ECE 4960 Fast Robots Seminar Course,
Instructor: Kirstin H. Petersen
Electrical and Computer Engineering
Cornell University
December 8, 2020

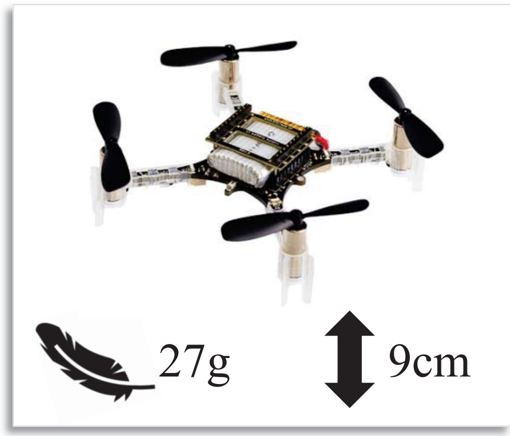
Fast Event-based Sensorimotor Planning and Control

PIs: Silvia Ferrari♣ and Robert Wood♠

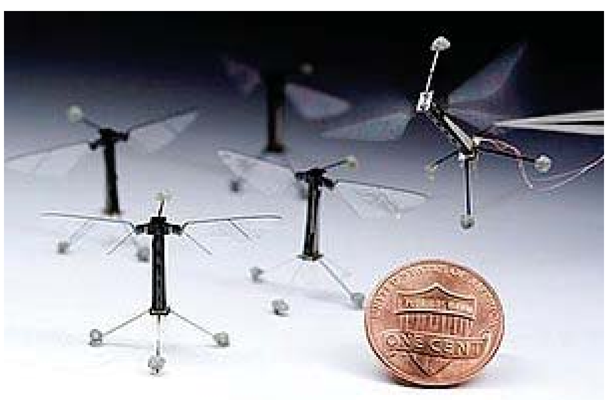
♣John Brancaccio Professor of Mechanical and Aerospace Engineering, Cornell
♠ Charles River Professor of Engineering and Applied Sciences, Harvard

Motivation: Insect-Scale Autonomous Flight

- Safer: weigh less than 1 pound and thus are safe to operate near humans
- Smaller and covert: can access narrow or unfriendly spaces inaccessible to other vehicles
- Autonomous flight expands the capability of a single operator to monitor previously inaccessible spaces
  - More effective search and rescue
  - Surveillance in complex environments
  - Security in densely populated, sensitive regions



Crazyflie 2.0 (https://www.bitcraze.io/crazyflie-2/)



RoboBee [Ma, 2013]

Challenges: Insect-scale Sensorimotor Control

- Size, weight and power constraints
  - RoboBee power budget: ~21mW
  - Only ~2mW available for sensing and control
- Fast dynamics
  - Dominant timescales on the order of a few hundred milliseconds
- Physical parameter variations
  - Small wing asymmetries result in undesired torque during flight
- Highly susceptible to external disturbances such as wind gusts

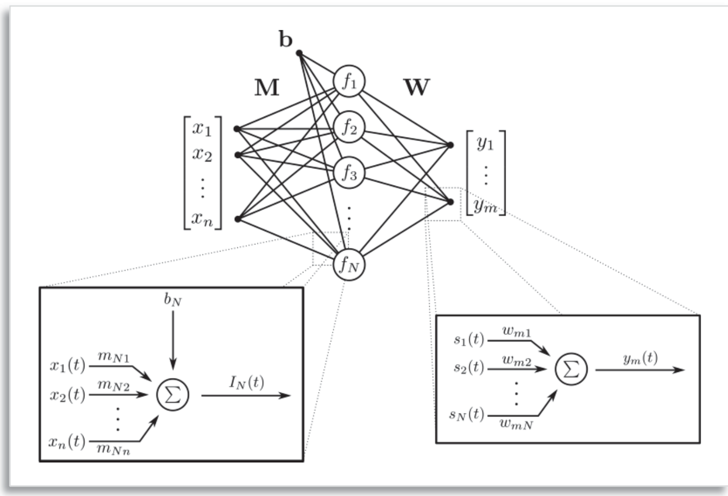
Open Loop Flight

Funded by the ONR Grant # N00014-17-1-2614

Neuromorphic Sensing and Control

Emerging Technologies

- Neuromorphic sensing and control algorithms for intelligent, energy-efficient autonomy



inivation (https://inivation.com/)

1

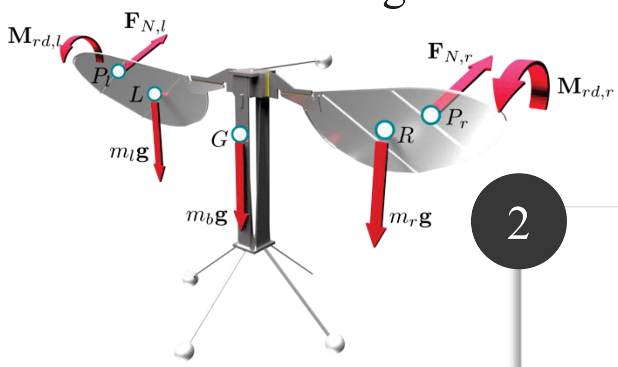
Spiking neural networks (SNNs), or neuromorphic chips, can learn online to improve performance or adapt to new conditions

2

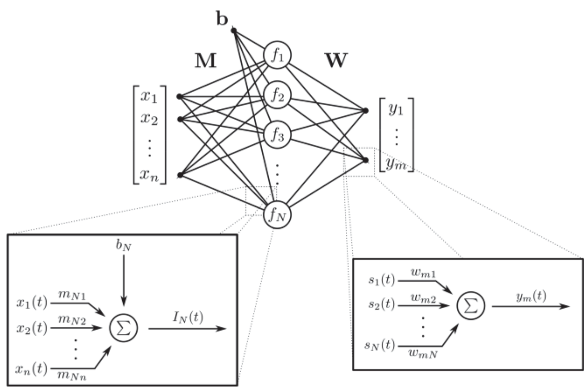
Neuromorphic cameras have 1µs temporal resolution and require at most a few milliwatts of power

Research Goals

1 Modeling



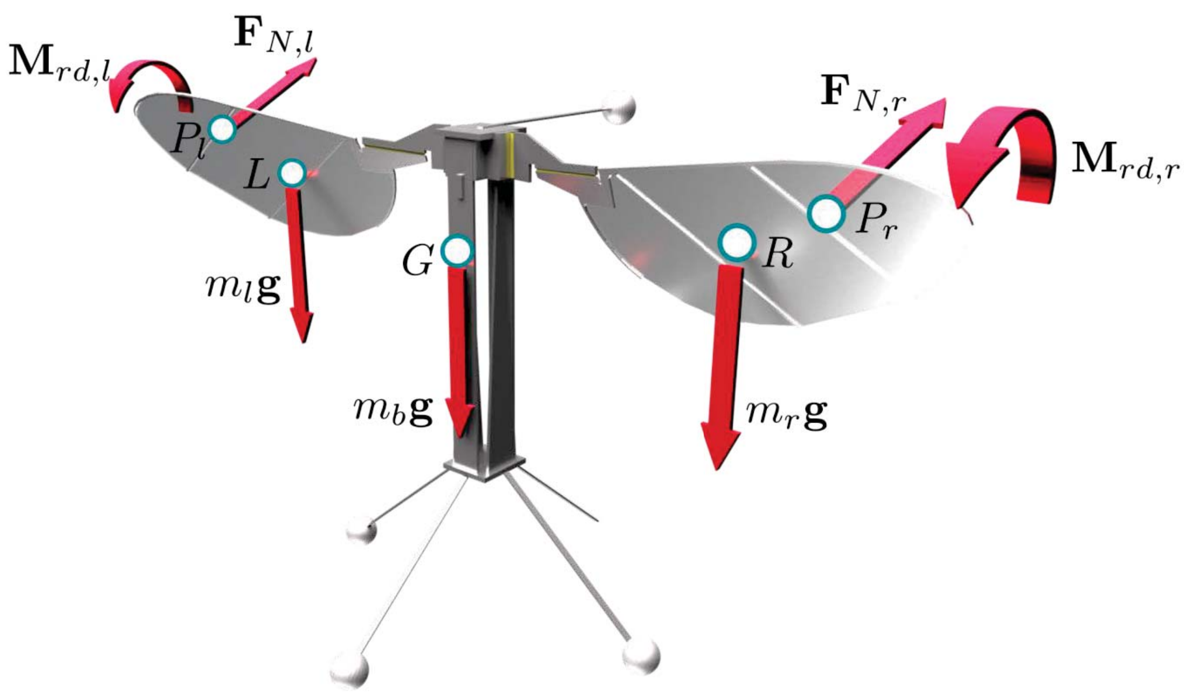
2 Adaptive Flight Control



3 Exteroceptive Sensing

1. Model the RoboBee flight dynamics, validate with experimental data
2. Develop adaptive flight controllers which account for physical variations
3. Develop sensing algorithms to perform target tracking and obstacle avoidance

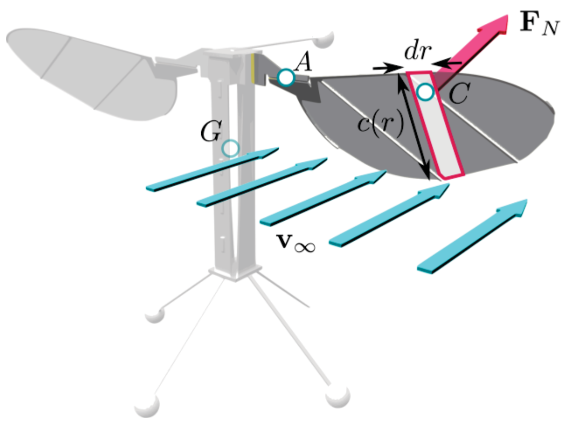
RoboBee Modeling



T.S. Clawson, S. Ferrari, E.F. Helbling, R.J. Wood, B. Fu, A. Ruina, and Z.J. Wang, "Full Flight Envelope and Trim Map of Flapping-Wing Micro Aerial Vehicles, " *AIAA JGCD,* Vol. 43, No. 12 (2020), pp. 2218-2236.

