Distribution Agreement

In presenting this thesis as a partial fulfillment of the requirements for a degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis in whole or in part in all forms of media, now or hereafter now, including display on the World Wide Web. I understand that I may select some access restrictions as part of the online submission of this thesis. I retain all ownership rights to the copyright of the thesis. I also retain the right to use in future works (such as articles or books) all or part of this thesis.

Xinyue An April 12, 2022

Using Deep-Learning Based Approaches to Quantify *Drosophila* Behaviors

By

Xinyue An

Gordon Berman
Adviser

Neuroscience and Behavioral Biology

Gordon Berman
Adviser

Anita Devineni
Committee Member

Davide Fossati
Committee Member

Astrid Prinz
Committee Member

2022

Using Deep-Learning Based Approaches to Quantify *Drosophila* Behaviors

By

Xinyue An

Gordon Berman

Adviser

An abstract of
a thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Science with Highest Honors

Neuroscience and Behavioral Biology

2022

Abstract

Using Deep-Learning Based Approaches to Quantify *Drosophila* Behaviors
By Xinyue An

Deep learning is an emergent theme in the field of computational neuroscience, and the increasing amount of behavioral data in the video form calls for better tools to quantify behaviors. Building on previous research by Cande et al. (2018) that used an unsupervised method to quantify *Drosophila* behaviors, we investigate the incorporation of a new animal tracking tool and an autoencoder, both deep-learning-based methods, to define animal behaviors with greater precision and accuracy. Comparing to published results, the behavioral representation from the new analysis pipeline is able to reproduce many aspects of the previous work but has limitations that require further investigation. These results show promising future directions towards linking behavior and the neural circuitry underlying it.

Using Deep-Learning Based Approaches to Quantify *Drosophila* Behaviors

By

Xinyue An

Gordon Berman

Adviser

A thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Science with Highest Honors

Neuroscience and Behavioral Biology

2022

Acknowledgements

I want to give my huge thanks to Dr. Berman for the opportunity he gave and all the weekly meetings and discussions we had. I am thankful for having him as my thesis advisor for his patience, kindness, and positivity, and of course his continuous guidance and support for me in all aspects. I also want to thank Kanishk Jain for always being there through the entire process, and I would not be able to overcome all the difficulties without his help.

I want to thank my committee members Dr. Prinz, Dr. Fossati, and Dr. Devineni. This is a starting point and an important milestone in my academic path. I am fortunate to have them chairing my committee, providing feedback and comments, and recognizing my work and efforts.

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Significance of Descending Neurons in Animal Motor Control

The highly conserved motor control system in animals composes of both a brain, which integrates contextual information and transmits motor commands, as well as an effective bottleneck between the brain and the circuitry that directly generates movements. In vertebrates, this bottleneck is the spinal cord, and in most arthropods, it is the ventral nerve cord (VNC). In arthropods, descending neurons (DNs) are a population of interneurons that connects the brain and the VNC by sending down motor commands, but they only take a small portion of the whole neuronal population.

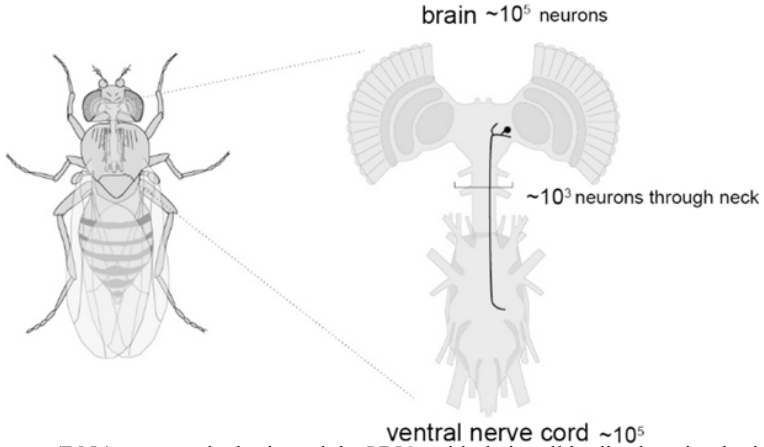
How descending neurons convey sufficient information about motor commands within their limited bandwidth remains unclear. One possibility is that the set of DNs decomposes the motor control signals from the brain to innervate individual motor modules in the central pattern generators of the VNC. If so, the activation of individual DNs should be sufficient to elicit observable motor behaviors, which is supported by previous studies (Kohatsu et al., 2011; von Philipsborn et al., 2011; von Reyn et al., 2014; Bidaye et al., 2014). However, previous studies also found that DNs may code behaviors in a combined or summed manner (Heinrich, 2002; Namiki et al., 2018), and the responses of DNs can have behaviorally dependent gating (Staudacher & Schildberger, 1998).

To understand how descending neurons code the full spectrum of animal behaviors and model the neural principle governing animal motor control, it is necessary to link all individual descending neurons' activity with its motor output. Toward that end, we would need a subjective and accurate measure of the animal's motor behaviors.

1.2 *Drosophila* Descending Neurons as the Model System

Figure 1

Drosophila Central Nervous System Composes of the Brain and the Ventral Nerve Cord



Note. Descending neurons (DNs) connect the brain and the VNC, with their cell bodies locating in the brain and axons projecting to VNC. An example of DN is drawn in black. From Death-Feigning in Insects (1st ed., p. 145-157), by S. Namiki, 2021, Springer. Copyright 2021 by Springer.

We are interested in understanding how descending neurons function as a set of neurons, which involves a system-wide analysis of descending neurons' activation phenotypes. *Drosophila melanogaster* is a good model system for this type of study, because (1) *Drosophila* is genetically tractable, with at most 550 pairs of descending neurons connecting the brain and the VNC (Hsu and Bhandawat, 2016; see Figure 1), and (2) with the library of split-GAL4 lines created by Namiki et al. (2018), individual *Drosophila* DNs became genetically accessible to external manipulation. The genetic tools, combined with a method to measure animal motor behaviors, together could allow researchers to decode the functional organization of descending neurons.

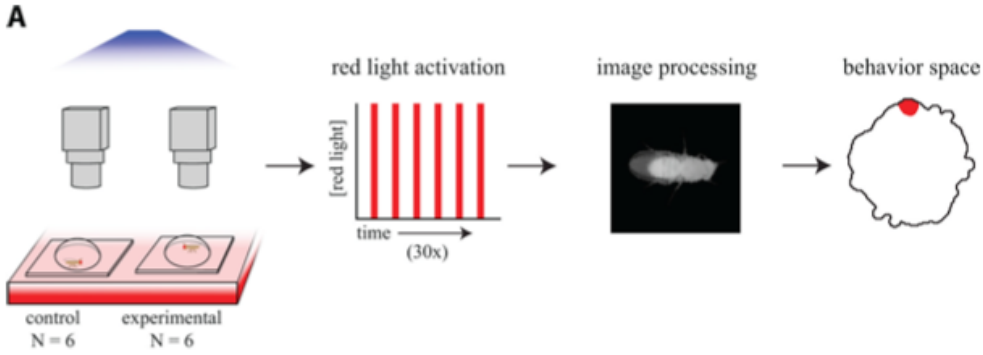
1.3 Using Optogenetics to Study Drosophila Descending Motor Control

While traditional approaches for studying fly behaviors involves human annotating of stereotyped behaviors, such as "front leg grooming", to figure the rules that govern behavior sequences (Dawkins & Dawkins, 1976), recent advancements in experimental tools empower researchers to achieve the same end with computational tools. In Cande et al. "Optogenetic dissection of descending behavioral control in *Drosophila*" (2018), authors demonstrated a novel experimental analysis pipeline that surveys the

activation effects of descending neurons in a system scale with unsupervised computational methods (see Figure 2).

Figure 2

Descending Neuron Phenotyping Pipeline

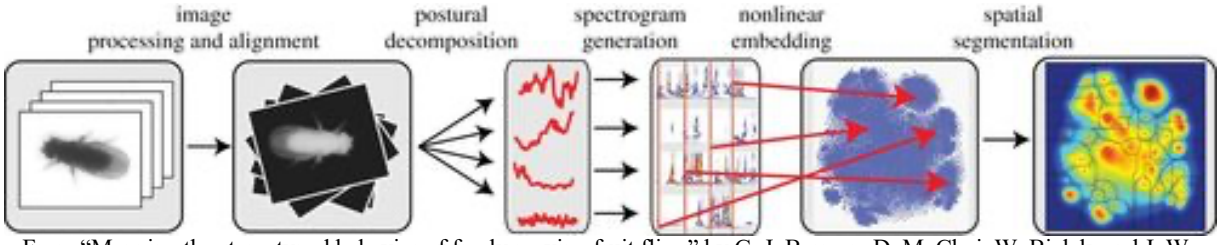


Note. From "Optogenetic dissection of descending behavioral control in *Drosophila*," by J. Cande, S. Namiki, J. Qiu, W. Korff, G. M. Card, J. W. Shaevitz, D. L. Stern, and G. J. Berman, 2018, *eLife*, 7, e34275 (http://doi.org/10.7554/eLife.34275). Copyright 2018 by eLife.

Using the split-GAL4 fly lines developed by Namiki et al. (2018), the authors crossed each line to enable neuronal activation with red light via CsChrimson, a red-shifted opsin protein (Klapoetke et al., 2014). They imaged flies that move freely in a chamber, applied red light activation during the recording, and processed images to phenotype the descending neurons labeled by each split-GAL4 line. To identify and define the behavioral phenotypes without human bias, they decomposed video images into low-dimensional data, producing a time-series that is then mapped onto a 'behavioral space' that can quantitatively define fly behaviors.

Figure 3

Fly Behavioral Analysis Pipeline



Note. From "Mapping the stereotyped behavior of freely moving fruit flies," by G. J. Berman, D. M. Choi, W. Bialek, and J. W. Shaevitz, 2014, *Journal of the Royal Society, Interface,* 11(99), 20140672. (https://doi.org/10.1098/rsif.2014.0672). Copyright 2014 by the Royal Society.

