Neuroinform (2010) 8:33–42
DOI 10.1007/s12021-010-9062-1

Automated Video Analysis of Animal Movements Using Gabor Orientation Filters

Daniel A. Wagenaar · Wiliam B. Kristan Jr

Published online: 2 February 2010

Abstract To quantify locomotory behavior, tools for determining the location and shape of an animal's body are a first requirement. Video recording is a convenient technology to store raw movement data, but extracting body coordinates from video recordings is a nontrivial task. The algorithm described in this paper solves this task for videos of leeches or other quasi-linear animals in a manner inspired by the mammalian visual processing system: the video frames are fed through a bank of Gabor filters, which locally detect segments of the animal at a particular orientation. The algorithm assumes that the image location with maximal filter output lies on the animal's body and traces its shape out in both directions from there. The algorithm successfully extracted location and shape information from video clips of swimming leeches, as well as from still photographs of swimming and crawling snakes. A Matlab implementation with a graphical user interface is available online, and should make this algorithm conveniently usable in many other contexts.

Keywords Image analysis · Animal detection · Natural scenes · Gabor filter · Shape extraction · Matlab

The article did not have an 'Information Sharing Statement', as per journal policy.

D. A. Wagenaar (✉)
Broad Fellows Program and Division of Biology, California Institute of Technology, Pasadena, CA, USA
e-mail: daw@caltech.edu

W. B. Kristan Jr
Section of Neurobiology, Division of Biological Sciences, University of California at San Diego, La Jolla, CA, USA

Introduction

Quantification of body motion is an essential element in many areas of neuroscientific study, including the characterization of central pattern generator motor output (Kristan et al. 2005), animal flight aerodynamics (Frye and Dickinson 2001; Sane 2003), even the study of hippocampal place fields (Buzsaki 2005). When only information about body position is needed, a variety of technologies may be used, from magnetic coils (Svensson and Thieme 1969) to the global positioning system (GPS) (Johnson et al. 2002; Wikelski et al. 2007). However, when information about body shape is needed as well, video recording is usually the most attractive option for its simplicity, ubiquitous availability, and good spatial resolution. The downside of video recording is that it requires image analysis for interpretation, which adds a layer of complexity. For studies on larger animals and humans, fluorescent beads placed on the body can greatly facilitate this analysis step, since they can be readily localized in video clips (Mazzoni et al. 2005). Unfortunately, for smaller animals—or when beads would interfere with the motion studied—this approach is not practical, and the dynamic conformation of the body must be reconstructed based on its image as recorded in the video sequence, a process known as object detection and tracking. A large number of algorithms of various degrees of generality and task specificity have been proposed to solve this problem (reviewed by Yilmaz et al. 2006). For example, a simple approach which works well when the contrast between the animal's body and the background is large, is to apply a brightness threshold to the video frames. This approach is commonly employed for the analysis of body motion of the worm *C. elegans*, which are filmed as they crawl on a flat agar substrate (Baek et al. 2002; Ramot et al. 2008). If the substrate is very uniform, applying a brightness threshold effectively reveals the animal's body outline. In practice, an adaptive threshold is usually needed to handle deviations from uniformity, especially when animals are filmed in an aquarium or in natural environments. Moreover, careful tuning is generally required for the image erosion, dilation, and skeletonization algorithms that reduce the outline of the animal to a single curve tracing its midline from head to tail (Geng et al. 2004; Cornea et al. 2007). An entirely different approach, which works well to detect objects of rigid shape is to apply a Hough transform (Duda and Hart 1972; Ballard 1981) to the images, which can be used to obtain a list of the straight lines present in the images. Although Hough transforms can be used to detect articulated objects (Leibe et al. 2008), it is less suited for arbitrarily flexing objects like leeches.

Here we demonstrate how a third approach, which uses a bank of Gabor filters (Gabor 1946; Kamarainen et al. 2006) to extract orientation information for all locations in the image, can reliably find the location and shape of quasi-linear animals such as snakes or worms in video sequences. This approach was inspired by the architecture in the mammalian visual cortex, which uses orientation-selective neurons ("simple cells") as a basis to represent complex scenes (Webster and Valois 1985; Field and Tolhurst 1986; Ringach 2002). We use a bank of Gabor filters to emulate orientation-selective cells which respond to the presence of a segment of an animal with a particular orientation in their receptive fields (Okajima 1998). Similar to the functioning of the simple-cell network in the cortex, where excitatory connections between cells with similar orientation preference and partially overlapping receptive fields help longer contours stand out (Ts'o et al. 1986; Malach et al. 1993; Kapadia et al. 1995), our algorithm uses the Gabor filter output to trace the animal's body midline.

We test our approach on video recordings of swimming leeches and on still images of swimming and crawling snakes.

Methods

Leech Behavior and Recording Setup

The main focus of this paper is the analysis of video clips of swimming leeches. The test data set is available from the authors' website.[1] Adult medicinal leeches (*Hirudo verbana*) were placed in shallow water, 2 cm deep: deep enough that the animals preferred swimming over crawling (Kristan et al. 2005), yet shallow enough that they were forced to swim in the horizontal plane. Medicinal leeches swim by undulation in the dorsoventral plane (approx. two full cycles per second) with a sinusoidal wave propagating from head to tail (Kristan et al. 1974). We used animals that were about 10 cm long and 0.5–1 cm in diameter. We recorded their swimming from above using a video camera with 640 × 480 (interlaced) resolution at 30 frames per second (Hitachi model HV-C20), zoomed in such that the length of a leech was typically about 200 pixels on the video. Because the leeches swam in the horizontal plane, the vertically mounted camera could capture their dorsoventral swim waves clearly. Clips were stored on VHS tape and later digitized for computer analysis.

