

Monocular distance estimation from optic flow during active landing maneuvers

This content has been downloaded from IOPscience. Please scroll down to see the full text.

2014 Bioinspir. Biomim. 9 025002

(<http://iopscience.iop.org/1748-3190/9/2/025002>)

View [the table of contents for this issue](#), or go to the [journal homepage](#) for more

Download details:

IP Address: 128.114.59.156

This content was downloaded on 05/02/2015 at 00:58

Please note that [terms and conditions apply](#).

Monocular distance estimation from optic flow during active landing maneuvers

Floris van Breugel¹, Kristi Morgansen² and Michael H Dickinson²

¹ California Institute of Technology, Pasadena, CA, USA

² University of Washington, Seattle, WA, USA

E-mail: floris@caltech.edu

Received 21 July 2013, revised 1 October 2013


Accepted for publication 4 October 2013

Published 22 May 2014

Abstract

Vision is arguably the most widely used sensor for position and velocity estimation in animals, and it is increasingly used in robotic systems as well. Many animals use stereopsis and object recognition in order to make a true estimate of distance. For a tiny insect such as a fruit fly or honeybee, however, these methods fall short. Instead, an insect must rely on calculations of optic flow, which can provide a measure of the ratio of velocity to distance, but not either parameter independently. Nevertheless, flies and other insects are adept at landing on a variety of substrates, a behavior that inherently requires some form of distance estimation in order to trigger distance-appropriate motor actions such as deceleration or leg extension. Previous studies have shown that these behaviors are indeed under visual control, raising the question: how does an insect estimate distance solely using optic flow? In this paper we use a nonlinear control theoretic approach to propose a solution for this problem. Our algorithm takes advantage of visually controlled landing trajectories that have been observed in flies and honeybees. Finally, we implement our algorithm, which we term *dynamic peering*, using a camera mounted to a linear stage to demonstrate its real-world feasibility.

Keywords: range finding, landing, insect flight, observability

 Online supplementary data available from stacks.iop.org/BB/9/025002/mmedia

Introduction

Many animals rely heavily on vision to gather information about their position and velocity relative to objects in the world around them. As an animal moves, it perceives apparent motion of these objects, with nearby objects moving faster than distant ones. This phenomenon is referred to as optic flow and essentially provides a measure of the ratio of forward movement to the distance of surrounding objects [1]. This coupled relationship intuitively suggests that estimating either absolute distance or velocity from optic flow is challenging. Indeed, it is not immediately clear whether an accurate estimate of both position and velocity from optic flow is possible at all.

Optic flow is only one of many sensory modalities available to most animals, and given additional sensory information it is possible to derive accurate estimates of velocity and position. For example, terrestrial animals might count strides, as supported by experiments in desert ants [2]. Another theoretical possibility for walking animals is to use visual odometry based on ventral optic flow, with the system calibrated by proprioceptive information that accurately measures the distance between the eye and the ground [3]. Animals with high acuity vision and long-term memory might estimate distance to recognizable objects by remembering their typical size [4]. Perhaps the most straightforward strategy is stereopsis, which can provide a distance estimate based on parallax [4, 5]. As will become clear in our subsequent discussion, however, none of these approaches are plausible for small flying insects. How is it then possible for a fruit fly or honeybee to avoid some objects



Content from this work may be used under the terms of the [Creative Commons Attribution 3.0 licence](http://creativecommons.org/licenses/by/3.0/). Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.

and land on others, without knowing either how fast they are going or how far away the obstacles are located?

Growing evidence from studies of insects and birds reveals a clever algorithm that solves this problem, at least in the context of landing. Although the details vary across the studies, the core strategy is that by maintaining a constant rate of optic flow, an animal can decelerate to a safe touchdown velocity without needing to directly measure either its velocity or distance to target [6–8]. Because optic flow essentially provides a measure of the ratio of velocity to distance (units of s^{-1}), it is often instead referred to as the inverse of time-to-contact (the time it would take for the animal to hit an object assuming it is on a collision course and maintains its initial speed and heading). A controller implementing this simple law would cause an animal to decelerate in such a way that its velocity would be inversely proportional to the distance to nearby objects. These observations have inspired the implementation of similar algorithms on robotic aircraft to achieve smooth automated landings [9].

Although the constant optic flow algorithm offers a robust strategy for deceleration, other components of landing behavior may require different sensory mechanisms. For example, at some point before contact the animals need to extend their legs to help touch down safely. For animals that tuck their legs tightly during flight, this motion would ideally happen at a short and consistent distance from the landing target. Indeed, both flies and honeybees appear to have this capacity [6, 10]. Furthermore, tethered flight experiments with fruit flies have demonstrated that leg extension behavior is triggered by visual cues [11, 12]. Honeybees have also been shown to rely exclusively on visual cues to accurately determine the distance to targets such as artificial flowers [13–15]. These observations suggest that somehow insects are able to measure some feature that is tightly correlated with distance using only vision, yet the precise details of how this might be accomplished are not known.

In this paper we present a novel algorithm, termed *dynamic peering*, that we propose as a potential model for how insects accomplish this task as well as being a useful algorithm for small scale and computationally limited robotics applications where traditional forms of distance estimation such as stereo, sonar, and laser rangefinders are too large, heavy, costly, or computationally intensive. Image based sensing also has the advantage that it is a passive system, rather than needing to send out or receive active signals.

