

1 Open-Source Thermometer, Temperature Controller, and Light 2 Meter for Use in Animal Facilities and During Experiments

3
4 Andrey Andreev^{1*}, Pavee Vasnarungruengkul², Daniel A. Wagenaar³, David A. Prober¹
5

6 ¹ Division of Biology and Biological Engineering, California Institute of Technology
7

8 ² Protein Expression Center, California Institute of Technology
9

9 ³ Neurotechnology Laboratory, California Institute of Technology
10

*Correspondence: aandreev@caltech.edu

11 Abstract

12 Experiments with biological samples require precise control of environmental conditions. In our
13 work we use zebrafish (*Danio rerio*) to understand the neurobiology of sleep, which requires
14 precise control of temperature and lighting. Like many labs, lighting and temperature in the animal
15 facility are centrally controlled in the building. During behavioral experiments and microscopy
16 sessions, we use custom-built heating systems and perform occasional manual checks of
17 conditions. However, without a system to precisely record conditions, gradual changes in
18 temperature can go unnoticed for a long time, and temporary failures may be missed entirely. Here
19 we present the design and characterization of affordable open-source tools to record temperature
20 and light conditions during animal experiments using an Arduino microcontroller or a Raspberry
21 Pi compact computer. The waterproof temperature sensor has high stability over 50 days of
22 recording and is precise to 0.1°C. The Arduino device can be used through a common serial port
23 interface for which we present code in Python and MATLAB. The Raspberry Pi version can be
24 accessed through a web interface, for which we provide an installation guide. We use the device
25 to record and review temperature and lighting conditions in two zebrafish animal facilities. We
26 use our platform to add a water heating system to maintain temperature at 28°C during *in vivo*
27 light-sheet imaging of larval zebrafish. We show that a change in temperature from 28°C to 32°C
28 affects resting heart rate of the animal, highlighting the importance of maintaining and recording
29 conditions. The protocols presented here do not require advanced engineering, fabrication, or
30 software skills, and provide an approach to accurately record and report experimental conditions.
31

32 Introduction

33 Animal facilities and experiments that use animals require precise monitoring and control of
34 environmental conditions such as light and temperature. This is especially true for sleep studies
35 where experiments can last for several days and environmental conditions affect behavior.
36 Zebrafish and other animals that do not maintain body temperature require control of these
37 conditions by external means. In zebrafish, light and temperature affect several bodily functions
38 such as brain activity, heart rate, and behavior [1-5]. Animal facilities often rely on daily manual
39 data recordings, but this usually only provides a single data point each day and is susceptible to
40 human error. Light measurements are especially important in experiments that involve visual
41 function, where even a small amount of light can perturb rhodopsin, and thus light levels must be
42 continuously monitored during experiments that can last for several weeks [6]. Measurements with
43 automatic continuous sensors provide independent information about the state of the conditions in
44 the animal facility or during an experiment.

45

46 Tracking conditions and independent verification of set points and schedules such as cyclical light
47 conditions allows more precise experiments as well as reporting of the actual data, rather than
48 estimates and assumptions about the conditions. Commercially available thermometers with data
49 logging can cost up to \$250, require specialized software, and usually cannot have additional
50 sensors or interface for remote access. Here we present an open-source platform for a temperature
51 and light data logger with several options for the interface, including remote web access. Device
52 cost starts at \$40. This platform can also be used to regulate temperature in a sample chamber
53 using a simple heating element and proportional control scheme, similar to previously reported
54 devices to keep and rapidly change temperature conditions under a stereomicroscope [7] or
55 confocal microscope [8].