The Wormfinder Algorithm

The Wormfinder algorithm takes video clips as its input, and computes the locations of 100 equally spaced points along the body of the animal found in each of the video frames. Conceptually, the algorithm may be divided in three stages (Fig. 1):

Preprocessing:	Decoding the video clip into individual frames and removing static backgrounds;
Detection:	Finding the locations and shapes of animals in each video frame;
Postprocessing:	Combining the results obtained in each frame and identifying which end of the animal is the head and which is the tail.

In the following, we describe the details of each stage.

Preprocessing

Video images recorded on VHS tape are stored at 30 full frames of 640 × 480 pixels per second in an interlaced format: the even scan lines in a frame are recorded 1/60 s later than the odd scan lines. When fast-moving animals are recorded, this delay results in displacements between the subimages formed by even and odd scan lines respectively. Therefore, the first step

[1] http://www.danielwagenaar.net/software/wormfinder

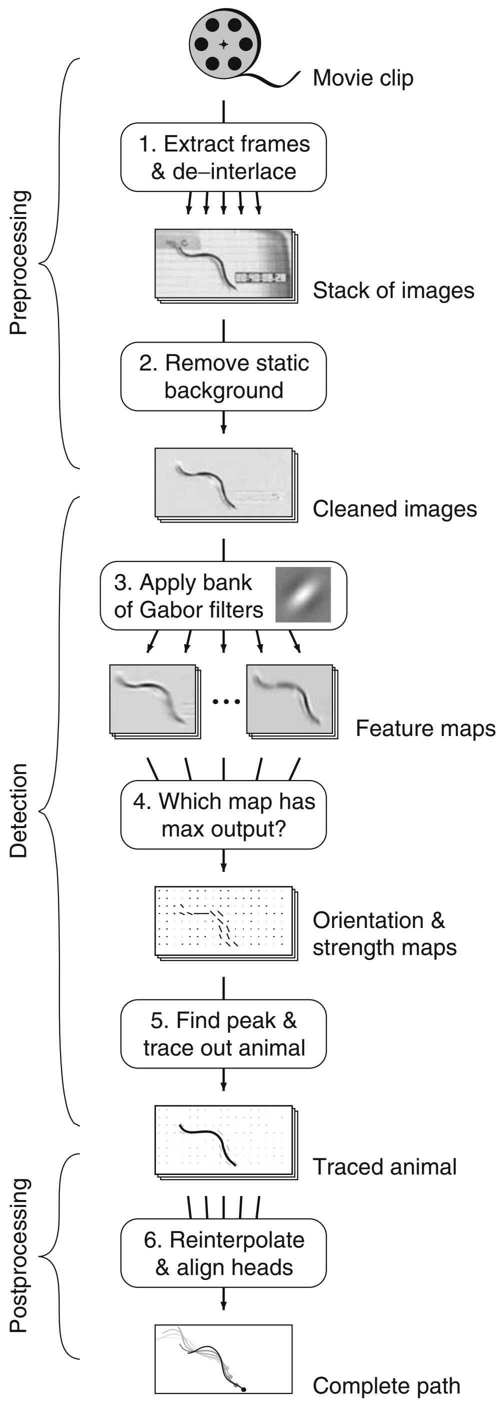

Movie clip
1. Extract frames & de–interlace
Stack of images
2. Remove static background
Cleaned images
3. Apply bank of Gabor filters
Feature maps
4. Which map has max output?
Orientation & strength maps
5. Find peak & trace out animal
Traced animal
6. Reinterpolate & align heads
Complete path
Preprocessing
Detection
Postprocessing

Fig. 1 Flow chart of the Wormfinder algorithm. Steps are explained in "Methods"

of preprocessing is to separate the subimages, i.e., to de-interlace (step 1 in Fig. 1). The first subimage is constructed by interpolating between the odd scan lines from the stored VHS image; the second by interpolating between the even scan lines. The net result is a sequence of images at 60 frames per second with 640×480 resolution. One image from such a sequence is shown in Fig. 2a, with insets illustrating the importance of de-interlacing. To further clean up the images, we remove any stationary background from the recording by determining the brightest value reached by each individual pixel in any frame in a recording (clips in our test data were between 34 and 106 frames long), and subtracting these values out (step 2; Fig. 2b).

Detection

The core of the Wormfinder algorithm processes images individually, in the following way. Let an image be represented by its brightness values $I(\vec{x})$, where $\vec{x}$ is the two-dimensional location in the image. Even though digital images have discrete pixels, the algorithm operates in continuous space, linearly interpolating between pixels as necessary. From the image we extract eight feature maps, $F_\phi(\vec{x})$, for orientations $\phi = 0°$ (horizontal), $22\frac{1}{2}°$, $45°$, etc., through $157\frac{1}{2}°$, by convolution with a bank of Gabor filters (step 3 in Fig. 2):

$$F_\phi(\vec{x}) = \iint G_\phi(\vec{x} - \vec{x}')\, I(\vec{x}')\, \mathrm{d}\vec{x}',$$

where

$$G_\phi(\Delta\vec{x}) = \mathrm{e}^{-\frac{1}{2}\xi^2/\sigma_\parallel^2 - \frac{1}{2}\eta^2/\sigma_\perp^2} \cos(2\pi\eta/\lambda),$$

and

$$\begin{pmatrix} \xi \\ \eta \end{pmatrix} = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \Delta\vec{x}.$$

Here, ξ and η are coordinates along the Gabor wave and perpendicular to it, $\sigma_\parallel$ represents the size of the Gabor patch along the wave, $\sigma_\perp$ represents the size perpendicular to the wave, and λ represents the wavelength of the patch. (For certain classes of images, a Log-Gabor filter (Field 1987) might yield even better results; we have not explored this avenue.)