Review of visual distance estimation in biological systems

Before beginning the derivation of the dynamic peering algorithm, we present a brief review of experimentally confirmed mechanisms for vision based distance estimation in biological systems (see also [16]). Perhaps the most familiar mechanism for estimating distance from visual information, on which humans heavily rely, is object recognition. Given a recognizable object and knowledge of its typical size, it is possible to estimate the distance to the object. This process is, however, a cognitively complex task that relies on high

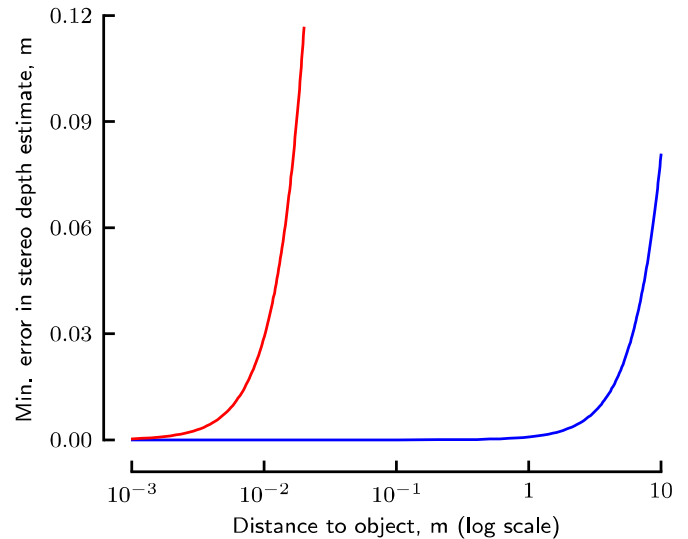


Figure 1. Theoretical errors in stereoscopic absolute distance estimates for humans (blue) and fruit flies (red), calculated using (3).

acuity vision, many layers of visual processing, and a large memory of objects. It is thus an unlikely general mechanism for insects, which have neither the high acuity vision nor the neural capacity necessary for this approach. Another strategy is to use an image-forming lens with a shallow depth of field, which makes it possible to calculate depth from the level of defocus [17]. Such a mechanism, however, is incompatible with the optics of compound eyes [18]. The simplest, and most widely used mechanism for estimating distance—both in biology as well as in computer vision and robotics—is stereopsis. Stereopsis works by calculating the parallax between two (or more) simultaneous images of the same object taken from different viewpoints to triangulate the absolute distance to the object. Many factors play a role in the accuracy of this method, however a simplified distance estimate error (e_d) from a stereoscopic camera pair is given by:

$$e_d \geq \frac{d^2 \xi}{l f}, \quad (1)$$

where d is the distance to the target, l is the interocular distance, f is the focal length of the lens, and ξ is the pixel resolution [19]. To use this equation with biological systems, we rewrite the equation using the relationship:

$$\frac{\xi}{f} = 2 \tan\left(\frac{\theta}{2}\right), \quad (2)$$

where θ is the angular resolution of the visual system. Substituting (2) in (1) yields:

$$e_d \geq 2 \tan\left(\frac{\theta}{2}\right) \frac{d^2}{l}. \quad (3)$$

To gain intuitive insight into this relationship, consider the human visual system, which has an interocular distance of approximately 65 mm and a stereoscopic angular resolution of 0.003° [20]. Given these parameters, the minimum theoretical error in distance estimates is described by the blue curve in figure 1. Next, consider the visual system of the fruit fly, which has an interocular distance of approximately 0.3 mm.

Although the precise angular resolution of the flies' visual system is not known, we can approximate it by the ommatidial acceptance angle of 5° , yielding the red curve in figure 1. Given these results, stereopsis might provide useful information for behaviors that involve operating at very close distances, for example when a male chases a female during courtship. However, it is unlikely that flies could use stereopsis to trigger leg extension during flight, which occurs when they are approximately 1 cm from the target [6], at which distance the minimum error using stereopsis would be approximately 3 cm. Although details vary, such limitations are general for most insects given their small values of interocular distance.

A related, time domain approach to stereopsis is to use sequential images from a single moving eye that has traversed some known distance between acquisitions. Some insects, such as locusts, use this approach to estimate distances before jumping or attacking prey by moving their heads back and forth in a regular fashion, a behavior known as 'peering' [21–23]. For this approach to work, the distance between the positions at which the two images are acquired must be known. This distance could, for example, be calculated if the velocity and time interval are known, or determined directly via proprioceptive sensory feedback [24]. However, because a flying animal has no accurate measure of its true groundspeed (only its airspeed), this method is not feasible for estimating distance for landing behaviors.

Suppose that rather than moving an eye or camera at some known velocity, it is accelerated at a known rate. The acceleration could either be produced along the direction of travel in the case of a straight trajectory, or by changes in direction. As we will show in the following section, knowledge of the acceleration and the time interval between image acquisitions is sufficient to estimate both velocity and absolute distance. This approach, which we term *dynamic peering*, has the intuitive functionality of the nonlinear observer we will develop more formally in the following sections.

Modeling and observability analysis

Of the approaches for vision based distance estimation described above, only dynamic peering has the potential for use by a small flying insect, and thus also by a similarly-scaled robot. In this section, we will take a control theoretic approach to formalize this concept and construct a nonlinear observer that can estimate distance if, and only if, certain controls (actuation patterns) are applied. In order to simplify the system so that the principles are as transparent as possible, we will focus on the problem of landing on a large flat target given a system with simple linear dynamics limited to a single translational degree of freedom. In the final section, we describe a physical implementation of the proposed observer, which demonstrates its feasibility in the real world. Finally, we will propose some methods that would allow our algorithm to be implemented by higher degree of freedom systems in complex environments. Throughout the following sections, we will use the term 'camera' and 'robot' to represent a simple imaging device and some moving agent, but the terms 'eye' and 'animal' could be used as well.

