CART) analysis, *The 2000 Annual Meeting of the Society for Academic Emergency Medicine, San Francisco, CA*, 2000.

[27] D. Grossman, Short course in data warehousing and data mining, (online material http://www.ir.iit.edu/~dagr/DataMiningCourse/Spring2001/Notes/Data_Preprocessing.pdf), 2002.

[28] R. Duda, P. Hart, and D. Stork, Pattern classification, 2nd ed, Wiley, NY, 2002.

[29] R. Tibshirani, G. Walther, and T. Hastie, Estimation the number of clusters in a dataset via the Gap statistic, *Journal of Royal Statistical Society Series B*, Vol. 63, pp. 411-423, 2001.

[30] J.M. Gray, J.J. Hill, and C.I. Bargmann, A circuit for navigation in *Caenorhabditis elegans*, *Proceedings of the National Academy of Sciences USA*, Vol. 102, pp. 3184-3191, 2005.

[31] S.A. Glantz, Primer of Biostatistics, 4th ed, McGraw-Hill, NY, 1996.

[32] L. Avery and J.H. Thomas, Feeding and defecation, *C. elegans* II, Cold Spring Harbor, NY, 1997.

[33] W.F. Schafer, Genetics of egg-laying in worms, *Annual Review of Genetics*, Vol. 40, pp. 487-509, 2006.

[34] M.M. Barr and L.R. Garcia, Male mating behavior, WormBook, (online material http://www.wormbook.org/chapters/www_malematingbehavior/malematingbehavior.html), 2006.

[35] B.L. Chen, D.H. Hall, and D.B. Chklovskii, Wiring optimization can relate neuronal structure and function, *Proceedings of the National Academy of Sciences USA*, Vol. 103, pp. 4723-4728, 2006.

[36] J. Karbowski, G. Schindelman, C.J. Cronin, A. Seah, and P.W. Sternberg, Systems level circuit model of *C. elegans* undulatory locomotion: mathematical modeling and molecular genetics, *Journal of Computational Neuroscience , in press*, 2007.

[37] K.S. Kindt, V. Viswanath, L. Macpherson, K. Quast, H. Hu, A. Patapoutian, and W.R. Schafer, *Caenorhabditis elegans* TRPA-1 functions in mechanosensation, *Nature Neuroscience*, Vol. 10, pp. 568-577, 2007.

[38] C.E. Metz, Basic Principles of ROC Analysis, *Seminars in Nuclear Medicine*, Vol. 8, No. 4, pp. 283-298, 1978.

[39] P.W. Welch, The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms, *IEEE Transactions on Audio Electroacoustics*, Vol. 15, No. 2, pp. 70-76, 1967.

[40] D. Manolakis, V. Ingle, and S. Kogon, Statistical and Adaptive Signal Processing, McGraw-Hill, 2000.

[41] M. O'Mahony, Sensory Evaluation of Food: Statistical Methods and Procedures, CRC Press, 1986.

[42] R. Dhawan, D.B. Dusenbery, and P.L. Williams, Comparison of lethality, reproduction, and behavior as toxicological endpoints in the nematode *Caenorhabditis elegans*, *Journal of Toxicology and Environmental Health Part A*, Vol. 58, No. 7, pp. 451-462, 1999.

[43] M.d. Bono, D.M. Tobin, M.W. Davis, L. Avery, and C.I. Bargmann, Social feeding in *Caenorhabditis elegans* is induced by neurons that detect aversive stimuli, *Nature*, Vol. 419, No. 6910, pp. 899-903, 2002.

[44] K.S. Liu and P.W. Sternberg, Sensory regulation of male mating behavior in *Caenorhabditis elegans*, *Neuron*, Vol. 14, No. 1, pp. 79-89, 1995.

[45] T. McInerney and D. Terzopoulos, Deformable models in medical image analysis: a survey, *Medical Image Analysis*, Vol. 1, No. 2, pp. 91-108, 1996.

[46] T.F. Cootes and C.J. Taylor, Statistical models of appearance for medical image analysis and computer vision, *Proceedings of SPIE Medical Imaging*, Vol. 4322, pp. 236-248, 2001.

[47] M. Kass, A. Witkin, and D. Terzopoulos, *Active contour models*, in *International Journal of Computer Vision*. 1987. p. 321-331.

[48] G.L. Scott, The alternative snake – and other animals, *3rd Alvey Vison Conference, Cambridge, England*, pp. 341-347, 1987.

[49] L.H. Staib and J.S. Duncan, Boundary finding with parametrically deformable models, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 11, pp. 1061-1075, 1992.