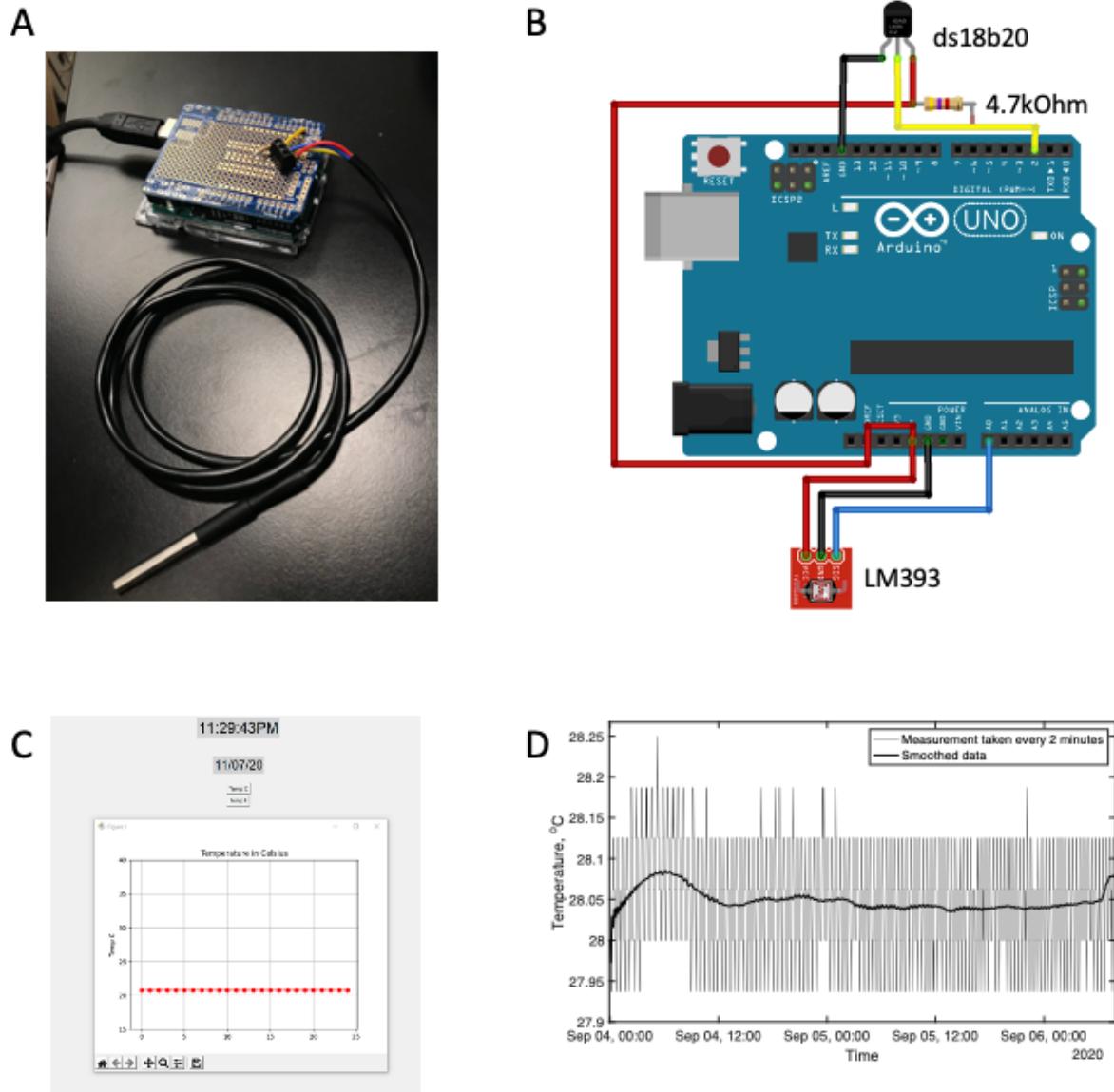
56

57 **Results**

58 Our temperature recorder (Figure 1A) consists of a microcontroller (Arduino Uno, see Table 1 for
59 list of parts) and a waterproof digital temperature sensor (DS18B20). Assembly consists of the five
60 parts listed in Table 1 soldered onto a “shield” (general purpose soldering circuit board). The
61 Arduino code for measuring temperature and light levels is presented in Code Snippet 1.

62

63 The Arduino Uno device records temperature measurements from the sensor and returns these via
64 a USB port (Figure 1B) at a preset rate between once per second to once every few minutes. To
65 collect and analyze the data, we developed a custom graphical user interface using Python (Figure
66 1C, Code Snippet 2-1 and 2-2), and MATLAB (Figure 1D, Code Snippet 3). We first tested the
67 sensor by recording temperature in a high-precision water bath set at 28°C. Continuous recording
68 for 48 hours (Figure 1E) yielded temperature measurements of $28.04^{\circ}\text{C} \pm 0.06^{\circ}\text{C}$ (mean \pm SD). We
69 did not determine whether this variation comes from the heater or the sensor. Thus the sensor is
70 stable within $<0.1^{\circ}\text{C}$, and is more precise than a previously reported similar thermometer [8].



71
72 **Figure 1. Arduino device for measuring temperature and associated interface allows long-term**
73 **recording with a PC interface.** (A) Arduino Uno microcontroller with waterproof temperature sensor
74 DS18B20 in a steel jacket. (B) Wire diagram of Arduino Uno with temperature sensor and light detector
75 attached. The light detector provides analog output that is connected to an analog input of the microcontroller
76 (blue wire), 5V (red wire), and ground (black wire). The temperature sensor DS18B20 uses a 1-wire protocol
77 and is connected to ground (black wire), and a digital pin 2 (yellow wire) with a pull-up resistor to 5V (red
78 wire). (C) Screenshot of Python graphical user interface used to read out temperature measurements using a
79 computer. (D) Continuous recording of water temperature with 2-minute resolution for over 48 hours. Set
80 point was 28°C. The thick line shows smoothed data (sliding window over 200 measurements).

81
82 The open Arduino platform allows use of multiple types of sensors. The photodiode (Osram SFH
83 203, Table 1) that produces current as a function of illumination intensity can be directly connected
84 to analog input to provide measurement of the ambient light level. Continuously monitoring light
85 level or temperature can identify irregularities due to unscheduled entrances by facilities personnel.

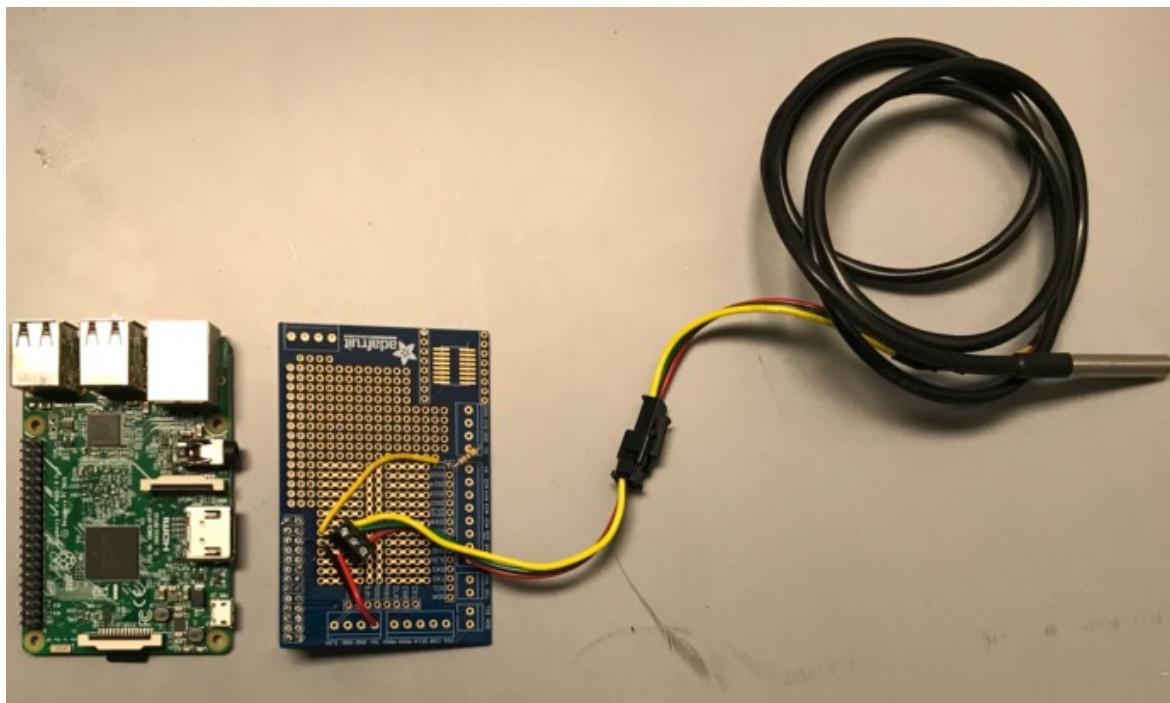
86 We have also tested an analog sensor (ALS-PT19, Table 1) that is calibrated to reflect the human
87 visual spectrum, and a photoresistor (LM393, Table 1).