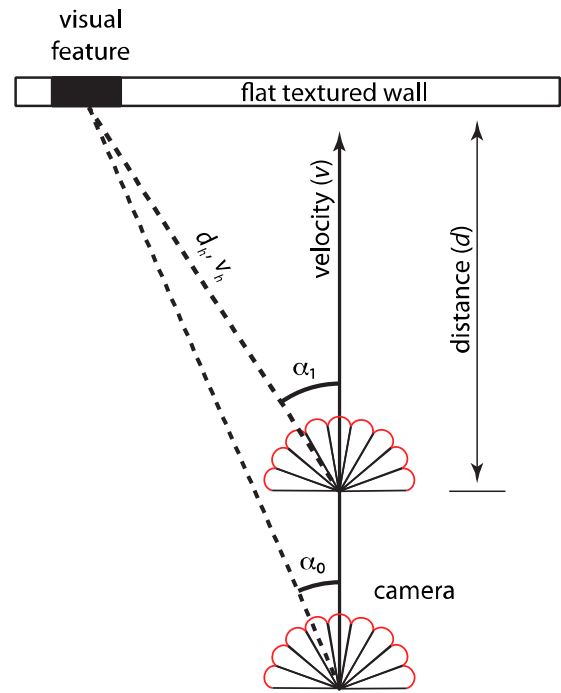


Figure 2. Geometric relationship of the moving camera and a reference object, as referenced in (4)–(5).

Problem statement

Given a single camera and control over forward acceleration, an agent flying toward a flat surface must decelerate to a safe speed and estimate the distance in order to prepare for touchdown.

Analysis

As the camera approaches the surface, from its perspective, textures on the wall will appear to move across its visual field at some angular velocity that is a function of its forward flight speed, the distance to the wall, and the heading angle between the texture and the agent's trajectory. If we assume perfectly spherical optics, this relationship can be described as follows,

$$\lim_{t \rightarrow 0} \left(\frac{\alpha_1 - \alpha_0}{\Delta t} \right) = \frac{d\alpha}{dt} = -\frac{v_h}{d_h} \tan(\alpha), \quad (4)$$

where d_h is the distance to the object, v_h is the velocity in the direction of d_h , α is the position of the object on the camera's retina, and $\dot{\alpha}$ is the angular velocity (e.g. optic flow) of the object relative to the camera, see figure 2.

Assuming spherical lens geometry, α corresponds directly to the heading of the object relative to the camera. Thus, for each direction α , the ratio v_h/d_h is directly proportional to $\dot{\alpha}$ by a constant of $-1/\tan(\alpha)$. In the case that the camera is moving directly toward a flat wall, we can relate all such measurements for different α 's and rewrite (4) in terms of the forward velocity (v) and the distance to the wall (d):

$$-\frac{\dot{\alpha}}{\tan(\alpha)} = \frac{v_h}{d_h} = \frac{v}{d}. \quad (5a)$$

By only using measurements corresponding to small α (e.g. from the center of its field of view), the equation further simplifies to:

$$\frac{v}{d} = -\frac{\dot{\alpha}}{\tan(\alpha)} \approx -\frac{\dot{\alpha}}{\alpha}. \quad (5b)$$

These simplifications make it possible to estimate v/d as the mean of $-\dot{\alpha}/\alpha$ across all (small) α . In the following sections we use this relationship to describe optic flow as v/d . Note that although this relationship appears to be poorly defined at $\alpha = 0$, $\dot{\alpha}$ also tends toward zero in this direction, and the limit remains well-defined. In a real-world implementation where noise is unavoidable, this relationship is prone to producing large errors. In our implementation section, we discuss a simple solution to this problem.

Next consider a robot equipped with a single camera flying straight toward a static object. The equations of motion can be written as:

$$\begin{aligned} \begin{bmatrix} \dot{d}(t) \\ \dot{v}(t) \end{bmatrix} &= \begin{bmatrix} v(t) \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \\ &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} d(t) \\ v(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t), \end{aligned} \quad (6)$$

where d is the distance to the object, v is the forward flight speed, and u is the control input (which is equivalent to acceleration with these dynamics). With optic flow as the system's only sensory input, we can write the observations as:

$$y(t) = [v(t)/d(t)]. \quad (7)$$

Although the dynamics are linear, the observation equation, y , is nonlinear. In order to use linear systems analyses, we begin by linearizing the system about a nominal trajectory, $(d_t(t), v_t(t))$. This choice allows us to write the system in the canonical state space form:

$$\dot{x}(t) = Ax(t) + Bu(t)$$

$$y(t) = Cx(t) + Du(t),$$

where, for our system, the terms are defined as follows:

$$x(t) = \begin{bmatrix} d(t) \\ v(t) \end{bmatrix} \quad (8a)$$

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} C = [-v_t/d_t^2 \quad 1/d_t] D = [0]. \quad (8b)$$

To address the question of whether or not it is possible to estimate distance and velocity using only optic flow, we examine the system's observability (a measure for how well the states of a system, such as position and velocity, can be inferred given the available sensory measurements [25]). First, we check the linear observability condition for an arbitrary nominal trajectory by calculating the rank of the observability matrix:

$$\begin{bmatrix} C \\ CA \end{bmatrix} = \begin{bmatrix} -v_t/d_t^2 & 1/d_t \\ 0 & -v_t/d_t^2 \end{bmatrix}. \quad (9)$$

This matrix is full rank along any trajectory provided that neither the velocity nor distance is zero, suggesting that the system is observable. Because in practice we are concerned

with non-zero velocities at non-zero distances, this limitation is not critical. This conclusion, however, goes against the intuition presented in the introduction, which suggested that it should be difficult, if not impossible, to extract either distance or velocity from optic flow. To explore this discrepancy, we use an alternative check for calculating the observability of a linear time-varying system, the observability Gramian:

$$P(t_0, t_f) = \int_{t_0}^{t_f} e^{A^T t} C^T C e^{A t} dt, \quad (10)$$

where C (defined in (8b)) is time varying. The advantage of this approach is that the condition number (the ratio of the minimum and maximum singular values of a matrix), termed the *local estimation condition number*, is a direct measure of the well-posedness of the estimation problem [26]. The smaller the condition number, the better posed the estimation problem. Calculating (10) analytically is often challenging for complex systems. Instead, it is possible to numerically estimate the observability Gramian, termed the *empirical local observability Gramian*, by simulating the system and comparing the outputs y for perturbations $\pm\epsilon$ of the initial condition [26, 27]. It can be shown that as $\epsilon \rightarrow 0$, the estimate converges to the result from (10). Using this approach on our system shows that the condition number for a constant velocity trajectory approaches infinity for any time interval (figure 3, black trace), suggesting the system is not in fact observable in a linear sense. However, significantly smaller condition numbers exist if we consider non-constant velocity trajectories (figure 3).

