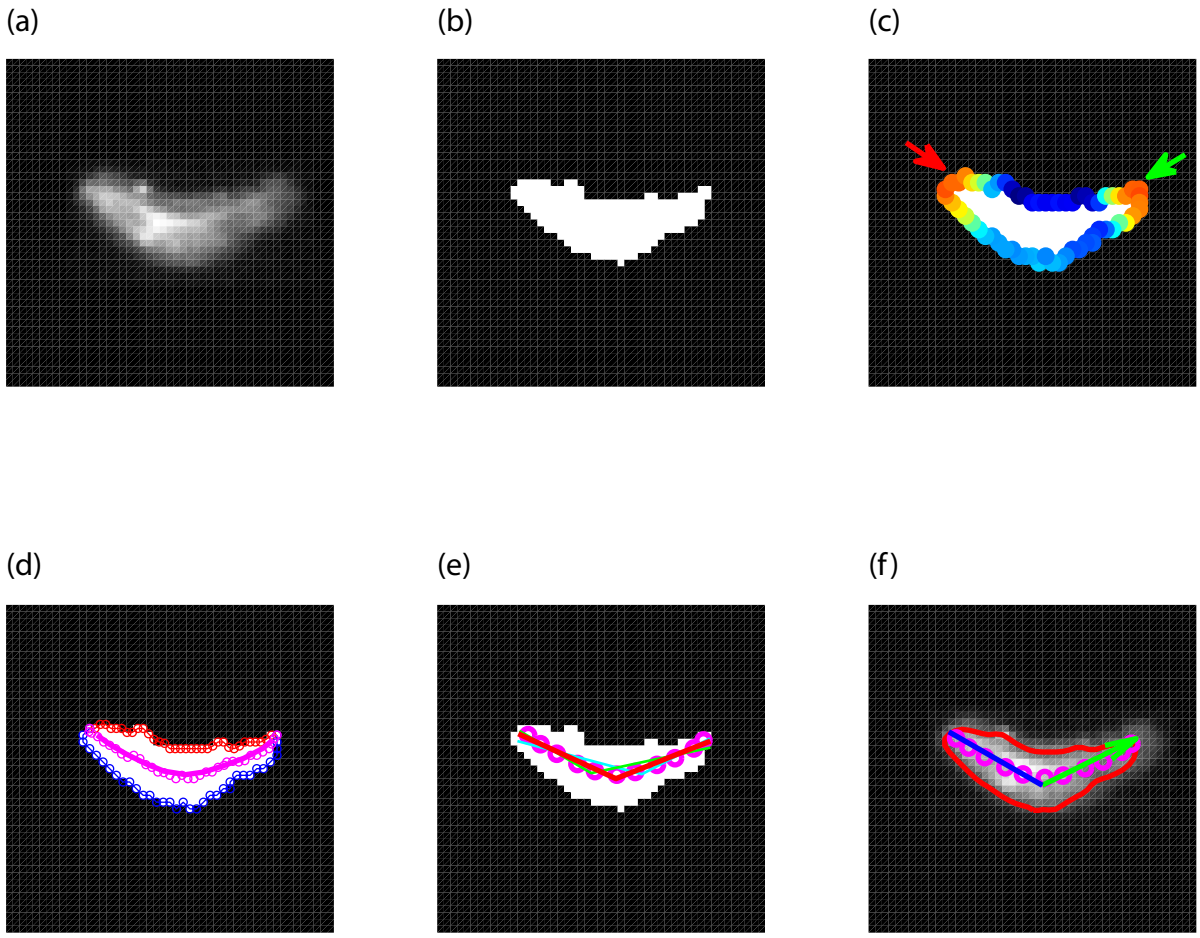
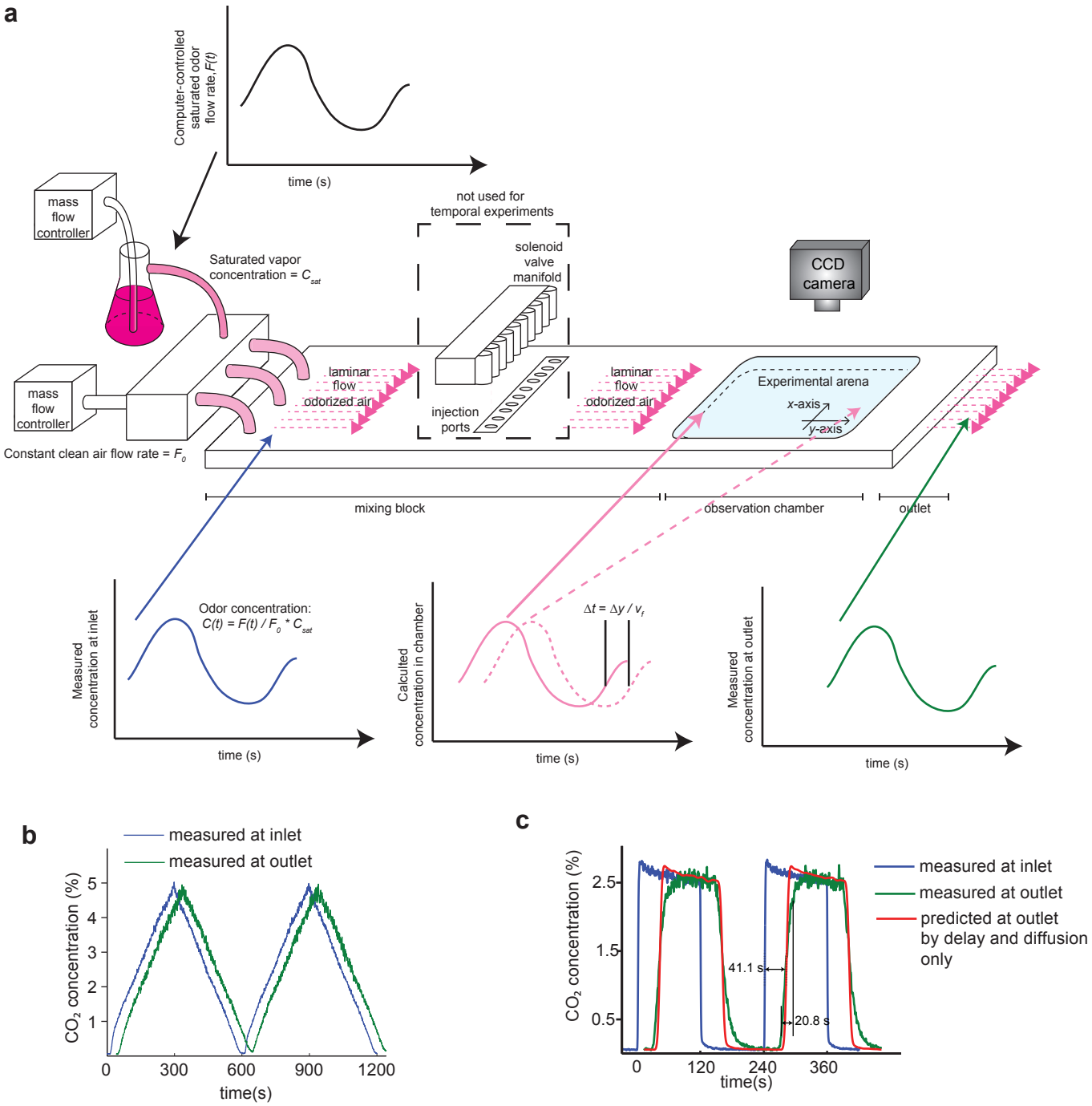
Supplementary Figure 1: Feature Extraction during MAGAT Analysis.