Modeling Flapping Wing Flight

- Aerodynamic forces in flapping flight differ from classic airfoil models
- Modeling aerodynamic effects on flapping wings
  - Computationally expensive CFD models [Liu, ’98], [Sun, ’02]
  - Simplified models can accurately predict stroke-averaged forces [Whitney, ’10], [Dickinson, ’99], [Wang, ’04]



- Modeling flight dynamics of the insect or robot body
  - Simple 2D models [Ristroph ’13]
  - Stroke-averaged models [Chirarattananon, ’16]
  - Kinematically-constrained wing trajectories
    - Limited wing pitch [Oppenheimer, ’10]
    - Kinematic models from experimental data [Wang, ’16], [Dickson, ’08]
- Finding hovering set point and analyzing modes of motion and stability [Wu, ’12]

Wing Modeling

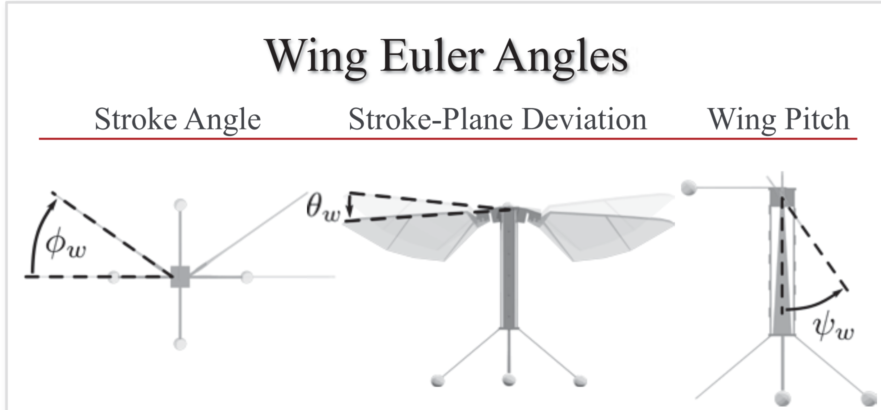
Assumptions:

- Rigid wings with passive pitching dynamics
- No stroke-plane deviation

$$\theta_w = 0$$

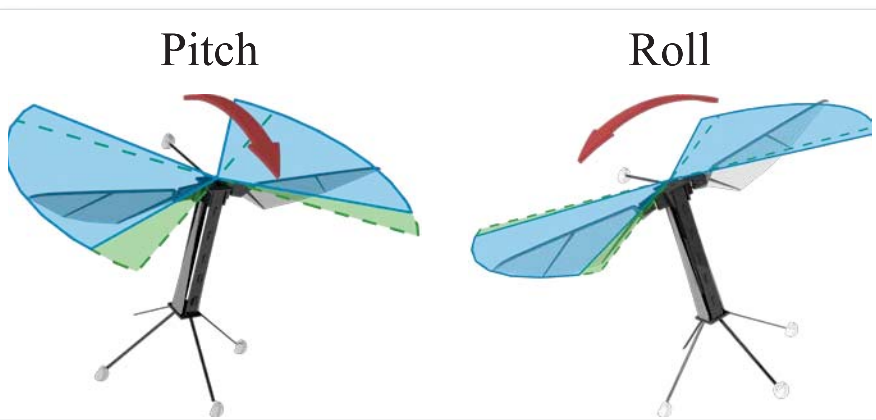
- Control inputs **u** affect stroke angle

$$\mathbf{u} = \begin{bmatrix} u_a & u_p & u_r \end{bmatrix}$$



- Stroke angle modeled by second order system

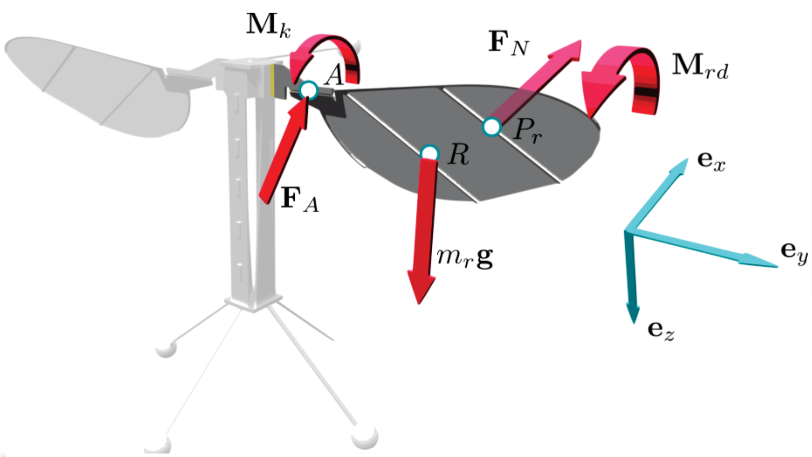
$$\ddot{\phi}_w(t) + 2\zeta\omega_n\dot{\phi}_w(t) + \omega_n^2\phi_w(t) = \frac{u_a \pm u_r}{2}\sin(\omega_f t) + u_p$$



ζ	Effective damping ratio	ω_f	Effective natural frequency
ω_n	Forcing frequency	u_a	Flapping amplitude input
u_p	Pitch input	u_r	Roll input

Passive Wing Pitch Dynamics

From angular momentum balance about wing hinge



$$\mathbf{e}_y \cdot \sum \mathbf{M}_A = \mathbf{e}_y \cdot \dot{\mathbf{H}}_A$$

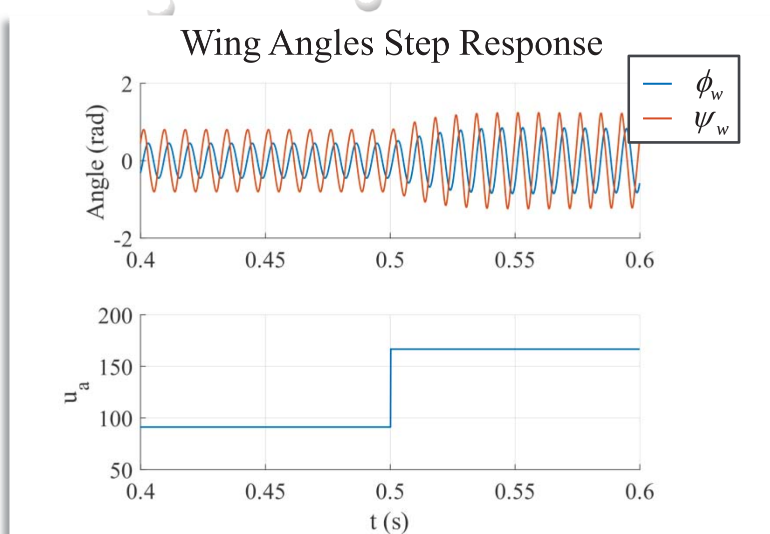
$$\sum \mathbf{M}_A = \mathbf{M}_{rd}(\alpha) + \mathbf{r}_{P_r/A} \times \mathbf{F}_N(\alpha) + \mathbf{r}_{R/G} \times m_r\mathbf{g} + \mathbf{M}_k$$

$$\dot{\mathbf{H}}_A = \mathbf{I}\dot{\boldsymbol{\omega}}_r + \boldsymbol{\omega}_r \times \mathbf{I}\boldsymbol{\omega}_r + \mathbf{r}_{R/A} \times m_r\mathbf{a}_R$$

- Center of pressure location:

$$\mathbf{r}_{P_r/A} = y_{CP}\mathbf{e}_y + z_{CP}(\alpha)\mathbf{e}_z$$

- Moment from spring: $\mathbf{M}_k = \kappa_h \psi_w$



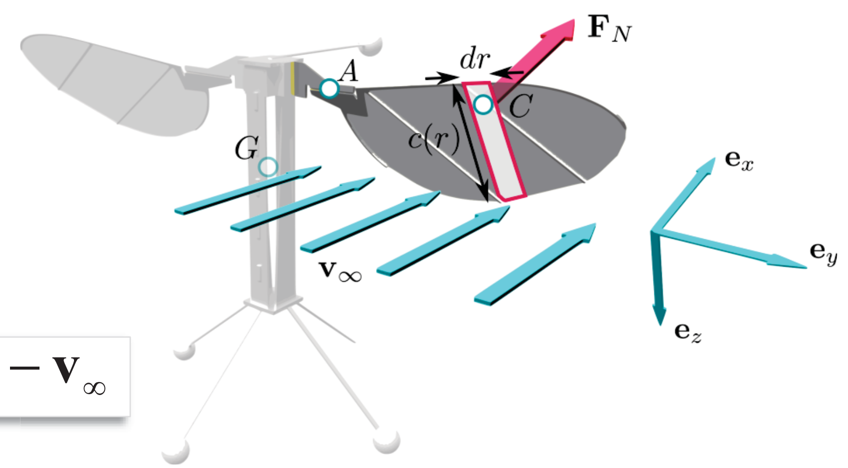
$\mathbf{M}_{rd}$	Rotational Damping	α	Angle of attack
$\mathbf{r}_{P_r/A}$	Position of right wing CP relative to hinge	$\mathbf{r}_{R/G}$	Position of right wing CG relative to body CG
m_r	Mass of right wing	$\mathbf{F}_N$	Aerodynamic normal force
ψ_w	Wing pitch	$\boldsymbol{\omega}_r$	Right wing angular rate
$\mathbf{I}$	Right wing inertia	$\dot{\mathbf{H}}_A$	Change in angular momentum about A
ϕ_w	Wing stroke angle	κ_h	Spring constant

[T. S. Clawson, S. B. Fuller, R. J. Wood, S. Ferrari "A Blade Element Approach to Modeling Aerodynamic Flight of an Insect-scale Robot," *American Control Conference (ACC)*, Seattle, WA, May 2017.]