For a more direct confirmation that non-zero control inputs are required for the system to be observable, we can employ a nonlinear observability analysis, which draws on Lie algebraic tools to account for the contribution of active controls.

A brief review of the key elements will be presented here; for a more detailed discussion see [25]. The Lie derivative of the observation equation, $y = h(x)$, with respect to a vector field, $f_i(x) \in \mathbb{R}^n$, where $x \in \mathbb{R}^n$, is defined as:

$$L_{f_i} h = \frac{\partial h}{\partial x} f_i. \quad (11)$$

Intuitively, the Lie derivative represents the change in a function or vector field along a vector field. Applied to the observability problem presented in this paper, $L_{f_i} h$ is the change in the observations ($y = h(x) = \text{optic flow}$) along either the drift dynamics, $f_0 = Ax$, or the control direction, $f_1 = B$. Because the drift dynamics cannot be turned off, we take a repeated Lie derivative with respect to both f_0 and f_1 in order to calculate the change in the observations with respect to the control. This repeated Lie derivative is defined as:

$$L_{f_0 f_1} h = L_{f_0} L_{f_1} h = \frac{\partial}{\partial x} (L_{f_1} h) f_0. \quad (12)$$

Next, define the *observability Lie algebra*, \mathcal{O} , which is the collection of Lie derivatives of the observations, h , with respect to the drift dynamics and each of the controls. In the case of our problem:

$$\mathcal{O} = \{h(x), L_{f_0} h(x), L_{f_0 f_1} h(x)\} = \left\{ \frac{v}{d}, \frac{-v^2}{d^2}, \frac{v}{d^2} \right\}. \quad (13)$$

If the Jacobian of \mathcal{O} is full rank (i.e. if the number of linearly independent terms is equal to the number of states in the

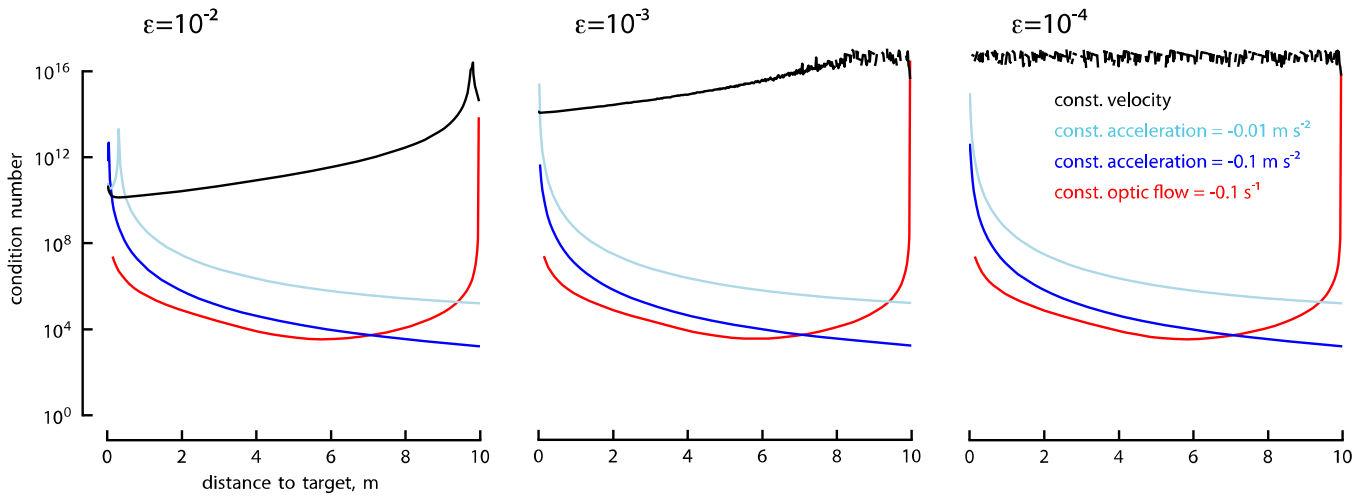


Figure 3. Condition number of the empirical local observability Gramian for a constant velocity trajectory (black) approaches infinity, whereas a constant acceleration trajectory (blue) or a constant optic flow trajectory (red) achieves significantly lower condition numbers. The constant optic flow trajectory is obtained by using the controller given by (14), and will be discussed at length later in the paper.

system) at all states, the system is said to be observable. If \mathcal{O} must include Lie derivatives between h and a control vector, f , in order to become full rank, this control dimension must be actuated in some way in order for the system to be observable. In our system, the terms $\{\frac{v}{d}, \frac{-v^2}{d^2}\}$ are linearly dependent, so \mathcal{O} reduces to $\{\frac{v}{d}, \frac{v}{d^2}\}$, which has a full rank Jacobian if and only if the term $L_{f_0 f_1} h(x)$ is included (and for $v \neq 0$ and $d \neq 0$). The presence of the term $L_{f_0 f_1} h(x)$ suggests that in order for the system to be fully observable, some acceleration must be applied. That is, if the camera-equipped robot were approaching the wall at constant velocity, we would not be able to observe d and v separately. If the camera accelerates, however, such an observation is possible. These results are consistent with our conclusions from the observability Gramian analysis in figure 3, and is the critical feature of our proposed method.

