Neural Computing on a Raspberry Pi: Applications to Zebrafish Behavior Monitoring

Lakshmi Narasimhan Govindarajan Tarun Sharma Ruth Colwill Thomas Serre
{lakshmi_govindarajan,tarun_sharma1,ruth_colwill,thomas_serre}@brown.edu
Brown University, Providence, RI, USA

Abstract

We describe a low-cost behavior monitoring system for zebrafish, which uses a single-board Raspberry Pi® computer for imaging and a dedicated Intel® Movidius™ Neural Compute Stick for computer vision. As a proof of concept, the system is shown to robustly track pose on larvae (5-7 dpf) that are typically very challenging to image. Open-source software and executable code are made publicly available to be re-used by other researchers.

Introduction

In contrast to human-derived annotations, automation promises to bolster ethology by increasing throughput, improving reproducibility and allowing novel types of quantitative analyses (Egnor & Branson, 2016). Pharmacological studies using small laboratory animals such as zebrafish are expected to particularly benefit. These animals have become popular screening tools because of their small size and their ability to reproduce easily under laboratory conditions, making them easy to maintain in the large numbers required for high-throughput screening.

Indeed, automated tracking systems have already been successfully used in a variety of zebrafish studies (see Rihel & Schier, 2012, for review). However, existing tracking systems (e.g., Kato et al., 2004; Hicks et al., 2006; Ramazani et al., 2007; Colwill & Creton, 2011; Maaswinkel et al., 2013; Pérez-Escudero et al., 2014; Wang et al., 2014; Barker & Baier, 2015; Nema et al., 2016) suffer several key limitations: (1) they rely on relatively simple error-prone image processing techniques (e.g., image subtraction, morphological operators, etc) that require manual threshold adjustments, severely limiting their applicability to large-scale studies; (2) they only track a single point on the animal body (but see Gomez-Marin et al., 2012; Pelkowski et al., 2011), which is sufficient for coarse motility measures (e.g., distance travelled, speed or space used) and circadian rhythm but does not allow for the analysis of finer reflexive behaviors such as optokinetic, optomotor or visual-motor responses, or finer motor and visual deficits. Existing systems are thus limited in scope, ease of use, and accuracy, and are only partially automated.

Learning-based methods offer an alternative approach with the potential to address several of the issues mentioned above. In recent years, progress has been particularly significant in the area of deep learning (LeCun et al., 2015). Several deep neural network systems have been developed that have been shown to approach and sometimes outperform humans in complex recognition tasks including object categorization (He et al., 2016) and face recognition (Kemelmacher-Shlizerman et al., 2016). Deep learning offers a unique opportunity to fully automate video analysis for behavioral studies via end-to-end learning – reducing the need for human supervision and, hence, significantly increasing throughput and statistical power. However, to our knowledge, these advances have not yet permeated zebrafish studies. One challenge with deep learning algorithms is that they typically require high-performance computing hardware (e.g., GPUs) which is typically not readily available in traditional animal laboratories.

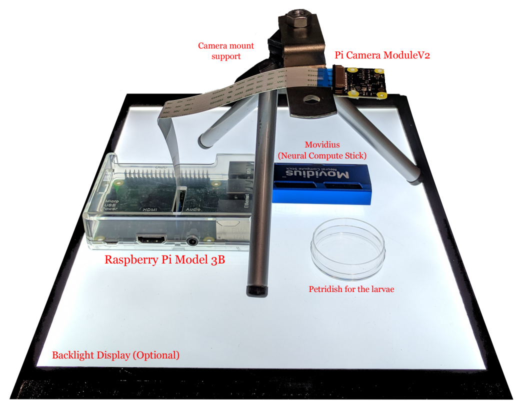

Figure 1: Proposed image acquisition system: The system includes a Raspberry Pi (model 3B) combined with a Pi camera module v2 for video imaging as well as an Intel Movidius Neural Compute Stick for tracking with a deep-learning based system. Though the back light display is optional, uniform lighting enhances video quality and potentially enables the presentation of stimuli for psychophysics studies.

