2007 American Control Conference
http://www.cds.caltech.edu/~murray/papers/eps+07-acc.html

Biologically Inspired Feedback Design for Drosophila Flight

Michael Epstein, Stephen Waydo, Sawyer B. Fuller, Will Dickson,
Andrew Straw, Michael H. Dickinson and Richard M. Murray

Abstract— **We use a biologically motivated model of the *Drosophila's* flight mechanics and sensor processing to design a feedback control scheme to regulate forward flight. The model used for insect flight is the Grand Unified Fly (GUF) [3] simulation consisting of rigid body kinematics, aerodynamic forces and moments, sensory systems, and a 3D environment model. We seek to design a control algorithm that will convert the sensory signals into proper wing beat commands to regulate forward flight. Modulating the wing beat frequency and mean stroke angle produces changes in the flight envelope. The sensory signals consist of estimates of rotational velocity from the haltere organs and translational velocity estimates from visual elementary motion detectors (EMD's) and matched retinal velocity filters. The controller is designed based on a longitudinal model of the flight dynamics. Feedforward commands are generated based on a desired forward velocity. The dynamics are linearized around this operating point and a feedback controller designed to correct deviations from the operating point. The control algorithm is implemented in the GUF simulator and achieves the desired tracking of the forward reference velocities and exhibits biologically realistic responses.**

I. Introduction

In this work we seek to use a biologically motivated model of the *Drosophila's* flight mechanics and sensor processing units to design a feedback control scheme to regulate forward flight. The goal is to design a stabilizing feedback control scheme that exhibits responses similar to observed flight behaviors. As is traditional in a control systems paradigm, this goal was broken into two roughly separable tasks:

1) Extract estimates of relevant state variables (i.e. velocity, orientation, etc.) from the sensory inputs available to the fly.
2) Design a feedback control law capable of stabilizing the fly in free flight using biologically correct control inputs.

Recent research efforts have been directed at developing models and control algorithms that utilize similar biologically inspired designs. Humbert, et al [6], [7] use idealized optical flow measurements to stabilize hovercraft. In addition to the obvious differences in the dynamics of the simulated hovercraft and rotorcraft, another significant difference with our work is that we chose to estimate the translational velocities based on visual velocity estimates from physically realistic optical simulations and biologically plausible neural operations. Humbert used a mathematically-derived description of perfect estimates of retinal velocities within pre-defined environmental geometries and linearized the results about a reference trajectory to arrive at a formal relation between such retinal velocities and control system inputs. Unfortunately, our choice of more physically realistic constraints makes such a formal analysis substantially more difficult. This is because we did not make the simplifying assumption that instantaneous estimates of retinal velocities across visual space were available, but rather that such velocity estimates would have to be computed from information falling on simulated light receptors. Deng, et al [2], [1] model a micro-aerial vehicle designed to mimic insect flight that has parallels to this work. One main difference appears to be that the authors were concerned with attitude control during hovering, while in this work we consider trajectory tracking of forward flight. The authors model the micromechanical flying insect [5] whose design is based on a typical housefly, while in this work we use the model developed in GUF [3] thus subtle differences arise in the models.

The paper is organized as follows. Section II briefly describes the GUF simulation model. The controller design is described in Section III. A method of obtaining velocity estimates from the visual system is described in Section IV. Simulation results of the controller implemented in GUF are shown in Section V. The paper concludes with a summary and description of future work in Section VI.

II. Insect Flight Model

The model used for insect flight is the Grand Unified Fly (GUF) [3]. This model is in the form of a numerical simulation comprised of five main components: an articulated rigid-body simulation, a model of the aerodynamic forces and moments, a sensory systems model, a control model and an environment model. A brief description of the GUF simulation environment is provided below; for a more complete description see [3].

In the context of this paper, the rigid body simulation of the fly and aerodynamic forces is the plant that we wish to control. It is represented by a system of three rigid bodies (two wings and a body) connected by a pair of actuated ball-joints. At each instant of the simulation, the aerodynamic forces and moments acting on the wings and body of the fly are calculated using an empirically derived quasi-steady state model. The model is based on experimental measurements of force and moment coefficients taken from a dynamically-scaled robotic model in a tow tank. The model

Support for this work was partially provided by AFOSR Grant FA9550-06-1-0079 as well as a Fannie and John Hertz Foundation Fellowship for S. Waydo and a National Science Foundation Graduate Fellowship for S. Fuller.

All authors are in the Division of Engineering and Applied Science at the California Institute of Technology epstein,waydo,minster,wbd,astraw,dickinson, murray@caltech.edu

wing kinematics are based on data captured from high-speed video sequences of real fly flight. The forces and moments produced by the wings are modulated by deforming these baseline kinematics along certain characteristic actuation modes believed to be those employed by the fly.