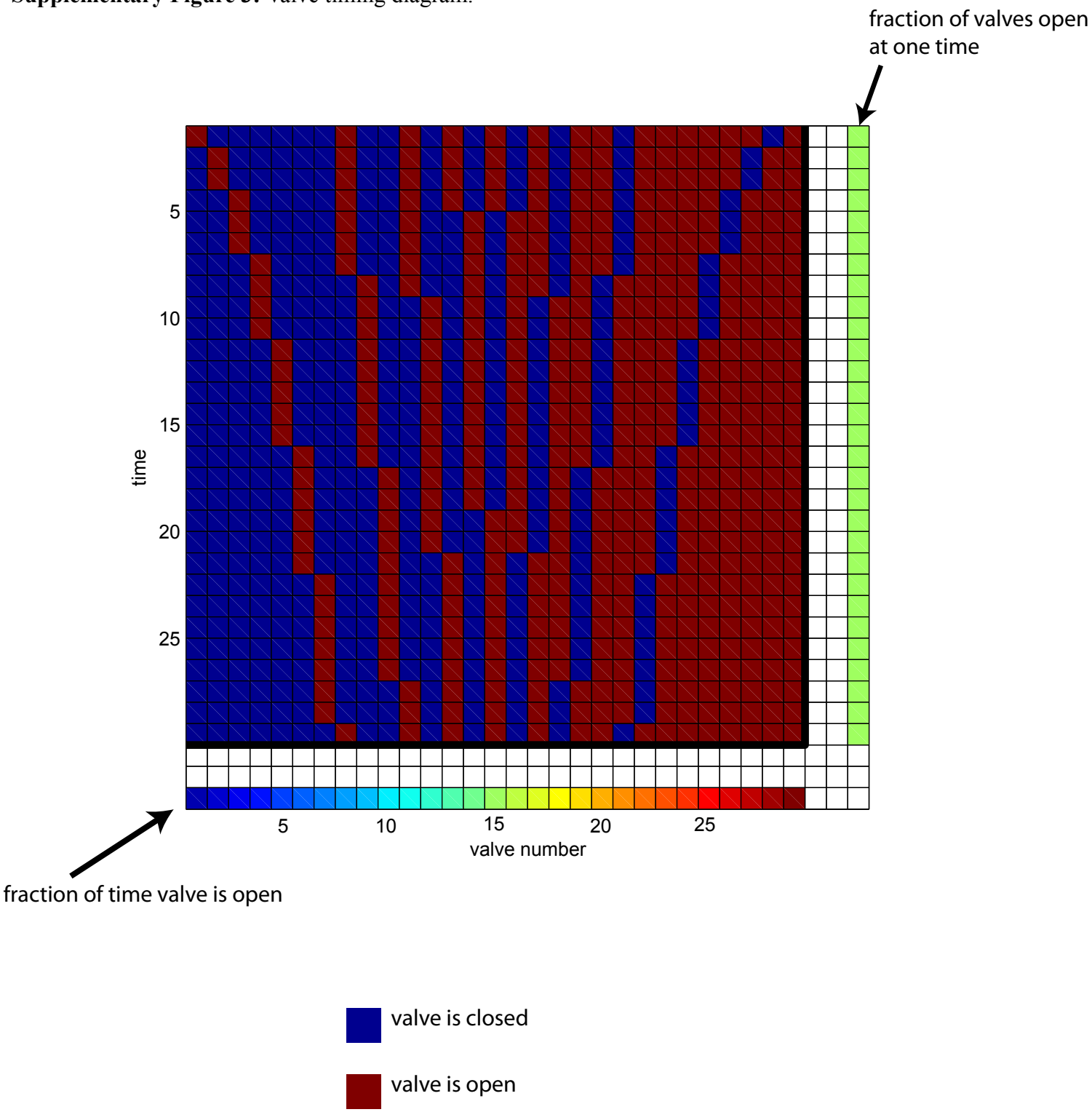
Supplementary Figure 1: Feature Extraction during MAGAT Analysis. (a) Grayscale image of a larva. (b) thresholded image. (c) Contour of the larva with curvature indicated by color (red = highest curvature). Arrows indicate determined head/tail locations (at this stage of the analysis, the head is not differentiated from the tail). (d) Contour divided into two segments (red and blue) resampled to have equal numbers of points, midline (purple) is the point by point average of these two segments (e) points along the midline (purple circles) and prospective fits of the midline to two straight lines; the thicker red line fits best (f) grayscale image of the the larva overlaid by the contour (red line), midline (purple circles), and prospective tail-mid and mid-head lines.