Our goal is thus to develop a low-cost deep learning-based system for zebrafish behavior monitoring. Several low-cost systems using Raspberry Pi® have already been developed for behavioral analysis including high-throughput, open-sourced platforms (Maia Chagas et al., 2017; Geissmann et al., 2017) and operant chambers (O'Leary et al., 2018). One of the main shortcomings of these systems is that they are usually limited by the poor processing power of the Raspberry Pi's CPU. A common workaround is to push processing to more powerful clusters. Here, instead, we explore the use of the recently released Intel® Movidius™ Neural Compute Stick to augment a single-board Raspberry Pi computer to build an open-source deep learning-based behavior monitoring platform (Fig. 1).

Contributions Our contributions include: 1) an easy-to-assemble low cost imaging system; 2) an application of neu-

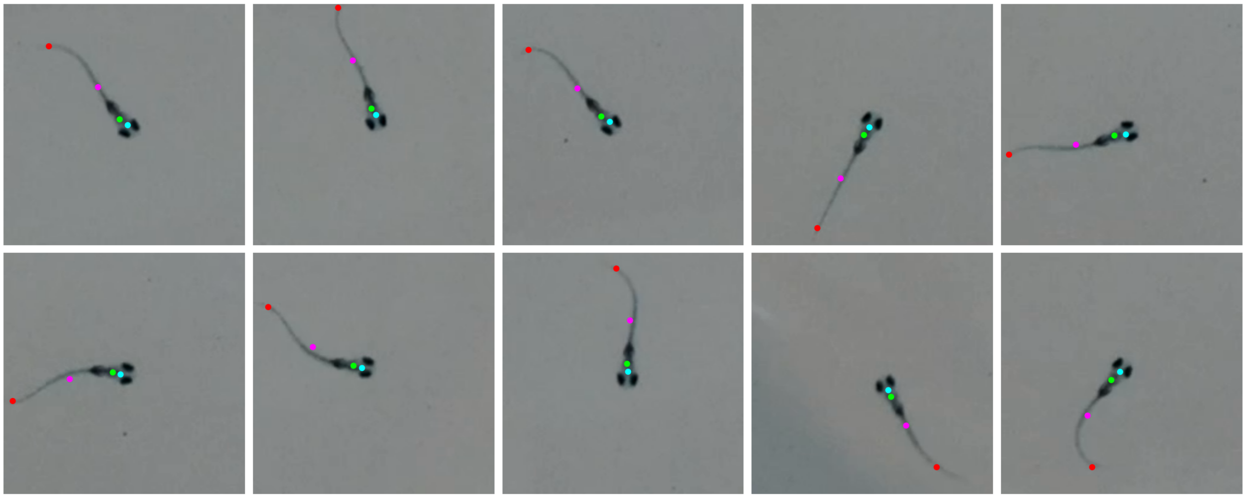

Figure 2: Sample pose estimation results on 7-dpf larvae. We parameterized our zebrafish skeleton model with 4 joints (shown as cyan, green, pink and red points on the images). Inputs to the pose estimation network are bounding box outputs from a fish detector centered on the head of the animal.

ral computing for the behavioral analysis of zebrafish ; 3) open-source software and executable code for pose estimation/tracking; 4) a large annotated zebrafish pose dataset (available on request) for other researchers to re-use.

Monitoring system overview

The imaging system includes a Raspberry Pi®3 Model B running a Raspbian Stretch operating system. A second generation Raspberry Pi camera module v2 with a fixed-focus lens was used. The Pi camera v2 module allows recording speed of up to 30FPS at a spatial resolution of 1920×1080 pixels, which is ideal for imaging small larvae.

Peripherals including monitor and keyboard/mice are optional. All devices can be controlled from a desktop via secure shell with a few simple commands. All software was written in Python. Data processing code (see below) was compiled to run inference on an Intel® Movidius™ Neural Compute Stick. The resulting system is both simple to assemble and affordable (Table 1).

Because of the small size of the Raspberry Pi camera, the system has the potential to be extended to a multi-well imaging system. Minimizing paraphernalia is especially desirable when stacking up multiple cameras to boost throughput. Relevant documentation, code and a disk image of our environment is released here `https://github.com/serre-lab/rpi-ncs-toolkit`.