The two key sensory systems of interest that are modeled in GUF are the visual system and the halteres. The halteres produce estimates of the rotational velocity of the fly. Since they provide complete information about the rotational velocity we assume that we have direct measurements of those states rather than using a detailed model of the halteres. The visual system can provide estimates of the fly's translational velocity relative to the external environment as well as the presence of approaching obstacles or distant targets. The environmental model provides input to both the sensory systems and the aerodynamics model. Different visual information can be presented to the sensory system allowing the flexibility of placing the insect in different virtual habitats, simulating both field and laboratory conditions. The environmental model can also impart wind gusts to provide disturbances.

The goal is to design a control law that uses the sensory outputs to make the appropriate modulations in the wing kinematics and produce stable flight behavior. A schematic of the feedback loop for fly flight is shown in Figure 1.

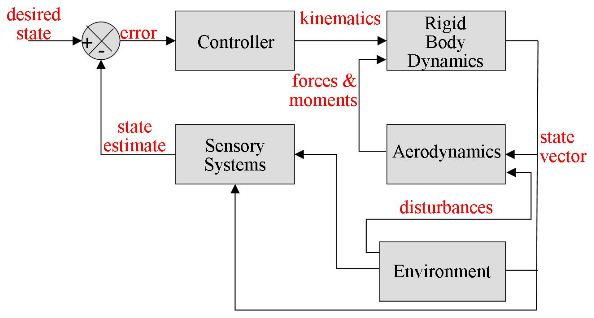

Fig. 1. Schematic of feedback control loop for fly flight.

III. Longitudinal Flight Model and Controller Design

A. Longitudinal Flight Model

To design a controller for the fly feedback loop we started with a longitudinal flight model based on the stroke averaged forces. That is, we only considered the averaged forces generated during a complete revolution of the wing, while the GUF simulation actually calculates the forces hundreds of times per wing beat. Rather then looking at the full 6 degree-of-freedom rigid body we also restricted our focus to motion in the $x-z$ plane including pitching about the y axis as shown in Figure 2. Note our coordinate system has x in the forward direction and z up, with y defined by the right-hand rule from these. In this setting the states of the fly are forward and vertical velocities $(\dot{x}, \dot{z})$, as well as pitch angle and pitch rate $(\theta, \dot{\theta})$. We have used the notation that (x_b, z_b) signify the body frame coordinates and (x, z) are the global inertial frame. We will use rotation matrices of the form

$$C_{i,j}(\xi) = \begin{bmatrix} \cos\xi & \sin\xi \\ -\sin\xi & \cos\xi \end{bmatrix} \tag{1}$$

to represent a vector in reference frame i that is given in frame j, where ξ is the angle between the two reference frames, i.e. $x_i = C_{i,j}(\xi)x_j$. The pitch angle θ defines the rotation matrices relating the two frames; to go from the body to global frame the rotation matrix is given by $C_{gb}(\theta)$. Likewise the body angle β defines the rotation between the parallel and normal axes and the body frame, so the rotation matrix going from the Normal-Parallel axes to the body frame is $C_{b,\mathrm{NP}}(\beta)$.

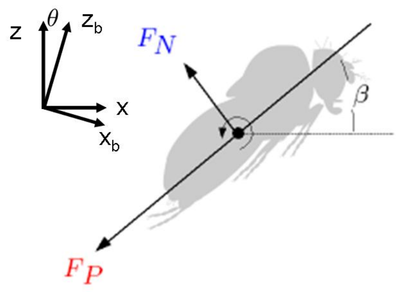

Fig. 2. Schematic of fly's longitudinal motion. Global reference frame is given by (x, z) and the body frame by (x_b, z_b). The pitch angle is θ and β is a fixed value and is termed the body angle.

The longitudinal dynamics of the fly are given by

$$\ddot{x} = \frac{1}{m}(F_{b,x} + F_{w,x}) \tag{2}$$

$$\ddot{z} = \frac{1}{m}(F_{b,z} + F_{w,z}) - g \tag{3}$$

$$\ddot{\theta} = \frac{1}{I}(M_{b,y} + M_{w,y}) \ , \tag{4}$$

where g is the acceleration due to gravity, m is the mass of the fly and I the inertia about the y axis. The forces and moments acting on the fly have been broken up into body and wing components. The body forces and moments are denoted by $(F_{b,\cdot}, M_{b,y})$ and the wing forces and moments by $(F_{w,\cdot}, M_{w,y})$, where the $\cdot$ signifies the corresponding component along a particular axis.