For best results, $\sigma_\perp$ and λ should be scaled in proportion to the thickness of the body of the animal to be detected, and $\sigma_\parallel$ in proportion to its typical radius of curvature (so that the scale of the Gabor patch matches the scale of the animal to be detected). Furthermore, λ should not be excessively large compared to $\sigma_\perp$, to avoid the risk of detecting false positives at the fringes of the Gabor filter. Since swimming leeches do not ordinarily curve their bodies with a radius smaller than their body width, and the typical width of the bodies of the leeches in the videos analyzed was approximately 8 pixels, we used $\sigma_\parallel = \sigma_\perp = 8$ pixels and $\lambda = 20$ pixels.

From the feature maps, we determine the directionality at every point of the image (step 4; Fig. 2d). Specifically, we construct an orientation map $\Phi(\vec{x})$ that

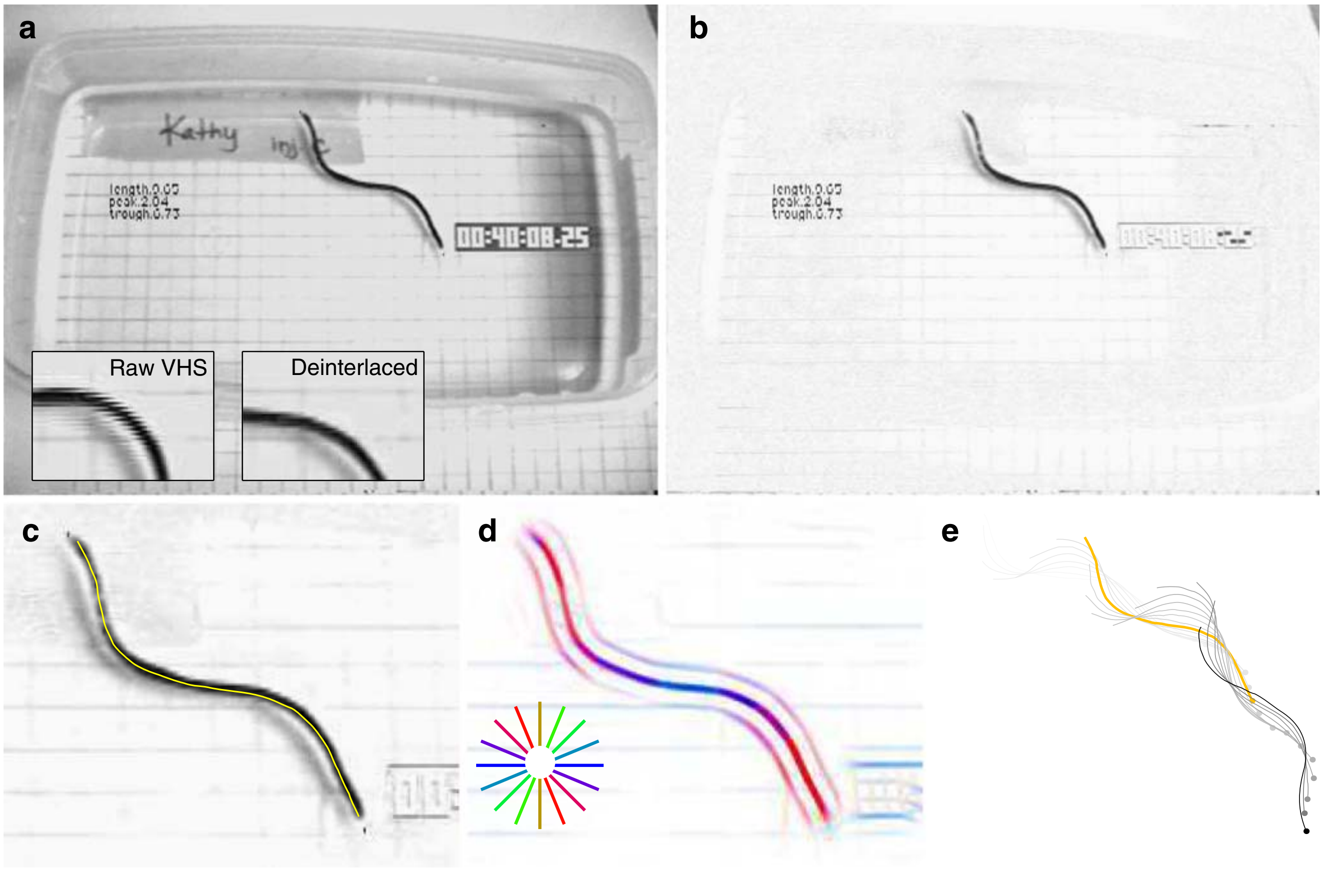


Fig. 2 Example results on a swimming leech. **a** Single frame from a video clip, after de-interlacing. *Insets*: detail before (*left*) and after (*right*) de-interlacing. Jagged edges in raw frame are due to motion between the odd and even subframes. **b** Background clutter is substantially reduced by subtracting out the median of all frames in the video sequence. **c** The output of the algorithm, i.e., the body shape of the leech (yellow curve) overlaid on the input. The animal's head is in the lower right. (Image cropped to show detail.) **d** The directionality maps used for tracing the animal. Intensity represents the strength map $S(\vec{x})$; hue represents the orientation map $\Phi(\vec{x})$. (See key on *bottom left*.) **e** The final result: the path of the swimming leech. The black trace is the animal's final position, gray traces are its positions at earlier times (at 0.05 s intervals). *Dots* mark the head of the animal. The body trace shown in (**c**) is *shaded orange*, and shown here in the context of the entire sequence

contains, for each point in the image, the orientation of the filter that responded most strongly there, and a strength map $S(\vec{x})$ that contains the amplitudes of those responses. Intuitively, $S(\vec{x})$ indicates how strongly the image neighborhood around the point $\vec{x}$ resembles a line segment (which could be a part of the target animal), and $\Phi(\vec{x})$ is the orientation of that line segment.