88

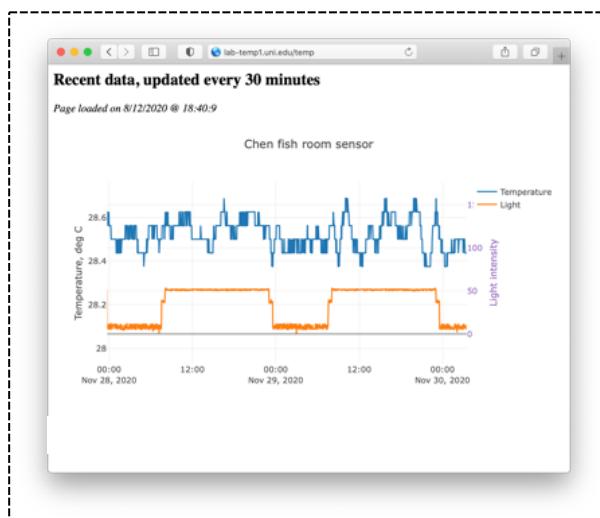
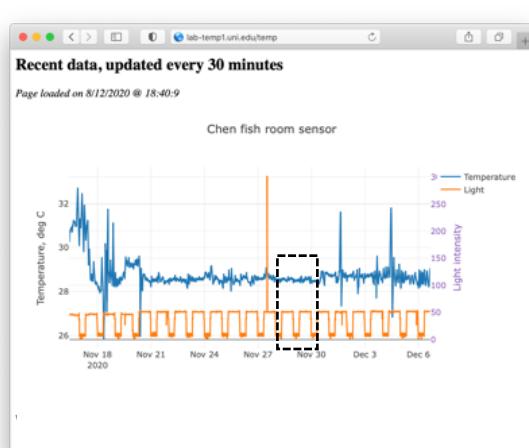
89 Arduino microcontrollers do not directly support internet connectivity that is necessary for remote
90 access to the data. However, they can be combined with an inexpensive and compact Raspberry
91 Pi computer. Using common functions, we implemented an open-source web interface (Figure 2A)
92 that allows access to raw recorded files, as well as interactive plotting of the data (Figure 2B). One
93 scenario where this device was useful for us is a newly established animal facility. Tracking for
94 more than two weeks identified unexpected variations in temperature control and lighting
95 conditions, and provided concrete data that could be used to determine the source of the
96 variations. We collected data on two different animal facilities, before and after the move to a new
97 building (Supplemental Figure 1).

98

A



B

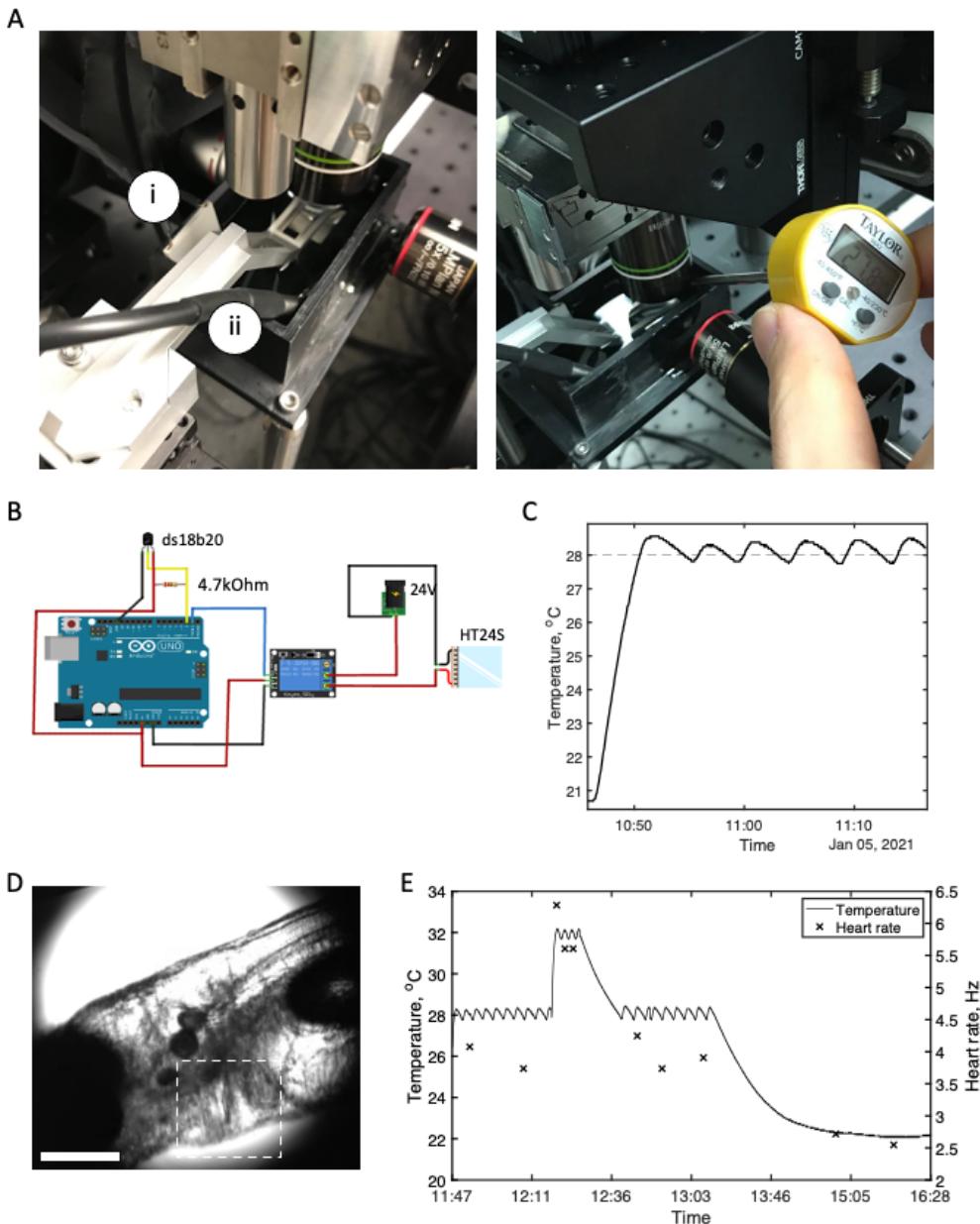


99

100 **Figure 2. Connection of a temperature sensor to a Raspberry Pi computer allows remote monitoring**
101 **of temperature and light levels with 2-minute resolution in an animal facility.** (A) Photograph of
102 Raspberry Pi 3B+ and a corresponding shield with installed DS18B20 water-proof temperature sensor. Green
103 wire connects to ground. Red wire connects to 5V. Yellow wire connects to IO4 pin through 4.7kΩ pull-
104 down resistor. (B) Screenshot of the web interface of conditions recorded over 20 days in a zebrafish animal
105 facility. Temperature and light intensity are reported and the user can interactively expand a region of interest
106 (e.g., dashed box in left panel is enlarged in right panel) and save the plots as an image.