The image processing and the behavioral mapping of processed images utilize the technique developed by Berman et al. (2014). In this analysis pipeline (Figure 3), images are first segmented to detect the outline of the fly, re-scaled and rotationally aligned so that the fly has a uniform body size with the same orientation in each image. Normalized images are then converted to 50-dimensional vectors by a principal component analysis (PCA) on the pixel values to describe the fly's postural modes. To obtain dynamic information about the postural modes, they decomposed those time series data into a spectrogram representation with Morlet wavelet transform so to extract changing patterns of vectors in the time-frequency domain and represent animal behaviors. The spectral feature vectors are then embedded into a behavior space by the dimensionality reduction of the t-SNE algorithm and spatially segmented with watershed transformation – each behavioral region represents a distinguishable set of postural dynamics.

In Cande et al.'s phenotyping pipeline, when descending neurons are photo-activated, observable behaviors are performed by the flies, leading to a reduction of entropy in the behavior space. The elicited behaviors are thus quantitatively characterized by which regions in the behavior space are occupied upon photo-activation and measured by the density change of behavior space. The analysis pipeline allows them to process a huge amount of video data and eliminate the impact of human bias in measuring behaviors, so they were able to test several hypotheses on descending motor control as well as construct a systematic report of 58 descending neuron cell types' behavioral effects.

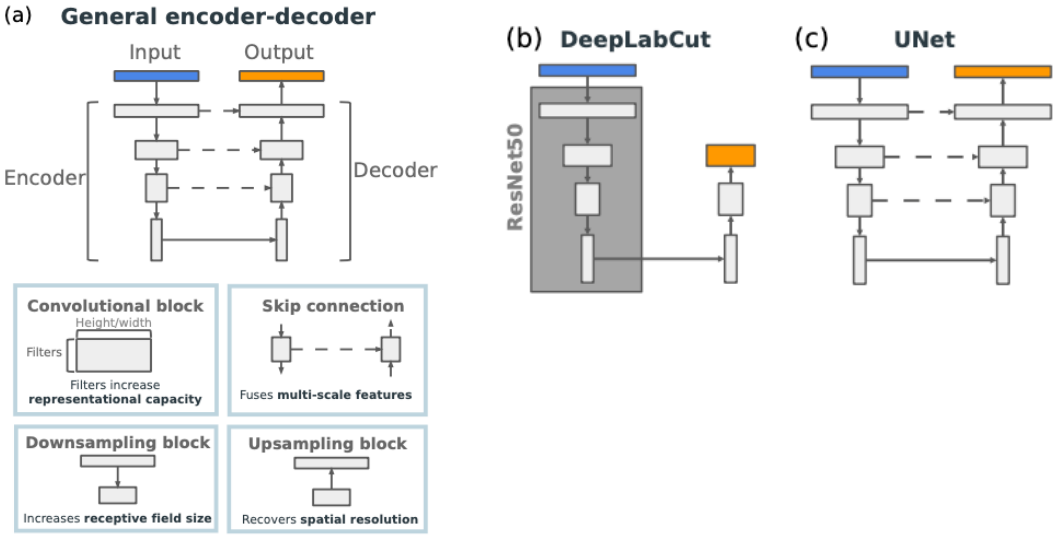
1.4 Recent Advances in Experimental Analysis Tool and Application

One potential limitation in Cande et al. (2018) exists in the image processing technique, in which they define flies' postural modes with the discriminative features in video frames' pixel values to then estimate postural dynamics. A more intuitive way is to decompose images with respect to the fly's anatomical body parts and describe postural modes as the angles between each pair of the body parts, but reliably identifying and tracking animal positions remained a computer vision challenge at the time of their study. The new computational methods in the field, however, allow an explicit tracking of joints and

limbs throughout videos, thus potentially providing a finer-scale representation of behaviors based on postural dynamics.

Figure 4

Neural Network Architectures Generated by the Encoder-Decoder Motif



Note. From "SLEAP: A deep learning system for multi-animal pose tracking," by T. D. Pereira, N. Tabris, A. Matsliah, *et al.,* 2022, *Nat Methods.* (https://doi.org/10.1038/s41592-022-01426-1). Copyright 2022 by Nature Methods.

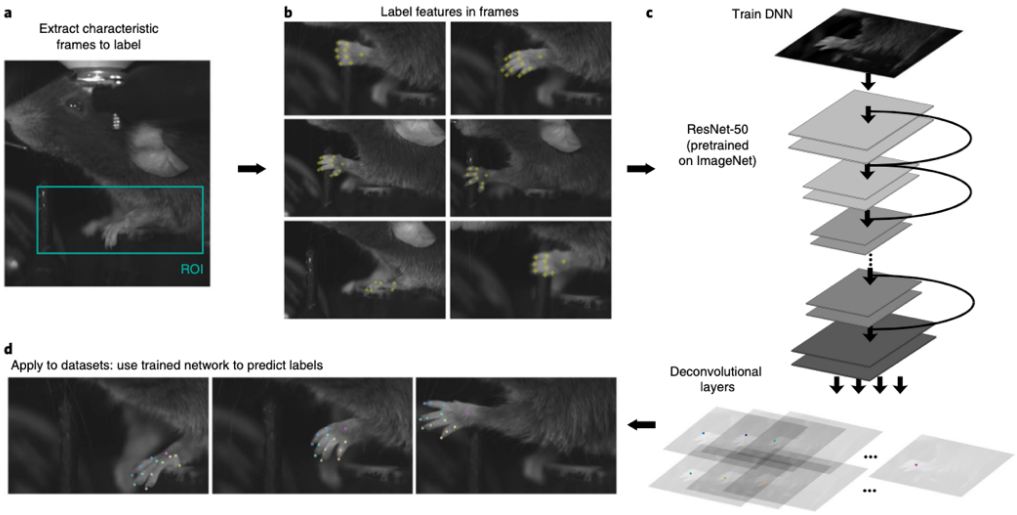
DeepLabCut and SLEAP are two of the most widely used toolboxes that track animal poses in an experimental setting with computer programs (Mathis et al., 2018; Pereira et al., 2020). With computer vision and deep learning algorithms, they both can reliably infer animals' body-part positions from surface video recordings without physical markers that constrain animals' behaviors. The general neural network architectures can be explained by the encoder-decoder meta-architecture (Figure 4.a). The "encoder" has a contracting path to extract relevant information about the labeled body parts from images, and the "decoder" has an expansive path to combine information and recover the input. Video frames with manually labeled body parts train the weights of the network to then predict the body-part locations in an unseen frame. Under the same meta-architecture, specific backbones serve to learn feature extraction with different approaches.

DeepLabCut (Mathis et al., 2018; Figure 5) extends from the feature detectors of DeeperCut (Insafutdinov et al., 2016), a multi-human pose estimation model, and adopts transfer learning toward the

task of animal pose estimation. Its architecture composes of (1) an encoder as a variation of deep residual neural networks (ResNet) that is pretrained on ImageNet to achieve the object recognition benchmark, and (2) a decoder that, in substitution of the object classification layer, constitutes of a readout per body part that has a confidence map of the body part's location.

Figure 5

Procedure for Using the DeepLabCut Toolbox



Note. From "DeepLabCut: markerless pose estimation of user-defined body parts with deep learning," by A. Mathis, P. Mamidanna, K. M. Cury, *et al.,* 2018, *Nat Neurosci* 21, 1281–1289 (https://doi.org/10.1038/s41593-018-0209-y). Copyright 2018 by Nature Neuroscience.

While DeepLabCut implements the ResNet backbone for feature extraction, SLEAP (Pereira et al., 2020) allows implementation with two different backbones: ResNet, as a replicate of the architecture in DeepLabCut (Figure 4.b), and UNet (Figure 4.c). UNets are commonly used in biomedical image segmentation tasks, and its network architecture has (1) an encoder that increases feature information of the images and decreases spatial information with convolutional layers, and (2) a decoder that uses up-convolutions to recover feature information and fuses with spatial information from the encoder to combine into the output – the confidence map of body parts' locations.

With the same training data, SLEAP can implement training with both ResNet and UNet backbones to compare for accuracy. Furthermore, SLEAP has a "human-in-the-loop" GUI-driven training style that expedites the workflows and demands fewer computation and time. After initial labeling of

minimal frames, predictions are incrementally refined and guide further labeling until desired accuracy is reached. Those features of SLEAP make it more desired in the execution of this project.

1.5 Hypotheses

The raw output of SLEAP composes of tracked body parts' coordinates in the video frames, which does not directly describe animal behaviors. To estimate animal behaviors with the tracking data, Pereira et al. (2019) calculated the relative position of each body part to the fly thorax and generate spectrograms of the trajectories of body-part positions. After embedding the feature vectors of spectrograms to the behavior space with the method by Berman et al. (2014), the tracking data can estimate the fly's body dynamics and describe locomotor behaviors. Extending from here, this project presents a new methodology that processes the video tracking data into time-series joint angles to estimate animal behaviors.