If the background does not contain any bright, strongly oriented, line segments that resemble the target animal, we can assume that the point in the image where directionality is strongest lies on the animal. (See under *Refinements*, below, for a remedy if this condition is violated.) We may then trace out the animal's body in both directions starting from that point (step 5), as follows. Let $\vec{x}_0 = \operatorname{argmax}(S(\vec{x}))$ be the described starting point, and $\phi_0 = \Phi(\vec{x}_0)$ be the orientation at that point. From there, we iteratively trace the body of the animal out to one end, by first taking a tentative step in the direction ϕ_0:

$$\vec{x}_0' = \vec{x}_0 + \epsilon\widehat{n}_{\phi_0},$$

where $\widehat{n}_{\phi_0}$ is a unit vector in the direction ϕ_0, and ε is the step size. The exact value of ε is not critical; we used $\varepsilon = 4$ pixels.

If the animal were straight, and perfectly oriented in the direction ϕ_0, the point $\vec{x}_0'$ would also lie on the midline of its body. However, in reality the animal will be curved, so the midline may pass at a slight distance of $\vec{x}_0'$. The algorithm thus scans a short line segment (of length 2ε) perpendicular to the step direction ϕ_0 at $\vec{x}_0'$. We then find the point on that line segment where the strength $S(\vec{x})$ is maximal, and continue from there, each time taking a tentative step in the direction of local orientation $\Phi(\vec{x})$ and scanning a short line segment to find the local maximum of the strength S, until that maximum falls below a threshold at some point $\vec{x}_k$, which is then deemed to be one end of the animal. Good results were obtained by setting the threshold at 0.2 to 0.3 times the maximal observed strength.

Noting that we have thus far only traced part of the animal (from the arbitrary starting point $\vec{x}_0$ somewhere along its body to one of its ends), we repeat the procedure, again starting out from $\vec{x}_0$, but in the reverse direction: $\phi_0 + 180°$, to trace out the rest of the animal (Fig. 2c).

Refinements

In this section we present several refinements to the basic algorithm described above, which prevent potential problems in difficult images. First, animals could theoretically cross back on themselves, although that did not happen in the recordings used as test data. The algorithm as described so far would get trapped in an infinite loop in this situation. To prevent this from happening, tracing is automatically terminated as soon as a self-crossing is detected. Second, near the head and the tail of the animal, as the orientation strength S drops to background levels, the orientation Φ becomes increasingly uncertain. This could result in loops in the orientation map. To keep the algorithm from getting stuck, such loops were cut at their weakest point or at the first point where the orientation changed by more than 90°, even if the strength had not yet fallen below threshold. Finally, the algorithm might get confused by patches of background with strong directionality that resemble short animals. Thus, when the core algorithm comes up with a candidate animal shorter than ten steps, the candidate is rejected, the area around the rejected candidate is grayed out by setting the strength map S to zero, and the search is automatically restarted in the rest of the image.

Postprocessing

After animals have been detected in all images, their body curves are reinterpolated to yield 100 equally spaced points in each frame (step 6). These curves are then aligned in two steps. First, corresponding ends are identified in consecutive frames. This is based on the assumption that corresponding points on the body curves in consecutive frames are closer together if both animals are parametrized head-to-tail than if they are parametrized in opposing directions. Second, the head of the animal is identified based on the heuristic that, on average, the animal moves forward (which holds true for both swimming and crawling in leeches), so that the front 99% of the body curve in one frame should resemble the rear 99% of the body curve in the next frame more closely than the converse pairing, at least on average across all frames in a clip. This technique works even if animals move backward some of the time, as long as integrated over the entire video clip they move forward. Moreover, movements of head or tail perpendicular to the body axis—an integral part of the undulatory swimming pattern of leeches—do not affect the heuristic.

The final result is a family of curves corresponding to the position and shape of the animal in each of the video frames (Fig. 2e).

Quality Assessment and Comparison with a Reference Algorithm

To better assess the merits of the Wormfinder algorithm, we compared its output and that of a reference algorithm to the assessment by human experts of the positions of the animal's head and tail, and of a third point on the midline, approximately equidistant between head and tail. For a reference algorithm we chose a skeletonization algorithm optimized for this particular set of video clips: First, the same background subtraction used for Wormfinder was performed. Then, the images were inverted (so that leeches appear bright on a dark background), and the average brightness of each individual image was subtracted out. The resulting images were then thresholded at 0.33× their maximal brightness. Two dilation steps (turning on "off" pixels if any of their eight direct or diagonal neighbors are "on" (Haralick et al. 1987; Russ 2006)) helped ensure that animals ended up as contiguous areas of "on" pixels. Multiple thinning steps (dropping "on" pixels if any of their four direct neighbors are off provided that this does not disconnect contiguous "on" areas or shorten the end of a line) were used to reduce this area to a skeleton (Lam et al. 1992). The longest direct (non-cyclic) path in this skeleton was interpreted as the midline of the animal to be detected.

Results

Application to Video Clips of Swimming Leeches

Wormfinder was used to analyse the motion of leeches in 18 video clips of freely swimming leeches (four different animals). Parameters were kept the same for all clips, and were not adapted to the variable sizes of the animals or to lighting conditions. Figure 2 shows the results on a typical clip.