Animal handling

Eggs were collected in a shallow tray for two hours following light onset from a breeding population of WT adult zebrafish. The eggs were transferred to a 1L plastic breeding tank (Aquatic Habitats) containing approximately 500 ml of egg water (EW, 60 mg/l Instant Ocean in deionized water and 0.25 mg/l methylene blue) buffered to a pH of 7.0 and incubated at 28.5^o C under a 14/10 h light/dark cycle until 7 days post-fertilization (dpf). The medium was partially refreshed every other day and any debris was removed to maintain water quality.

Component	Price (in US$)
Raspberry Pi®3B	35
Pi camera v2	30
Intel® Movidius™ Neural Compute Stick	77
USB hub	8
Total	150

Table 1: Cost of the system broken down by component.

Automated pose estimation

We built a two-stage deep learning-based processing pipeline: An initial neural network was used for coarsely detecting the animal body which was then passed to a second network for precise pose estimation. We have implemented a single shot detector (SSD) with MobileNet (Howard et al., 2017) for producing bounding box predictions around the head of the animal. MobileNets are a class of efficient convolutional architectures for embedded vision applications. The entire pipeline was implemented in Caffe (Jia et al., 2014) and compiled to run on Intel® Movidius™ Neural Compute Stick.

To produce ground-truth annotations, we used an in-house shape-based tracker (Guo et al., 2018) which is very accurate but exceedingly slow (about 5 frames per min on a modern workstation). Machine outputs were manually inspected and errors were corrected by hand (corresponding to a fraction of a percent). We used a total of 10,415 such machine annotated images, and augmented 5$\times$ (application of random brightness perturbation sampled from a normal distribution on the HSV

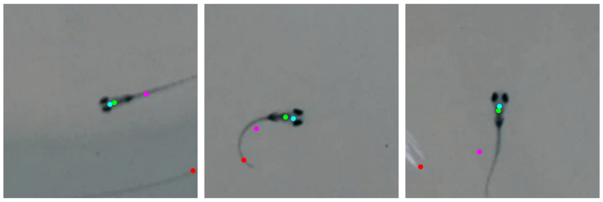

Figure 3: Sample failure cases. Though these sort of errors are rare, background artifacts, motion blur and random lighting changes can cause failure of the pose estimation system.

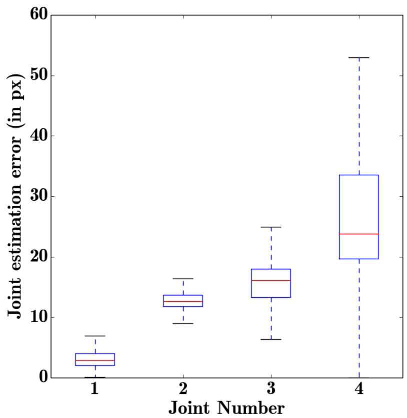

Figure 4: A box plot of per-joint estimation errors from our test data set. The average joint error is 14.33 pixels. Numbering starts from the head (1) through to the tail (4).

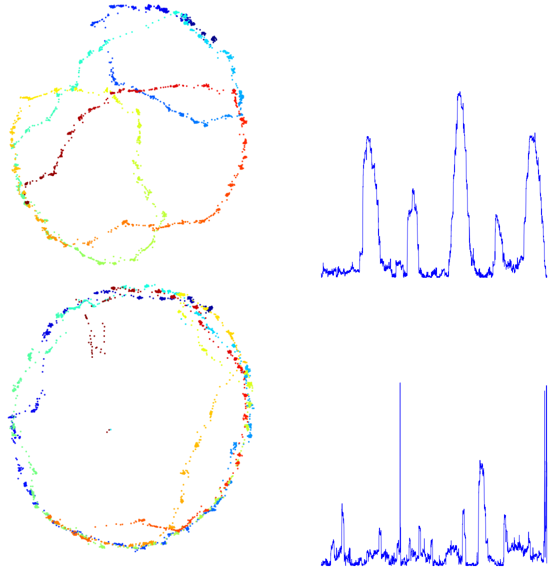

Figure 5: Profiling distance from the wall of the arena. (Left) Trajectory from a single individual color coded for time (blue:start, red:end) (Right) Corresponding distance to the wall over time.

color space) to yield a total of 62,490 images. We trained the MobileNet-SSD network from scratch using the RMSProp optimizer with a learning rate of $5e{-4}$ and weight decay of $5e{-5}$ for a total of 38,000 iterations.