[50] R. Bajcsy and A. Kovacic, Multi-resolution elastic matching, *Computer Graphics and Image Processing*, Vol. 46, pp. 1-21, 1989.

[51] M.C. Burl, M. Weber, and P. Perona, A probabilistic approach to object recognition using local photometry and global geometry, *European Conference on Computer Vision, Freiburg, Germany*, pp. 628-641, June, 1998.

[52] M.A. Fischler and R.A. Elschlager, The representation and matching of pictorial structures, *IEEE Transactions on Computers*, Vol. 22, No. 1, pp. 67-92, 1973.

[53] D. Hogg, Model based vision: a program to see a walking person, *Image and Vision Computing*, Vol. 1, No. 1, pp. 5-20, 1983.

[54] K. Rohr, Incremental recognition of pedestrians from image sequences, *IEEE Conference on Computer Vision and Pattern Recognition, New York, USA*, pp. 9-13, June, 1993.

[55] J. Deutscher, A. Blake, and I. Reid, Articulated body motion capture by annealed particle filtering, *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 2, pp. 126-133, 2000.

[56] L. Kakadiaris and D. Metaxas, Model-based estimation of 3D human motion, pattern analysis and machine intelligence, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 12, pp. 1453-1459, 2000.

[57] H. Sidenbladh, M.J. Black, and D.J. Fleet, Stochastic tracking of 3D human figures using 2D image motion, *European Conference on Computer Vision, D. Vernon (Ed.), Springer Verlag LNCS 1843, Dublin, Ireland*, pp. 702-718, June, 2000.

[58] Y. Bar-Shalom, T. Fortmann, and M. Scheffe, Joint probabilistic data association for multiple targets in clutter, *Proceedings Conference on Information Sciences and System*, 1980.

[59] M. Isard and J. MacCormick, BraMBLe: A Bayesian Multiple-Blob Tracker, *Proceedings International Conference on Computer Vision*, pp. 34-41, 2001.

[60] J. MacCormick and A. Blake, A probabilistic exclusion principle for tracking multiple objects, *International Journal of Computer Vision*, Vol. 39, No. 1, pp. 57-71, 2000.

[61] S. Ioffe and D.A. Forsyth, Human tracking with mixtures of trees, *International Conference on Computer Vision, Vancouver, Canada*, Vol. 1, pp. 690-695, 2001.

[62] G. Mori and J. Malik, Estimating human body configurations using shape context matching, *European Conference on Computer Vision, Copenhagen, Denmark*, Vol. 3, pp. 666-680, May, 2002.

[63] J. Sullivan and S. Carlsson, Recognizing and tracking human action, *European Conference on Computer Vision, Copenhagen, Denmark*, Vol. 1, pp. 629-644, May, 2002.

[64] A. Mittal and L. Davis, M2 tracker: a multi-view approach to segmenting and tracking people in a cluttered scene, *International Journal of Computer Vision*, Vol. 51, No. 3, pp. 189-203, 2003.

[65] Z. Khan, T. Balch, and F. Dellaert, MCMC-based particle filtering for tracking a variable number of interacting targets, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27, No. 11, pp. 1805-1819, 2005.

[66] W. Qu, D. Schonfeld, and M. Mohamed, Real-time interactively distributed multi-object tracking using a magnetic-inertia potential model, *IEEE Transactions on Multimedia*, Vol. 9, No. 3, pp. 511-519, 2007.

[67] N. Roussel, C.A. Morton, F.P. Finger, and B. Roysam, A computational model for *C. elegans* locomotory behavior: application to multiworm tracking, *IEEE Transactions on Biomedical Engineering*, Vol. 54, No. 10, pp. 1786-1797, 2007.

[68] K. Huang, P. Cosman, and W.R. Schafer, Automated tracking of multiple *C. elegans* with articulated models, *IEEE International Symposium on Biomedical Imaging*, pp. 1240-1243, 2007.

[69] M.A. Trick, A tutorial on dynamic programming, (online material http://mat.gsia.cmu.edu/classes/dynamic/node1.html#SECTION00010000000000000000), 1998.

[70] A. Amini, T. Weymouth, and R. Jain, Using dynamic programming for solving variational problems in Vision, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 9, pp. 855-867, 1990.

[71] E. Angel and R. Bellman, Dynamic programming and partial differential equations, Academic Press, Orlando, FL, 1972.

[72] D.G. Lowe, Fitting parameterized three-dimensional models to images, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, No. 5, pp. 441-450, 1991.