Wormfinder's report of the locations of the head and the tail of the animals in each frame were compared to annotations by human experts. Without specific instructions, humans tended to mark spots approximately in the middle of the head and tail, whereas the curves found by Wormfinder extended to the very tips. This discrepancy, which could easily be eliminated by dropping 1% of each end of the curves, and which, in any case, is a matter of taste, was not counted against the algorithm as a demerit. With that average longitudinal shift taken out of the equation, Wormfinder's reports of head and tail location differed by more than 5% of animal length in only 10 and 17 frames respectively (0.85% and 1.45%; Fig. 3a). In contrast, the reference algorithm's output for heads and tails was off by more than 5% in as many as 99 and 109 frames respectively (8.4%

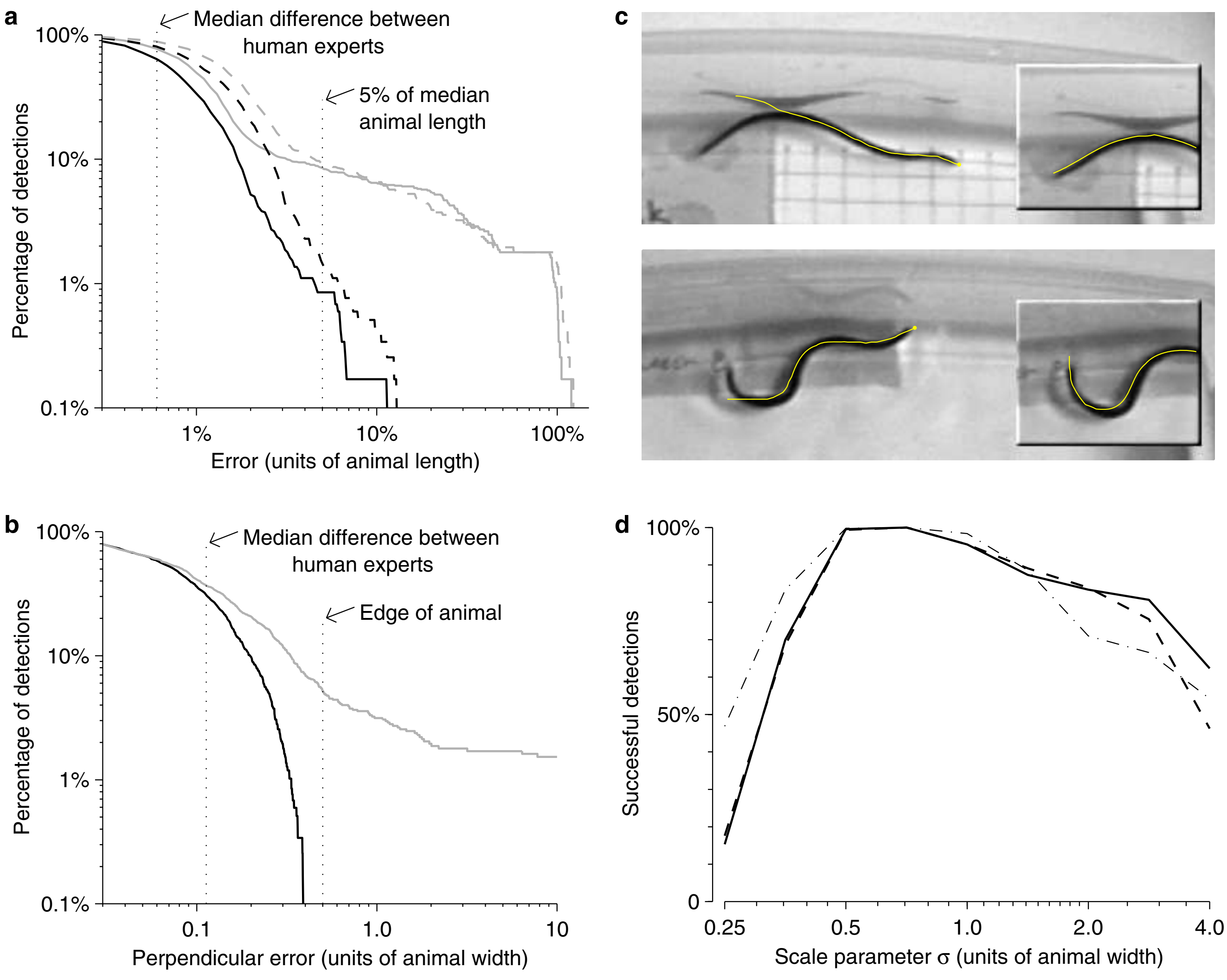


Fig. 3 Analysis of the quality of Wormfinder's output **a** Cumulative distribution of the error in the location of the animal's head (*solid*) or tail (*dashed*), as reported by Wormfinder (*black traces*) or Skeletonize (*gray traces*). **b** Cumulative distribution of the error in the location of the animal's midline (at a point about halfway between head and tail), as reported by Wormfinder (*black trace*) or Skeletonize (*gray trace*). **c** Examples of cases where Wormfinder produced erroneous output. The algorithm was occasionally confused by a reflection in the wall of the dish in which the leech swam (*top*), and once by a shadow (*bottom*). In adjacent frames, Wormfinder produced correct output (*insets*). **d** Quality of detection as a function of parameter values. We set the scale parameter $\sigma = \sigma_\perp = \sigma_\parallel = 0.4\lambda$ to different values, in relation to the width of the animal in each video. Plotted are the fraction of frames in which: the error in the location of the head was less than 5% of the animal's length (*solid*); the error in the location of the tail was less than 5% of the animal's length (*dashed*); the error in the location of the midline was less than half the animal's width (*dot-dashed*)

and 9.3%), and by as much as 100% of animal length or more in 9 and 17 frames. The 95% confidence limits of errors by Wormfinder were 2.1% and 3.0% of animal length for head and tail respectively, versus 23% and 16% for the reference algorithm. For all error thresholds greater than 0.5% of animal length, Wormfinder made significantly fewer errors than did the reference algorithm in either heads or tails ($p<.01$, Fisher Exact test).