The body forces are only a function of the flight aerodynamics, that is they do not directly depend on the fly's inputs but rather only on its current state and physical and aerodynamic parameters according to

$$F_N = -\frac{1}{2}\rho S C_N v_N v$$

$$F_P = -\frac{1}{2}\rho S C_P v_P v$$

where ρ is the air density, S is the surface area, (C_N, C_P) are the aerodynamic coefficients along the normal and parallel axes respectively, $v = \sqrt{v_{x_b}^2 + v_{z_b}^2}$ the velocity magnitude and (v_N, v_P) represent the velocity components along the normal and parallel axes respectively. To express these forces in the global frame simply apply the transformation

$$\begin{bmatrix} F_{b,x} \\ F_{b,z} \end{bmatrix} = C_{gb}(\theta) C_{b,\mathrm{NP}}(\beta) \begin{bmatrix} F_N \\ F_P \end{bmatrix} \tag{5}$$

The body moment is calculated according to

$$M_{b,y} = \frac{1}{2} \rho S L C_M v^2 \tag{6}$$

where L is the body reference length, C_M is the moment coefficient given by

$$C_M = \sum_{k=1}^{4} a_k \sin k\alpha + b_k \cos k\alpha \ , \tag{7}$$

where $\alpha = \tan^{1} \frac{v_N}{v_P}$ is the angle of attack and (a_k, b_k) are fixed coefficients determined experimentally from a dynamically scaled model in a tow tank [3].

The wing forces and moments in the equations of motion are dependent on the set point of the wings. The inputs that affect the fly in the longitudinal model are the wing beat frequency f and mean stroke angle ϕ. To design a controller for this system we need an input-output map that describes how changes in (f, ϕ) affect the states. To generate this map the GUF simulation was run by fixing the wing beat frequency and mean stroke position to various values and recording the forces and moments produced by the fly's wings. The simulation model calculates the forces and moments hundreds of times during each wing beat, but for the purposes of designing a controller we took the average of these across each wing stroke. It was assumed the produced forces and moments were dependent on the forward and vertical velocities and the inputs (mean stroke position and wing beat frequency). The dependence was assumed to be up to third order and a least squares fit of the simulated data was used to find the coefficients. Expressed in global frame coordinates we have

$$\begin{bmatrix} F_{w,x} \\ F_{w,z} \\ M_{w,y} \end{bmatrix} = C_{gb}(\theta) A(v_{x_b}, v_{z_b}, f, \phi) b \tag{8}$$

where b represent the coefficients that were fit by least squares and A is the matrix whose elements are third order polynomials of the velocities and input controls.

B. Linearization About an Operating Point

To clarify, our model has the states $\mathbf{X}$ and inputs $\mathbf{U}$

$$\mathbf{X} = \begin{bmatrix} \dot{x} & \dot{z} & \theta & \dot{\theta} \end{bmatrix}^T \quad \mathbf{U} = \begin{bmatrix} \phi & f \end{bmatrix}^T \ . \tag{9}$$

We seek to design controllers that stabilize longitudinal flight at fixed speeds along the x-axis only. That is we seek an operating point of the form

$$(\mathbf{X}^*, \mathbf{U}^*) = \left(\begin{bmatrix} \dot{x}_{\text{des}} \\ 0 \\ \theta^* \\ 0 \end{bmatrix}, \begin{bmatrix} \phi^* \\ f^* \end{bmatrix} \right) \ , \tag{10}$$

which is defined by $\dot{x}_{\text{des}}$. A forward velocity in the world frame is specified and from that the other states and inputs are determined. The dynamics can then be linearized about this operating point and a feedback controller can be designed for the linearized dynamics. The process of finding an operating point and linearizing about it for the stroke-averaged model was done numerically using MATLAB's "findop" and "linmod2" commands.

The parameter values used in the design and simulation are shown in Table I.

TABLE I
PHYSICAL PARAMETER VALUES.

Parameter	Value
m	0.9315 mg
I	0.4077 mg $\cdot$ mm^2
β	25 deg
S	1.8 mm^2
L	2.5 mm
C_N	1.3
C_P	0.6
ρ	0.0013 mg / mm^3
$[a_1, a_2, a_3, a_4]$	$[0.0167, 0.0878, -0.005, -0.0063]$
$[b_1, b_2, b_3, b_4]$	$[-0.0494, -0.0136, -0.0028, 0.0027]$