As a post processing step, any bounding box prediction greater than two standard deviations away from the mean, calculated using a sliding window of length 30 frames, was deemed an outlier and removed. Detections were then smoothed by enforcing temporal continuity via mean filtering using a sliding window of size 10. Evaluation on an annotated video (3,585 frames) of a single held-out animal yielded a nearly perfect accuracy of 99.7%.

We then implemented a patch-based classification scheme for fine pose estimation, with $C = 5$ classes (4 joints and background). We considered the head, the neck, the midpoint of the tail, and the tip of the tail as our 4 joints and refer to them thereafter as joints $1-4$, respectively. A 5-layer convolutional architecture was trained with sliding patches of size 28×28. We refrained from using any pooling operation to retain spatial selectivity. Convolutional layers with stride=2 were used instead with ReLU as an activation function. The network was trained from scratch using an Adam Optimizer (Kingma & Ba, n.d.) with a learning rate of $1e{-4}$ and a weight decay of $5e{-3}$. Standard softmax cross-entropy was used as the loss function. This implementation was done in Tensorflow (Abadi et al., 2016) and compiled to run on the Intel Movidius as with the bounding-box detector. Note that we prefer a patch classification approach to direct coordinates regression simply because of the ability to obtain multiple competing hypotheses for every joint location by sampling from the heatmaps produced. This in turn allows for application of explicit structural constraints.

After obtaining the per-pixel joint classification results, we implemented a simple skeleton constraint to enforce logical positioning of the joints. Mode seeking from the confidence values of joint classification was done using the Mean-shift algorithm (Fukunaga & Hostetler, 1975) for the first three joints starting from the *head*. Detection of the tail tip was done by maximizing the distance from the first three joints along the medial axis obtained from patches classified as the tail.

A hand-annotated data set of 5,495 images was created, with four joint locations per frame. Of this, we used 1,900 images for training our pose estimation model. The remaining 3,595 formed a continuous sequence which was used for evaluation purposes. Training data samples were augmented via random image flips and brightness/contrast adjustments. Sample visual results from our pose estimation pipeline are shown in Fig. 2, while error cases are show in Fig. 3

We evaluated the pose estimator on our test data set. The standard euclidean joint error metric was used for pose estimation. A full per-joint box plot of joint errors is shown in Fig. 4. The average joint error over the full test data set is 14.33 pixels. Following intuition, we observe that it is much harder to predict the location of the tail since it appears faint – often blending with the background. Additionally, inaccuracy in human annotations may have also contributed to the increased error observed in the latter two joints. Our method is robust to minor changes in camera setup, with retraining required for different image acquisition systems. Also, a more principled way to introduce structural constraints during pose estimation is part of our plan for future work.

Our detection and pose estimation pipeline can be utilized to extract a number of useful behavioral measures. Here, as proof-of-concept, we demonstrate the extraction of distance to wall profiles. This is an important metric for assessing thigmotactic behavior. For example, in caffeine dose-response relationship analysis proximity to a wall could serve as a reliable indicator of anxiety and/or unwillingness to explore. Fig. 5 shows two examples; one where a larva is freely exploring the central region of the arena, and the other where another individual is keeping to the walls.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., et al. (2016). TensorFlow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX conference on operating systems design and implementation* (pp. 265–283). Berkeley, CA, USA: USENIX Association.

Barker, A. J., & Baier, H. (2015, November). Sensorimotor decision making in the zebrafish tectum. *Curr. Biol.*, *25*(21), 2804–2814.

Colwill, R. M., & Creton, R. (2011). Imaging escape and avoidance behavior in zebrafish larvae. *Rev. Neurosci.*, *22*(1), 63–73.

Egnor, S. E. R., & Branson, K. (2016, July). Computational analysis of behavior. *Annu. Rev. Neurosci.*, *39*, 217–236.

Fukunaga, K., & Hostetler, L. (1975). The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Trans. Inf. Theory*, *21*(1), 32–40.