In principle, we could conclude our analysis at this point and simply implement an unscented Kalman filter (UKF) or particle filter to estimate the unobserved states [28] and ensure that the system accelerates sufficiently often. Unfortunately, however, the observability Lie algebra calculation does not give any indication of *what* that control needs to be; only that it cannot be identically zero. To address this limitation, previous studies have developed methods that use the condition number of the observability Gramian to rank potential trajectories [27]. In this paper, we take a bio-inspired approach as described below.

Consider the landing algorithm presented in the introduction, where an animal maintains a constant rate of optic flow in order to guarantee a safe landing velocity at touchdown. This motion can be accomplished by using a simple proportional feedback controller on the velocity:

$$u = k(r - r_d), \quad (14)$$

where $r = v/d$, and r_d is the desired ratio of v/d . To understand the implications of such a controller we write:

$$\text{optic flow} \propto r = v/d, \quad (15)$$

$$\frac{d}{dt} r = \frac{d}{dt} v d^{-1} = \frac{\dot{v}}{d} - \frac{v^2}{d^2} = \frac{\dot{v}}{d} - r^2 = \frac{u}{d} - r^2. \quad (16)$$

Rewriting this equation yields the following equation for distance:

$$d = \frac{\dot{v}}{\dot{r} + r^2} = \frac{u}{\dot{r} + r^2}. \quad (17)$$

Thus, given non-zero control, it is possible to estimate distance directly from the control input and a measurement of r along with its derivative, both of which can be extracted from measurements of optic flow. It is important to note, however, that this relationship has several pitfalls, particularly for a noisy system. As we will describe in the implementation section, current computer vision methods for calculating optic flow are indeed noisy, so these pitfalls present a real problem. In the case that $\dot{r} + r^2 = 0$, the distance estimate approaches infinity. Although this will theoretically not happen if the acceleration is sufficiently large, there is no such guarantee in a noisy system. Given the bio-inspired controller that maintains a constant rate of optic flow ($\dot{r} = 0$), however, we can simplify the distance estimate in (17) to:

$$d = \frac{u}{r_d^2}, \quad (18)$$

where r_d is the desired rate of optic flow, u is the control input (acceleration), and d is the distance to the target. This equation evolves with time according to:

$$d = d_0 e^{r_d t}, \quad (19)$$

where t is time, and d_0 is the (unknown) initial distance. In this way, we have removed the potential for dividing by zero, and all the effects of noise reside within the relative safety of the numerator. Furthermore, by using (19) we have reduced our estimation problem to that of estimating a single parameter—the initial distance—allowing us to calculate a clean distance estimate with a sequential least squares (SLS) filter [28]. This approach is, of course, limiting for general applications. However, in the context of landing, where a constant rate of optic flow controller is ideal, these constraints do not present a critical problem. On the contrary, they provide an exceptionally

elegant method for triggering behaviors such as leg extension based on the internal state of control (or the measured value of acceleration). In the discussion, we explore the possibilities for expanding this approach to more general cases.

Implementation

As a proof-of-concept physical demonstration of the distance estimation algorithm presented in the previous section, we implemented it using a camera (Basler Ace 640—100 gm) equipped with a 1.4 mm fisheye lens (Fujinon C Mount 1.4 mm CCTV Fish-Eye) mounted to a linear stage. The camera was driven by a computer controlled servo along a 1.5 m track toward a panoramic image of a forested scene. All the image processing and estimation were done on a desktop computer running Ubuntu Linux. Images were acquired with the open source camera aravis driver, and the rectified images were published on a Robot Operating System (ROS fuerte) network. All subsequent processing was done in Python. We used OpenCV's implementation of the Lucas–Kanade algorithm to calculate optic flow over a region of interest that corresponded to an approximately 45° field of view in the horizontal direction. For simplicity, we restricted our analysis to the optic flow along the horizontal dimension.

Figure 4 shows a representative measurement of optic flow along this axis. Recall from (5) that optic flow ($\dot{\alpha}$) is equal to $-\tan(\alpha)v/d$. In order to extract v/d , we fit a line to the central portion of this curve using a RANSAC algorithm [29], and used the slope of this line as the estimate for v/d . Note that this approach circumvents the potential for dividing by zero at small α and also makes the algorithm robust to poor camera alignment and noisy optic flow estimates.

In order to use these optic flow estimates for our estimation problem, we first had to calibrate the system, since the Lucas–Kanade algorithm provides normalized values between ± 1 . To calibrate the system, we drove the camera at various known rates of v/d and recorded the associated slope. After collecting several of these points, we did a least squares fit to determine the relationship between our slope estimate and the true v/d .

Next, we implemented a simple proportional controller (14) with gain $k = 6$ to adjust the acceleration of the camera such that it maintained the desired value of optic flow. The acceleration commands were turned into updated velocity commands based on the operating frequency of the control loop and were sent to an Arduino Uno board over USB. The Arduino continuously generated step and direction commands that were sent to the stepper motor controller, which in turn moved the camera. These step commands served as our ground truth of the actual distance and velocity of the camera to which we could compare our algorithms' estimates. The rate limiting step of our implementation was the communication to the Arduino board, resulting in a 50 Hz operating frequency, which was sufficient for our demonstration purposes.

To estimate the distance, we used a two-step process. For each cycle of the 50 Hz control loop, we estimated distance using (18). These distance estimates were then run through a SLS filter to estimate the initial distance d_0 in (19). Because (18) is only valid when the system is moving

with the desired rate of optic flow, we kept the covariance in the SLS filter artificially high until the desired rate of optic flow was reached. Our software, as well as the data collected using this system, are freely available online at www.github.com/florisvb/dyneeye.

Results and discussion

The camera started out at zero velocity 1.5 m away from the target, quickly accelerated up to the desired optic flow set-point, and subsequently began gently decelerating as it approached the target so as to maintain the desired constant rate of optic flow. Figure 5 shows a comparison between the actual position, the nonlinear observer estimates at each time point calculated using (18), and the SLS filtered estimates. Figures 4 and 5 are also provided as a movie (see supplemental materials (available from stacks.iop.org/BB/9/025002/mmedia)). We also implemented a square-root UKF [30] and found very similar performance to the SLS results but with much greater computational overhead (not shown for the sake of graphical clarity).