The human experts also marked the midline of the animal in each frame, at a location approximately halfway between head and tail, and we measured how close the midline curves reported by the two algorithms passed by these points. In the majority of cases, both algorithms did very well on this test, producing median errors less than the median difference between the assessments by two humans, who were generally not able to achieve subpixel precision (Fig. 3b). Wormfinder never produced errors in excess of 0.5× animal width, meaning that its output always stayed within the boundaries of the body (at least in the vicinity of the test points). By contrast, the reference algorithm strayed by 0.5× animal width or more in 5.2% of cases. The 95% confidence limit of errors by Wormfinder was 0.25× animal width, vs. 0.51× animal width for the reference algorithm.

We manually inspected each of the 27 cases in which Wormfinder made an error in head or tail position of 5% of animal length or more, and found that in ten of these cases (0.9% of total), reflections in the wall of the container in which the leech swam confused the algorithm. In eight cases (0.7%) tracing was prematurely terminated near the tail; in 6 cases (0.5%) tracing was prematurely terminated near the head; in one case tracing extended past the tail by 5.0% of animal length; in one case the algorithm was confused by a shadow; in one final case the algorithm was confused by a time stamp placed in the image by the VHS recorder. Two examples of failures are presented in Fig. 3c.

To determine how sensitive the algorithm is to changes in parameter values, we reran it several times with different values of the key variable $\sigma_\perp$, scaled to different multiples of the body width of the animal in individual clips. In all cases, $\sigma_\parallel$ was set to equal $\sigma_\perp$, and λ was set to $2.5\times\ \sigma_\perp$. We quantified a successful detection of head or tail location as one where the error was less than 5% of animal length relative to human annotations, and a successful detection of midline location as one where the error was less than 0.5× animal width. For values of the parameter between 0.5× animal width and 1.0× animal width, well over 95% of detections were successful by these criteria; outside of this range, success rates dropped markedly (Fig. 3d).

Application to Still Images of Snakes

To show that the algorithm has utility beyond studying leeches on the relatively clean background of our experimental tanks, we also applied it to two photographs (found on the internet) of snakes in outdoor settings.

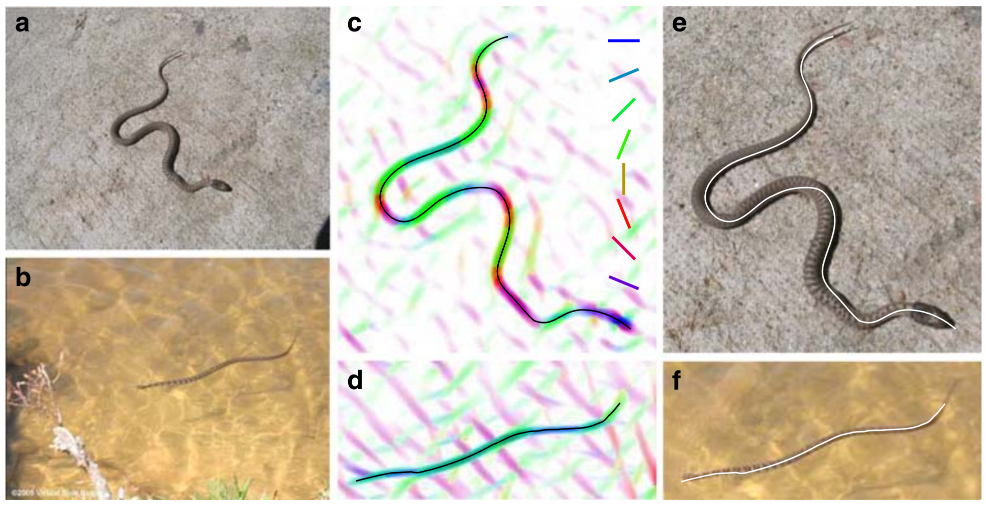
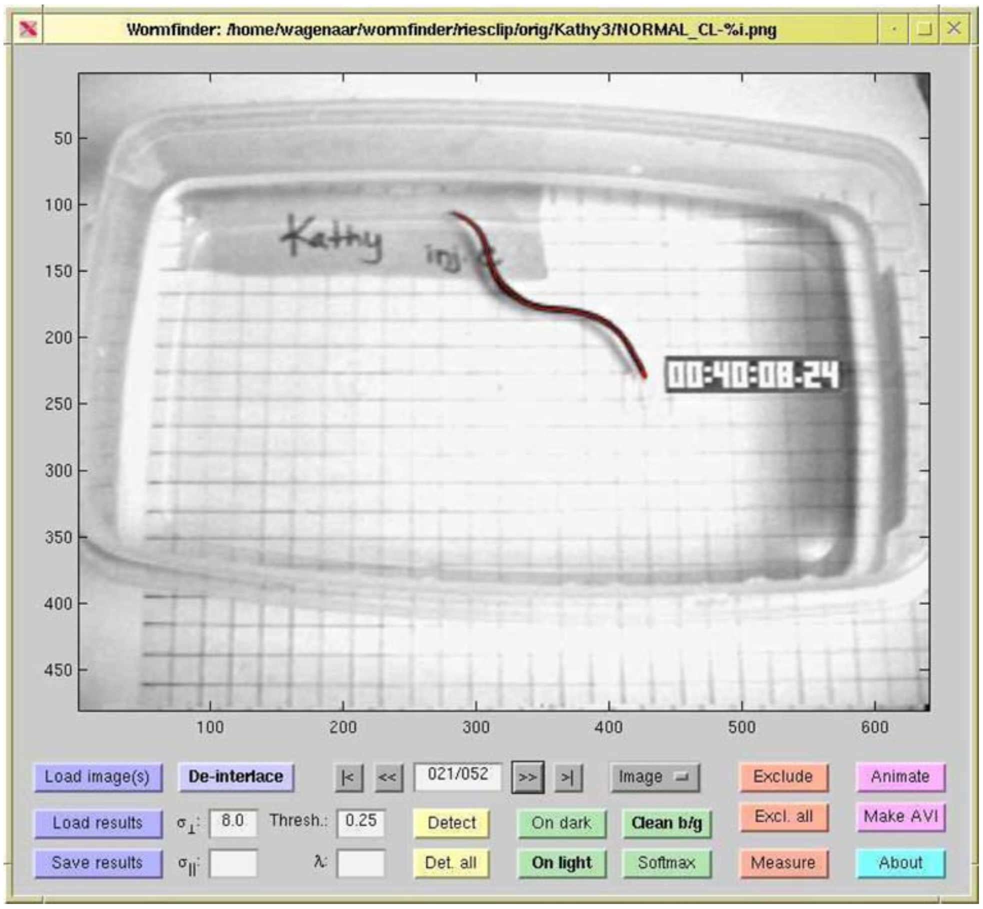