Aerodynamic Forces and Moments

- Aerodynamic forces on wing caused by translational motion
- Locally, lift and drag are proportional to the square of the incident velocity $\mathbf{v}_C$

$$F_L(\alpha) = \frac{1}{2}\rho \int_0^R C_L(\alpha)\mathbf{v}_C^T \mathbf{v}_C c(r) dr$$



- Where

$$\mathbf{v}_C = \mathbf{v}_G + \boldsymbol{\omega}_b \times \mathbf{r}_{A/G} + \boldsymbol{\omega}_r \times \mathbf{r}_{C/A} - \mathbf{v}_\infty$$

$$C_L(\alpha) = C_{L_{max}} \sin(2\alpha)$$

- Rotational damping $\mathbf{M}_{rd}$ caused by span-wise rotation of wing

$$\mathbf{M}_{rd} = -\frac{1}{2}\rho C_{rd} \int_0^R \int_{z_0}^{z_1} (\boldsymbol{\omega}_y{}^2 z^2) \,|\, z \,|\, dz dr$$

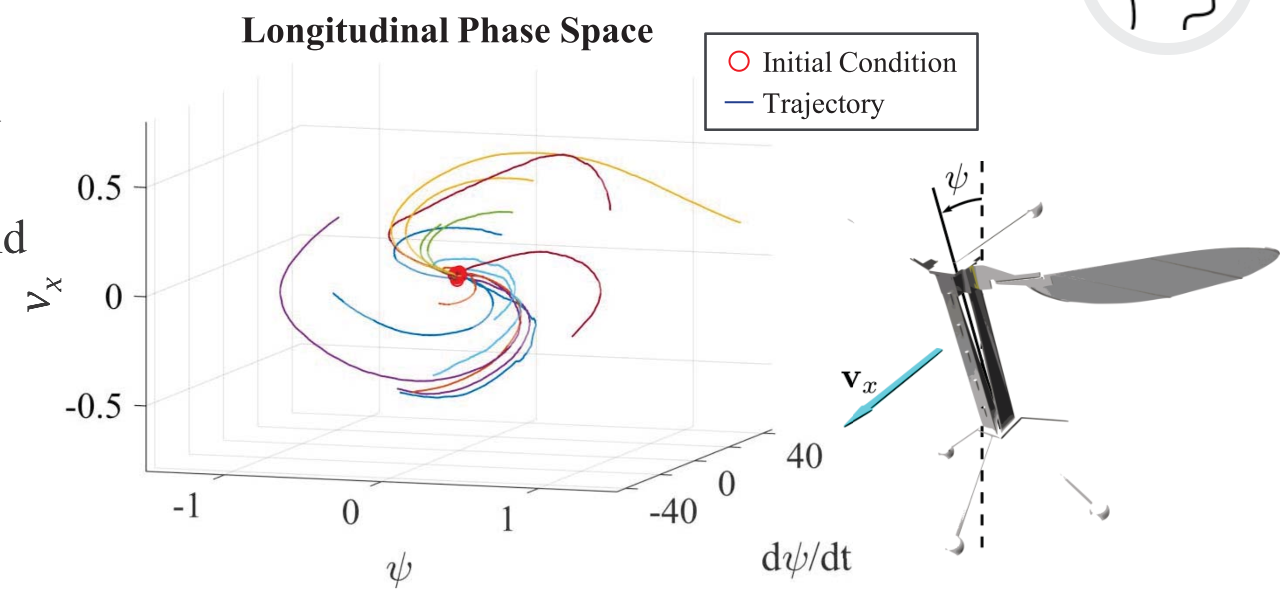
F_L	Lift force	α	Angle of attack
$\mathbf{r}_{A/G}$	Position of hinge relative to body CG	$\mathbf{r}_{C/A}$	Position of blade element relative to hinge
$\mathbf{v}_C$	Velocity of element	$\mathbf{F}_N$	Aerodynamic normal force
$\boldsymbol{\omega}_b$	Body angular rate	$\boldsymbol{\omega}_w$	Wing angular rate
$\mathbf{v}_\infty$	Free stream velocity	$c(r)$	Chord length
$\mathbf{M}_{rd}$	Rotational damping moment	C_L	Lift coefficient
C_{rd}	Rotational coefficient		

[T. S. Clawson, S. B. Fuller, R. J. Wood, S. Ferrari "A Blade Element Approach to Modeling Aerodynamic Flight of an Insect-scale Robot," *American Control Conference (ACC)*, Seattle, WA, May 2017.]

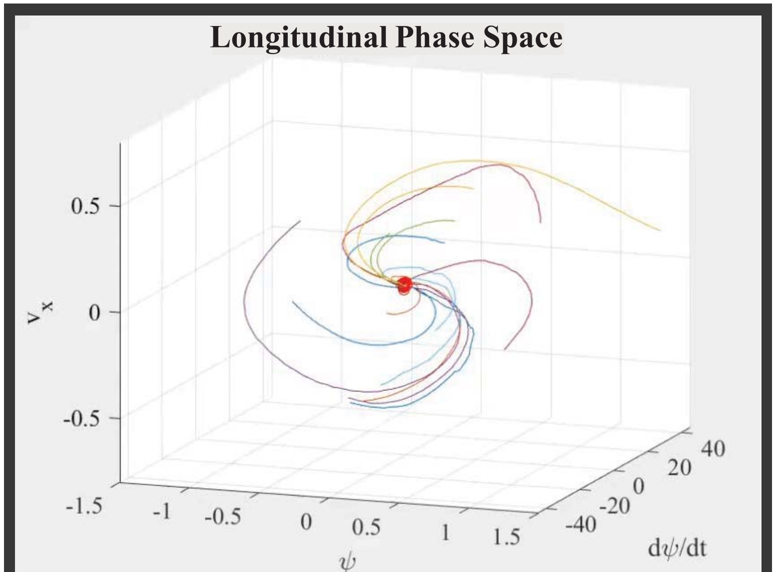
Model Validation: Challenges

Model Validation

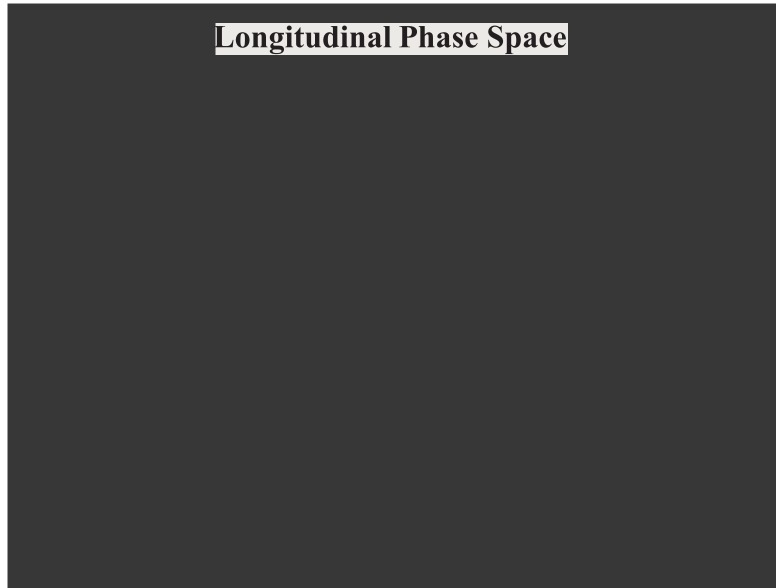
- Validate model with open loop flight tests
- Dominant longitudinal and lateral modes visible in experimental data
- Model predicts the same dominant modes



Longitudinal Instability



Experimental Trajectories



Simulated Trajectories

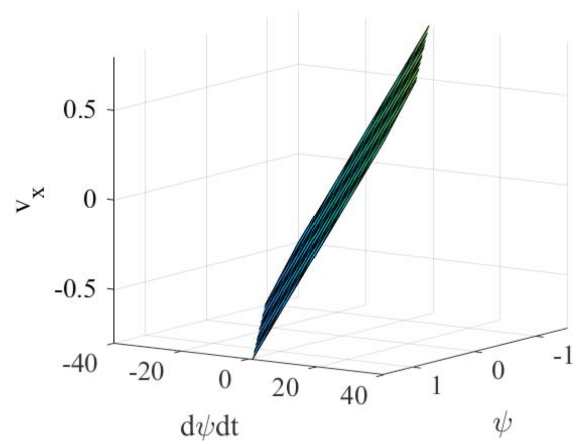
Period T and time constant τ for longitudinal mode:

$$T = 0.38\text{s} \approx 45 \text{ wing beats}$$

$$\tau = -0.24\text{s}$$

Trajectories in state space tend to lie on plane defined by dominant mode

Plane of Dominant Longitudinal Mode



Mode Subspaces

- Analyze modes of system $\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}, \mathbf{u})$ by linearizing about hovering set point $\mathbf{x}^*$, $\mathbf{u}^*$

Linear system: $\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t)$

Eigenvalues: $\lambda_i = \sigma_i \pm i\omega_i$
Eigenvectors: $\mathbf{v}_i = \mathbf{u}_i \pm i\mathbf{w}_i$

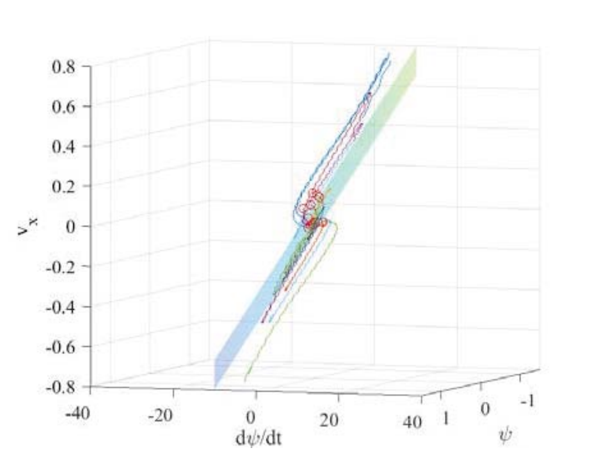
- Solution $\mathbf{x}(t) = \sum_{i=1}^{n} \mathbf{x}_i(t)$ of linear system is a summation of the modes $\mathbf{x}_i(t)$

$$\mathbf{x}_i(t) = \alpha_i e^{\lambda_i t}\mathbf{v}_i = \alpha_i e^{\sigma_i t}(\cos\omega_i t + i\sin\omega_i t)(\mathbf{u}_i \pm i\mathbf{w}_i)$$