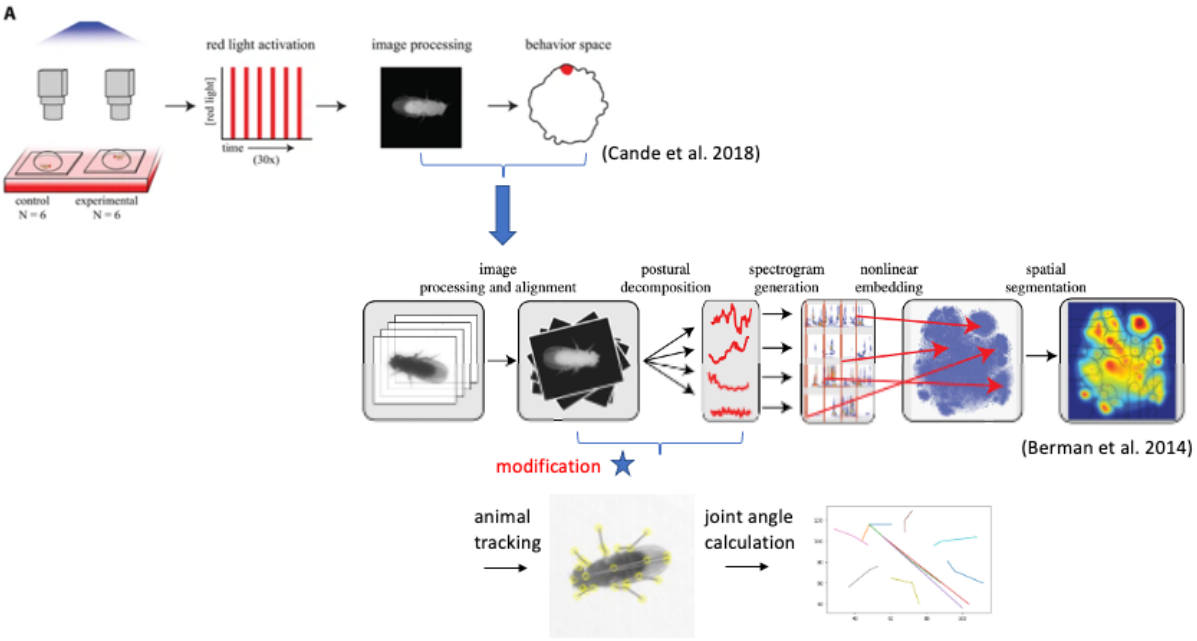
Our hypothesis is that the surface tracking of animal joints and the calculated joint angles, at least for arthropods such as flies, entails their actual skeletal kinetics and thus locomotor behaviors. Previous studies presented the complications with inferring rodents' limb kinematics from video tracking because of their overlying soft tissues' movements (Bauman and Chang, 2010). Comparing to vertebrates, fruit flies have relatively unobscured limbs and minimal soft tissues, so we might overcome such challenge, and the success of it was demonstrated by Pereira et al. (2019). With our attempt, we wish to validate the accuracy of deep learning tracking methods in the use toward measuring animal behaviors, and we hypothesize that the time-series data of joint angles are sufficient to capture flies' motor behaviors.

To validate the efficacy of our animal video processing methods, we substitute a section of the Cande et al.'s (2018) analysis pipeline and aim to re-analyze their original video data (Figure 6). We hypothesize to yield similar results as Cande et al.'s (2018) major findings, including: (1) activation of most descending neurons leads to stereotyped behaviors in flies, (2) activation of multiple DNs have similar behavioral output, and (3) behaviors activated by optogenetic stimulation depend on the prior behavioral state. We also hope to explore potential

new findings that were previously masked by the limited specificity in previous image processing techniques. With time-series data of joint angles, our findings could move beyond a coarse description of body dynamics and in the scale of specific body parts' kinematics.

Figure 6

Modified Descending Neuron Phenotyping Pipeline



Note. The top figure is from "Optogenetic dissection of descending behavioral control in *Drosophila*," by J. Cande, S. Namiki, J. Qiu, W. Korff, G. M. Card, J. W. Shaevitz, D. L. Stern, and G. J. Berman, 2018, *eLife*, 7, e34275 (http://doi.org/10.7554/eLife.34275). Copyright 2018 by eLife. The middle figure is from "Mapping the stereotyped behavior of freely moving fruit flies," by G. J. Berman, D. M. Choi, W. Bialek, and J. W. Shaevitz, 2014, *Journal of the Royal Society, Interface,* 11(99), 20140672. (https://doi.org/10.1098/rsif.2014.0672). Copyright 2014 by the Royal Society.

2. Methodology

2.1 Dataset

With the split-GAL4 intersectional system, Namiki et al. (2018) created a collection of transgenic *Drosophila* strains that target descending neurons. Using this collection of strains, Cande et al. (2018) screened 130 of the sparsest line and crossed them with the channel rhodopsin CsChrimson to enable red light activation of the targeted descending neurons. For each fly line, they imaged six retinal-fed experimental animals and six non-retinal-fed control animals and recorded their behaviors from the activation of descending neuron split-GAL4 lines. (For detailed data collection methods, see Cande et al.

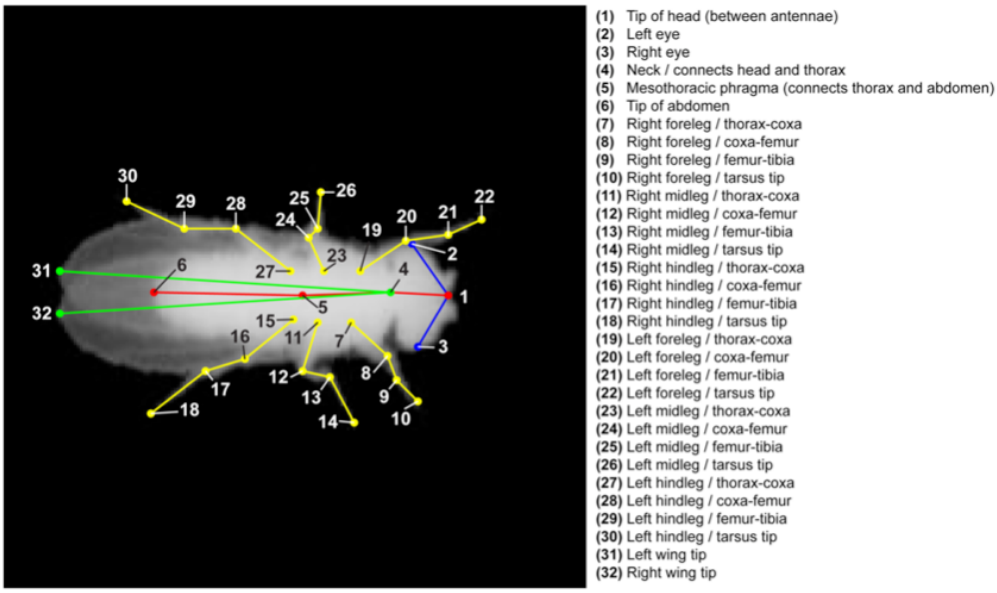
2018) Each video recorded for the animal has 30 LED cycles each composes of 15 seconds LED-on and 45 seconds LED-off programs.

The dataset for this project is a subset of the videos collected by Cande et al. (2018), and I screened 15 out of the 130 descending neuron split-GAL4 lines. In total, I processed 173 fly videos that are in total approximately 30 million frames long.

2.2 SLEAP Model for Animal Tracking

Figure 7

User-Defined Skeleton for Fly



Note. From "Fast animal pose estimation using deep neural networks," by T. D. Pereira, D. E. Aldarondo, L. Willmore, *et al.,* 2019, *Nat Methods* 16, 117–125. (https://doi.org/10.1038/s41592-018-0234-5). Copyright 2019 by Nature Methods.

The fly's skeleton (Figure 7) for labeling and tracking adopts the skeleton used in the original method paper by Pereira et al. (2019). The complete skeleton consists of 32 total nodes (joints or body parts) that are connected by 25 edges.

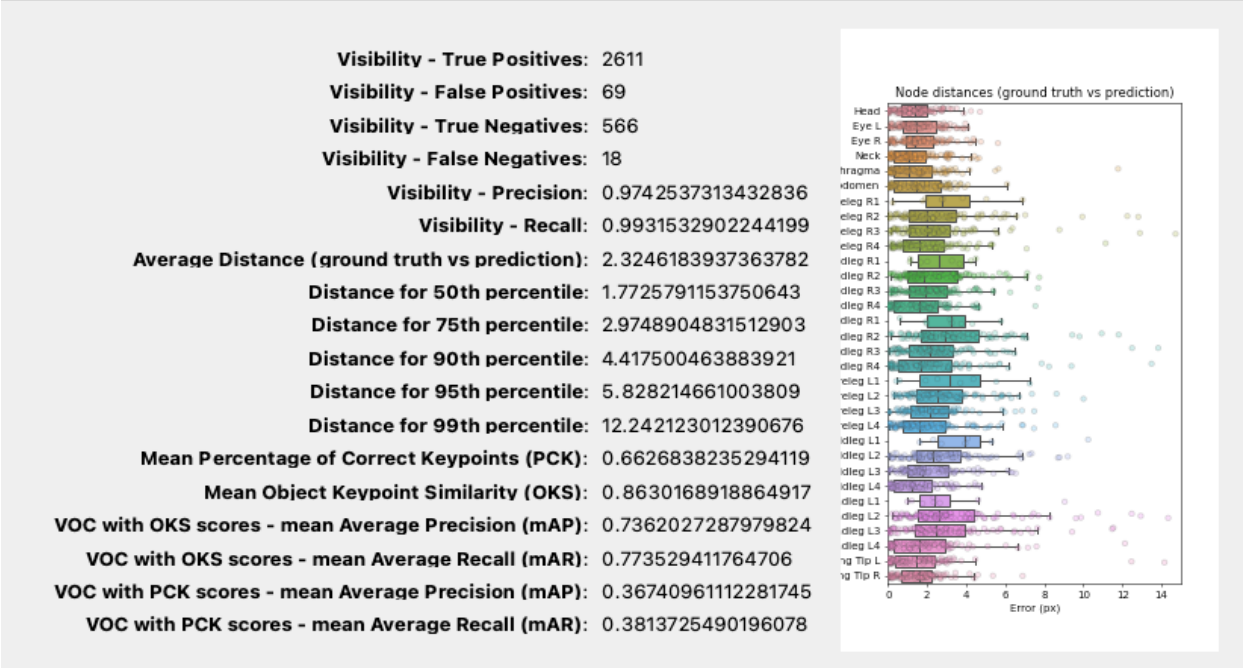
The training cycle started with 100 frames of initial labeling and continued until desired labeling accuracy is achieved. The trained model in each stage was iteratively used to assist further labeling in the next stage. By the end of the training cycle, 1000 frames were labeled, sampled from 40 randomly chosen videos and 25 frames came from each video. The labeled frames were primarily chosen by random sampling. In the last training stage, to achieve a more robust model that tackles the tracking of difficult

frames, labeling frames were both chosen by the image feature method from SLEAP (algorithms that suggest visually distinctive frames for labeling) and manually chosen (frames of flies' grooming behaviors that require higher resolution and were previously poorly tracked by the model).