Supplementary Figure 2: Temporal Gradients



Supplementary Figure 2: Temporal Gradients. (a) Schematic of setup for temporal gradient experiments. The valve array is bypassed. Odor/gas is combined with clean air prior to the device inlet. Gas concentration is manipulated by setting the flow rate of odor/gas relative to the flow rate of clean air. The concentration in the chamber is constant in the x-direction and varies in the y-direction as the temporal waveform flows across the chamber. The concentration at any point in the chamber can be calculated using the measured inlet concentration, the y-position and the gas flow rate (v_f). (b) Carbon dioxide concentration during triangle wave (same conditions as Figure 5a, reproduced from Figure 1). (c) Carbon dioxide concentration measured during square wave (same conditions as Figure 5b). 2 complete periods are shown. Blue line – concentration measured immediately upstream of the mixing block inlet . Green line – concentration measured immediately downstream of flow chamber outlet; Orange line – expected measurement at outlet taking into account diffusion and flow time from inlet to outlet. The time delay (41.1 s) agrees with the prediction obtained by dividing the chamber volume by the gas flow rate. The rise time (20.8 s) is longer than that predicted by diffusion.

Supplementary Figure 3: Valve timing diagram.



Supplementary Figure 3: Valve timing diagram. Time in the cycle progresses vertically from top to bottom, valve number increments horizontally from left to right. Blue squares indicate the valve is closed at a particular time in the cycle, red squares that the valve is open.

Supplementary Table 1: Fit of statistical model to reorientation after turning decisions

Description of null model	Parameter(s) set to zero	Change in log-likelihood of fit to navigational data	
		EtAc gradients (Figure 3)	CO_2 gradients (Figure 4)
No bias in turn direction	*A*	-19.3 ($p<0.01$)	-12.3 ($p<0.01$)
No bias in turn magnitude	*B,C*	-30.2 ($p<0.01$)	-4.8 ($p<0.01$)
No skew	α,*C*	-23.5 ($p<0.01$)	-132.0 ($p<0.01$)
No bias	*A,B,C*	-51.8 ($p<0.01$)	-17.8 ($p<0.01$)

Recall:

$$P(\Delta\theta|\theta_i) = (\frac{1}{2} - A\sin(\theta_i - \theta_0)) \times SN(\Delta\theta, \mu - B\cos(\theta_i - \theta_0), \sigma, \alpha - C\cos(\theta_i - \theta_0))$$

$$+ (\frac{1}{2} + A\sin(\theta_i - \theta_0)) \times SN(-\Delta\theta, \mu - B\cos(\theta_i - \theta_0), \sigma, \alpha - C\cos(\theta_i - \theta_0))$$

where

$$SN(x, \mu, \sigma, \alpha) = \frac{e^{-\frac{(x-\mu)^2}{2\sigma^2}}}{(\sigma\sqrt{2\pi})} Erfc\,(\frac{-\alpha(x-\mu)}{\sigma\sqrt{2}})$$

Supplementary Note 1: Construction Details and Logistics

The apparatus consists of these major components: a computer controllable source of carbon dioxide or odorant, a microcontroller actuated valve array for generating a stepwise linear gradient, a linear flow chamber where experiments are carried out, camera and lighting for observing and recording behavior, and ancillary structures for support, light management, and gas handling. Plans and schematics for these components are available from the Samuel lab website: http://worms.physics.harvard.edu.

Computer controlled odor/carbon dioxide source

The odor source is based around an AALBORG mass flow controller (MFC) and interfaces to the computer via a LabJack USB data acquisition device. We use Labview to control the LabJack, but drivers are available for a number of programming languages. For carbon dioxide delivery, the MFC is calibrated for CO_2 and used to meter gas directly to the valve array device (for spatial gradients) or to the flow chamber inlet (for temporal gradients). For odor delivery, the MFC is calibrated for and used to meter clean air into a bubbler, which then connects to the valve array or the flow chamber inlet. A second MFC controlled by the same LabJack is used to provide carrier air flow.

The odor source can be made from off-the-shelf components in less than one day. Familiarity with compression fittings (e.g. Swagelok) is required as is some basic computer programming.

Microcontroller actuated valve array

The valve array consists of 29 Lee company LHD Series miniature solenoid valves mounted on a custom-machined gas handling manifold, a custom designed circuit board, and a flow block to mix the output of the valves with air prior to delivery to the flow chamber. The valve manifold is connected to the mixing block via Teflon tubing and secured using 10-32 flat bottom fittings, both from Upchurch Scientific. Plans for the circuit board, manifold, and mixing block area available on the website.

We estimate that machining of the manifold, mixing block, outlet and accessories will require 20-30 hours of machine shop time. The circuit board can be ordered from a PCB manufacturing company (we favor Advanced Circuits) with a lead time of 5 days and requires a few hours of soldering to assemble. Source code to program the microcontroller is included on the website.