- Imaginary component is zero – each mode must have a purely real solution

$$\mathbf{x}_i(t) = (\alpha_i e^{\sigma_i t}\cos\omega_i t)\mathbf{u}_i - (\alpha_i e^{\sigma_i t}\sin\omega_i t)\mathbf{w}_i$$

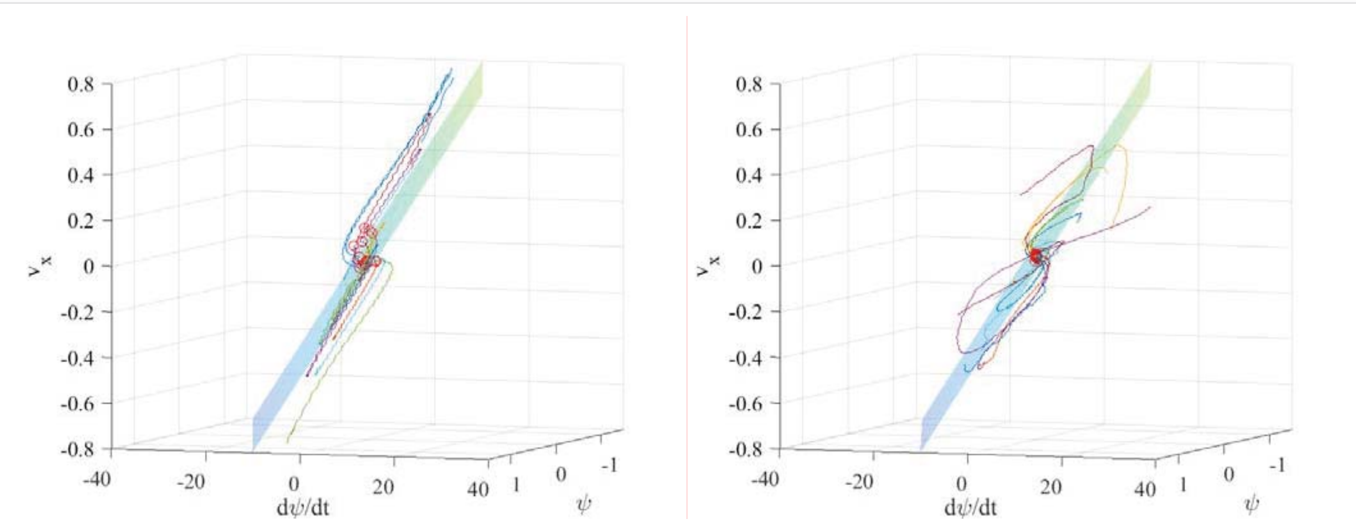
- The solution for a single mode shape $\mathbf{x}_i(t)$ is spanned by $\mathbf{u}_i$ and $\mathbf{w}_i$
  - $\mathbf{u}_i$ and $\mathbf{w}_i$ define a plane in 3D phase space



Longitudinal Instability

- Trajectories in state space tend to lie on plane of dominant longitudinal mode
- Longitudinal instability from dynamic model matches experimental data closely

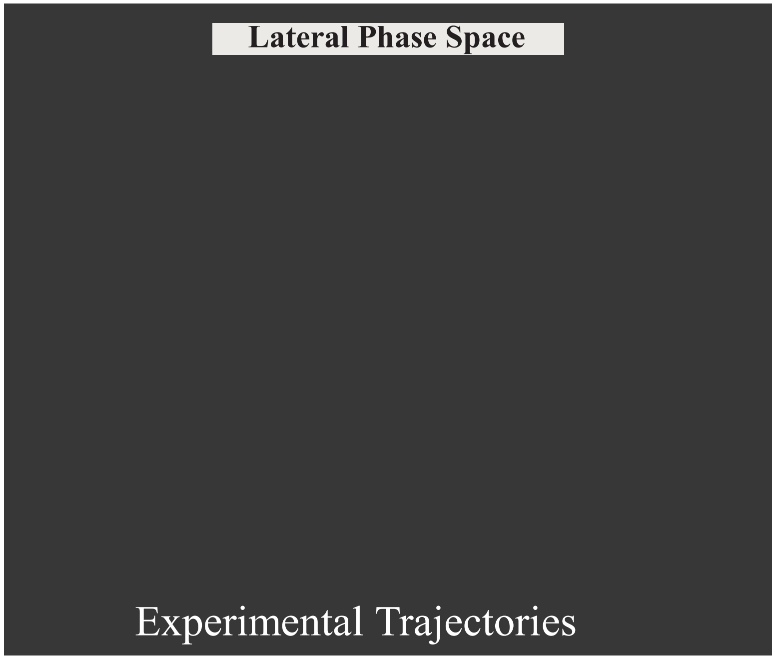
$$\mathbf{x} = \begin{bmatrix} \theta \\ \psi \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ v_x \\ v_y \\ v_z \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} v_2 \\ v_3 \\ v_7 \\ v_8 \\ v_9 \\ v_{10} \\ v_{11} \\ v_{12} \end{bmatrix} = \begin{bmatrix} 0 \\ -0.004 - 0.0145i \\ 0 \\ 0 \\ 0.218 - 0.126i \\ 0.008 + 0.0004i \\ 0 \\ 0 \end{bmatrix}$$

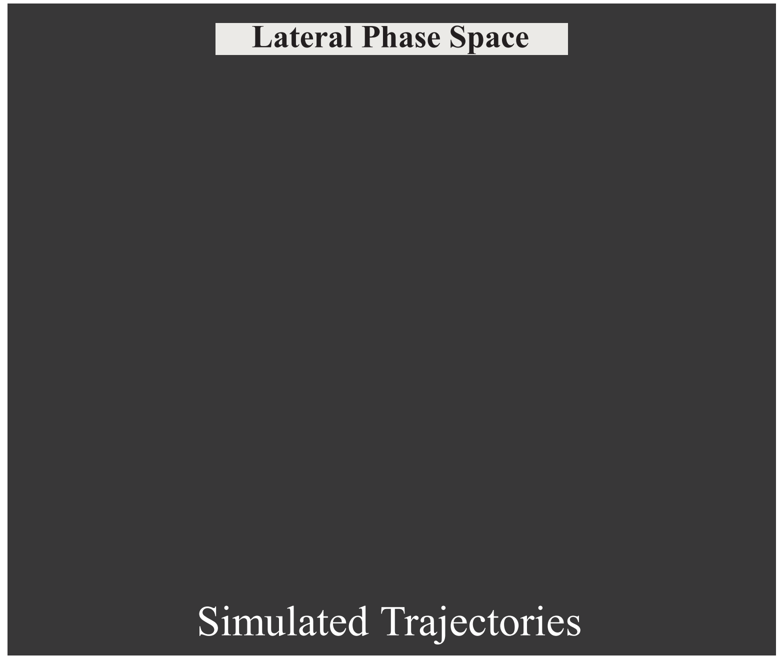


Data from Model

Experimental Data

Lateral Instability Comparison



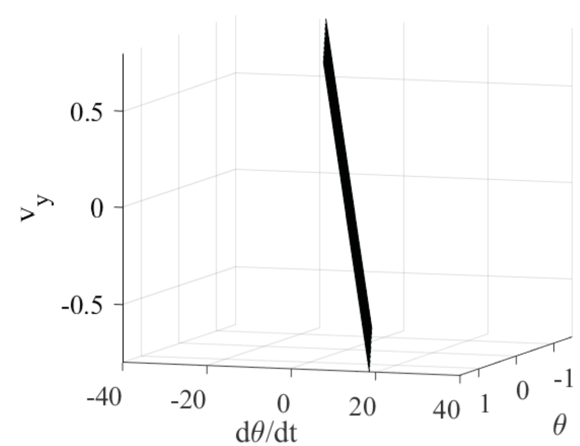


- Model can reproduce lateral instability observed in open loop flight experiments
- Period T and time constant τ of mode:

$$T = 0.83\text{s} \approx 100 \text{ wing beats}$$

$$\tau = -0.88\text{s}$$

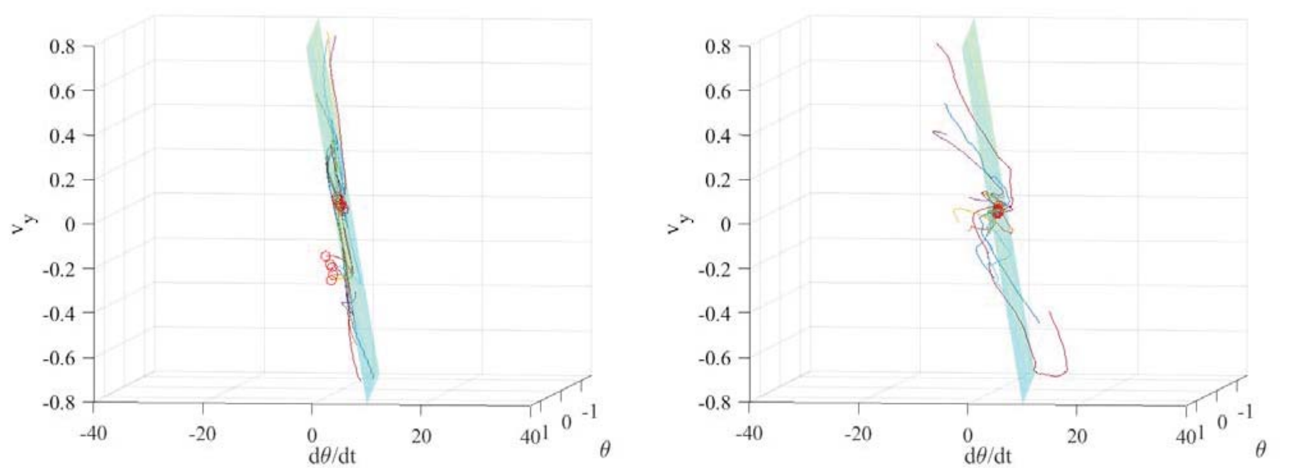
Plane of Dominant Lateral Mode



Lateral Instability

- Trajectories in lateral state space tend to lie on plane of dominant lateral mode
- Model lateral instability closely matches the experiments

$$\mathbf{x} = \begin{bmatrix} \theta \\ \psi \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ v_x \\ v_y \\ v_z \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \end{bmatrix} = \begin{bmatrix} 0.0004 - 0.041i \\ 0 \\ 0.445 + 0.106i \\ 0.357 - 0.036i \\ -0.002 + 0.001i \\ 0 \\ -0.051 - 0.013i \\ 0 \end{bmatrix}$$