Since SLEAP supports training with different backbones, I was able to train the labeled images on both the ResNet50 backbone (with pretrained weights on ImageNet) and the UNet backbone (baseline) to compare for the tracking accuracy. Training profiles with hyperparameters of the training process are created by SLEAP and can be found on SLEAP's GitHub page.

Figure 8

Training Results with the Pretrained Training Profile (ResNet50)



Note. This figure includes the output from using "SLEAP: A deep learning system for multi-animal pose tracking," by T. D. Pereira, N. Tabris, A. Matsliah, *et al.,* 2022, *Nat Methods.* (https://doi.org/10.1038/s41592-022-01426-1).

The ResNet50 backbone was chosen because it has a superior accuracy for every accuracy metric, including lower percentile distances and higher mean Average Precision and mean Average Recall. The final SLEAP model trained on ResNet50 has an average distance of 2.32 pixels for each tracked node across all video frames, with <6 pixels distance for 95th percentile of the frames (Figure 8). The average pixel distance (~2 px) is relatively small comparing to the body length of flies in video frames which is

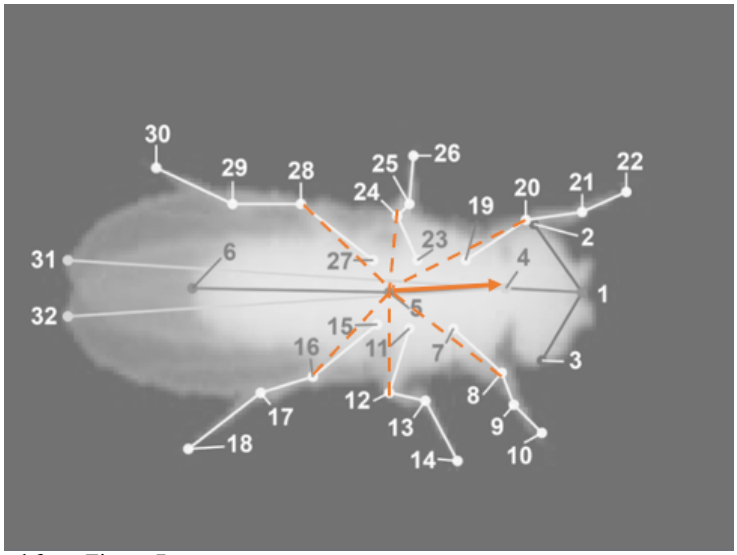
~100 pixels. The distribution of the node distance suggests that the model can predict most of the frames well, but not as well for some difficult frames. The errors resulted from tracking will be further addressed by the following methods in Section 2.4.

2.3 Joint Angle Calculation

The output shape of SLEAP is a time series of tracked node positions (x, y coordinates) throughout the video. With 32 tracked nodes, the representation of the fly's pose in each frame is a 64-dimensional data. Converting the joint-position representation into the joint-angle representation of animal poses is beneficial for several reasons: (1) joint angles are independent from the fly's body orientation and size, so they can act as a measure to standardize the data, (2) the joint angle calculation method that I apply here reduces the dimensionality of the data from 64 dimensions to 22 dimensions, which preserves the shape of the fly skeleton yet demands fewer computation power, (3) joint angles present more human interpretable data, so the results could be used for more potential applications and lead future research directions.

Figure 9

Description of the Joint Angle Calculation Method on the Fly's Skeleton



Note. This figure is adapted from Figure 7.

The objective for the joint angle conversion is to account for and faithfully represent the fly's pose in each frame by calculating enough yet minimal number of angles. Toward that end, angles that are

fixed and independent from the fly's postural dynamics are not included in the calculation (e.g., the angles between the eyes' and the head's positions, formed by nodes 2-1-4 and 3-1-4 in Figure 9).

The detailed method for joint angle calculation is as follows. For each of the fly's six limbs (e.g., nodes 7 to 10 represent the fly's right forelimb), two angles around each joint ought to be calculated (e.g., angles formed by nodes 10-9-8 and 9-8-7). However, six proximal joints tend to be obscured by the fly's trunk and thus poorly tracked by the SLEAP model (nodes 7, 11, 15, 19, 23, 27). Because the proximal joints have relatively fixed positions in the fly's body, an alternative solution is to use another fixed node, the methoracic phragma (node 5), to substitute the proximal joints (e.g., calculate angles formed by nodes 10-9-8 and 9-8-5, instead of 9-8-7).

In a fly, the thorax connects all other body parts of the fly including three pairs of legs, two wings, head, and abdomen, thus acting as a pivot in the fly's anatomy. Taking the thorax as a pivot, I choose the line formed by node 4 and 5 (solid arrowed line in Figure 9) as a base axis of the fly and calculate all other body parts' angles with it. For example, the angle formed by nodes 31-4-5 accounts the left wing's relative position, and the angle formed by 1-4-5 accounts for the head's relative position. Taken together, a total of 22 joint angles are calculated from each image to approximate the fly's pose.

The discontinuity in an angle's magnitude during a brief time period can disrupt the approximated postural dynamics measured by the following steps. Thus, according to the range of each angle, I adjusted it to either reside in the range of $[-\pi, \pi]$ or $[0, 2\pi]$ to make the joint angle time series maximally continuous during the fly's natural behaviors. For example, avoiding a jump of an angle's magnitude from $-\pi$ to π between two consecutive frames when the fly is performing a continuous movement.

2.4 Autoencoder and Median Filter for Data Denoising

The time series of joint angles will then pass through an autoencoder and a median filter to denoise the data and reduce errors from tracking, as will be described below.

2.4.1 Autoencoder Training

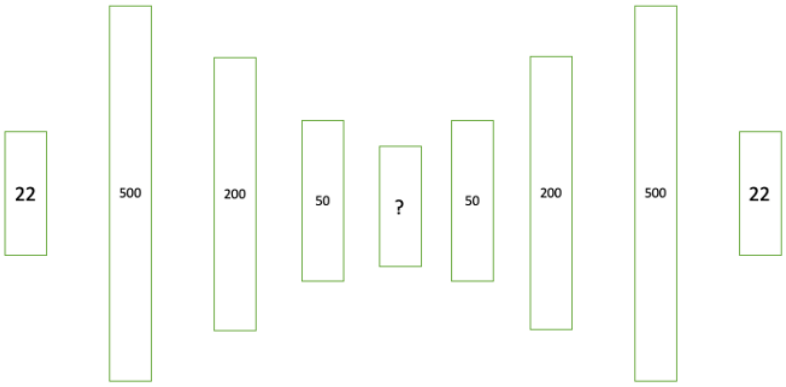
An autoencoder is an unsupervised artificial neural network that can be trained to produce an output that is a replicate of its input. A stacked autoencoder composes of an encoder that compresses the

input data into a low dimensional representation (encoding dimension), and a decoder that reconstructs the original data from the low dimensional representation. The weights of the network are iteratively trained through training epochs for the output to maximally resemble the input.

For this dataset, an autoencoder is especially useful because some of the joint positions might be falsely tracked or not tracked by the SLEAP model, leading to missing or error values in the 22 joint angles calculated. Thus, a denoising autoencoder structure is adopted here to correct those data (Figure 10).

Figure 10

Stacked Autoencoder Structure



For the autoencoder training, I randomly sampled 260 frames from each of the 173 videos that have all 22 joint angles calculated, totaling approximately 45,000 frames, which should be sufficient to capture most of the possible fly poses. All data points have a shape of [22,] for 22 joint angles. The dataset is randomly shuffled and split into the training dataset and the test dataset by 40,000 : 5,000. Before inputting into the autoencoder, each joint angle is adjusted to the range [0, 1] with min-max normalization for better computation of the activation function in the autoencoder. Min-max normalization uses the formula:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

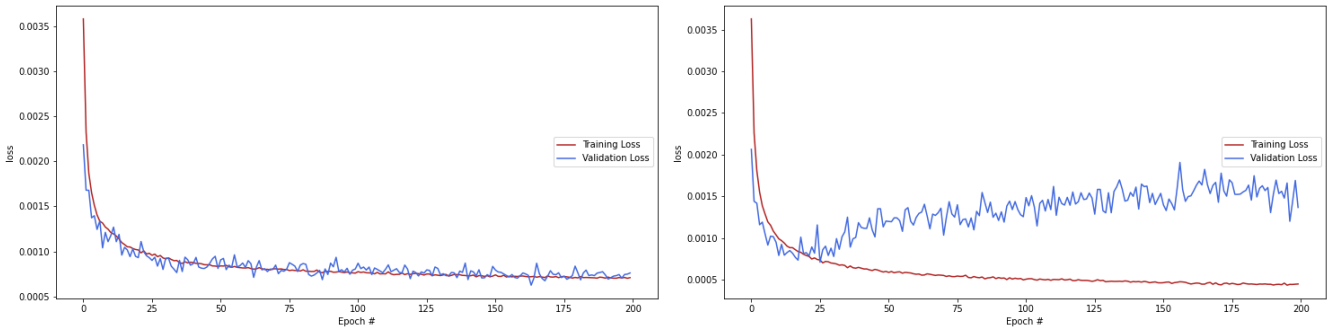
where the maximum and minimum value of a given joint angle is found across 45,000 frames and adjusted up or down by 10 percent to account for possible actual values out of the range of the sampled frames.