Because our nonlinear observer (18) only provides valid information when the system is close to the desired trajectory, the position and velocity estimates show large initial errors. These initial errors are an artifact of our system starting out at zero velocity and initially needing to accelerate to reach the desired optic flow rate. The initial errors are more pronounced in experiments with higher rates of optic flow, as the system took longer to reach the target level. This left little space on our limited track to operate at the final desired rate. In a freely moving system, deceleration could be triggered when the optic flow reaches the desired threshold, and subsequent distance and velocity estimates would be accurate so long as the system maintained the target level of optic flow. For very slow rates of optic flow ($r > -0.01$), the poor estimates are most likely due to insufficient changes in pixel values between sequential image acquisitions to calculate accurate measures using the Lucas–Kanade algorithm. This limitation could potentially be solved by using longer delays between image acquisitions.

Applications to robotic systems

The approach presented here is best suited for landing applications due to our trajectory choice; however, by integrating the algorithm into a more complex trajectory it could provide more general utility in navigation tasks. For example, a flying robot could periodically approach the ground below it with constant optic flow to estimate its altitude and use this measurement as a calibration for other optic flow estimates. Recall that optic flow alone can provide relative measurements to different objects, so if the distance to any one of these objects is known, the others can be calculated as well. Between these bouts, the robot could use other bio-inspired visual cues to maintain a constant altitude, such as local horizon following [31]. Although our implementation made use of a full desktop computer, this use was purely for convenience. Optic flow can be calculated with simple parallel analog circuits, without the need for extensive memory [32].

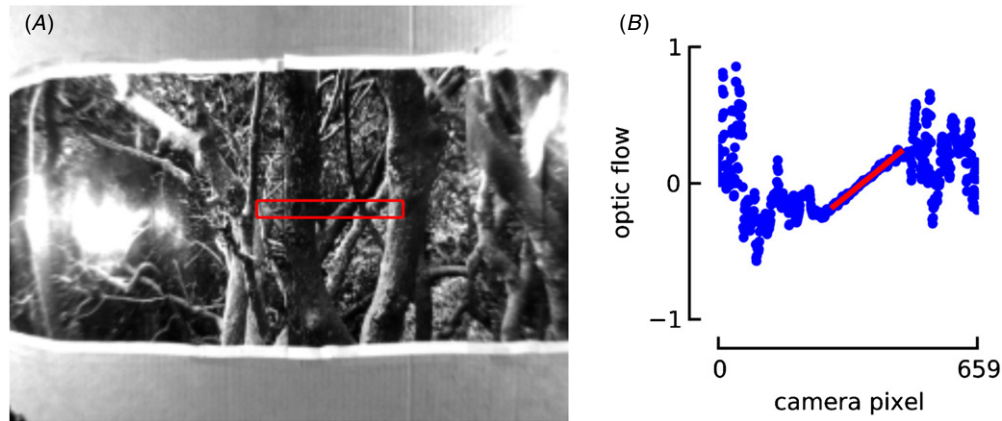


Figure 4. Visual target and optic flow estimation. (A) Example camera image showing the visual target and region of interest (red box). (B) Optic flow as a function of camera pixel from two successive frames, calculated using OpenCV's Lucas-Kanade algorithm. For the purposes of control, we calculated a linear fit of the data (red line) over the region of interest indicated in (A).