The valve array as we constructed it requires a good deal of custom work. However, there are opportunities for later iterations to use more off-the-shelf or user-friendly components. In particular, one might take advantage of the increasing availability of USB-programmable microcontroller development boards (e.g. the Arduino and the Teensy) and also of the number of manifolds the lee company makes to fit its LHD Series solenoid valves.

Assembly of the components into the completed valve array will require a day of work, with an hour or two devoted to cutting the tubing to length and attaching the fittings securely in a somewhat confined space. Some soldering, microcontroller programming, and mechanical assembly are required.

Valve timing algorithm

The valves are timed so that each valve is open for one continuous block of time during each duty cycle (1 second). To create a linear gradient, the length of time each valve is open is linearly proportional to its position in the array. The valves are opened and closed at specific times during the duty cycle so that exactly half are open at any one point. A simple graphical algorithm for generating a valve timing sequence that achieves these goals for N valves is as follows:

Take a sheet of graph paper and mark off an NxN block of squares. Each column represents a valve and each row represents a fraction (1/N) of the duty cycle. Place a single X in the upper left box. This indicates valve 1 is open for the first fraction of the duty cycle only. Now move down and to the right one box and place two Xs in the second column in rows two and three, indicating that valve 2 is open for the second and third fractions of the duty cycle. Repeat this process, marking M boxes in column M, beginning with the box to the lower right of the last box marked in column M-1. When you reach the bottom row, wrap back to the top row and continue marking. The finished graph paper is a valve timing diagram; each valve is open for the portion of the duty cycle marked with Xs. An example of this timing diagram for 29 valves is shown in **Supplementary Fig. 3**.

Flow/Observation Chamber

The flow chamber is machined from a solid block of aluminum and anodized. Plans are available on the website. We estimate 10 hours of shop time to machine the flow block. A glass lid (available from any glass shop) is sealed to the roof of the flow chamber using an o-ring seal. To compress the glass against the o-ring we built a custom pneumatic hinge from 80/20 t-slot extrusion (plans available in the supplement).

Assembly of the pneumatic hinge and connection of the flow cell to the mixing block and outlet will take less than half a day and requires no special expertise.

Camera and Lighting

Lighting is provided by a set of red LEDs (624 nm, 30 degree, CREE C503B-RCN-CW0Z0AA1) mounted on the pneumatic hinge. We were unsatisfied with commercially available led light sources, so we fabricated our own. Plans for these LED light bars are included in the supplement and can be ordered from Advanced Circuits with a lead time of 5 days. We used a 5 megapixel USB 2.0 camera from Mightex, but any high pixel density camera that supports a frame rate of at least 5 Hz may be used. We favored the Mightex camera for its low cost, but others might choose a more user friendly camera that features better integration with third party software.

Assembly of the light bars requires cutting the PCB and spending a few hours soldering. Mounting the light bars and the camera to the setup requires less than an hour.

Support Structure

To exclude light and provide rigid support for the camera and apparatus, we constructed an enclosure from 80/20 t-slotted extrusion and ABS paneling. Plans and a list of materials for this enclosure are included on the website. The camera was connected to the enclosure using standard optical mounting components from Thorlabs. We found that a blackout curtain attached to the front of the enclosure with Velcro was the most convenient way to block light from the experiment while maintaining ease of access.

Depending on one's familiarity with t-slot extrusion, assembly of the enclosure will take between an hour and half a day.

Gas handling and additional materials

The apparatus requires at a minimum a source of clean compressed air at approximately 20 psi. Additionally, for carbon dioxide experiments, a source of compressed carbon dioxide at similar pressure is required. For carbon dioxide experiments, no special handling of the exhaust is required, but for ethyl acetate or other volatile odorants, it may be desirable to vent the exhaust outside the lab to avoid disturbing other experiments. If toxic odorants are used, special handling is required, probably involving placing the apparatus in a fume hood.

In addition to the materials mentioned in the preceding section, assembly of the apparatus will require a standard assortment of fittings, valves, tubing, and hardware. Low pressure regulators can be desirable, especially as building compressed air and carbon dioxide are often delivered at higher pressures than desirable. We filter our building's compressed air with an activated charcoal filter and use a photoionization detector to verify it is free of volatile organic compounds. Finally, a flow meter on the outlet of the observation chamber is essential for verifying a proper seal, and other flow meters may be useful during setup.