C. Control Design

A convenient feature of the dynamics is that the linearized system takes a very similar form across a wide range of forward velocity operating points, enabling us to design a single controller at a representative operating point that works well throughout the flight envelope. Our chosen operating point was $\dot{x} = 0.25$ m/s, in the middle of the positive velocity operating range of 0 to 0.5 m/s. At this operating point, the linearization takes the form

$$\frac{d}{dt} \begin{bmatrix} \dot{x} \\ \dot{z} \\ \theta \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} -0.0076 & -0.0004 & 0.0102 & 0 \\ 0.0025 & -0.0115 & -0.0001 & 0 \\ 0 & 0 & 0 & 1 \\ -0.0055 & -0.0037 & 0.0003 & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{z} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0.0034 & 0.0135 \\ 0.0009 & 0.0773 \\ 0 & 0 \\ 0.0320 & 0.0242 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{11}$$

where we abused notation slightly to let the linearized states represent the deviations from the states of the operating point and $[u_1, u_2]$ the deviations from the control inputs of the operating point. Our control objective is to regulate about a desired forward velocity. Two key features of Equation 11 inform this design. The first is that the $\dot{x}$ dynamics are much more strongly coupled to θ than to $\dot{z}$, and so $\dot{z}$ can be safely ignored in the control design. The second is that the two control inputs u_1 and u_2 affect $\dot{x}$ and $\dot{\theta}$ differently, so we can choose a control vector v along which we can control $\dot{\theta}$ (and hence θ) without directly affecting $\dot{x}$. We do this by finding the null space of the first row of the control matrix, setting

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} 0.9691 \\ -0.2467 \end{bmatrix} v.$$

We can now develop an inner-loop controller to regulate the single input-single output transfer function from v to θ_l. We will then design an outer loop controller to regulate $\dot{x}$ using the pseudo-control θ, taking advantage of the strong coupling between θ and $\dot{x}$. For the moment we will set aside the control of the altitude rate $\dot{z}$.

1) Pitch Control: At the design operating point of $\dot{x} = 0.25$ m/s, the transfer function from v to θ is

$$H_{v\theta} = \frac{0.025(s+0.0142)(s+0.0075)}{(s+0.0425)(s+0.0132)(s^2-0.0366s+0.0013)}.$$

This transfer function has two poles and zeros on the negative real axis and a pair of unstable complex poles. Our controller must pull the unstable poles to the left-half plane, which is not possible with proportional feedback only. A simple PD controller with a zero at $s = -0.01$ has this desired effect. However, as we are working with a stroke-averaged model of the fly dynamics, we also want to ensure that the control signals remain well inside the 200 Hz wingbeat frequency to avoid producing controls that have no meaning within the context of the full GUF model. As our units of time are milliseconds, 200 Hz corresponds to $\omega = 1.26$ rad/ms. Good performance was achievable with a rolloff frequency of 0.2 rad/ms, comfortably inside the wingbeat frequency. Choosing the gain to maximize phase margin we arrive at the pitch controller

$$C_\theta(s) = 1.2\frac{s+0.01}{s+0.2}, \tag{12}$$

which is a standard lead compensator.

2) Forward Velocity Control: We control the forward velocity by altering the desired pitch angle. A simple integral controller of the form

$$C_{\dot{x}}(s) = \frac{0.01}{s}$$

so

$$\delta\theta(s) = C_{\dot{x}}(s)\dot{x}(s) \tag{13}$$

provides adequate, stable performance and good tracking of velocity set points away from the design point.

3) Altitude Rate Control: Up until this point we have neglected the altitude rate portion of the dynamics. This is a reasonable approach because the altitude rate does not couple strongly into the forward velocity dynamics. However, if we wish to fly for long periods of time without significant excursions in altitude we need to provide some control of this state. Because $\dot{z}$ depends much more strongly on the wingbeat frequency u_2 than on u_1, we simply increase the frequency command by an amount proportional to the error in $\dot{z}$, or

$$\delta f = -\dot{z}. \tag{14}$$

This controller drastically reduces excursions in z. For better performance an integrator could be included here as well, but for the purposes of our simulations performance is quite adequate.

4) Feedforward Terms: So far we have only found the *change* in inputs needed to control the insect. We need to add this to a feedforward term that makes the desired forward velocity an equilibrium point. This feedforward term can be computed from a polynomial fit to the linearization of the longitudinal model at various operating points. These fits are as follows:

$$\begin{aligned} \phi^* &= 0.1915\dot{x}_{des} - 0.1254 \\ f^* &= -0.0534\dot{x}_{des}^2 - 0.0054\dot{x}_{des} + 0.2209, \end{aligned} \tag{15}$$

where $\dot{x}_{des}$ is the desired forward velocity.