107
108 With a single Arduino device, a temperature sensor can be bundled together with a heater to create
109 a control device to keep conditions at a given temperature during microscopy sessions (Figure 3A).
110 We applied a simple non-linear proportional control process (Code Snippet 5) to regulate the
111 temperature of a 50-mL imaging sample chamber (40 mm x 60 mm x 20 mm, Figure 3A) in a
112 custom light-sheet microscope similar to a previously published design [9]. We used a
113 commercially available ceramic heating element and the thermometer described in the previous
114 section (Arduino-based thermometer). This device brought the temperature within the sample
115 chamber to 28°C from a room temperature of 21°C in less than 5 minutes (Figure 3D). The sample
116 temperature was maintained at $28^{\circ}\text{C} \pm 0.3^{\circ}\text{C}$. We verified temperature values using second
117 calibrated thermometer (Figure 3B). We used Arduino code to read the temperature (T), and if it
118 fell below the target setpoint (T_{ref}), the relay was switched on, closing the circuit and turning the
119 heater on for a given period of time. We found that fast and accurate control is achieved if the
120 minimal time of heater activation is set to be 10 ms, with an additional 100 ms per degree of
121 temperature difference between the reading and the setpoint:

122
123 If $T < T_{\text{ref}}$: turn relay ON for $10 \text{ ms} + 100 \text{ ms}/^{\circ}\text{C} * (T_{\text{ref}} - T)$;
124 If $T \geq T_{\text{ref}}$: wait 10 ms
125



126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141

Figure 3. The Arduino temperature sensor can be modified and programmed to drive proportional control of temperature inside a sample chamber. (A) Assembled device attached to the sample chamber of a custom light-sheet microscope. (i) Heating element (ceramic plate) and (ii) thermometer are submerged in water. Recorded temperature deviated from independent lab thermometer (shown here) by less than 0.3°C. (B) Wiring diagram of the heating device. Thermometer DS18B20 is attached as described in Figure 1B. In addition, the electromechanical relay is connected to 5V (red wire) and ground (black wire) and activated by a digital signal from pin 1 (blue wire). Activation of the relay causes the connection of the heating element to a 24 V power source. By regulating the duration of relay activation, the amount of heat that is added to the chamber is controlled. (C) Ramp of temperature in the chamber from a room temperature of 23°C to 28°C takes less than 5 minutes and temperature is subsequently maintained within $\pm 0.5^\circ\text{C}$. The target temperature is indicated by the dashed line. (D) 5 day old zebrafish (5 dpf) whose heart is imaged using transmitted light. The heart region is indicated with a dashed box, which outlines the region analyzed using Fourier transformation to extract frequency of image change. Scale bar: 200 μm . (E) Simultaneous recording of temperature in the sample chamber and larval zebrafish heart rate. The temperature was first maintained at 28°C, then stepped up to 32°C, and subsequently decreased to room temperature

142 (22°C). Heart rate was recorded as described in Supplemental Figure 2. Without active cooling, it took ~10
143 minutes for the temperature to decrease from 32°C to 28°C.

144
145 Heating a 50 mL water chamber from 28°C to 32°C took less than 1 minute (Figure 3D). Return
146 to 28°C took 10 minutes without any active cooling device with a room temperature of 22°C. We
147 observed effects of temperature on zebrafish heart rate using imaging at different temperatures.
148 We used a behavioral camera on our light-sheet microscopy setup to register heart movement,
149 imaging at 40 frames per second (Supplemental Figure 2). Extraction of heart rate from the
150 imaging data was done using an image difference approach and a Fast Fourier Transform. Heart
151 rate correlated with temperature as previously described [10], changing from 4 Hz at 28°C to 6 Hz
152 at 32°C, and then to 2.5 Hz at 25°C (Figure 3F).

153

154 Discussion

155 We described an open-source temperature recording and control scheme based on Arduino
156 microcontroller architecture using commodity parts and open-source software. We showed how
157 this platform can be expanded to provide long-term recording of light and temperature conditions
158 in an animal facility. Furthermore, an Arduino platform can be used as a basis for a custom
159 temperature control device to keep a light-sheet microscope sample chamber at a constant
160 temperature.

161

162 Accurate reporting of temperature is important during animal experiments. By specification, the
163 DS18B20 thermometer has $\pm 0.5^\circ\text{C}$ accuracy, and might need calibration. We have not done
164 extensive calibration, but temperatures reported by the DS18B20 sensor consistently agreed with
165 other thermometers to within 0.5°C.

166

167 Open-source hardware allows expansion of the set of sensors. This allows continuous collection
168 of values of multiple environmental parameters, unlike the usual practice of occasional checks.
169 Temperature and humidity can be monitored using a single sensor (e.g., DHT22). A vibration
170 sensor (using an analog accelerometer such as ADXL335) can be used to record and validate a
171 vibration-based assay, such as tapping a plate containing zebrafish larvae, which can be used as an
172 arousal threshold assay [11]. A microphone can be installed for experiments that involve sound
173 stimulation. Sensors can be combined according to the needs of a particular experiment to provide
174 continuous recording and calibration of the assay, as well as provide data for reporting
175 experimental methods.

176

177 The sample chamber temperature described here used a simple proportional control algorithm. It
178 can be improved by adding a power-regulator (e.g., using a motor driver) and adding integrated
179 and differential control. Heating elements and thermometers come in different form-factors (such
180 as rods, circular elements, or flexible films), and potentially can be adapted for various
181 microscopes, imaging chambers, or experimental platforms, without changing the principle of
182 operation.