Final assembly and testing of the apparatus can be completed in one to two days depending on the number of problems (e.g. leaky connections) revealed by testing.

Supplementary Note 2 Calibration of Temporal Gradients

The measurement scheme used to calibrate the spatial gradient cannot be used to measure the temporal gradients as the equilibration time of detectors placed in the ceiling of the flow chamber is roughly one minute, significantly longer than the time it takes to change odor concentration in the flow chamber. However, the detectors integrated directly into the flow stream at the inlet and outlet can achieve fast response times. For all temporal experiments, we continuously monitored the odor concentration at the inlet. These measurements are shown (averaged over repeated trials) as the concentration in **Fig. 5**.

To calibrate the effects of diffusion and mixing in the chamber we used carbon dioxide and placed a second detector on the outlet of the chamber. We provided a square wave of carbon dioxide varying from 0 to 2.5% with a 240 second period (**Supplementary Fig. 2c,** the conditions of **Fig. 5b**). At the inlet to the chamber we measured a 10-90% rise time of 1.4 seconds, and a 90-10% fall time of 2.2 seconds (both of these measurements likely also reflect the response times of the NDIR detector, which is not designed to operate at high speeds). Based on the volume of the chamber between the inlet and outlet and the set flow rate, we would expect 41.3 seconds to pass between detection of the carbon dioxide step at the inlet and outlet detectors. We measure this time to be 41.1 seconds. Based on the concentration at the inlet, the flow speed of gas in the chamber, and the diffusion constant of carbon dioxide, we would expect a rise time of 7.3 seconds at the outlet and a fall time of 9.0 seconds. In fact, we measure these times to be 20.8 seconds and 33.5 seconds. This likely reflects the presence of unswept (dead) volumes in the inlet and outlet reservoirs. Future iterations of Lady Gaga will remove this dead volume to eliminate these lags. In any case, the rise time at the outlet is greater than the rise time in the flow chamber as the latter involves only the dead volume in the inlet chamber. The data presented in **Fig. 5** has a minimum temporal bin size of 24 seconds, so the bin size is approximately equal to or greater than the rise time within the flow chamber.

We also produced a 10 min triangle wave of carbon dioxide with a peak concentration of 5% (the conditions of **Fig. 5a**). The concentration at the outlet registered with the same delay as for the square wave, and we could see no distortion of the wave form due to diffusion or mixing (**Supplementary Fig. 2b, Fig. 1d**).

Supplementary Note 3 MAGAT Analyzer Algorithms

Analysis of larval behavior proceeds in two stages. In the first, the larva's position and posture are extracted from video records by a feature extractor written in C++ using the openCV library. In the second, descriptions of the paths, including speed, posture, heading angle, and segmentation into runs and turns are done using custom MATLAB scripts. Separating the tasks in this fashion allows the most computationally intensive tasks to run quickly as compiled software while allowing rapid development and flexible access using the scripted MATLAB language.

Machine Vision Algorithm (C++)

Raw data is supplied to the feature extraction software as a sequence of still images. In the experiments described here, frames were acquired at a rate of 5 Hz, but because analysis takes place offline, any frame rate is supported.

The feature extractor assumes dark field illumination – larvae are always brighter than the surface on which they travel. Thus the first step of feature extraction is to determine a background by taking the minimum value of each pixel over a sequence of images. This sequence could be the entire recording, but to avoid errors from gradual shifts in light levels we recalculate the background every 40 seconds (200 frames). Once the background is determined, we subtract it from each image in the sequence.

In the first pass of feature extraction, bright spots in the background subtracted image are assembled into individual animal tracks. In every frame, each spot whose brightness and area exceed user defined thresholds is either added to an existing track or forms the start of a new track. A bright spot is added to an existing track if that track's terminus at the previous frame was closer to this point than any other and if that distance is below a user-defined threshold (this prevents tracks that end from jumping to new, unrelated points). In the case where two tracks converge to the same bright spot, we attempt to rethreshold the single bright spot to produce two independent spots. When this can be done, both tracks continue to their respective spots. When this proves impossible, both tracks are terminated. After this sequence, any remaining bright spots begin their own tracks. Each track consists of a sequence of points representing the center of mass of the spot and a sequence of images of the bright spot and surrounding region.