The forward velocity controller only outputs the change in pitch angle desired to adjust the forward velocity. We also must calculate a feedforward pitch angle from a polynomial fit to the operating point of the longitudinal model as

$$\theta^* = 0.7115\dot{x}_{des} + 0.0391. \tag{16}$$

5) Estimation: We have thus far used three state variables as the inputs to our controllers: $\dot{x}$, $\dot{z}$, and θ. Of these, $\dot{x}$ and $\dot{z}$ can be estimated from optical flow. The pitch angle θ, however, is more likely to be estimated by the fly from the pitch rate $\dot{\theta}$ as sensed by the halteres. For this initial implementation we simply use the integral of $\dot{\theta}$ to estimate θ. This provides performance similar to pure state feedback, but in the future a true estimator could be used to improve this behavior.

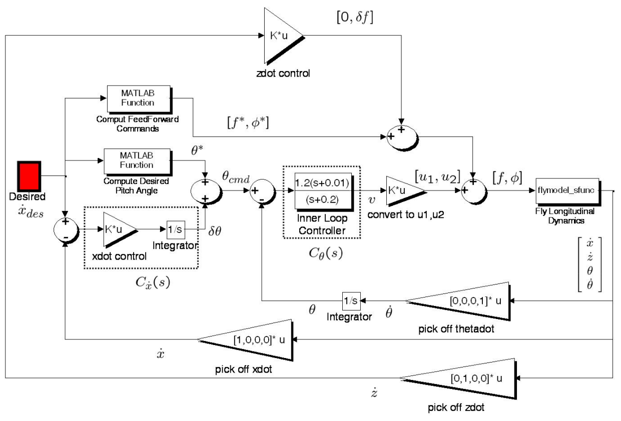

Fig. 3. Simulink diagram of control implementation. The S-function block is a wrapper around the stroke-averaged nonlinear fly model. The MATLAB function blocks compute the feedforward terms from the polynomial fits based on the stroke-averaged model. The innermost loop regulates pitch angle, the next loop regulates forward velocity using desired pitch, and the outermost loop adjusts the wingbeat frequency to regulate altitude rate.

IV. Sensory Model and Visual Processing

The sensory systems provide estimates $\hat{\dot{x}}$, $\hat{\dot{z}}$, $\hat{\theta}$, $\hat{\dot{\theta}}$ of the fly's state variables $\dot{x}$, $\dot{z}$, θ, $\dot{\theta}$. To simplify our analysis we restrict this discussion to known world geometries and sinusoidal visual patterns. We assume the head maintains a constant attitude regardless of body pitch, which is plausible as the fly has the ability to pitch its head. We expect the foundations of this work to be able to generalize to more complex environment geometries.

A. *Pitch Rate Estimation*

An estimate of the fly's pitch rate is available from the fly's halteres, small vestigial wings that beat in anti-phase with the wings and function as rate gyroscopes. In this work we use the actual pitch rate as the estimate, but future work may incorporate a more elaborate model of the halteres, as described in [3].

B. *Velocity Estimation by Retinal Velocity Pattern*

Retinal velocity is a geometric quantity that measures the rate of deformation of scenery projected onto the retina as it passes by. Electrophysiology performed on the blowfly has found a class of cells, the lobula plate tangential cells (LPTC's), that are stimulated by specific velocity patterns over large fields of the retina. Certain of these cells appear to be activated by flow patterns corresponding to rolling or linear translation.[8] This suggests that flies use wide-field velocity field patterns to aid in flight navigation. In this work we estimate the state variables $\dot{x}$ and $\dot{z}$ by observing the retinal velocity flow field of the fly during flight.

1) Visual Rendering in GUF model: Our model consists of a single eye with 642 facets arranged in a sphere, each of which performs a Gaussian visual blur to emulate the optics of flies.[3] This number is comparable to the approximately 1400 visual sensors of the fruit fly's eyes. Real flies utilize two eyes, but there is little stereo overlap so a single spherical eye suffices for the model. The visual field is rendered in OpenGL using the Open Scene Graph library as six scenes assembled into a cube map and projected onto the visual sphere (Figure 4).

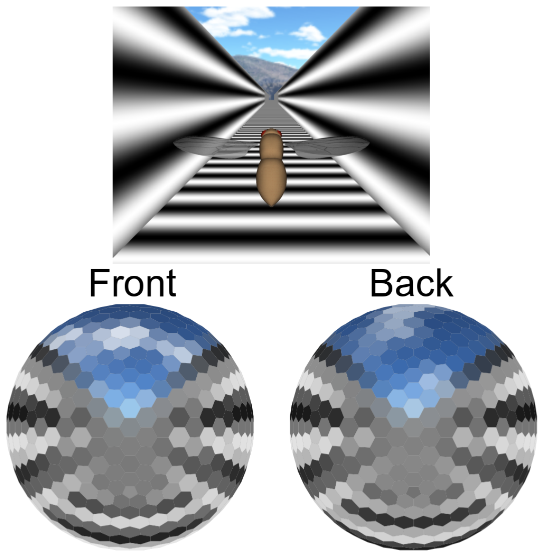

Fig. 4. Image of a visual environment encountered by the fly (top) and a rendering of the world as seen through the receptors of the fly model (bottom). Sinusoidal grating stripes on the floor enable forward velocity estimates, and horizontal stripes on the wall enable vertical velocity estimates.