Fig. 4 Locating snakes in still images. **a** Photograph of a crawling snake. **b** Photograph of a swimming snake. **c–d** Resulting directionality maps (as in Fig. 2d), with Wormfinder's output overlaid in black. **e–f** Details of original images, with Wormfinder's output overlaid in white. Parameters used for **a**, **c**, **e** were (in pixels): $\sigma_\parallel = 6$, $\sigma_\perp = 14$, $\lambda = 24$; for **b**, **d**, **f**: $\sigma_\parallel = 11$, $\sigma_\perp = 10$, $\lambda = 18$. Exact values were not critical in either case. Credits: Photo of crawling snake (**a**) reproduced from http://rogerwendell.com with permission from Roger Wendell. Swimming snake (**b**) photographed by Foster Hunt, reproduced with permission from VirtualBlueRidge.com


Wormfinder: /home/wagenaar/wormfinder/resclip/orig/Kathy3/NORMAL_CL-%i.png

021/052

$\sigma_\perp$: 8.0 Thresh.: 0.25

$\sigma_\parallel$: λ:

Fig. 5 Screenshot of the Matlab implementation of Wormfinder. The user interface allows for quick, automated discovery of animal positions in individual images or in an entire movie clip in one go

One photograph (Fig. 4a) features a snake crawling over a driveway, the other (Fig. 4b), a snake swimming in a small lake. Still images are more challenging than video clips because subtracting out the median across a large number of frames obviously does not work for single images. This means that the algorithm must contend with the large amount of contrast present in the background and with the small brightness difference between the snakes and their surroundings. Nevertheless, the Gabor filter approach lifted both snakes out of the background (Fig. 4c–d), allowing the algorithm to successfully identify the positions and forms of the animals (Fig. 4e–f). Note, however, that in both cases the tails, which taper to narrow ends and—in the case of the swimming snake—suffered some motion blur in the image, were cut short by a few percent of the animals' lengths.

Matlab Implementation

A Matlab (The Mathworks, Natick, MA) implementation of the Wormfinder algorithm with a graphical user interface (Fig. 5) is available from the authors' website.[2] The user interface allows importing of single images or sequences of images and movie files. Detection parameters can be specified, and annotations can be made in individual frames to prevent spurious detection in unusually noisy areas. After detection, an animation can be shown of the results in each frame, and corrections can be made as necessary by modifying detection parameters and reprocessing individual frames or all frames. Final results can be saved as AVI files or as Matlab data files for further analysis. A user manual, supplied with the program, explains the operation of the Matlab program, including the precise correspondence between its settings and the parameters described in this paper. In addition, online help is provided in the form of tool tips. On a modern PC with a 2-GHz Quad Core Intel CPU, the Matlab implementation can process about two frames of 640 × 480 pixels per second (0.5 frames per second per core). Most of the time is spent tracing the animal midline (calculating the Gabor filter outputs is not very

[2] http://www.danielwagenaar.net/software/wormfinder

time-consuming in Fourier space). The current version of the code is not optimized for speed, and, by translating the algorithm to a compiled language such as C++, real-time operation could become a possibility. (With the one caveat that the disambiguation of head vs. tail can only be performed reliably once several seconds of video have been processed.)

Discussion

We have described an algorithm for detecting the location and body shape of leeches and other quasi-linear animals in video sequences based on determining directionality for every pixel of the image, and tracing out the animal starting from the point where directionality is strongest. Directionality was computed in a manner inspired by the architecture of the mammalian visual system: using a bank of Gabor filters. The algorithm successfully located leeches in over 95% of video frames, without tuning any parameters for individual video clips. It also successfully located snakes in two natural scenes, despite the substantial visual clutter in these images. We expect that it would perform equally well on still images or movie clips of other similarly shaped animals such as eels or lampreys.

The described approach offers significant advantages over the use of beads stitched to the body wall: First, it does not require surgery, simplifying experimental procedures and eliminating any concerns that the observed behavior is distorted by the observation method. Second, it allows for measurement along the entire body of the animal rather than only on a few discrete locations. By validating results against human judgment, we determined that in nearly all cases Wormfinder produced smaller errors than a brightness threshold-based skeletonization method used as a reference algorithm, and that its worst-case behavior was better by an order of magnitude. Wormfinder required no fine tuning of parameters and was robust against background clutter in the image.

By making a user-friendly Matlab implementation of the algorithm publicly available, we hope to have provided an immediately useful tool as well as a platform for further improvement of the algorithm. For instance, the current implementation is geared toward detecting a single animal in each image. Provided animals do not cross over each other, it would be straightforward to modify the algorithm to detect multiple animals, by zeroing the strength map $S(\vec{x})$ in the immediate vicinity of previously detected animals and iterating the portion of the algorithm that finds the peak in the map and traces out the animal. Keeping track of the identities of multiple animals across frames can be achieved analogously to our current heuristic for head-to-tail alignment.