183

184 Attaching sensors to a computer allows more external software to be run, such as email alerts and
185 remote access to data on a network. We integrated a simple web interface to observe temperature
186 and light conditions in an animal facility, but issues of security and continuous operation of the
187 computer are often not trivial. We coordinated with our campus to block all incoming internet

188 access to our device, only allowing access via a virtual private network. We also disabled access
189 using passwords, relying only on public/private key pair authentication. Attaching simple devices
190 to outdated campus networks can potentially compromise IT security.
191

192 The design and approach presented here allows continuous automatic reporting of environmental
193 data, and calibration of a wide range of experimental setups that can result in decreased
194 experimental variability. We hope that open-source devices such as the device presented here will
195 enable scientists to track and report data about experimental conditions in their publications.
196

197 Methods

198 The DS18B20 waterproof temperature sensor uses a OneWire protocol [12] that requires only one
199 data pin in addition to a power (5V) and ground connection. In order to create the Arduino-based
200 temperature and light recorder, we connected the DS18B20 sensor probe onto the Arduino shield.
201 As shown in Figure 1A, the yellow data wire was linked to digital port 2, the red wire to 5V, and
202 the blue wire to GND. The OneWire protocol requires the use of a 4.7-k Ω pullup resistor that
203 connects the data wire of the sensor to the 5V wire. For the Raspberry Pi version of the device, the
204 data wire was connected to IO4 pin, otherwise wiring is unchanged. We followed a tutorial
205 accessible at <https://bit.ly/3u5e0uJ>.
206

207 To record ambient light levels, we used a LM393 photo-resistive light sensor. Similar to the
208 temperature probe, the LM393 photo-resistive light sensor has 3 wires, COM, 5V, and GND. The
209 COM wire was connected to the Arduino A0 analog input pin, the 5V wire was connected to the
210 board 5V pin, and the GND wire was connected to the ground pin.
211

212 Another light sensor that we tested, ALS-PT19, is an analog RoHS-compliant sensor with ground,
213 3–5 V, and data pins. It was wired in the same way as LM393.
214

215 Wire diagrams were created using Fritzing [13]. We used MATLAB versions 2018b and 2020b,
216 and Python version 3.1. We tested Python and MATLAB code on Windows 7, Windows 10, and
217 Mac OS X 10.15.
218

219 The Raspberry Pi version of the thermometer/light-meter was assembled similarly to the Arduino
220 version. The source code is available at <https://github.com/aandreev0/raspb-temp-logger>.
221 The installation process consists of these steps:

- 222 1. Install recent version of Raspbian operating system
- 223 2. Connect to the Raspberry Pi using an external display and keyboard
- 224 3. Get the MAC address and communicate with the IT services of your institution to allow
225 access via a separate domain name, for example “lab-thermo1.uni.edu”.
 - 226 a. MAC address is shown on the start-up screen or using the `ifconfig` command
227 and looking for line `HWaddr: AA:BB:CC:DD:EE:FF`

```
[pi@retropie:~ $ ifconfig
eth0      Link encap:Ethernet HWaddr b8:27:eb:b6:65:eb
          inet addr:131.215.52.121 Bcast:131.215.52.255 Mask:255.255.255.0
          inet6 addr: fe80::5fb1:e273:761:9afe/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:7434728 errors:0 dropped:7724 overruns:0 frame:0
          TX packets:176005 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:596614613 (568.9 MiB) TX bytes:191593702 (182.7 MiB)
```

- 228
229 4. On a personal or lab workstation configure password-less SSH authentication using a
230 private/public key pair:
231 a. Follow the guide for generation of a private/public key pair on your system:
232 i. On Linux-type systems and Mac OS X use “ssh-keygen -t rsa”
233 command
234 ii. On Windows systems use PuTTy or OpenSSH, and follow instructions for
235 the corresponding terminal
236 b. Add a public key to Raspberry Pi’s ~/.ssh/authorized_keys file
237 5. Connect the Raspberry Pi to the internet using an ethernet cable.
238 6. Continue working directly in the Raspberry Pi terminal or connect to it using SSH (step 4).
239 7. Copy code from GitHub repository to ~/raspb-temp-logger using “git clone
240 https://github.com/aandreev0/raspb-temp-logger.git”.
241 8. Install the necessary Python packages using the “pip install” command.
242 9. Install Nginx server using “sudo apt install nginx”.
243 10. Configure Nginx server to serve the content of ~/raspb-temp-logger:
244 a. Edit the Nginx configuration file using “sudo nano /etc/nginx/sites-
245 available/default”
246 b. Change the line “root /var/www/html/;” to “root /home/raspb-temp-
247 logger/”
248 c. Save and close the file using CTRL+X
249 d. Restart Nginx using the command “sudo systemctl restart nginx”
250 11. Configure crontab to automatically run the temperature-recording script every 2 minutes
251 using “sudo crontab -e”
252 a. Add line:
253 “*/2 * * * * python /home/pi/raspb-temp-logger/report_temp.py”
254 12. Configure crontab to automatically run the generation of plots every 30 minutes using the
255 same command
256 a. Add line “*/30 * * * * python /home/pi/raspb-temp-
257 logger/csv_to_json.py”
258 b. Close crontab by pressing CTRL+X
259 13. Test by accessing the domain name or IP address
260
261 To generate plots, we used the open-source Python library *plotly* and custom code that generates
262 a javascript data structure file (JSON file) from CSV files with recorded data.
263
264 The light meters we used are analog, with output voltage between 0V and 5V. The Raspberry Pi
265 3B+ lacks a built-in analog-to-digital converter (ADC) and cannot work with these devices

266 directly. We therefore used an Arduino as an ADC. The light-measuring device was plugged into
267 the analog input of Arduino Uno, and the Arduino was connected to a Raspberry Pi through a USB
268 cable. We used similar code to write light measurement to serial port, and read serial port using
269 Python script (Code Snippet 6).

270

271 Heart rate recording was performed on larval zebrafish at 5 days post fertilization by imaging using
272 infrared trans-illumination with an LED (780nm, Thorlabs) and a CMOS camera at 40 frames per
273 second. We used an air 4x objective with 0.1 NA (Edmund Optics). To process the video, we first
274 calculated image difference between adjacent frames (temporal differentiation), followed by a Fast
275 Fourier Transform of average intensity within a region of interest containing the heart (see
276 Supplemental Figure 2 for details). Code of the plugin for Fiji image analysis software is available
277 at <https://github.com/aandreev0/fiji-heartrate>.