In the second pass of feature extraction, we determine the outer contour, midline, and head and tail locations for each tracked larva. Proceeding in two steps allows us to use the entire sequence of images to determine features that are common to all images in the sequence, e.g. the size of the animal. This also allows flexibility in development. To analyze the posture of a different animal, e.g. *C. elegans*, we only need to modify this portion of the code as the first-pass track extraction is indifferent to the type of animal.

The first step of larval posture analysis is to use the entire set of images to determine the size of the larva (in pixels). Subsequently each image in the track is thresholded to produce a white region of the appropriate size on a black background (**Supplementary Fig. 1b**). Thus the analysis is insensitive to changes in overall illumination that may occur as the larva explores a large surface. The border of the white thresholded region is the contour of the larva. We determine the head and tail locations by finding the two "pointiest" regions on the contour separated by at least ¼ the total perimeter of the contour (**Supplementary Fig. 1c**). If the tracker cannot find exactly two separated pointy regions (as for instance if the larva touches its head to its tail forming a circle), the point is flagged. The midline is determined by dividing the contour up into two segments with an equal number of points between the head and tail and then taking the point-by-point average of these two segments (**Supplementary Fig.1d**). Head-tail identity is determined by minimizing the distance between corresponding midline points from one frame to the next and by the rule that the tail to head direction is generally parallel to the direction of motion, ignoring head-tail indeterminate images. The body bend angle is calculated by finding two lines that best fit the midline and taking the angle between them (**Supplementary Fig. 1e**).

Behavioral Analysis (MATLAB)

When experimental data is loaded into MATLAB, we supply a calibration file (generated by taking an image of a 1 cm checkerboard in the behavioral arena) to turn pixel locations in the image into real units, removing lens distortion in the process. We also supply a metadata file containing information about externally supplied stimuli, e.g. when odor concentration varies temporally over the course of an experiment.

Experimental data is loaded into MATLAB as a sequence of tracks. Each track has a sequence of points. Each point describes the position and posture of a larva at a single point in time. The MATLAB analysis scripts use these sequences of points to generate time varying descriptions of the animals' behavior. The velocity vector is a lowpassed derivative of the midpoint position, the heading angle is the four quadrant arctangent of the velocity vector, the body bend angle is the angle between the tail-mid and mid-head lines, and so on.

Optionally, we discard tracks that likely contain invalid data. For instance, tracks with near zero speed and frequently indeterminate head-tail positions may represent dust that has fallen on the arena lid or a larva that has found a crevice in the agar into which it can burrow. Animals that continuously circle in one direction may have motor defects that interfere with their ability to navigate.

Finally, the sequence of behavioral data is turned into a sequence of behaviors by segmenting the tracks into runs and turns. Runs are defined as continuous periods of forward movement with the head direction aligned with the direction of forward travel. Turns separate successive runs. The initiation of each head sweep during a turn is flagged when the body bend angle between the anterior and posterior of the animal exceeds 20°. Each head sweep ends when the body bend

angle drops below 10°, changes sign (the head swept to the other side of the body), or a new run begins. Each turn ends at the start of a new run. Thus, each turn can involve zero or more head sweeps. Head sweeps that begin a new run are termed "accepted." Head sweeps that are followed by another head sweep or by the straightening of the body without forward movement are "rejected."

The MATLAB script keeps track of the time indices defining each run, turn, and headsweep and calculates metrics like run length, mean run heading, and number of head sweeps and change in angle achieved by a turn. Thus we can determine the answers to questions like "what is the mean speed of runs headed perpendicular to the gradient and lasting at least 15 seconds?" or "how likely is a head sweep to the left to be accepted given the larva was previously headed at 30 degrees to the gradient direction?" We can also return to the original sequence of images that determine a particular track and play back all or part of the image sequence overlaid by our analysis results. The software is capable of arbitrary playback – thus we can ask to see all rejected headsweeps to the left and the 3 seconds preceding them or all points at which the tracker thinks the larva executed a turn with no headsweeps. This feature allows us to make sure that our analysis has produced results that align with what we would produce were we to score the behavior by hand (an impossible task, given the hundreds of hours of maggot tracks analyzed here).