Simulated Data

Experimental Data

Dominant Unstable Modes in Hovering

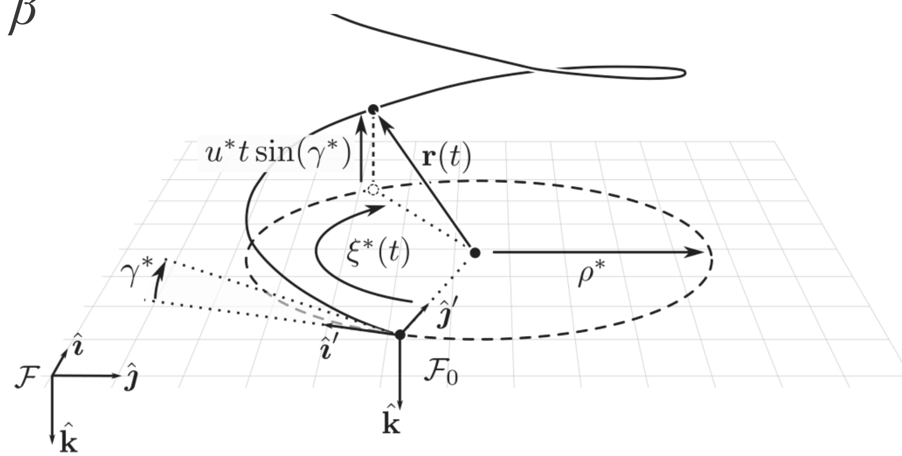
Steady Maneuvers and Flight Envelope

Steady Maneuvers

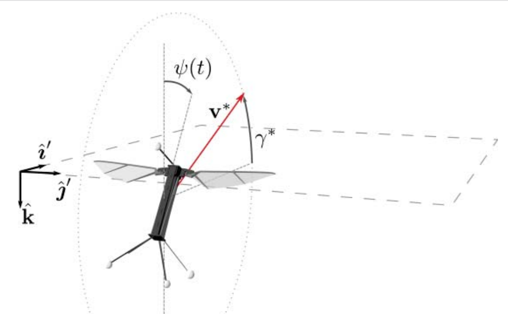
- *Steady* maneuvers are trajectories with minimum period equal to the flapping period T and constant control inputs
- Command input $\mathbf{y}^*$ defines maneuvers in terms of commanded speed u^*, climb angle γ^*, turn rate $\dot{\xi}^*$, and sideslip angle β^*

$$\mathbf{y}^* = \begin{bmatrix} u^* & \gamma^* & \dot{\xi}^* & \beta^* \end{bmatrix}$$

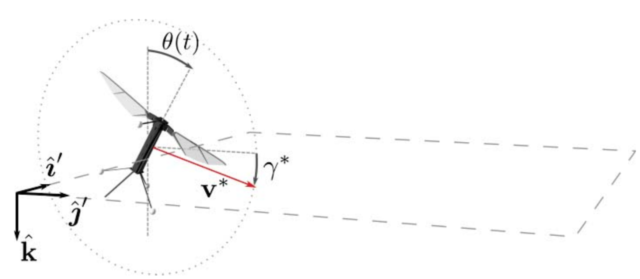
- The most general steady maneuver is the coordinated turn



- Other steady maneuvers include:



Longitudinal Flight



Lateral Flight

Coordinated Turn Constraints

Maneuver constraints derived by relating command input $\mathbf{y}^*$ to state $\mathbf{x}(t)$

After each period T:

- Wing state $\mathbf{x}_w(t)$ is constant:

$$\mathbf{x}_w(T) - \mathbf{x}_w(0) = 0$$

- Yaw advances by commanded turn angle:

$$\mathbf{\Theta}(T) - \mathbf{\Theta}(0) - \begin{bmatrix} T\dot{\xi}^* & 0 & 0 \end{bmatrix}^T = 0$$

- Position is on helical path given by $\mathbf{y}^*$

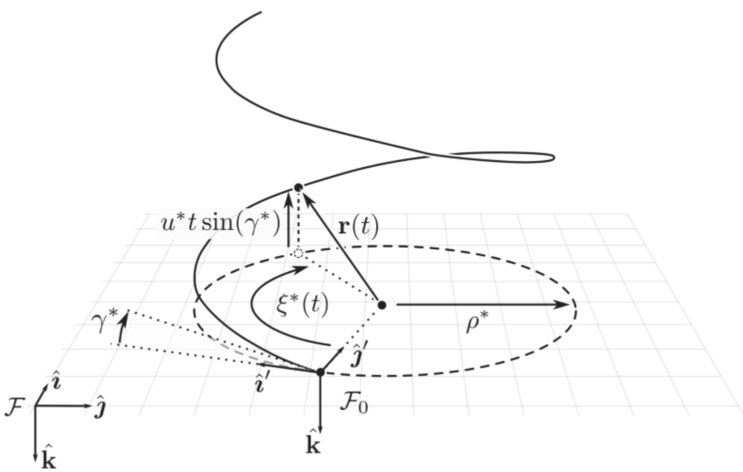
$$\mathbf{r}(T) - \mathbf{r}(0) + u^* T \sin(\gamma^*)\hat{\mathbf{k}} - \rho^* \cos(\gamma^*)(\hat{\jmath}' - \mathbf{R}^* \hat{\jmath}') = 0$$

- Body angular rate $\boldsymbol{\omega}$ and velocity $\mathbf{v}$ rotate by the commanded turn angle:

$$\mathbf{R}^* = \mathbf{R}^{\hat{\mathbf{k}}}_{T\dot{\xi}^*}$$

$$\boldsymbol{\omega}(T) - \mathbf{R}^* \boldsymbol{\omega}(0) = 0$$

$$\mathbf{v}(T) - \mathbf{R}^* \mathbf{v}(0) = 0$$



$\dot{\xi}^*$	Commanded turn rate	u^*	Commanded speed
γ^*	Commanded climb angle	ρ^*	Commanded turn radius
T	Flapping period	$\mathbf{\Theta}$	Euler angles (ϕ, θ, ψ)
$\mathbf{r}$	Body position	$\boldsymbol{\omega}$	Body angular rate
$\mathbf{v}$	Body velocity		

Longitudinal Flight Constraints

Longitudinal flight is a special case of a coordinated turn where:

$$u^* \neq 0 \qquad \dot{\xi}^* = 0 \qquad \beta^* = 0$$

After each period T:

- Body orientation $\boldsymbol{\Theta}(t)$ is constant:

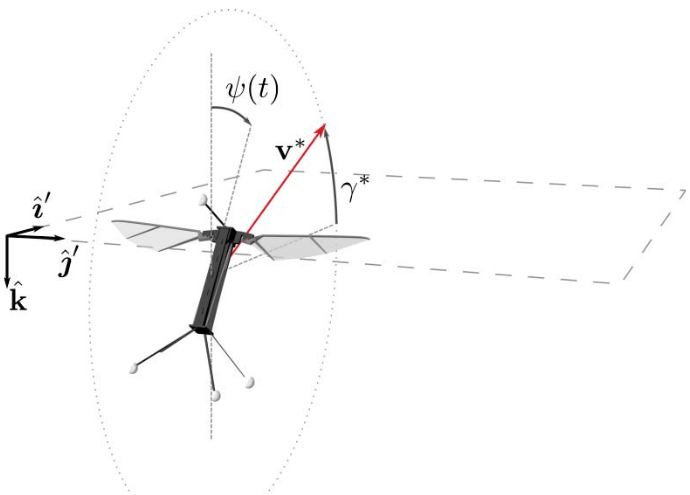
$$\boldsymbol{\Theta}(T) - \boldsymbol{\Theta}(0) = 0$$

- Position is on the straight path defined by $\mathbf{y}^*$:

$$\mathbf{r}(T) - \mathbf{r}^*(0) + u^* T \sin(\gamma^*)\hat{\mathbf{k}} + u^* T \cos(\gamma^*)\hat{\imath}' = 0$$

- Body angular rate $\boldsymbol{\omega}$ and velocity $\mathbf{v}$ are periodic:

$$\boldsymbol{\omega}(T) - \boldsymbol{\omega}(0) = 0$$
$$\mathbf{v}(T) - \mathbf{v}(0) = 0$$



$\dot{\xi}^*$	Commanded turn rate	u^*	Commanded speed
γ^*	Commanded climb angle	ρ^*	Commanded turn radius
T	Flapping period	$\boldsymbol{\Theta}$	Euler angles (ϕ, θ, ψ)
$\mathbf{r}$	Body position	$\boldsymbol{\omega}$	Body angular rate
$\mathbf{v}$	Body velocity	β^*	Commanded sideslip

Solving for Maneuver Set Points

- To find set points corresponding to steady maneuvers, solve equations of motion subject to maneuver constraints $\mathbf{c}_m = \mathbf{0}$

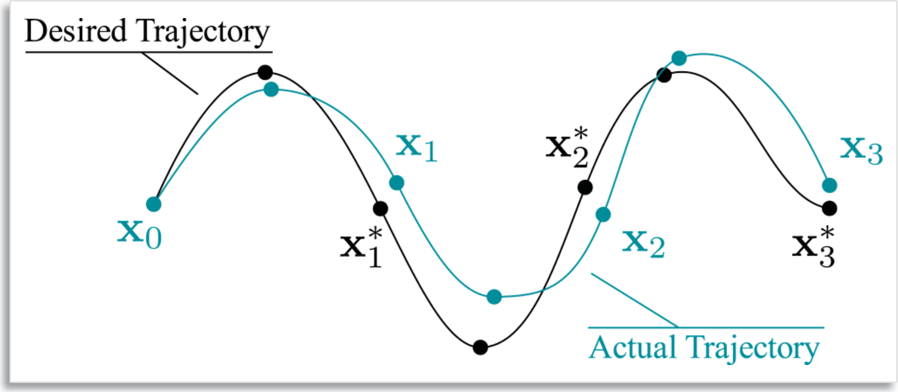
$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) \qquad \mathbf{c}_m \triangleq \mathbf{x}^*(T) - \mathbf{x}(T)$$