278

279 Acknowledgements

280 We thank Prober lab members for providing animals for the experiments, and especially Dr. Amina
281 Kinkhabwala for testing the design and instructions. The first version of the code used to extract
282 heart rate from images was developed by A.A. during work in Truong/Fraser lab at University of
283 Southern California. This work was supported by grants from the NIH to D.A.P. (R35 NS122172
284 and R01 MH121601).

285

286 **References**

287

- 288 [1] G. Folguera *et al.*, “An experimental test of the role of environmental temperature variability
289 on ectotherm molecular, physiological and life-history traits: Implications for global
290 warming,” *Comp. Biochem. Physiol. A. Mol. Integr. Physiol.*, vol. 159, no. 3, pp. 242–246,
291 Jul. 2011, doi: 10.1016/j.cbpa.2011.03.002.
- 292 [2] K. Lahiri, D. Vallone, S. B. Gondi, C. Santoriello, T. Dickmeis, and N. S. Foulkes,
293 “Temperature Regulates Transcription in the Zebrafish Circadian Clock,” *PLOS Biol.*, vol.
294 3, no. 11, p. e351, Sep. 2005, doi: 10.1371/journal.pbio.0030351.
- 295 [3] A. Abozaid, B. Tsang, and R. Gerlai, “The effects of small but abrupt change in temperature
296 on the behavior of larval zebrafish,” *Physiol. Behav.*, vol. 227, p. 113169, Dec. 2020, doi:
297 10.1016/j.physbeh.2020.113169.
- 298 [4] H. Dvir, I. Elbaz, S. Havlin, L. Appelbaum, P. C. Ivanov, and R. P. Bartsch, “Neuronal noise
299 as an origin of sleep arousals and its role in sudden infant death syndrome,” *Sci. Adv.*, vol. 4,
300 no. 4, p. eaar6277, Apr. 2018, doi: 10.1126/sciadv.aar6277.
- 301 [5] J. Schaefer and A. Ryan, “Developmental plasticity in the thermal tolerance of zebrafish *Danio*
302 *rerio*,” *J. Fish Biol.*, vol. 69, no. 3, pp. 722–734, 2006, doi: <https://doi.org/10.1111/j.1095-8649.2006.01145.x>.
- 303 [6] M. Haeri, P. D. Calvert, E. Solessio, E. N. P. Jr, and B. E. Knox, “Regulation of Rhodopsin-
304 eGFP Distribution in Transgenic Xenopus Rod Outer Segments by Light,” *PLOS ONE*, vol.
305 8, no. 11, p. e80059, Nov. 2013, doi: 10.1371/journal.pone.0080059.
- 306 [7] N. D. Testa, S. Kaul, K. N. Le, M. Zhan, H. Lu, and A. B. Paaby, “A portable, low-cost device
307 for precise control of specimen temperature under stereomicroscopes,” *bioRxiv*, p.
308 2019.12.19.882605, Dec. 2019, doi: 10.1101/2019.12.19.882605.
- 309 [8] B. D. Knapp, L. Zhu, and K. C. Huang, “SiCTeC: An inexpensive, easily assembled Peltier
310 device for rapid temperature shifting during single-cell imaging,” *PLOS Biol.*, vol. 18, no.
311 11, p. e3000786, Nov. 2020, doi: 10.1371/journal.pbio.3000786.
- 312 [9] K. Keomanee-Dizon, S. E. Fraser, and T. V. Truong, “A versatile, multi-laser twin-microscope
313 system for light-sheet imaging,” *Rev. Sci. Instrum.*, vol. 91, no. 5, p. 053703, May 2020, doi:
314 10.1063/1.5144487.
- 315 [10] J. Gierten *et al.*, “Automated high-throughput heartbeat quantification in medaka and
316 zebrafish embryos under physiological conditions,” *Sci. Rep.*, vol. 10, no. 1, Art. no. 1, Feb.
317 2020, doi: 10.1038/s41598-020-58563-w.
- 318 [11] S. Chen, S. Reichert, C. Singh, G. Oikonomou, J. Rihel, and D. A. Prober, “Light-Dependent
319 Regulation of Sleep and Wake States by Prokineticin 2 in Zebrafish,” *Neuron*, vol. 95, no. 1,
320 pp. 153–168.e6, Jul. 2017, doi: 10.1016/j.neuron.2017.06.001.
- 321 [12] R. Downs, “Using 1-Wire I/O for distributed system monitoring,” in *Wescon/98. Conference
322 Proceedings (Cat. No.98CH36265)*, Sep. 1998, pp. 161–168. doi:
323 10.1109/WESCON.1998.716439.
- 324 [13] A. Knörig, R. Wettach, and J. Cohen, “Fritzing: a tool for advancing electronic prototyping
325 for designers,” in *Proceedings of the 3rd International Conference on Tangible and
326 Embedded Interaction*, New York, NY, USA, Feb. 2009, pp. 351–358. doi:
327 10.1145/1517664.1517735.
- 328

329 **Code availability:**

330

- 332 • Code for the Raspberry Pi-based web interface is available at
333 <https://github.com/aandreev0/raspb-temp-logger>
334 • Code of the heart rate extraction plugin for Fiji is available at
335 <https://github.com/aandreev0/fiji-heartrate>

336

337

338 **Code**

339 **Code snippet 1: Arduino interaction with DS18B20 and analog light sensor (LM393)**

```
340
341     void setup(void)
342     {
343         Serial.begin(9600);
344         sensors.begin();
345     }
346     void loop(void)
347     {
348         sensors.requestTemperatures();
349         byte lightValue = analogRead(0); // Read light value from port A0
350         Serial.print(sensors.getTempCByIndex(0)); // Read temperature value
351         Serial.print(",");
352         Serial.println(lightValue);
353         delay(800); // wait 800ms
354     }
355
```