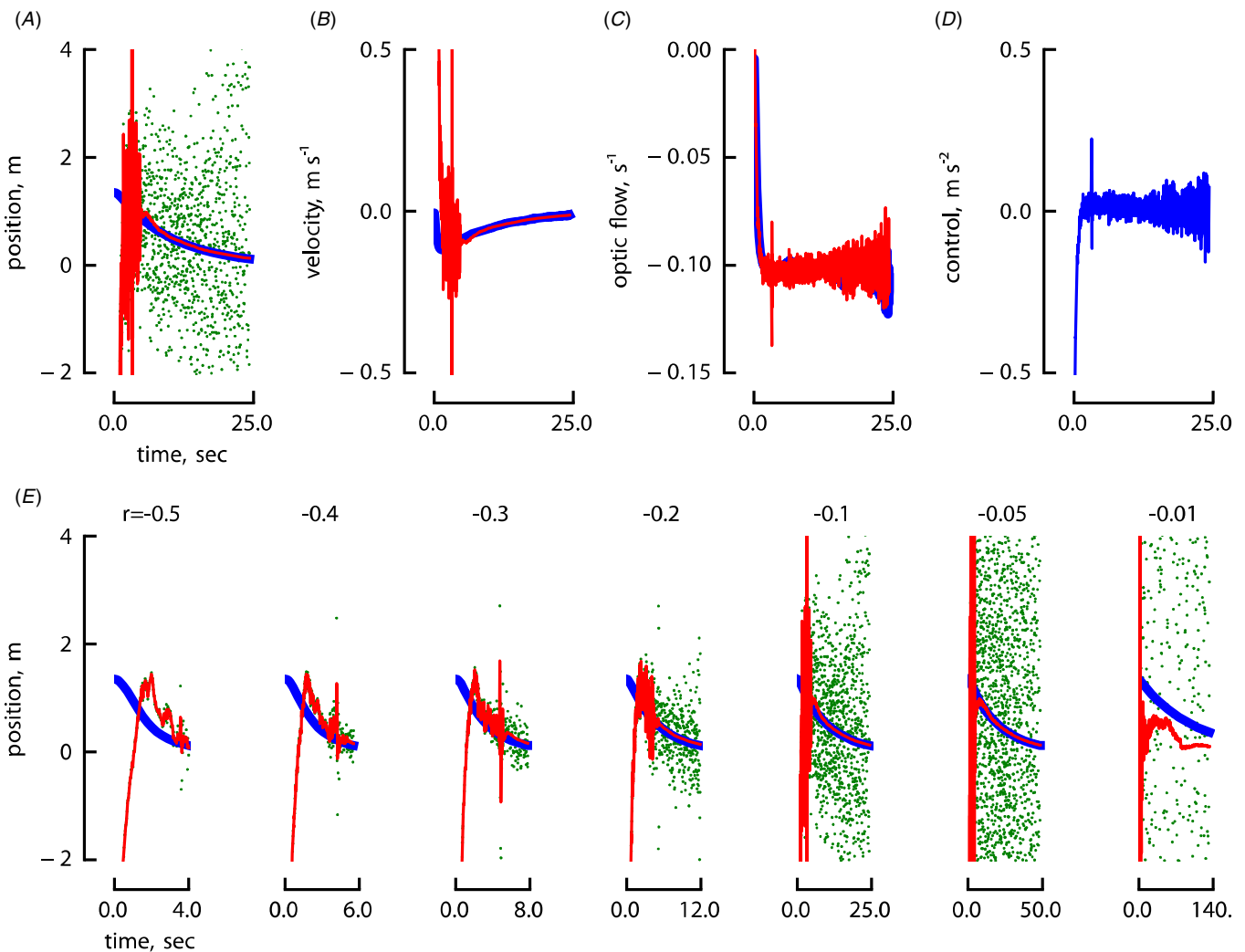


Figure 5. Performance of the dynamic peering estimation algorithm. Panels (A)–(D) show results for a desired optic flow rate of -0.1 s^{-1} . (A) Actual position (blue), raw dynamic peering estimate (green), and the SLS filtered dynamic peering estimate (red). (B) Actual velocity (blue), dynamic peering with SLS filtering (red). (C) Actual velocity/distance (blue) and optic flow measurement based on Lucas-Kanade calculations (red). (D) Control commands, equivalent to acceleration (blue). (E) Same figure as (A) repeated for different desired optic flow rates (indicated above each plot).

We restricted our analysis to a single degree of freedom system for simplicity, however the principles presented here can be generalized to three-dimensional motion. In three dimensions, the non-zero acceleration requirement can also be satisfied by non-zero angular accelerations, i.e. changes in direction [33].

Implications for landing insects

The dynamic peering algorithm presented here provides a plausible mechanism for how insects might estimate distance in order to trigger leg extension or any other behavior during landing maneuvers. Once the animal starts decelerating in preparation for landing by maintaining a constant rate of optic flow, it could trigger leg extension when either its internal control signal, or a measured value of acceleration, reaches a particular threshold. This threshold would always correspond to the same distance if the animal uses the same rate of optic flow for regulating its deceleration, a hypothesis supported by behavioral observations [6, 7]. Such a threshold calculation could be easily implemented by the nervous system with as little as a single neuron. To our knowledge, this is a novel and plausible principle. Insects, such as flies [34–36], are known to use optic flow to regulate their velocity by adjusting acceleration, suggesting that such an internal control signal is indeed present.

To experimentally test whether insects use dynamic peering, one could construct an experiment in which animals are tricked into decelerating faster or slower than usual by using a virtual reality system. If the insects use an approach similar to what we describe, their leg extension behavior would be correlated with their internal control commands, and thus the observed acceleration behavior. Unfortunately this type of experiment is challenging to perform, as it requires free flight observations of leg extension behavior and simultaneous control of the visual stimulus in closed loop. As reported previously, flies landing on a target typically extend their legs after having first initiated a deceleration [6]. However, flies that approached the target but did not decelerate, did not extend their legs. These results suggest that leg extension is not simply triggered by object size or image velocity, but rather it depends on some aspect of the flies' internal state, or their deceleration behavior, an observation that is consistent with our model for distance estimation. Tethered flight experiments in fruit flies, as well as other insects, have shown that visual stimuli are sufficient to elicit strong leg extension responses [11, 12]. These results from tethered flies seem contradictory to our hypothesis because the flies extend their legs without physically decelerating. However, in these tethered flight experiments it is impossible to know if the flies were attempting to decelerate. Assuming they were, this behavior could be explained if they used an internal signal (e.g. efferent copy) of the control output. It is also possible that there are multiple sensory-motor pathways that can elicit leg extension.

Our analysis focused on a single degree of freedom trajectory, which is consistent with the landing behavior of fruit flies which do not make significant changes in heading

after initiating deceleration prior to landing [6]. Additional degrees of freedom in the system would make it possible for an insect to extract a true depth estimate from optic flow simply by changing direction by a *known* amount. Previously published results from experiments with honeybees are consistent with this option, having shown that they do actively change direction in a stereotypical fashion in order to determine the height of objects above the ground using optic flow information [14].

Summary

To summarize, in this paper we used tools from control theory to show that non-zero acceleration is necessary and sufficient to estimate distance from optic flow with a single camera. There are, however, an infinite number of possible trajectories that satisfy these requirements, many of which will fail in real-world noisy implementations. In order to choose a trajectory that provides an accurate estimate of distance, we turned to biological inspiration from landing insects, which decelerate so as to keep their optic flow at a constant value. This choice of trajectory simplifies the estimation problem to a single parameter (initial distance), resulting in an accurate estimate of distance from SLS filtering. Our dynamic peering algorithm provides a plausible, and testable, mechanism for how insects might trigger leg extension prior to landing, as well as being a novel method for estimating distance with a single camera in robotic systems.

Acknowledgments

The authors wish to thank Dr Michael Elzinga for help with constructing the linear rail mechanism used in our robotic implementation, and Nathan Powell for providing MATLAB code to run the square root implementation of the unscented Kalman filter. Funding was provided by the Hertz Foundation Graduate Research Fellowship (awarded to FvB), NSF Graduate Research Fellowship (awarded to FvB), Air Force Office of Scientific Research (FA9550-10-1-0368), and the Paul G. Allen Family Foundation Distinguished Investigator Award (awarded to MHD). The funders had no role in study design, data collection and interpretation, or the decision to submit the work for publication.