The input layer of the autoencoder has 22 joint angles calculated from a fly's image. Between the input layer and the first layer of the encoder, a dropout layer of 0.1 is added to simulate a partially destroyed input (10% of the input data is corrupted). Every layer in the neural network is a dense layer, which means every single neuron receives output from every neuron in its preceding layer. By combining results from the preceding layer, a dense layer is often used to change the dimension of the vectors yet retain information. Each layer before the output layer uses the Scaled Exponential Linear Unit (SELU) activation function that can give a faster convergence of the network. The output layer uses the Sigmoid function to scale the result between 0 and 1 and reconstruct the input.

By iterative training on the training dataset, the network will correct its weights and learn to reconstruct the original input in an unsupervised manner despite of the destroyed input. Sizes of the encoder layers (n = 500, 200, 50) are arbitrarily chosen. The idea is to first expand the low-dimensional input and gather enough information about the input, and gradually compress into the latent space representation. The decoder has a symmetrical architecture to reconstruct the input. The key of the autoencoder architecture is the size of the encoding dimension, so I experiment to find the best suited size for this specific dataset.

Figure 11

Training Curves of Example Models



Note. (a) Training plots for encoding dimension = 9, batch size = 128, epochs = 200. (b) Training plots for encoding dimension = 16, batch size = 128, epochs = 200.

The loss function of the network is the mean squared error between the output and the input, and Adam optimizer is used in training. Parameters of the model are constantly updated by the optimizer

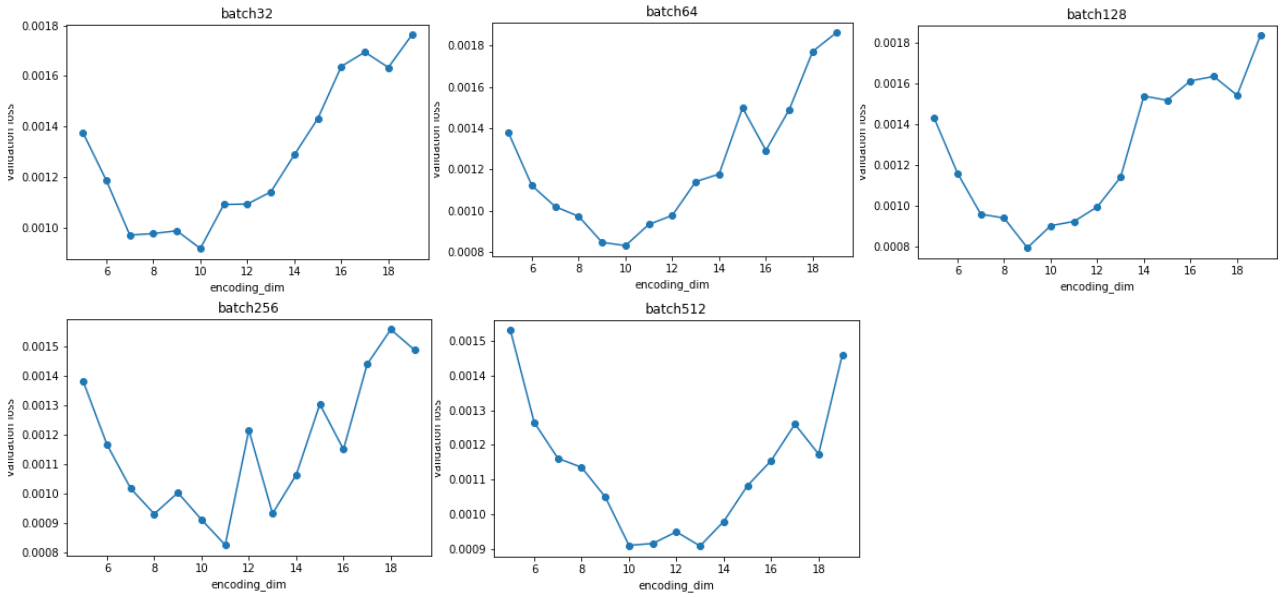
through training, and I experiment to choose hyperparameters for training, including the batch size of training and the number of training epochs.

To choose the most appropriate size of the encoding dimension, I trained the autoencoder on a combination of varying encoding dimension size (in the range of 5 to 20) and batch size (n = 32, 64, 128, 256, 512) for 200 epochs. A good autoencoder model, by the end of the training, should have a minimal validation loss value. Also, the validation loss (the prediction loss on un-trained data) should converge with the training loss (the prediction loss on trained data) to avoid model overfitting. Figure 11 shows an example of a better-fitted model (Figure 11.a) and an overfitted model (Figure 11.b).

After experimenting, I observed that the model's loss after 100 epochs do not further decrease, independent of the fitting of the model (Figure 11). The model's loss reaches a minimum value for encoding dimensions 8 to 11, independent of the batch size (Figure 12). By plotting the loss versus batch size for encoding dimensions 8 to 11, I chose the combination of encoding dimension = 9 and batch size = 128 that has the minimum validation loss (Figure 13) and an optimal fitting curve (Figure 11.a).

Figure 12

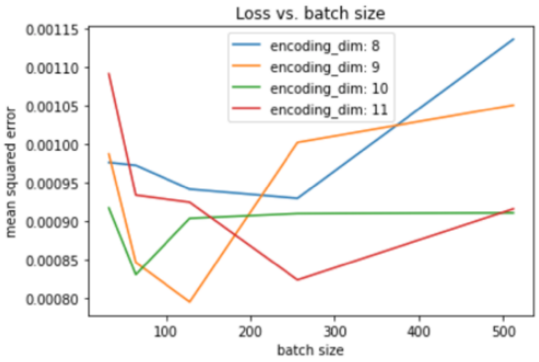
Validation Loss of Varying Encoding Dimensions with Different Batch Sizes



Note. Each loss value is averaged among three independent training events with the same encoding dimension, batch size, and epochs = 200.

Figure 13

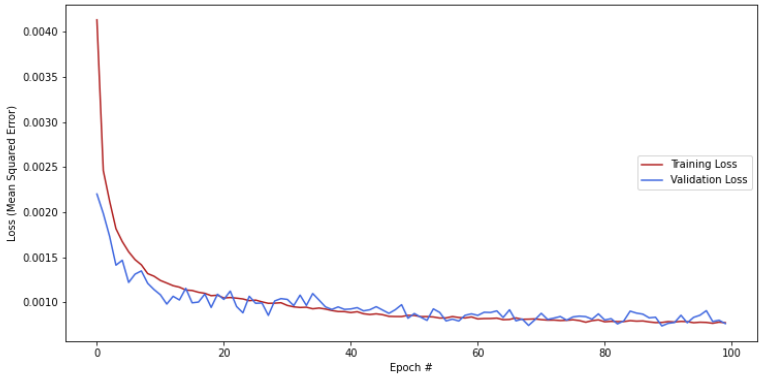
Validation Loss of Encoding Dimensions 8 to 11 with Different Batch Sizes



Note. Each loss value is averaged among three independent training events with the same encoding dimension, batch size, and epochs = 200.

Figure 14

Training Curve of the Final Model



Note. Model has encoding dimension = 9, batch size = 128, epochs = 100. The validation loss (Mean Squared Error) = 0.000764, converging with the training loss. R^2 of the model is 0.975156.

The final model is trained on encoding dimension = 9, batch size = 128, and epochs = 100 (Figure 14), which has a validation loss of 0.00076 (the mean squared error between the prediction value and the ground truth on un-trained dataset). R-squared of the model is 0.975, which means ~97.5% of the variance in the test dataset can be explained by the model prediction, further proving the strength of this autoencoder model on predicting the actual data.

2.4.2 Joint Angle Data Processing Pipeline

All 173 time series of joint angles go through a processing pipeline as follows. For each time series (video), all joint angles are first normalized by the min-max normalization, where minimum and maximum

value for each joint angle is obtained across all data points. Frames with some but not all NaN joint angle values are screened and predicted by the autoencoder to reconstruct the data. Each time series will go through a median filter of window size 5, with the purpose of reducing tracking glitches as well as filling in some frames where all joint angle values are NaN's. The video is in the form of 100 frames per second (FPS), thus a median filter with the window size of 5 frames will sacrifice some temporal resolution of the fly's movement but mostly preserve the movement dynamics. All joint angles in the time series will again be predicted by the autoencoder to reduce tracking errors.

By the end of the processing pipeline, 6 out of 173 videos still have NaN gaps of lengths mostly < 50 frames. Since the following computational methods to extract postural dynamics does not allow the presence of NaN values, for each NaN gap < 100 frames, I use linear interpolation on the gap's preceding and succeeding 'good' data points to fill in the gap. For the two NaN gaps > 100 frames, which means more than 1 second of the video has missing tracking data, I split the dataset at the NaN gap to avoid over-interpolation on missing data. The presence of NaN gaps often suggest that the fly is performing an abnormal movement that poses challenge to the tracking method. The use of linear interpolation here aims to identify (not to homogenize) NaN gaps from other parts of the tracking, because postural dynamics in a linear mode can be separated with other natural dynamics by the following method in Section 2.5.

Finally, I obtained time series of joint angles that have eliminated tracking glitches and errors and without NaN joint angle values in each frame.