356 **Code snippet 2-1: Python code to get recording via serial port on PC**

```
357
358     import serial
359     import serial.tools.list_ports
360
361     def findArduino(ports):
362         commPort = 'None'
363         numPorts = len(ports)
364         for i in range(0, numPorts):
365             strPort = str(ports[i])
366             if 'Arduino' in strPort:
367                 splitPort = strPort.split(' ')
368                 commPort = (splitPort[0])
369         return commPort
370
371     ports = serial.tools.list_ports.comports()
372     connectPort = findArduino(ports)
373
374     if connectPort != 'None':
375         s = serial.Serial(connectPort,baudrate= 9600)
376         print ('Connected to ' + connectPort)
377     else:
378         print ('Can\'t find Arduino device')
379
380     dataStr = s.readline().decode("utf-8")
381     dataArray = dataStr.split(',')
382     temperature = float(dataArray[0])
383     light = float(dataArray[1])
```

```
384
385     print([temperature, light])
386     s.close()
387
```

388 **Code snippet 2-2: Python code to make GUI with data plot**

```
389
390     def drawn():
391         count = 0
392         while True:
393             arduinoString = s.readline().decode("utf-8")
394             dataArray = arduinoString.split(',')
395             temp = float(dataArray[0])
396             L = float (dataArray[1])
397             tempC.append(temp)
398             light_sensor.append(L)
399             drawnow(makeFig)
400             plt.pause(.000001)
401             count = count + 1
402             if (count > 50):
403                 tempC.pop(0)
404                 light_sensor.pop(0)
405
```

406 **Code snippet 3: MATLAB code to get recording via serial port**

```
407
408     port = 'COM4';
409     s = serial(port, 'Terminator', 'LF');
410     fopen(s)
411     getString = fgets(s);
412     data = regexp(getString, ',', 'split');
413     temperature = str2num(data{1}) % temperature in C
414     light      = str2num(data{2}) % light in a.u.
415     fclose(s)
416
```

417 **Code snippet 4: Python code to read temperature from Raspberry Pi**

```
418
419     import os
420     import glob
421     import sys
422     import serial
423
424     os.system('modprobe w1-gpio')
425     os.system('modprobe w1-therm')
426
427     light_arduino = '/dev/'
428
429
430     def read_temp_raw():
431         base_dir = '/sys/bus/w1/devices/'
432         devs = glob.glob(base_dir + '28*')
433         if len(devs) > 0:
434             device_folder = devs[0] # only picks 1st device
435             device_file = device_folder + '/w1_slave'
```

```
437         f = open(device_file, 'r')
438         lines = f.readlines()
439         f.close()
440         return lines
441     else:
442         return False
443
444     def read_temp():
445         lines = read_temp_raw()
446         if not lines:
447             return 0
448
449         while lines[0].strip()[-3:] != 'YES':
450             time.sleep(0.2)
451             lines = read_temp_raw()
452             equals_pos = lines[1].find('t=')
453             if equals_pos != -1:
454                 temp_string = lines[1][equals_pos+2:]
455                 temp_c = float(temp_string) / 1000.0
456                 temp_f = temp_c * 9.0 / 5.0 + 32.0
457                 return temp_c
458
459
```

Code snippet 5: Arduino code for the proportional control of sample chamber heater

```
460
461     #include <OneWire.h>
462     #include <DallasTemperature.h>
463     #define ONE_WIRE_BUS 2 // DS18B20 COM is plugged into pin A2
464     int relayPin = 5;      // digital pin used to switch relay on/off
465
466     OneWire oneWire(ONE_WIRE_BUS);
467     DallasTemperature sensors(&oneWire);
468     float temp;
469     float set_temp = 28.5; // target temperature
470     int relayStatus; // status of the switch: 1=open, 0=closed
471
472     void setup(void)
473     {
474         Serial.begin(9600);
475         sensors.begin();
476         pinMode(relayPin,OUTPUT);
477         digitalWrite(relayPin, LOW);
478         relayStatus = 0;
479     }
480
481     void loop(void)
482     {
483         sensors.requestTemperatures(); // get temperature reading
484         temp = float(sensors.getTempCByIndex(0));
485
486         // print current time, current remperature, and relay position
487         Serial.println(String(millis())+":"+String(temp)+","+relayStatus);
488
489         if(temp < set_temp){
490             digitalWrite(relayPin,HIGH);           // turn heater ON
```

```
491     relayStatus = 1;
492     delay(10 + int((set_temp - temp)*100)); // wait 10ms+100ms/deg C
493 }else{
494     digitalWrite(relayPin,LOW);           // turn heater OFF
495     relayStatus = 0;
496     delay(10);
497 }
498 }
499
```

500 **Code snippet 6: Python code for light measurement from serial port in Raspberry Pi**

```
501
502     import serial
503     def read_light():
504         ser = serial.Serial('/dev/ttyACM0') # open serial port
505         avg_light = 0;
506         for i in range(0,5):
507             line = ser.readline()    # read a '\n' terminated line
508             if i>2:
509                 avg_light += int(line.strip())
510
511         avg_light = avg_light / 2.0;
512         ser.close()
513         print("Averaged light:" + str(avg_light))
514         return avg_light
515
```

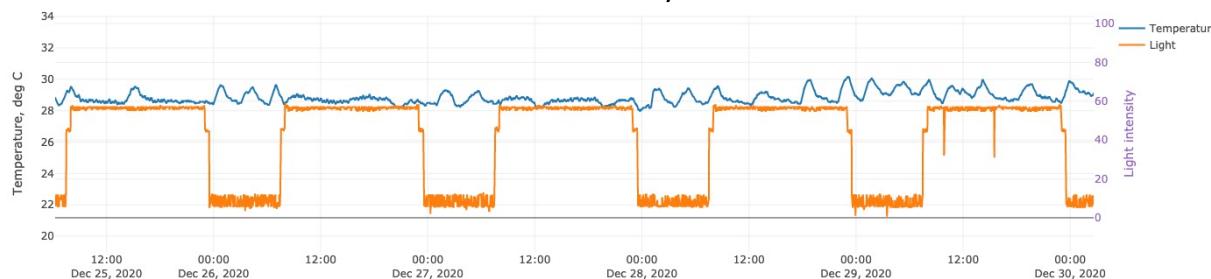
516 **Table 1. List of parts.**

Part name	Distributor ID	Cost
Arduino Uno	Amazon: A000066	\$23.00
Waterproof temperature Sensor	Adafruit: DS18B20	\$9.95
4.7 kΩ Resistor	Digikey: 294-4.7K-RC	\$0.15
Arduino shield	Digikey: 2077	\$9.95
USB B cable	Adafruit: 62	\$2.95
<i>Same as Arduino Temperature Recorder, plus:</i>		
24-W Metal Ceramic Heater, 20x20 mm	Thorlabs: HT24S	\$47.00
Power supply, 24 V	Digikey: QFWB-65-24-US01	\$17.00
Electromechanical switch or	Digikey: DFR0017	\$5.00
Solid-state relay or	Digikey: 101020603	\$6.00
DC motor driver	Digikey: 1528-1719-ND	\$7.50
<i>Same as Arduino Temperature Recorder, plus:</i>		
Raspberry Pi Model 3B+	Adafruit: 3775	\$35
Raspberry Pi Shield	Adafruit: 801	\$15.95
<i>Same as Arduino Temperature Recorder, plus:</i>		
Analog light sensor or	ALS-PT19	\$2.50
Photodiode or	Osram SFH 203	\$0.80
Analog light sensor	LM393	\$1.15