Geissmann, Q., Garcia Rodriguez, L., Beckwith, E. J., French, A. S., Jamasb, A. R., & Gilestro, G. F. (2017, 10). Ethoscopes: An open platform for high-throughput ethomics. *PLOS Biology*, *15*(10), 1-13.

Gomez-Marin, A., Partoune, N., Stephens, G. J., Louis, M., & Brembs, B. (2012, January). Automated tracking of animal posture and movement during exploration and sensory orientation behaviors. *PLoS One*, *7*(8), e41642.

Guo, Y., Govindarajan, L. N., Kimia, B., & Serre, T. (2018). Robust pose tracking with a joint model of appearance and shape. *arXiv:1806.11011v1 [cs.CV].*

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Computer vision and pattern recognition.*

Hicks, C., Sorocco, D., & Levin, M. (2006). Automated analysis of behavior: A computer controlled system for drug screening and the investigation of learning. *J. Neurobiol.*.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., et al. (2017, April). MobileNets: Efficient convolutional neural networks for mobile vision applications.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., et al. (2014). Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on multimedia* (pp. 675–678). New York, NY, USA: ACM.

Kato, S., Nakagawa, T., Ohkawa, M., Muramoto, K., Oyama, O., Watanabe, A., et al. (2004, March). A computer image processing system for quantification of zebrafish behavior. *J. Neurosci. Methods*, *134*(1), 1–7.

Kemelmacher-Shlizerman, I., Seitz, S. M., Miller, D., & Brossard, E. (2016). The megaface benchmark: 1 million faces for recognition at scale. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4873–4882).

Kingma, D. P., & Ba, J. L. (n.d.). Adam: a method for stochastic optimization. In *International conference on learning representations.*

LeCun, Y., Bengio, Y., & Hinton, G. (2015, May). Deep learning. *Nature*, *521*(7553), 436–444.

Maaswinkel, H., Zhu, L., & Weng, W. (2013, December). Using an automated 3d-tracking system to record individual and shoals of adult zebrafish. *J. Vis. Exp.*(82), 50681.

Maia Chagas, A., Prieto-Godino, L. L., Arrenberg, A. B., & Baden, T. (2017, 07). The 100 lab: A 3d-printable open-source platform for fluorescence microscopy, optogenetics, and accurate temperature control during behaviour of zebrafish, drosophila, and caenorhabditis elegans. *PLOS Biology*, *15*(7), 1-21.

Nema, S., Hasan, W., Bhargava, A., & Bhargava, Y. (2016, September). A novel method for automated tracking and quantification of adult zebrafish behaviour during anxiety. *J. Neurosci. Methods*, *271*, 65–75.

O'Leary, J. D., O'Leary, O. F., Cryan, J. F., & Nolan, Y. M. (2018, Mar 08). A low-cost touchscreen operant chamber using a raspberry pi$^{\mathrm{TM}}$. *Behavior Research Methods*.

Pelkowski, S. D., Kapoor, M., Richendrfer, H. A., Wang, X., Colwill, R. M., & Creton, R. (2011, September). A novel high-throughput imaging system for automated analyses of avoidance behavior in zebrafish larvae. *Behav. Brain Res.*, *223*(1), 135–144.

Pérez-Escudero, A., Vicente-Page, J., Hinz, R. C., Arganda, S., & Polavieja, G. G. de. (2014, July). idtracker: tracking individuals in a group by automatic identification of unmarked animals. *Nat. Methods*, *11*(7), 743–748.

Ramazani, R. B., Krishnan, H. R., Bergeson, S. E., & Atkinson, N. S. (2007, May). Computer automated movement detection for the analysis of behavior. *J. Neurosci. Methods*, *162*(1-2), 171–179.

Rihel, J., & Schier, A. F. (2012, March). Behavioral screening for neuroactive drugs in zebrafish. *Dev. Neurobiol.*, *72*(3), 373–385.

Wang, J., Liu, C., Ma, F., Chen, W., Liu, J., Hu, B., et al. (2014, April). Circadian clock mediates light/dark preference in zebrafish (danio rerio). *Zebrafish*, *11*(2), 115–121.