Intelligent Pattern Recognition of Moving Organisms

1. Introduction

In the past two decades, the recognition of *Caenorhabditis elegans*–commonly referred to as *C. elegans*–as a model organism has greatly attracted the attention of the scientific community. Extensive research has been conducted to better understand similarities between the human genome and the genome of this model organism, knowing that they both share more than 90% of genes. The similarities between the two genomes have significantly advanced our understanding of human diseases and provided us with a simple model platform for the discovery of new pharmaceuticals. Major advancement in the area of genomics have led to supercomputers that can handle massive amounts of genomics data, but, unfortunately, our understanding of the behavioral facets of these organisms is limited to manual observation. Besides understanding the crucial interplay of individual genes and gene networks within modern organisms, fundamental understanding of the different behavioral features is important towards building a complete understanding of how genes, proteins, cells, and organs interact inside any organism.

Currently, behavioral data of *C. elegans* is gathered mostly during their movement on agar plates or microscale fluidic chips. During such experiments, simple MATLAB programs identify moving worms and extract standard movement parameters such as velocity and centroid position. In this regard, our project aims to develop intelligent pattern recognition software that facilitates tracking of randomly moving *C. elegans*, extracting intelligent data that accurately identifies and predicts the behavior of given worms within one or multiple genetic strains. This will eventually produce user-friendly data that biologists can use to interpret experimental outcomes without the need of manual qualitative observation.

Several attempts have been made by researchers towards designing an effective automated software platform to track *C. elegans* and extract their behavioral data. The most recent work in this field has been from researchers at MBFBioscience (Vancouver, Canada). Unlike their predecessors, the WormLab (MBFBioscience) software offers a robust workflow in which worm coordinate data are extracted from pre-recorded and real-time videos of *C. elegans*.

Some important contributions in this area of automated worm tracking have come from William Schafer (UCSD), Miriam Goodman (Stanford), and Pamela Cosman (UCSD). Upon review of some of the existing worm tracking programs, we found certain commonalities among them that helped us frame our design goals, which we will outline below.

1. All available programs are written using MATLAB-based platforms.

2. All software was primarily tested on wild-type (with all the naturally occurring genes) *C. elegans*, and extracted the worm centroid coordinates over the length of the video.
3. All software was tested on worms moving on macroscopic agar plates that inherently avoided complex issues during the tracking process, such as collisions within restricted device geometries.

1.1 Design Goals

Our goal is to develop a software solution that facilitates analysis of worm behavior either from single or multiple videos. Initially, the software will extract centroid coordinates of multiple worms over the length of the video. Later, this raw data will be intelligently filtered to identify, interpret, and predict behavioral patterns unique to worms under a given experimental condition.

2. Architecture

2.1 Background

The system is a Windows-based application. It need not interface with other systems, or offer an API, though it may be possible to integrate such features at a later date. At a large scale, it features one main input of a video or videos, and one major output, the extracted data. Selection of the video or videos as well as other user information will be gained through a graphical user interface (GUI), built using an existing library. There are no networking or multi-threading requirements for the project, though multithreading or CUDA technology may be utilized later for an increase in performance.

2.2 Specifications

Our goal is to develop software that uses novel techniques in computer vision. Our program must include the features of the software that already exist in the market and it also must include new features to make our program distinct and advantageous compared to other existing platforms. We are focusing on delivering software that provides more accurate performance, given less user input, as well as contains more features that will provide the user with meaningful data and visual representations for different work tracking scenarios.

After our initial software requirements and specifications are met, we hope to expand our functionality with performance enhancements and more features. Our core specifications can be found below.

2.2.1 Video Support

- The software shall support the following video compression codecs:
    - Microsoft Video 1

  - Intel Indeo
- The software shall support input video frame rates between 1 and 30 frames-per-second

2.2.2 Interface

- The software shall provide a graphical user interface that allows the user to select worms and track data only for the selected worm(s)
- The interface shall prompt the user to enter video parameters
- The interface shall support batch processing of multiple video files

2.2.3 Tracking

- The tracking system shall identify *C. Elegans* in pre-recorded videos with a 99% success rate, as confirmed by human judgment of the video
- The tracking system shall filter out background noise that could be incorrectly identified as a worm, including, but not limited to, walls and shadows
- The software shall provide a confidence interval for worm identification after the event of a worm collision based on the frame rate.

2.2.4 Data

- The analysis system shall provide x-y coordinates of worm centroid, head, and tail over the duration of a given video
- The analysis system shall be able to derive velocity and acceleration from the mentioned worm data
- The analysis position shall be extrapolated as a spline vector for each frame, with interpolations given between frames
- The analysis system shall draw worm location and movement graphically on the video frame after it has been processed
- The analysis system shall identify each selected worm's state of locomotion over time

2.2.5 Non-Functional Specifications

- The software shall process video at no less then 1 GB of video data per minute
- The software and any support files shall be wrapped into a single installer
- The GUI shall be intuitive and functional

2.2.6 Coding Style

- In order to build maintainable software, the code shall adhere to coding guidelines
- The guidelines are outlined here: http://code.google.com/p/isu-ecpe491-1203/wiki/CodingStyle

2.3 Data

The large-scale data flow of our system is relatively simple. Video, saved on disk and provided by user input, is streamed into memory, where it goes through a series of algorithms, each either preparing the video for other algorithms, or extracting data from the video. The data is saved within its own class for every video and is then exportable via user interface.