- Discretize ODE and write dynamics as constraints using Hermite-Simpson rule

$$\mathbf{0} = \bar{\mathbf{x}}_{k+1} - \frac{1}{2}(\mathbf{x}_{k+1} + \mathbf{x}_k) - \frac{\Delta t}{8}(\mathbf{f}_k - \mathbf{f}_{k+1}) \qquad \mathbf{0} = \mathbf{x}_{k+1} - \mathbf{x}_k - \frac{\Delta t}{6}(\mathbf{f}_{k+1} + 4\bar{\mathbf{f}}_{k+1} + \mathbf{f}_k)$$

- Dynamics constraints can be written in terms of constant matrices **A**, **B**:

$$\mathbf{c}_d(\mathbf{x}, \mathbf{u}) \triangleq \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{q}(\mathbf{x}, \mathbf{u})$$
$$\mathbf{q}(\mathbf{x}, \mathbf{u}) \triangleq \Delta t \begin{bmatrix} \mathbf{f}_1 & \bar{\mathbf{f}}_2 & \mathbf{f}_2 & \bar{\mathbf{f}}_3 & \dots & \mathbf{f}_M \end{bmatrix}^T$$



- Use nonlinear program to numerically solve:

$$\begin{bmatrix} \mathbf{c}_m(\mathbf{x}) \\ \mathbf{c}_d(\mathbf{x}, \mathbf{u}) \end{bmatrix} = \mathbf{0}$$

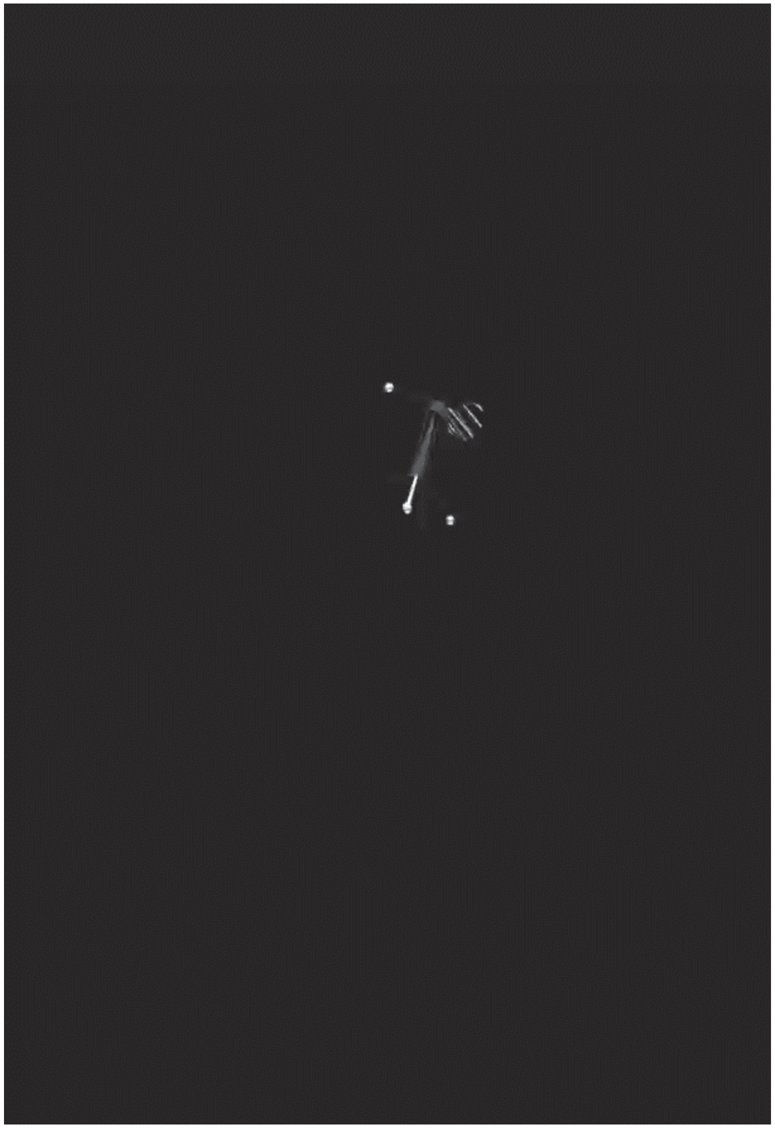
Stability of Modes in the Model

- 4 Oscillatory modes for each set point
- 2 Highly damped, coupled attitude oscillations
- 1 dominant longitudinal and 1 dominant lateral mode

Hovering Modes

Forward Flight Modes

Hovering – Longitudinal Mode



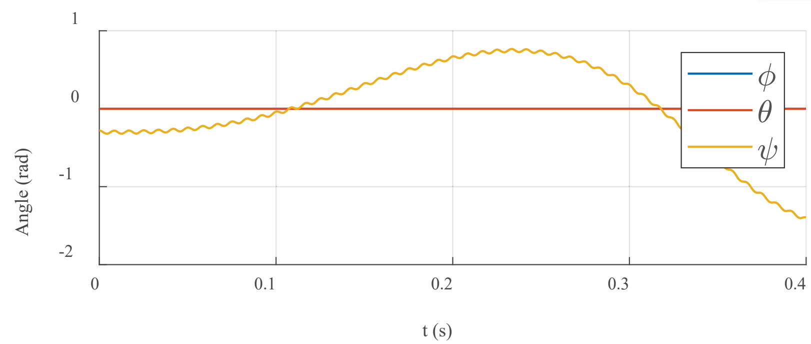
Unstable longitudinal mode in hovering

Longitudinal mode shape **v** dominated by pitching

$$\mathbf{x} = \begin{bmatrix} \theta \\ \psi \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ v_x \\ v_y \\ v_z \end{bmatrix} \quad \mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \end{bmatrix} = \begin{bmatrix} 0 \\ -0.004-0.0145i \\ 0 \\ 0 \\ 0.218-0.126i \\ 0.008+0.0004i \\ 0 \\ 0 \end{bmatrix}$$

Time constant τ and frequency f of longitudinal mode:

$$\tau = -0.24\text{s}$$
$$f = 2.64\text{Hz}$$



Hovering – Lateral Mode

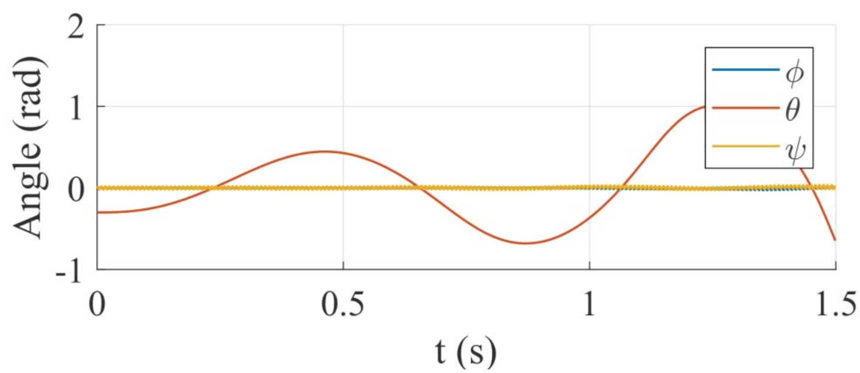
$$\mathbf{x} = \begin{bmatrix} \theta \\ \psi \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ v_x \\ v_y \\ v_z \end{bmatrix} \quad \mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \end{bmatrix} = \begin{bmatrix} 0.0004 - 0.041i \\ 0 \\ 0.445 + 0.106i \\ 0.357 - 0.036i \\ -0.002 + 0.001i \\ 0 \\ -0.051 - 0.013i \\ 0 \end{bmatrix}$$

Unstable lateral mode in hovering

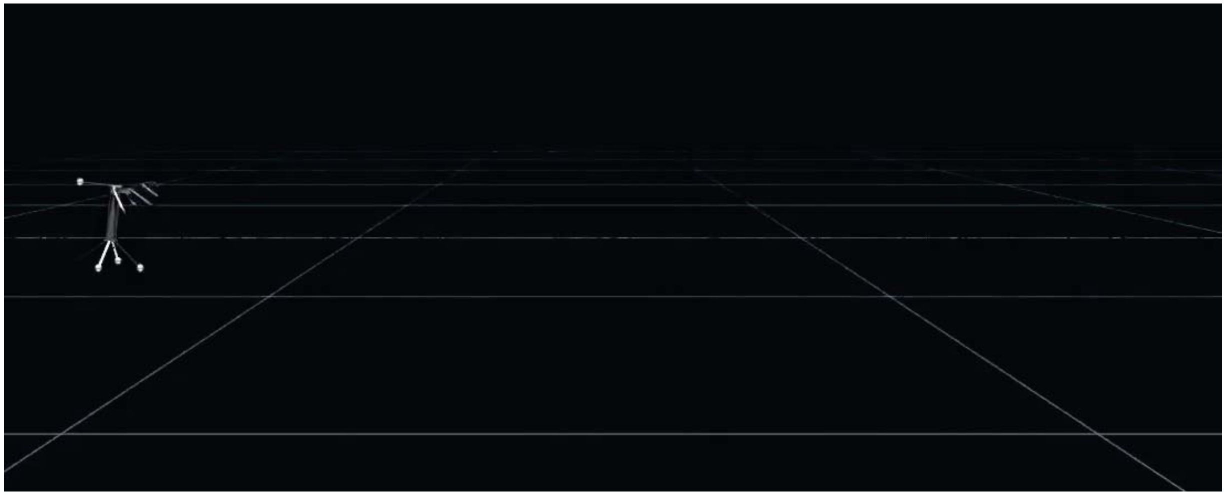
Lateral mode shape **v** dominated by roll and lateral velocity