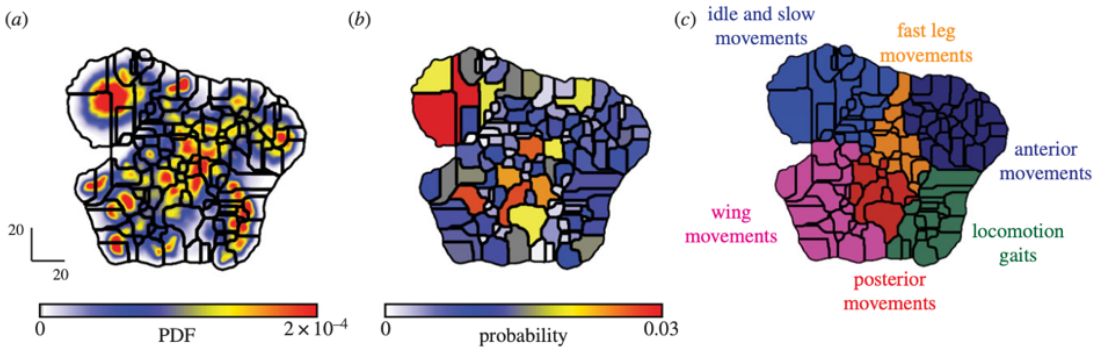
2.5 Motion Mapper for Building the Behavior Map

The method to build the behavior map uses the MotionMapperPy package, which adapts the original MotionMapper methodology described by Berman et al. (2014). In the first step, time series data undergoes wavelet transform to generate a spectrogram representation of the postural modes. By computing amplitudes of the Morlet continuous wavelet transform for the time series of joint angles, the method can identify representative postural dynamics (in this context, stereotyped behaviors) of flies in different time scales. Since the videos have the sampling frequency (or FPS) of 100 Hz, the maximum frequency for Morlet wavelet transform is set as 50 Hz, and we set the minimum frequency to be 0.5 Hz.

Following the wavelet transformation, a behavior map is created using the t-distributed Stochastic Neighbor Embedding (t-SNE). t-SNE is a dimensionality reduction algorithm that can transform the high-dimensional spectrogram data from the previous step into a two-dimensional visual map. Inside the map, data points will be clustered if the animal poses defined by their feature vectors are close in the high-dimensional space, meaning their transition probability is high. For each dataset, I trained the t-SNE on 4,500 number of points to generate a two-dimensional behavior map ("mini-tSNEs"). The algorithm with trained weights can learn to embed all data points into a single behavior map.

Figure 15

Segmentation into Behavioral Regions



Note. From "Mapping the stereotyped behavior of freely moving fruit flies," by G. J. Berman, D. M. Choi, W. Bialek, and J. W. Shaevitz, 2014, *Journal of the Royal Society, Interface,* 11(99), 20140672. (https://doi.org/10.1098/rsif.2014.0672). Copyright 2014 by the Royal Society.

The behavior map can be then segmented into multiple behavior regions where each region corresponds to a particular behavior performed by the fly (Figure 15.a). The image segmentation method uses the watershed transformation technique which finds watershed regions based on probability density peaks present in the map. By adjusting the number of regions to segment inside a map, each map region can provide a finer to more coarse description of the behaviors in the behavior repertoire, and the definitions of regions can be determined by visual assessment of videos (15.c). For this project, I choose to identify around 150 behavior regions in the map.

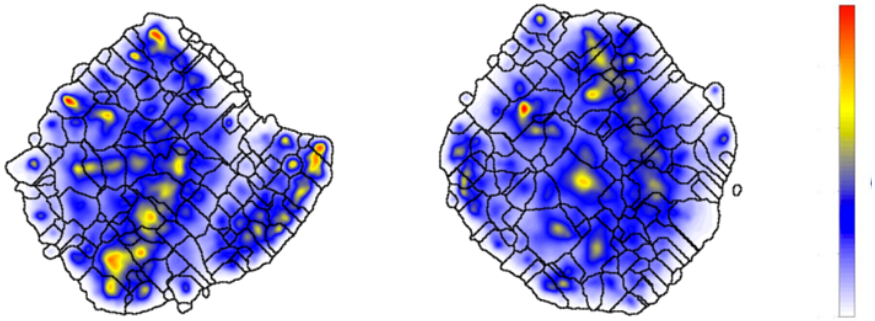
3. Results

3.1 Choosing the Behavior Map

With the data as time series of joint angles (from Section 2.4), I made two different attempts to generate the behavior space. First, I can feed the raw time series of joint angles (22 dimensional) to directly build the behavior map (Figure 16.a). Else, I can generate the map based on the time-series latent state data of the joint angles, to compress the 22-dimensional data into a lower dimension and enable more efficient computation (Figure 16.b). Toward which, I adopt the encoder architecture and weights of the trained autoencoder described in Section 2.4.1. and yield time series of 9-dimensional data by extracting the output of the encoding layer.

Figure 16

Behavior Maps Generated from Time-series Data



Note. (a) uses raw time-series of joint angles, probability density $\in [0, 2.66\times10{-}19]$ (b) uses encoded time-series of joint angles, probability density $\in [0, 1.98\times10{-}19]$.

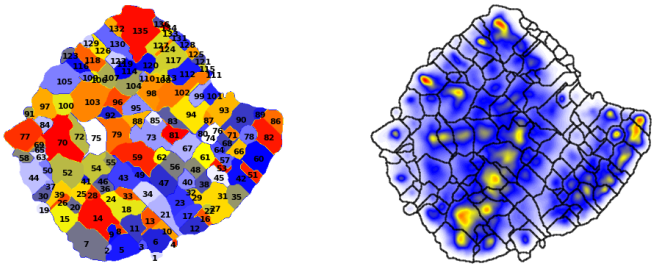
I used the output behavior map from the first method (Figure 16.a) to proceed with the result analysis because it shows more probability density peaks (shown as yellow and red regions) inside the map. More probability density peaks mean that more video frames are identified as “similar” by the algorithm, and in the context of animal behaviors, it suggests the method is more capable of identifying stereotyped behaviors from the raw time-series data.

3.2 Labeling the Behavior Map

The raw behavior map is not easily understandable by human eyes (Figure 17). To make sense of the map and measure the accuracy of my methods, I manually annotate the map with human-curated labels of fly behaviors.

Figure 17

Final Behavior Map Generated and Segmented



Note. (a) Map is segmented into 136 regions with watershed transformation, sigma = 0.8 for Gaussian probability curves. (b) Probability density ∈ [0, 2.66×10−19].

For each region in the behavior map, the MotionMapper can identify frame ranges across all videos that are allocated to this region. Ideally, flies across videos should be performing comparable behaviors in the identified frame regions. By adapting the video correspondence method created by Kanishk Jain for mice videos, I created a composite video for each region in the behavior map (Figure 18). Each composite video is composed of at most 25 video clips for the behavior region they are contained to. The selected video clips are at least 30 frames long and at most 300 frames long, corresponding 0.3 seconds to 3 seconds in the original experimental video.

Figure 18

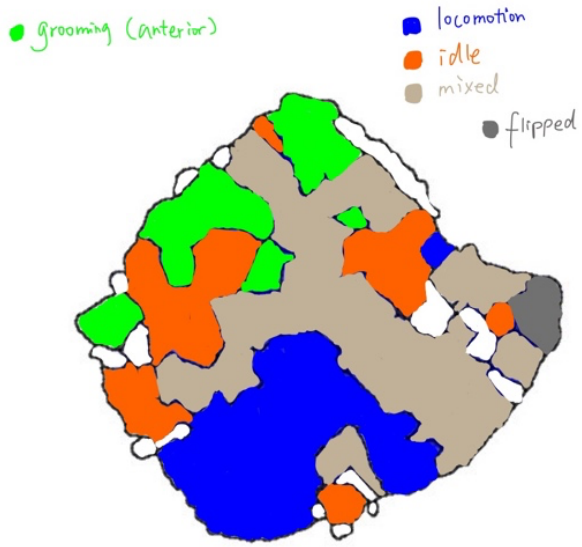
Screenshot of a Composite Video Created for Region 5 in the Behavior Map

Note. All composite videos can be accessed at http://drive.google.com/drive/folders/1fVz_2SDwc48HIz0r4kt-TH4IMnsQFgEZ.

By observing the composite videos, I labeled the map to describe stereotyped fly behaviors in each region by “idle”, “locomotion”, and “grooming (anterior)” (Figure 19), similar to the labels used in Berman et al (2014). Regions labeled as “mixed” describe two or more labels of stereotyped behaviors. Regions labeled as “flipped” suggest that the flies are downside up in the frames.

Figure 19

Human-Labeled Behavior Map with Stereotyped Behaviors



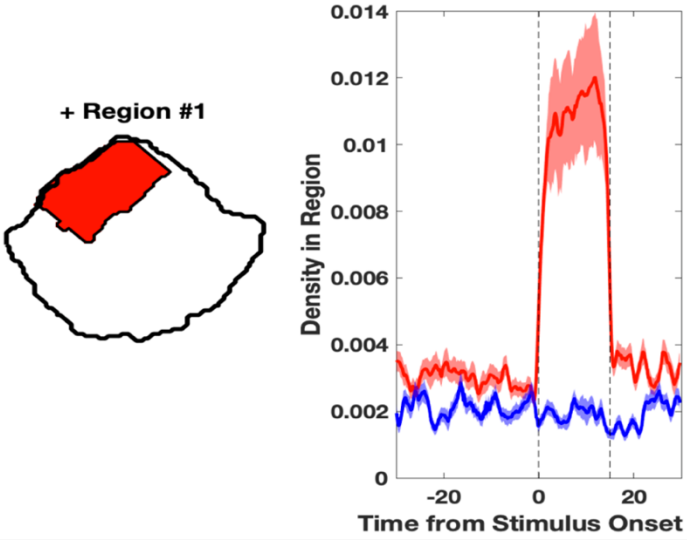
3.3 Optogenetic Analysis

After generating the behavior space, I used the same methodology described by Cande et al. (2018) to measure the activation phenotypes of 13 split-GAL4 lines that I have full access to for the 6 experimental and 6 control animals’ video tapes.

If the targeted descending neurons of a split-GAL4 line elicits or modifies the fly’s motor behaviors upon red light activation, the density of the behavior map will significantly shift to a particular region for the experimental but not control flies. For each fly line, the method isolates behavior regions that have a significant shift in density during red-light stimulation (as an example, see Figure 20). To determine that a behavior region is affected, Cande et al. assessed significance both within experimental flies and between experimental and control flies (Figure 21).