For the input data video, it must be in the format specified in the requirements, with a desire for a wider variety of file-types if possible. Bi-products of the algorithmic processes may be stored on disc temporarily, in a given video or image format, with an option to delete them when finished analyzing. Data shall be provided in a comma-separated value (CSV) file in early prototypes, but may shift to a more encompassing file type at a later date.

2.4 Modules

2.4.1 Worm Video

The worm video encapsulates pertinent constructs and represents a single video of *C. elegans*. This is the primary interface for processing the videos and extracting meaningful behavioral data from worms.

2.4.2 Background Subtraction

The background subtraction module performs all of the segmentation and separation of worms from the background of a video. This unit encapsulates several useful algorithms for removing noise and background pixels:

- Unimodal Analysis
    - Mean
    - Mode
    - Covariance
- Codebooks and Connected Components
- Principal Component Analysis

2.4.3 Centroid Finder

The centroid finder module makes use of a data in which worms have already been separated from their background and identifies centroids of each worm across the video. It achieves this by means of a centroid detection algorithm:

- Convert image to gray and blur frame-by-frame
- Detect edges
- Find contours of each detected object

- Get moment of each detected object
- Find mass center with moment analysis

2.4.4 Centroid

The centroid module is a containment structure used to represent a singular worms centroid information across a video.

2.5 Interfaces

Current prototypes require the user to interact with the program by calling it from the command line with the name of the video to be processed. In future iterations we plan to strip this command line interface away entirely and instead create a graphical user interface that will enable the user to process and analyze data in a way that is intuitive.

3. Operation

3.1 User types

The software shall be used by lab technicians to analyze worm videos. The return data format will be the same regardless of video type, and so the number of use-cases is limited. User types such as administrator are unnecessary.

3.2 Scenarios

Within our normal operating environment by a lab-technician, there are many different scenarios presented for the software to handle. Worms' movement can occur in an infinite number of ways, in which worms may collide, stop moving, reverse direction, enter frame, leave frame, die, or any number of other possibilities. To deal with these eventualities, the software will assess its confidence in its assessment of videos and shall provide the user with feedback according to its confidence level, in such a way that ensures reasonable data collection.

3.3 Installation

The project will be self-contained in a single *.msi installation file. It shall install all necessary *.dlls, *.libs, and other files during the installation process. The installation is only for Windows 7 environment. An uninstall of the software should be possible through the Windows 7 Control Panel. The uninstall process should remove all traces of the application, save dlls and static libraries.

3.4 Licensing

One of the tangible goals put forth by the client is a research general publication in the next semester. During the review process and subsequent publication, we plan to approach ISURF through our client for submitting a pre-application for a patent through Iowa State University.

3.5 Upgrades

In order to increase performance, it may be possible to accelerate algorithms through the use of the OpenCL or CUDA platforms. However, these platforms are dependent upon the graphics-card manufacturer, and would only be available for certain users.

Additional interfaces or API's may be implemented so that other applications may interface with ours, but this is currently not a priority.

4. Development

Currently we have an initial prototype of tracking single worms from pre-recorded videos. As a next step, we plan to modify this prototype to enable the tracking of multiple worms with more accuracy. Towards making the computation more effective and organized, we plan to shift towards a model-based approach for tracking these randomly-moving objects. We plan to establish a well-defined workflow within which to intelligently analyze data and filter meaningful behavioral patterns.

During the next semester, we will develop software prototypes under an iterative process to improve the robustness and accuracy with which we identify, interpret, and predict behavioral patterns. Unlike testing the developed software merely on worms in agar plates, we plan to prove the robustness of the application when analyzing worms in varied physical conditions, such as in microscale fluidic chips.

Furthermore, a graphical user interface will be created for interfacing with the software solution that would allow the user to automatically upload multiple videos, select desired worms to analyze, and that allow them to define the behavioral parameters to be displayed.

5. Resources

Links to related papers, as well as other information about the project can be found at the project wiki: http://code.google.com/p/isu-ecpe491-1203/w/list

6. Team

Project Advisor and Client:

Dr. Santosh Pandey
Assistant Professor
Electrical and Computer Engineering
pandey@iastate.edu

Graduate Supervisor

Roy Lycke
Graduate Assistant-Researcher
Electrical and Computer Engineering
rylycke@iastate.edu

Team Members

Ryan Alley
Team Leader
Computer Engineering
rtalley@iastate.edu

Colin Ray
Communicator
Electrical Engineering
rcray@iastate.edu

Laith Abbas
Webmaster
Computer Engineering
laitha0@iastate.edu

Shan Zhong
Electrical Engineering
szhong@iastate.edu

Shusheng Xu
Electrical Engineering
shusheng@iastate.edu