Acknowledgements This work was supported by a fellowship from the Broad Foundations (to DAW), by grant IOS-0825741 from the NSF (to WBK), and by grant RO1 MH043396 from NIH/NIMH (to WBK). DAW holds a Career Award at the Scientific Interface from the Burroughs Wellcome Fund.

References

Baek, J. H., Cosman, P., et al. (2002). Using machine vision to analyze and classify *Caenorhabditis elegans* behavioral phenotypes quantitatively. *Journal of Neuroscience Methods, 118*(1), 9–21.

Ballard, D. H. (1981). Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recognition, 13*(2), 111–122.

Buzsaki, G. (2005). Theta rhythm of navigation: Link between path integration and landmark navigation, episodic and semantic memory. *Hippocampus, 15*(7), 827–840.

Cornea, N. D., Silver, D., & Min, P. (2007). Curve-skeleton properties, applications, and algorithms. *IEEE Transactions on Visualization and Computer Graphics, 13*(3), 530–548.

Duda, R. O., & Hart, P. E. (1972). Use of the Hough transformation to detect lines and curves in pictures. *Communications of the ACM, 15*(1), 11–15.

Field, D. J. (1987). Relations between the statistics of natural images and the response properties of cortical-cells. *Journal of the Optical Society of America. A, Optics, Image Science, and Vision, 4*(12), 2379–2394.

Field, D. J., & Tolhurst, D. J. (1986). The structure and symmetry of simple-cell receptive-field profiles in the cat's visual cortex. *Proceedings of the Royal Society of London, 228*(1253), 379–400.

Frye, M. A., & Dickinson, M. H. (2001). Fly flight: A model for the neural control of complex behavior. *Neuron, 32*(3), 385–388.

Gabor, D. (1946). Theory of communication. *Journal of The Institution of Electrical Engineers, 93*(3), 429–457.

Geng, W., Cosman, P., et al. (2004). Automatic tracking, feature extraction and classification of *C. elegans* phenotypes. *IEEE Transactions on Biomedical Engineering, 51*(10), 1811–20.

Haralick, R. M., Sternberg, S. R., & Zhuang, X. H. (1987). Image-analysis using mathematical morphology. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 9*(4), 532–550.

Johnson, C. J., Heard, D. C., & Parker, K. L. (2002). Expectations and realities of GPS animal location collars: Results of three years in the field. *Wildlife Biology, 8*(2), 153–159.

Kamarainen, J. K., Kyrki, V., & Kalviainen, H. (2006). Invariance properties of Gabor filter-based features—Overview and applications. *IEEE transactions on Image Processing, 15*(5), 1088–1099.

Kapadia, M. K., Ito, M., et al. (1995). Improvement in visual sensitivity by changes in local context: parallel studies in human observers and in v1 of alert monkeys. *Neuron, 15*(4), 843–856.

Kristan, W. B., Jr, Calabrese, R. L., & Friesen, W. O. (2005). Neuronal control of leech behavior. *Progress in Neurobiology, 76*(5), 279–327.

Kristan, W. B., Stent, G. S., & Ort, C. A. (1974). Neuronal control of swimming in medicinal leech. 1. Dynamics of swimming rhythm. *Journal of Comparative Physiology, 94*(2), 97–119.

Lam, L., Lee, S. W., & Suen, C. Y. (1992). Thinning methodologies—A comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 14*(9), 869–885.

Leibe, B., Leonardis, A., & Schiele, B. (2008). Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision, 77*(1–3), 259–289.

Malach, R., Amir, Y., et al. (1993). Relationship between intrinsic connections and functional architecture revealed by optical imaging and in vivo targeted biocytin injections in primate striate cortex. *Proceedings of the National Academy of Sciences of the United States of America, 90*(22), 10469–10473.

Mazzoni, A., Garcia-Perez, E., et al. (2005). Quantitative characterization and classification of leech behavior. *Journal of Neurophysiology, 93*(1), 580–593.

Okajima, K. (1998). The Gabor function extracts the maximum information from input local signals. *Neural Networks, 11*(3), 435–439.

Ramot, D., Johnson, B. E., et al. (2008). The parallel worm tracker: A platform for measuring average speed and drug-induced paralysis in nematodes. *PLOS One, 3*(5), e2208

Ringach, D. L. (2002). Spatial structure and symmetry of simple-cell receptive fields in macaque primary visual cortex. *Journal of Neurophysiology, 88*(1), 455–463.

Russ, J. C. (2006). *The image processing handbook*. CRC Press, Boca Raton FL.

Sane, S. P. (2003). The aerodynamics of insect flight. *Journal of Experimental Biology, 206*(23), 4191–4208.

Svensson, T. H., & Thieme, G. (1969). An investigation of a new instrument to measure motor activity of small animals. *Psychopharmacologia, 14*(2), 157–163.

Ts'o, D. Y., Gilbert, C. D., & Wiesel, T. N. (1986). Relationships between horizontal interactions and functional architecture in cat striate cortex as revealed by cross-correlation analysis. *Journal of Neuroscience, 6*(4), 1160–1170.

Webster, M. A., & Valois, R. L. D. (1985). Relationship between spatial-frequency and orientation tuning of striate-cortex cells. *Journal of the Optical Society of America A, 2*(7), 1124–1132.

Wikelski, M., Kays, R. W., et al. (2007). Going wild: What a global small-animal tracking system could do for experimental biologists. *Journal of Experimental Biology, 210*(2), 181–186.

Yilmaz, A., Javed, O., & Shah, M. (2006). Object tracking: A survey. *ACM Computing Surveys, 38*(4).