Figure 20

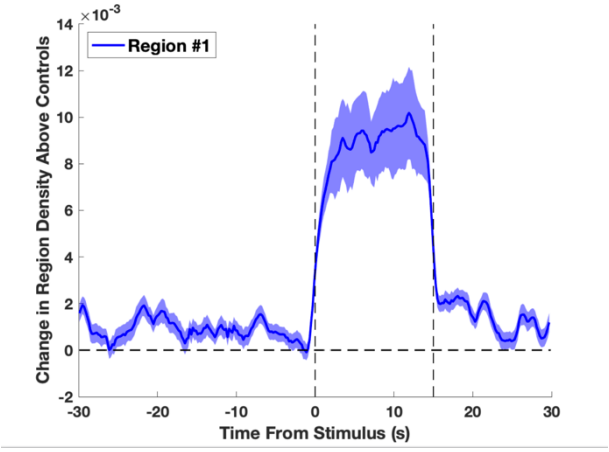
Analysis of Line SS02635 (Targeting DNg07 and DNg08)



Note. Image on the left is the behavior region that has significantly increased density during red-light stimulation. Image on the right compares the averaged density in this region for control (blue line) and experimental (red) animals. Experimental animals have a higher region density during the 15-second red-light stimulation (in between dotted lines). Similar to Cande et al. (2018), I reported the activation phenotype of line SS02635 as anterior grooming.

Figure 21

Averaged Change in Region Density of Experimental Animals above Controls for the Elicited Behavior Region of Line SS02635



Within experimental flies, for each animal in each LED cycle, the authors defined the average behavior space density during the 3 seconds of stimulation (on-period, t = 0s to t = 3s) and 15 seconds furthest from the stimulation (off-period, t = 30s to t = 45s) to be the same for normalization purposes. They used a Wilcoxon rank sum with Šidák correction to assess if the behavior space density during on-

and off-period is significantly different at each location in the space ($p<0.05$), where the Šidák correction is used to account for the loss in statistical significance with multiple comparisons.

Between experimental and control flies, the authors computed the difference in average behavior space density between the on-period and the mean of its two preceding off-periods, and used this quantity at each behavior space location to compare the behaviors among experimental and control flies. Similarly, they used a Wilcoxon rank sum with Šidák correction to assess if the behavior space density at each location is significantly different between two groups.

For a behavior region to be considered affected by the light stimulation, locations in this region show significant results by the assessments of both within experimental flies (light-on versus light-off periods) and between experimental and control flies (experimental versus control animals).

With the described approach, I measured the activation effects of 13 split-GAL4 fly lines that are created by Namiki et al. (2018). 11 out of 13 fly lines show significant region density increases during light stimulation, and affected regions are organized by broad behavior categories, labeled according to results from Section 3.2 (Figure 22). If multiple regions are significantly affected, I reported the most significant region. Targeted descending neurons by 11 split-GAL4 lines can be referenced by Table 1.

Figure 22

Summary of Activation Behavioral Effects of 11 Fly Lines

Note. Activation effects are labeled by line identifiers and organized by broad behavior categories.

Table 1

Line Identifiers and Targeting Descending Neurons Measured in This Project

Line identifiers	Targeting DNs
split_gal4.janelia.org:ss01052	DNp25
split_gal4.janelia.org:ss01060	DNb02
split_gal4.janelia.org:ss01540	DNp09
split_gal4.janelia.org:ss01580	DNp10
split_gal4.janelia.org:ss01581	DNp13, DNp30
split_gal4.janelia.org:ss01582	DNb04
split_gal4.janelia.org:ss01596	DNp03
split_gal4.janelia.org:ss01597	DNg07
split_gal4.janelia.org:ss02256	DNp06
split_gal4.janelia.org:ss02393	DNa08
split_gal4.janelia.org:ss02542	DNb01
split_gal4.janelia.org:ss02617	DNg11
split_gal4.janelia.org:ss02635	DNg07, DNg08

Note. The data are compiled from "The functional organization of descending sensory-motor pathways in *Drosophila*," by S. Namiki, M. H. Dickinson, A. M. Wong, W. Korff, and G. M. Card, 2018, *eLife* 7:e34272. (https://doi.org/10.7554/eLife.34272).

4. Discussion

In this project, I demonstrated a new analysis pipeline to represent fly behaviors in an experimental setting, with computational methods consisting of an animal tracking model, a joint angle calculation method, and an autoencoder. After that, I evaluated the accuracy of my methods by annotating the 'behavior map' generated from the processed data.

By revisiting the original experimental design of the dataset, I also measured the activation effects of 11 fly descending neuron lines. Similar to Cande et al.'s (2018) major results, I found that (1) activation of most descending neurons leads to stereotyped behaviors in flies, and (2) activation of multiple DNs have similar behavioral output.

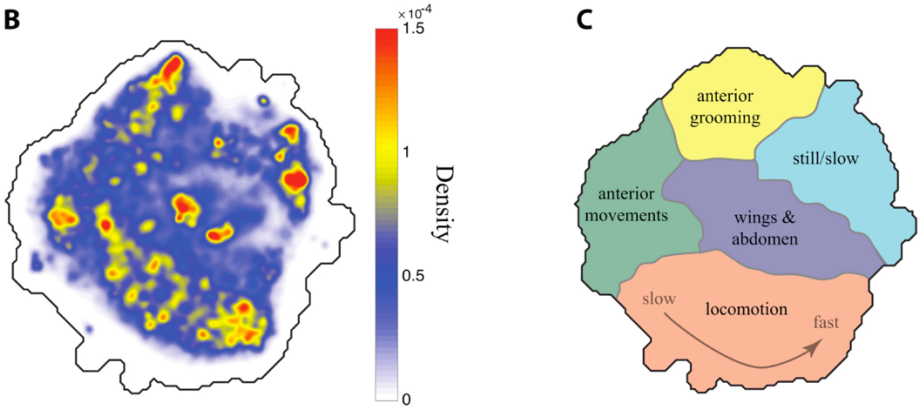
The analysis pipeline presented here highly depends on deep learning methods that flourish in the computational field in the past decades. The animal tracking model built by SLEAP uses the encoder-decoder architecture to track animal body parts through video frames; the autoencoder uses the encoder-decoder architecture to restore missing data and denoise errors; the MotionMapper package uses an unsupervised learning algorithm (t-SNE) to embed and cluster data points in a meaningful way onto the two-dimensional space. A major theme of the neural networks used in those methods lies in dimensionality reduction which extracts useful information from complex data and saves power in computation. Comparable dimensionality reduction techniques are also employed by biological systems to perceive real-world complex variables, compress into lower dimensions, correspond with previous experiences, and process into meaningful representations in the brain.

Results of this project indicate the potentially advantageous incorporation of deep learning methods in studying animal behaviors. Unlike traditional approaches, computational analysis pipelines free human labors and biases in annotating video datasets and present findings that are not readily apparent to researchers by visual assessment.

4.1 Limitations

Figure 23

2D Representation of Behaviors in the Descending Neuron Video Dataset and Human Curation of Watershedded Regions in the Behavior Space



Note. From "Optogenetic dissection of descending behavioral control in *Drosophila*," by J. Cande, S. Namiki, J. Qiu, W. Korff, G. M. Card, J. W. Shaevitz, D. L. Stern, and G. J. Berman, 2018, *eLife*, 7, e34275 (http://doi.org/10.7554/eLife.34275). Copyright 2018 by eLife.

As shown in the Results section, my analysis pipeline is able to identify and cluster some fly behaviors based on the video dataset. Comparing with the behavior map generated by Cande et al. (2018) (Figure 23), however, several stereotyped fly behaviors are missing from the behavior map. For example, most posterior movements (wings & abdomen region in Figure 23) are not identified.

A post-hoc visual inspection on the dataset suggests flies in my sampled videos do exhibit posterior movements, albeit in much lower frequencies than anterior and locomotion movements. As a result, they might not be well represented in all the training sets (of SLEAP, autoencoder, and MotionMapper) and thus poorly represented in the final behavior map. That could possibly explain the difference in completeness between my behavior map and the map generated by Cande et al. (2018) because my dataset is a subset of theirs, and flies in my sampled videos might not fully represent posterior and wing behaviors.

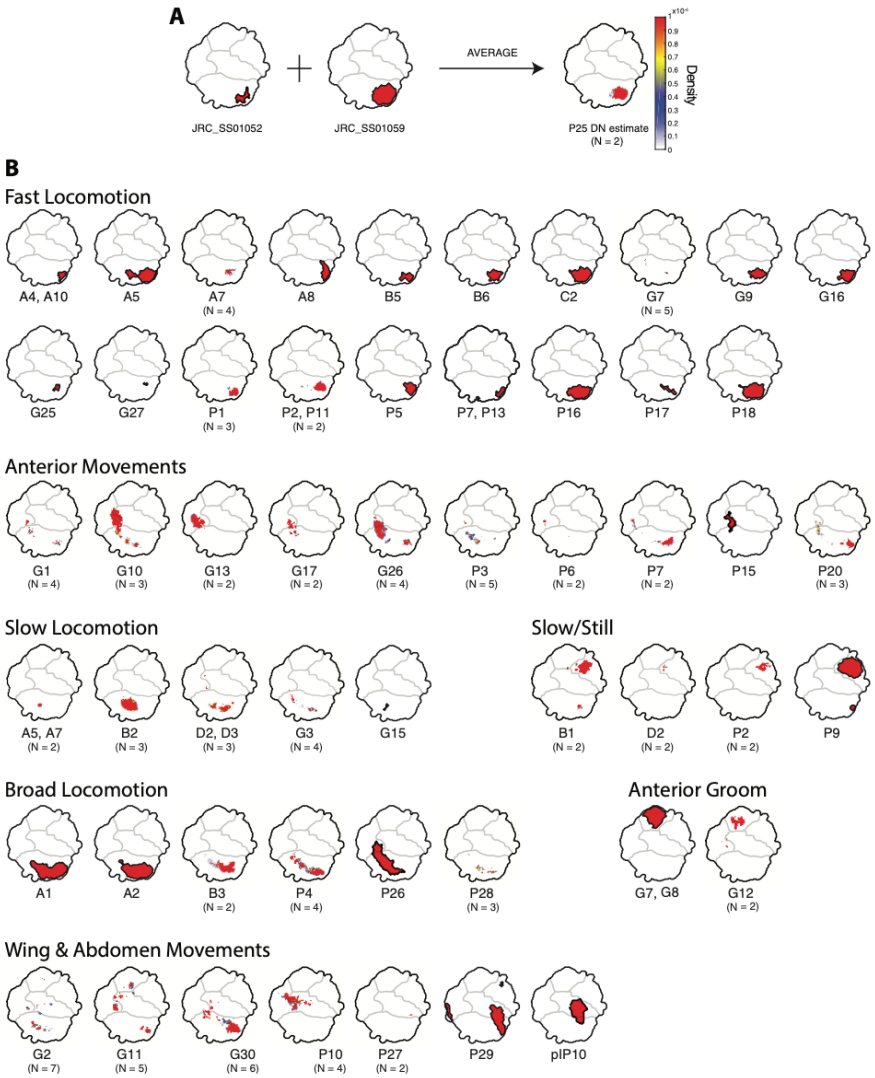
Another hypothesis to explain the missing behaviors in my behavior map is mainly due to the autoencoder implementation. As explained in Section 2.4.1, the training/validation set of the autoencoder composes of 45,000 data points with no missing joint angles. Those data may not fully capture all the fly