The authors declare that no competing interests exist.

References

- [1] Koenderink J J 1986 Optic flow *Vis. Res.* **26** 161–80
- [2] Wittlinger M, Wehner R and Wolf H 2006 The ant odometer: stepping on stilts and stumps *Science* **312** 1965–7
- [3] Gibson J J 1958 Visually controlled locomotion and visual orientation in animals *Br. J. Psychol.* **49** 182–94
- [4] Gibson J J 1950 *The Perception of the Visual World* (Cambridge, MA: Riverside Press)
- [5] Besl P J 1988 Active, optical range imaging sensors *Mach. Vis. Appl.* **1** 127–52
- [6] Van Breugel F and Dickinson M H 2012 The visual control of landing and obstacle avoidance in the fruit fly *Drosophila melanogaster* *J. Exp. Biol.* **215** 1783–98

- [7] Srinivasan M V, Zhang S W, Chahl J S, Barth E and Venkatesh S 2000 How honeybees make grazing landings on flat surfaces *Biol. Cybern.* **83** 171–83
- [8] Wang Y and Frost B J 1992 Time to collision is signalled by neurons in the nucleus rotundus of pigeons *Nature* **356** 236–8
- [9] Chahl J S, Srinivasan M V and Zhang S W 2004 Landing strategies in honeybees and applications to uninhabited airborne vehicles *Int. J. Robot. Res.* **23** 101–10
- [10] Evangelista C, Kraft P, Dacke M, Reinhard J and Srinivasan M V 2010 The moment before touchdown: landing manoeuvres of the honeybee *Apis mellifera* *J. Exp. Biol.* **213** 262–70
- [11] Tammero L F and Dickinson M H 2002 Collision-avoidance and landing responses are mediated by separate pathways in the fruit fly, *Drosophila melanogaster* *J. Exp. Biol.* **205** 2785–98
- [12] Borst A and Bahde S 1986 What kind of movement detector is triggering the landing response of the housefly *Biol. Cybern.* **55** 59–69
- [13] Lehrer M, Srinivasan M V and Zhang S W 1988 Motion cues provide the bee's visual world with a third dimension *Nature* **332** 356–7
- [14] Lehrer M 1996 Small-scale navigation in the honeybee: active acquisition of visual information about the goal *J. Exp. Biol.* **199** 253–61
- [15] Cartwright B A and Collett T S 1979 How honey-bees know their distance from a near-by visual landmark *J. Exp. Biol.* **82** 367–72
- [16] Collett T S and Harkness L I K 1982 Depth vision in animals *Analysis of Visual Behavior* ed D J Ingle, M A Goodale and R J W Mansfield (Cambridge, MA: MIT Press) pp 111–76
- [17] Pentland A P 1987 A new sense for depth of field *IEEE Trans. Pattern Anal. Mach. Intell.* **9** 523–31
- [18] Land M F and Nilsson D-E 2012 *Animal Eyes* (Oxford: Oxford University Press)
- [19] Verri A and Torre V 1986 Absolute depth estimate in stereopsis *J. Opt. Soc. Am. A* **3** 297
- [20] Walls G L 1943 Factors in human visual resolution *J. Opt. Soc. Am.* **33** 487–505
- [21] Sobel E C 1990 The locust's use of motion parallax to measure distance *J. Comp. Physiol. A* **167** 579–88
- [22] Wallace G K 1959 Visual scanning in the desert locust *Schistocerca gregaria* Forskal *J. Exp. Biol.* **36** 512–25
- [23] Collett T S 1978 Peering—a locust behavior pattern for obtaining motion parallax information *J. Exp. Biol.* **76** 237–41
- [24] Poteser M, Pabst M A and Kral K 1998 Proprioceptive contribution to distance estimation by motion parallax in a praying mantis *J. Exp. Biol.* **201** 1483–91
- [25] Nijmeijer H and Van der Schaft A J 1990 *Nonlinear Dynamical Control Systems* (New York: Springer)
- [26] Krener A J and Ide K 2009 Measures of unobservability *Proc. 48th IEEE Conf. on Decision and Control (Shanghai, Dec. 2009)* pp 6401–6
- [27] Hinson B T and Morgansen K A 2012 Flowfield estimation in the wake of a pitching and heaving airfoil *Proc. American Control Conf. (Montréal, June 2012)*
- [28] Crassidis J L and Junkins J L 2012 *Optimal Estimation of Dynamic Systems* (London: Chapman and Hall)
- [29] Fischler M A and Bolles R C 1981 Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography *Commun. ACM* **24** 381–95
- [30] Van der Merwe R and Wan E A 2001 The square-root unscented Kalman filter for state and parameter-estimation *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (Salt Lake City, UT, May 2001)* vol 6 pp 3461–4
- [31] Straw A D, Lee S and Dickinson M H 2010 Visual control of altitude in flying *Drosophila* *Curr. Biol.* **20** 1550–6
- [32] Stocker A A 2005 Analog integrated 2D optical flow sensor *Analog Integr. Circuits Signal Process.* **46** 121–38
- [33] Alaeddini A and Morgansen K A 2013 Autonomous state estimation using optic flow sensing *Proc. American Control Conf. (Washington, DC, June 2013)*
- [34] Fry S N, Rohrseitz N, Straw A D and Dickinson M H 2009 Visual control of flight speed in *Drosophila melanogaster* *J. Exp. Biol.* **212** 1120–30
- [35] Rohrseitz N and Fry S N 2011 Behavioural system identification of visual flight speed control in *Drosophila melanogaster* *J. R. Soc. Interface* **8** 171–85
- [36] Medici V and Fry S N 2012 Embodied linearity of speed control in *Drosophila melanogaster* *J. R. Soc. Interface* **9** 3260–7