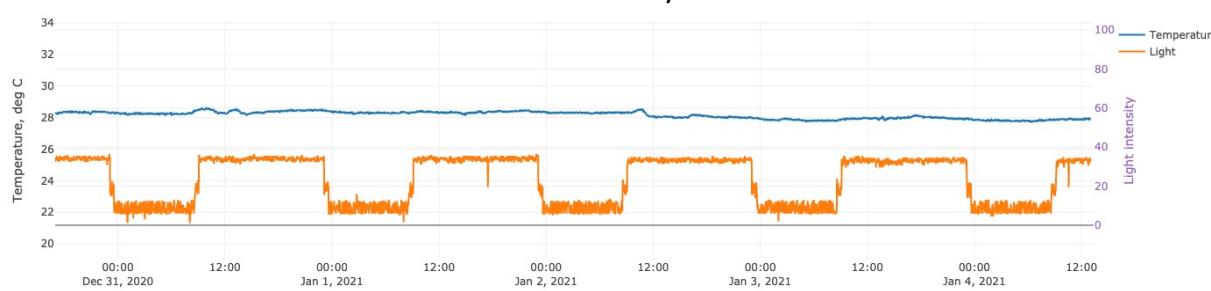
518 **Supplemental Figures**

519

Zebrafish facility #1



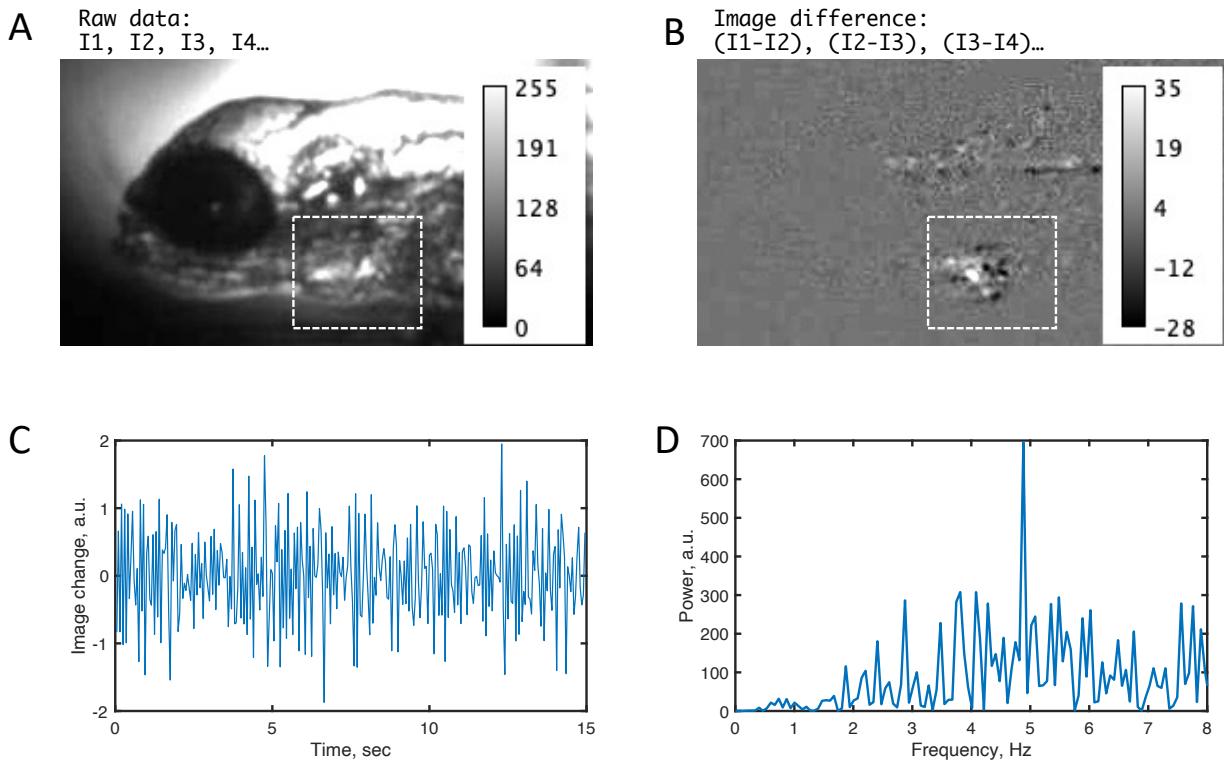
Zebrafish facility #2



520

521 **Supplemental Figure 1. Continuous recordings of temperature and light levels from two**
522 **zebrafish facilities.** Data was collected using the same Raspberry Pi device with web interface.
523 Light level has two values above dark: white lights that were turned on from 9 am until 11 pm,
524 and red lights that were turned on for 30 minutes immediately before and after white lights were
525 turned on during the day. The two facilities are located in different buildings on the same campus.
526 Data was collected in 1 minute intervals.

527



Supplemental Figure 2. Heart rate extraction from video recording of larval zebrafish heart.
 (A) Trans-illuminated microscopy images of a zebrafish heart were acquired at 20 frames per second as a sequence of frames ($I_1, I_2, I_3\dots$). The white box outlines the location of the heart. (B) Image difference between consecutive frames was generated using the Image Calculator function of Fiji. (C) Average image intensity in the region of interest (ROI) around the heart of the image-difference data was extracted. (D) Fourier-transformed data shows a peak at 5 Hz.

528
 529
 530
 531
 532
 533
 534
 535