2) Elementary Motion Detectors: The retinal velocity at each point in the visual field is estimated using Hassenstein-Reichardt Elementary Motion Detectors (HR-EMD's) between pairs of visual sensor elements. Behavioral studies have shown evidence that insects use HR-EMD's to estimate retinal velocity.[9] The differenced delay-and-correlate operation performed by the HR-EMD gives a constant response to a zero-mean sinusoid signal moving by the pair of sensors (Figure 5). Each EMD operates between pairs of nearest-neighbor facets, and each facet has six (occasionally five) neighbors, giving approximately 3-fold times as many EMD's as sensors, or 1920. The time constant of the low-pass filter in the HR-EMD in used this work is 35ms.

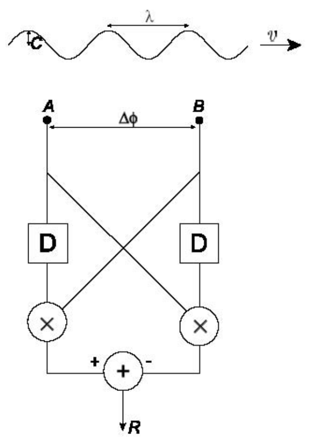

Fig. 5. The Hassenstein-Reichardt Elementary Motion Detector (HR-EMD). The luminance reading from sensor A is correlated with (multiplied by) a time-lagged (low-pass filtered) signal from sensor B and vice versa, then the two correlations differenced to generate an estimate of retinal velocity. C is the contrast, λ is the wavelength in radians, and v is the velocity of the sinusoidal luminance signal, and $\Delta\phi$ is the angle between the pair of sensors. Image credit: [4]

The HR-EMD suffers from a number of non-idealities that make its estimates differ from true retinal velocity. These include aliasing at high spatial frequencies and aliasing at high image velocity. The former is eliminated by Gaussian blurring the image sensors, eliminating high-frequency spatial components. Note that distant textures become a uniform gray in the faceted fly-eye view of Figure 4, a consequence of blurring. Velocity aliasing is avoided by moving fly slowly enough that the EMD's are operating in the low-speed monotonic regime, ensuring the function can be inverted to recover velocity. Sinusoidal gratings (stripes) on the floor and walls enable velocity estimates by HR-EMD. Refinements by [4] and others to the HR-EMD to make velocity estimates from more realistic images are planned for future work.

3) Matched Filters: Velocity estimates are made by correlating wide-field motion sensitivity patterns with retinal velocity estimates made by the EMD's. The correlation operation is analogous to the arrangement synaptic weights stimulating LPTC cells that constitute the motion sensitivity patterns.

A given motion sensitivity pattern is known as a filter. One approach to deriving filters might be to use spherical sinusoidal harmonics. But in this work, we derive filters based on EMD-estimated retinal velocity patterns experienced during stereotyped motions. Such filters, known as “matched filters”, may be a reasonable approximation to biological processes, which often derive patterns based on experience rather than theoretical constructs.

Two filters were generated. The $\dot{x}$ filter, $\mathbf{m}_{\dot{x}}$, was the time-averaged EMD response vector ϵ as the fly was moved over a floor of infinite extent covered by stripes. Similarly, the $\dot{z}$ filter, $\mathbf{m}_{\dot{z}}$, recorded the response to vertical motion between parallel striped vertical walls. Each is a 1920×1 vector. Figure 6 shows the matched filter $\mathbf{m}_{\dot{x}}$ generated for forward velocity. While the stereotyped environments used to derive the matched filters differ from the environment they were used in (a corridor), they yielded nearly correct estimates. The reason is that because of the sensors’ spatial blurring, the filter’s most responsive portion of the visual field was within the walls of the corridor.

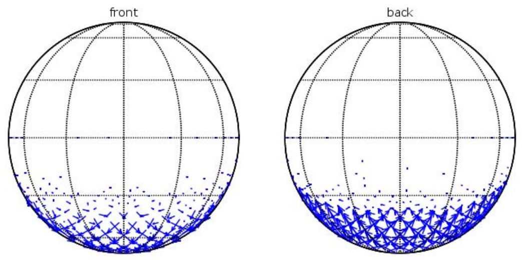

Fig. 6. An image of the matched filter $\mathbf{m}_{\dot{x}}$ rendered on the front (left) and rear (right) visual hemispheres. Each arrow represents the magnitude of sensitivity between each pair of visual receptors, drawn along the line connecting the two sensors. The apparent difference in “sensitivities” between the front and rear is only a consequence the arrowheads pointing generally upward versus downward.