poses and bias toward more easily tracked frames when flies have all body parts exposed (i.e., those in locomotion and anterior grooming behaviors). In the joint-angle data processing pipeline (Section 2.4.1), the restoration of any input with missing values depends on available joint angles, which uses the learnt model weights trained on the training set. As a result, the filling in of missing values, when some body parts are obscured, might bias toward poses present in the training set.

4.2 Future Directions

Figure 24

Activated Behavior Regions of Representative Lines



Note. (A) Example showing how the averaging of two lines to produce an estimated phenotype of DNp25. (B) Averaged activation effects of 53 individual descending neurons, organized by broad behavior categories. From "Optogenetic dissection of descending behavioral control in Drosophila," by J. Cande, S. Namiki, J. Qiu, W. Korff, G. M. Card, J. W. Shaevitz, D. L. Stern, and G. J. Berman, 2018, eLife, 7, e34275 (http://doi.org/10.7554/eLife.34275). Copyright 2018 by eLife.

Cande et al. reported the activation effect of 53 individual descending neurons by averaging across multiple fly lines that target the same DN (example shown in Figure 24.A). My report (Section 3.3) only surveys 11 fly lines and the limited dataset would not allow averaging calculations, so it is not feasible to directly compare my results with the results by the original authors.

A rough comparison of the activation effects of DNs targeted by my 11 fly lines based on DN cell types shows that most but not all activation effects of individual descending neurons are reported as the same (Table 2). An immediate future direction of this work, thus, is to apply the analysis pipeline on the whole dataset by Cande et al. (2018) and compare the resulting behavior map and measured activation effects of descending neurons. The broadening of dataset would also better allow the exploration of context dependency in activated behaviors by optogenetic stimulation.

Table 2

Comparing Activation Effects of 11 Surveyed Fly Lines with Cande et al.'s Report by Targeted DN Cell Types

Similar Report

Line Identifier (Targeted DN cell type(s))	Activation Effects Reported in this Project	Cande, et al.'s Report
ss01581 (P13, P30)	Fast locomotion	Fast locomotion (P13)
ss02542 (B1)	Slow/Idle	Slow/Idle
ss01540 (P9)	Slow/Idle	Slow/Idle
ss02635 (G7, G8)	Anterior groom	Anterior groom
ss01052 (P25)	Broad locomotion	Fast locomotion
ss01060 (B2)	Broad locomotion	Slow locomotion
ss01597 (G7)	Broad locomotion	Fast locomotion

Different Report

ss01582 (B4)	Fast locomotion	not reported
ss01596 (P3)	Fast locomotion	Anterior Movement
ss02256 (P6)	Fast locomotion	Anterior Movement
ss02617 (G11)	Slow/Idle	Wing & Abdomen Movements

Note. The column of reported activation effects in this project is adapted from Figure 22 and Table 1. The column of Cande et al.'s report is adapted from "Optogenetic dissection of descending behavioral control in Drosophila," by J. Cande, S. Namiki, J. Qiu, W. Korff, G. M. Card, J. W. Shaevitz, D. L. Stern, and G. J. Berman, 2018, eLife, 7, e34275 (http://doi.org/10.7554/eLife.34275).

While Cande et al. provided a systematic report for the activation effects of 53 descending neurons (Figure 24.B), those effects are defined by the activated behavior regions and do not provide finer details on the actual behaviors beyond a coarse description. An advantage of using joint angles to build

the behavior map is that one can reverse the data processing procedure and better define the activation effects with the average joint angle dynamics of the activated behavior regions, so to differentiate multiple descending neurons that are organized under the same broad category as in Figure 24.

Prior to achieving this goal, we need to rule out the causes of errors in the current results and make modifications to improve the analysis pipeline if necessary. First, one may impose the analysis pipeline on video datasets that span the whole behavior repertoire of fly, and the representation of more fly behaviors in the behavior map would suggest the current errors are generated from the incompleteness of the dataset. Also, one may generate the behavior map by skipping or substituting the autoencoder (with Principal Component Analysis or similar dimensionality reduction techniques) and comparing the results, which might suggest if the errors come from the autoencoder implementation.

We can imagine that flies, unlike human or other mammals, have relatively independently moving limbs. On the contrary, mammals have biomechanical constraints to maintain postural stability and movement control (Ivanenko & Gurfinkel, 2018). The failure of an autoencoder to predict missing joint angles with available joint angles in fly, if verified, might indicate the complexity in predicting fly poses when a significant amount of tracking data is missing. As another direction of future work, one could attempt implementing similar analysis pipeline in other experimental contexts on rodents or mammals to provide insights into the biomechanical aspect of animal behaviors.

References

Bauman, J. M., & Chang, Y. H. (2010). High-speed X-ray video demonstrates significant skin movement errors with standard optical kinematics during rat locomotion. *Journal of neuroscience methods*, *186*(1), 18-24.

Berman, G. J., Choi, D. M., Bialek, W., & Shaevitz, J. W. (2014). Mapping the stereotyped behaviour of freely moving fruit flies. *Journal of The Royal Society Interface, 11*(99), 20140672.

Bidaye, S. S., Machacek, C., Wu, Y., & Dickson, B. J. (2014). Neuronal control of Drosophila walking direction. *Science, 344*(6179), 97-101.

Cande, J., Namiki, S., Qiu, J., Korff, W., Card, G. M., Shaevitz, J. W., ... & Berman, G. J. (2018). Optogenetic dissection of descending behavioral control in Drosophila. *Elife*, *7*, e34275.

Dawkins, R., & Dawkins, M. (1976). Hierarchical organization and postural facilitation: Rules for grooming in flies. *Animal Behaviour*, *24*(4), 739–755.

Heinrich R. (2002). Impact of descending brain neurons on the control of stridulation, walking, and flight in orthoptera. *Microscopy research and technique, 56*(4), 292–301.

Hsu, C. T., & Bhandawat, V. (2016). Organization of descending neurons in Drosophila melanogaster. *Scientific reports*, *6*(1), 1-14.

Insafutdinov, E., Pishchulin, L., Andres, B., Andriluka, M., & Schiele, B. (2016, October). Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *European conference on computer vision* (pp. 34-50). Springer, Cham.

Ivanenko, Y., & Gurfinkel, V. S. (2018). Human Postural Control. *Frontiers in neuroscience, 12*, 171.

Klapoetke, N. C., Murata, Y., Kim, S. S., Pulver, S. R., Birdsey-Benson, A., Cho, Y. K., ... & Boyden, E. S. (2014). Independent optical excitation of distinct neural populations. *Nature methods*, *11*(3), 338-346.

Kohatsu, S., Koganezawa, M., & Yamamoto, D. (2011). Female contact activates male-specific interneurons that trigger stereotypic courtship behavior in Drosophila. *Neuron*, *69*(3), 498-508.

Mathis, A., Mamidanna, P., Cury, K. M., Abe, T., Murthy, V. N., Mathis, M. W., & Bethge, M. (2018). DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. *Nature neuroscience*, *21*(9), 1281-1289.

Namiki, S., Dickinson, M. H., Wong, A. M., Korff, W., & Card, G. M. (2018). The functional organization of descending sensory-motor pathways in Drosophila. *Elife*, *7*, e34272.

Namiki, S. (2021). Descending Neuron for Freezing Behavior in Drosophila melanogaster. In *Death-Feigning in Insects* (pp. 145-157). Springer, Singapore.

Pereira, T. D., Aldarondo, D. E., Willmore, L., Kislin, M., Wang, S. S. H., Murthy, M., & Shaevitz, J. W. (2019). Fast animal pose estimation using deep neural networks. *Nature methods*, *16*(1), 117-125.

Pereira, T. D., Tabris, N., Li, J., Ravindranath, S., Papadoyannis, E. S., Wang, Z. Y., ... & Murthy, M. (2022). SLEAP: A deep learning system for multi-animal pose tracking. *Nature Methods*.

Staudacher, E. R. I. C. H., & Schildberger, K. L. A. U. S. (1998). Gating of sensory responses of descending brain neurones during walking in crickets. *The Journal of experimental biology*, *201*(4), 559-572.

von Philipsborn, A. C., Liu, T., Jai, Y. Y., Masser, C., Bidaye, S. S., & Dickson, B. J. (2011). Neuronal control of Drosophila courtship song. *Neuron*, *69*(3), 509-522.

von Reyn, C. R., Breads, P., Peek, M. Y., Zheng, G. Z., Williamson, W. R., Yee, A. L., ... & Card, G. M. (2014). A spike-timing mechanism for action selection. *Nature neuroscience*, *17*(7), 962-970.