4) Estimating Velocity: The filter outputs are

$$\begin{aligned} f_{\dot{x}}(\epsilon) &= \mathbf{m}_{\dot{x}} \bullet \epsilon \qquad (17) \\ f_{\dot{z}}(\epsilon) &= \mathbf{m}_{\dot{z}} \bullet \epsilon \end{aligned}$$

where ϵ is the current EMD response vector and $\mathbf{m}_{\dot{x}}$ and $\mathbf{m}_{\dot{z}}$ are the matched filters for forward and vertical velocity, respectively. The operator $\bullet$ denotes the vector dot product.

The EMD response function of velocity in the monotonic region can be approximated by the sinusoid

$$f(\epsilon) = a[\cos(bv + \pi) + 1] = g(v) \qquad (18)$$

where v is the velocity $\dot{x}$ or $\dot{z}$ and a and b are scalars adjusted to maximize fit to true velocities. This function was inverted to find the velocity estimate $\hat{v} = g^{-1}(f(\epsilon))$ for $f(\epsilon)$ within the domain of g^{-1}.

Figure 7 shows a comparison of the velocity estimates by EMD matched filters to truth values for forward and vertical velocities. Velocity estimates approximate true velocity near the operating point of $\dot{x} = 0.25$ m/s and $\dot{z} = 0$ m/s and diverge near saturation. Different forward velocities had no effect on estimates of vertical velocity.

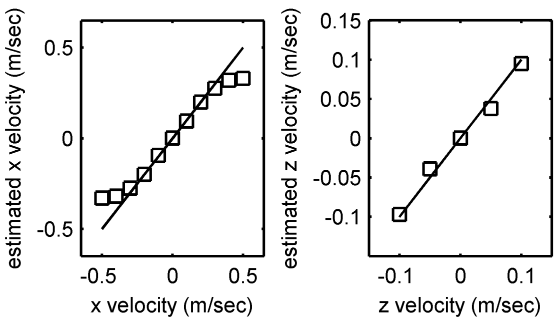

Fig. 7. Velocity estimates by EMD matched filters comparison to truth values. Estimate data points appear as boxes. The forward velocity estimate saturates as the velocity approaches the saturation velocity for the EMD’s. Because the operating point for vertical velocity was zero, lower velocities were tested and no saturation is apparent.

V. Simulations in GUF

We implemented the longitudinal controller with visual estimation scheme described above on GUF. This is a good test to see if our simplified controller for the longitudinal dynamics and stroke averaged model will work on the full articulated rigid body with the full wing dynamics (not the stroke averaged model) using the translational velocity estimates from the visual system. In addition, out of plane motion was constrained to be consistent with our focus on in plane motion.

The simulation was run with a step command in forward velocity from 0 to 0.25 m/s commanded at 1000 ms with a small initial error. The results are shown in Figure 8. As can be seen the performance is quite good. The forward velocity as estimated by the visual system converges to the commanded velocity (the true velocity is slightly offset). The vertical velocity deviation is kept very small. The high frequency content in the signals is due to the wing beats. Figure 9 shows the values the controller set for the inputs (f, ϕ). Plotted are both the feedforward component, based only on the desired forward velocity, and the feedback component. The nonzero feedback component of the input can be due to initial error (both at the start and when the desired set point is changed), using the longitudinal model to compute the feedforward commands as well as the offset in the visual estimates compared to the true velocity values.

VI. Conclusions and future work

We have demonstrated the feasibility of using estimates of the fly’s states based on biologically available sensor information to stabilize forward flight using biologically available actuation signals. A great deal of both short- and long-term work remains to fully explore even simple forward flight within the GUF model. A similar approach can be followed to design controllers for the lateral dynamics so we can stabilize the full 6 degree-of-freedom GUF model.

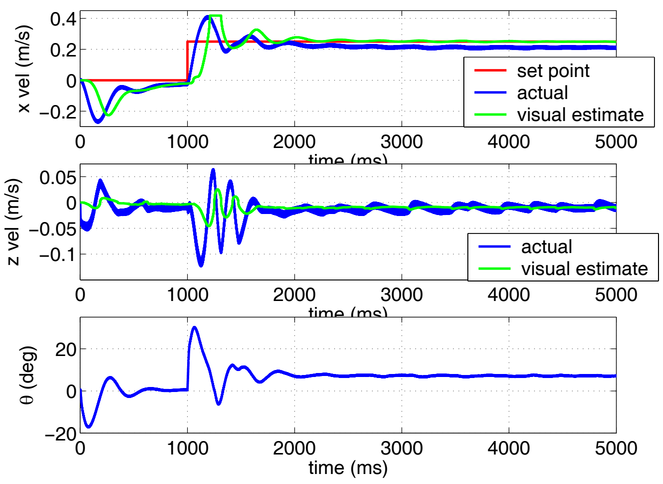

Fig. 8. Simulation response in GUF using the longitudinal controller with translational velocity estimates from the visual system and rotational velocity estimates from the halteres. The measured states are in blue, with the velocity estimates in green. The commanded forward velocity is in red.

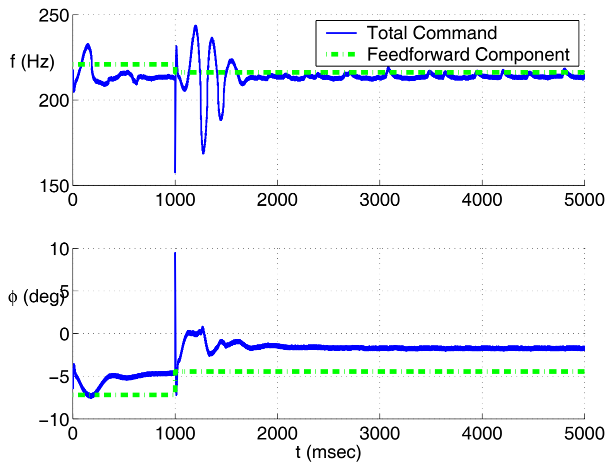

Fig. 9. Simulation control values in GUF. The applied control values are in blue, with the feedforward component in green.

The work on visual estimation used in the simulation estimated forward velocity in a simplified visual environment. As environments become more complex, for example by including less-regular textures or open environments, alternate estimation strategies may be required.

Once we have developed a stable flight control system within the GUF environment, many longer-term investigations will be possible. The behavior of this simple control system in more complex environments and under various external conditions will elucidate where this model is adequate and where it needs to be augmented to explain real fly behavior. It is our suspicion that some seemingly high-level behaviors such as the slowdown on landing approach may emerge from the basic properties of the sensory-motor system. The existence of a baseline stable, closed-loop model will enable investigation of numerous real-world behaviors.

It is also important to note that although there is no reason to think flies compartmentalize tasks in the way we did, this exercise has been useful to generate a proposed control scheme and visual estimator block which are independent and relatively simple. This allows analysis using conventional tools which would be significantly more difficult on a more coupled model. Nevertheless, interpreting the biological relevance of our findings using such an arbitrary division of the estimation-and-control problem into smaller sub-problems must be done with caution. For example, although altitude-coupling with visual estimates of forward velocity is an issue that animals have presumably confronted, it is entirely possible this feature is exploited to slow down while flying lower rather than 'overcome' to produce more accurate state estimates. Using these results to gain understanding in biology and vice-versa will continue.

References

[1] Xinyan Deng, Luca Schenato, and S. Shankar Sastry. Flapping flight for biomimetic robotic insects: Part ii - flight control design. *IEEE Transactions on Robotics*, 22(4):789–803, 2006.

[2] Xinyan Deng, Luca Schenato, Wei Chung Wu, and S. Shankar Sastry. Flapping flight for biomimetic robotic insects: Part i - system modeling. *IEEE Transactions on Robotics*, 22(4):776–788, 2006.

[3] William B. Dickson, Andrew D. Straw, Christian Poelma, and Michael H. Dickinson. An integrative model of insect flight control. In *AIAA*, 2006.

[4] Ron O. Dror, David C. O'Carroll, and Simon B. Laughlin. Accuracy of velocity estimation by reichardt correlators. *Journal of the Optical Society of America A*, 18(2):241–252, February 2001.

[5] R. Fearing, K. Chiang, M. Dickinson, D. Pick, M. Sitti, and J. Yan. Wing transmission for a micromechanical flying insect. In *IEEE Int. Conf. Robot. Autom.*, 2000.

[6] J. Sean Humbert, Richard M. Murray, and Michael H. Dickinson. A control-oriented analysis of bio-inspired visuomotor convergence. 2005.

[7] J. Sean Humbert, Richard M. Murray, and Michael H. Dickinson. Sensorimotor convergence in visual navigation and flight control systems. 2005.

[8] Holger G. Krapp and Roland Hengstenberg. Estimation of self-motion by optic flow processing in single visual interneurons. *Nature*, 384:463–466, 1996.

[9] Werner Reichardt. *Sensory Communication: Contributions to the Symposium on Principles of Sensory Communication*, chapter 17 "Autocorrelation, a principle for the Evaluation of Sensory Information by the Central Nervous System", pages 303–317. MIT Press, 1959.