Time constant τ and frequency f of longitudinal mode:

$$\tau = -0.88\text{s}$$
$$f = 1.20\text{Hz}$$



Steady Forward – Longitudinal Mode



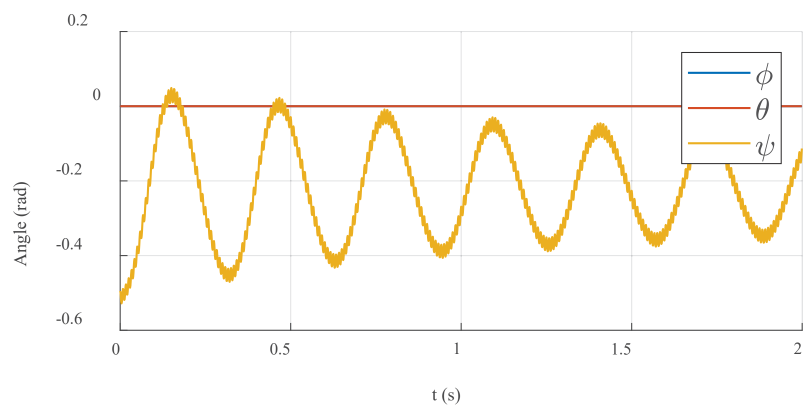
$$\mathbf{x} = \begin{bmatrix} \theta \\ \psi \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ v_x \\ v_y \\ v_z \end{bmatrix} \quad \mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \end{bmatrix} = \begin{bmatrix} 0 \\ 0.004+0.0001i \\ 0 \\ 0 \\ 0.008+0.078i \\ 0.006+0.002i \\ 0 \\ 0.004+0.0006i \end{bmatrix}$$

Longitudinal mode becomes stable in forward flight

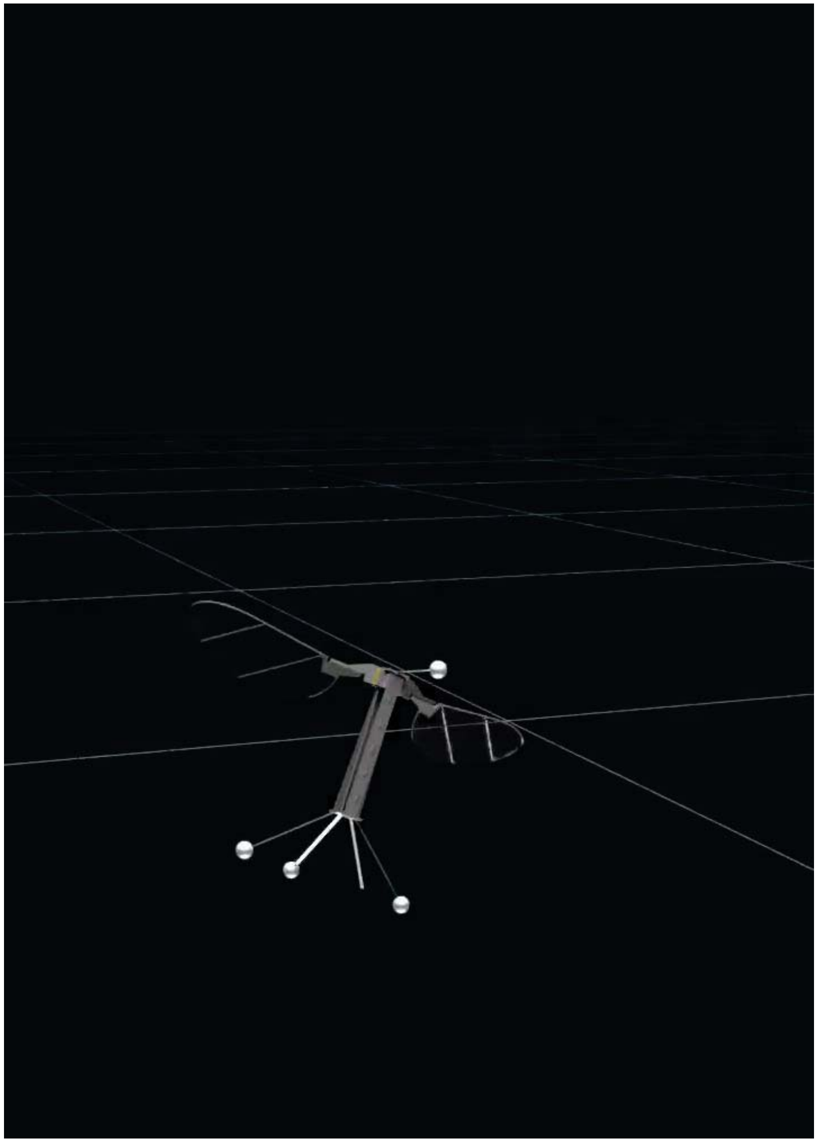
Mode shape **v** dominated by pitching

Mode has a very large time constant τ

$$\tau = 71.1\text{s}$$
$$f = 3.15\text{Hz}$$



Steady Forward – Lateral Mode



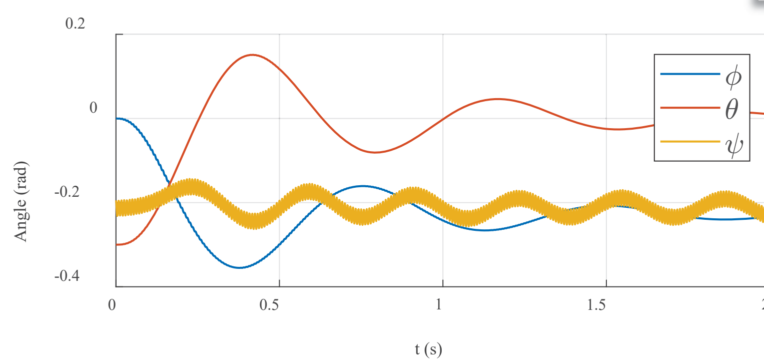
Lateral mode becomes stable in forward flight

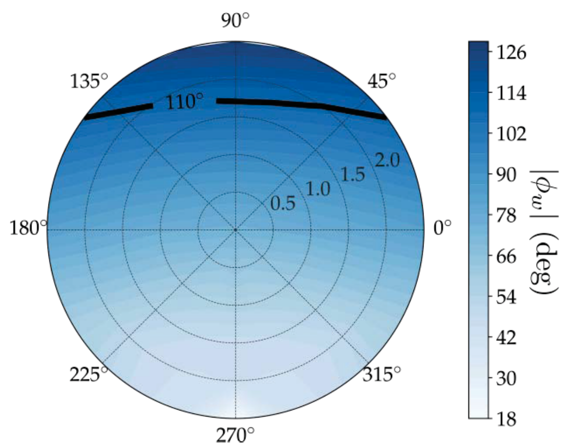
Lateral mode shape $\mathbf{v}$ shows coupling between yaw and roll

$$\mathbf{x} = \begin{bmatrix} \theta \\ \psi \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ v_x \\ v_y \\ v_z \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \end{bmatrix} = \begin{bmatrix} 0.044+0.001i \\ -0.002-0.001i \\ -0.070+0.039i \\ -0.045-0.322i \\ 0.002-0.022i \\ 0 \\ 0.027-0.035i \\ -0.002-0.002i \end{bmatrix}$$

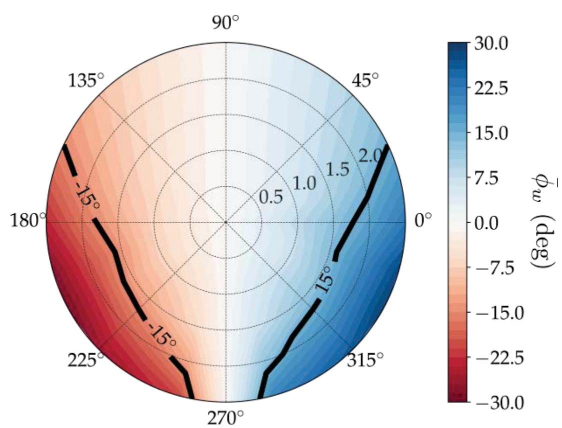
Time constant τ and frequency f of longitudinal mode:

$$\tau = 0.62\text{s}$$
$$f = 1.38\text{Hz}$$

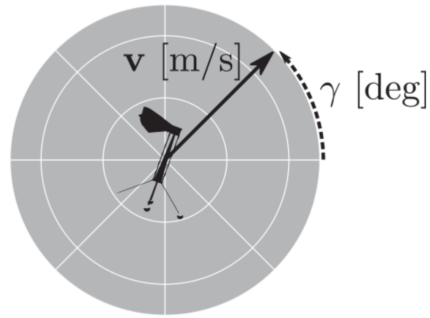


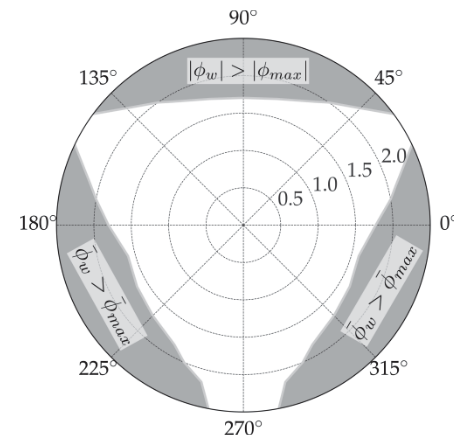


Peak-to-peak stroke amplitude



Mean stroke angle

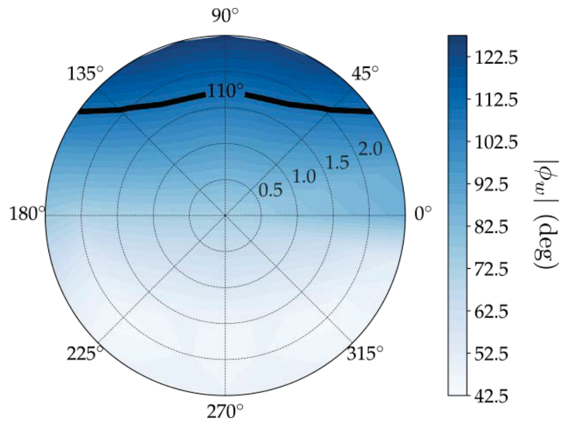




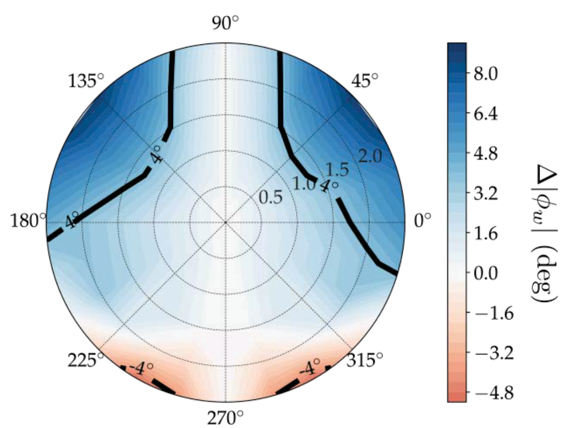
Flight Envelope

Longitudinal Flight Envelope

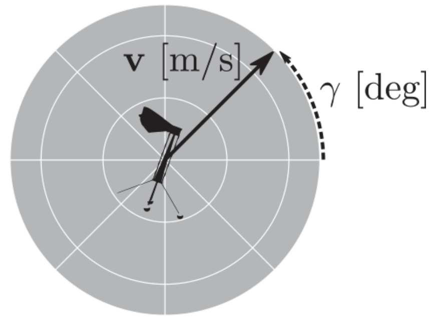
Peak-to-peak stroke amplitude and mean stroke angle as a function of speed and climb angle

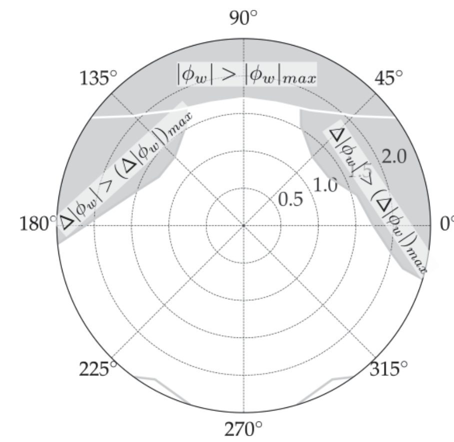


Peak-to-peak stroke amplitude



Stroke amplitude difference





Flight Envelope

Lateral Flight Envelope

Peak-to-peak stroke amplitude and right-left stroke amplitude difference as a function of speed and climb angle

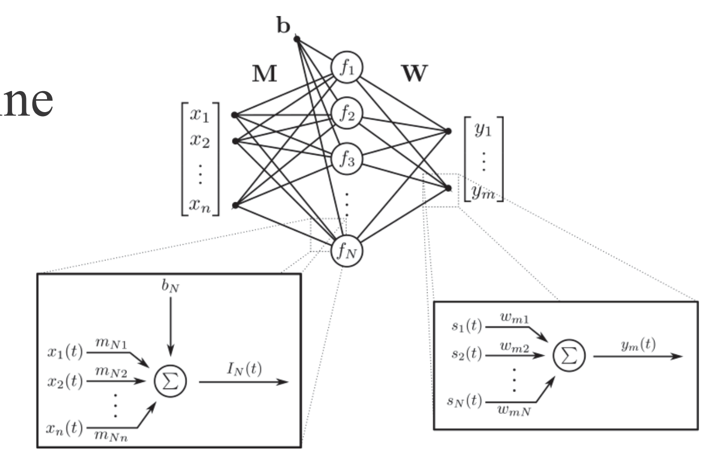
Flight Control

Flapping Wing Flight Control

- Fixed-gain controllers require hand calibration for each robot [Ma, ’13], [Dickson, ’08]
- Adaptive controller for wind gust disturbance rejection only stabilizes about hovering [Chirarattananon, P. ’17]
- Hovering control of simplified 2D model with SNN [Clawson, T.S. ’16]

Goal:

- Develop a full envelope flight controller, which can adapt online to physical parameter variations
- Spiking neural networks (SNNs) can adapt online and can be implemented in power-efficient neuromorphic chips



Single-layer SNN Controller

- SNN function approximation by connection weights **M**, **W**, and **b**

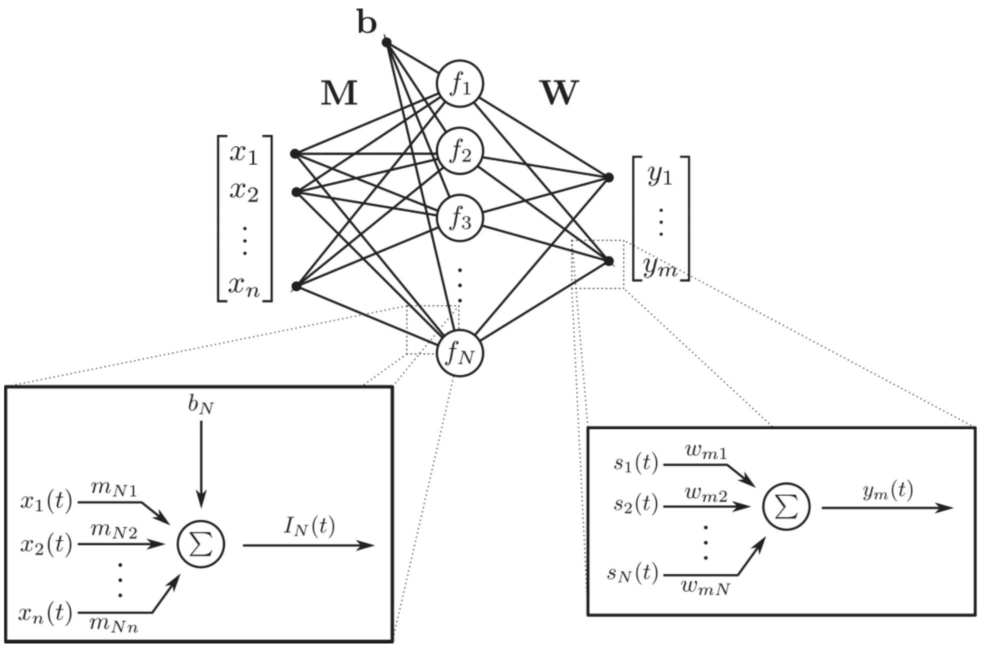
$$\mathbf{y}(t) = \mathbf{W}\mathbf{s}(t) = \mathbf{W}F(\mathbf{M}\mathbf{x}(t) + \mathbf{b})$$

- Output connection weights **W** determined offline by supervised learning

$$\mathbf{W} = \arg\min_{\mathbf{V}} \sum_j \left\| \mathbf{f}(\mathbf{x}_j) - \mathbf{V}F(\mathbf{M}\mathbf{x}_j + \mathbf{b}) \right\|^2$$

- Training data set $\mathcal{D}$ generated by a stabilizing target control law (e.g. optimal PIF controller)

$$\mathcal{D} = \left\{ \left(\mathbf{x}_j, \mathbf{f}(\mathbf{x}_j) \right) \middle| \; j = 1, \ldots, M \right\}$$



M	Input Connection Weights
W	Output Connection Weights
b	Input bias
$\mathbf{s}(t)$	Post-synaptic current
F	Nonlinear activation function
$\mathbf{f}(\mathbf{x}_j)$	Target control law data
M	Number of training data points

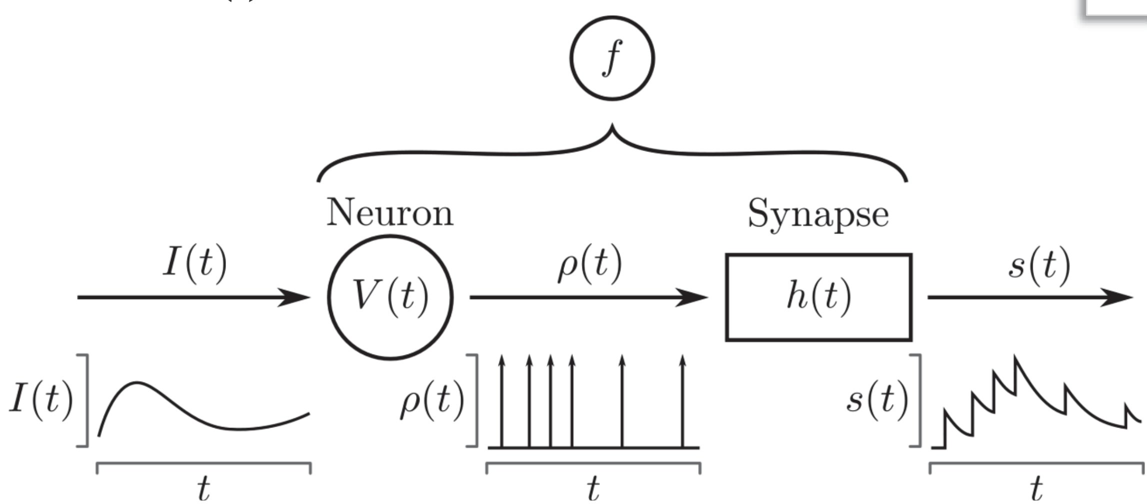
SNN Control Model

- Neurons generate spike trains $\rho(t)$ based on input current $I(t)$
- Synapses filter the spikes and generate post-synaptic current $s(t)$
- Synapses modeled as first-order low-pass filters $h(t)$

$$\rho(t) = \sum_{k=1}^{M} \rho_k(t) = \sum_{k=1}^{M} \delta(t - t_k)$$

$$s(t) = \int_0^t h(t-\tau)\rho(\tau)d\tau$$

$$h(t) = \frac{1}{\tau_s} e^{-t/\tau_s}$$



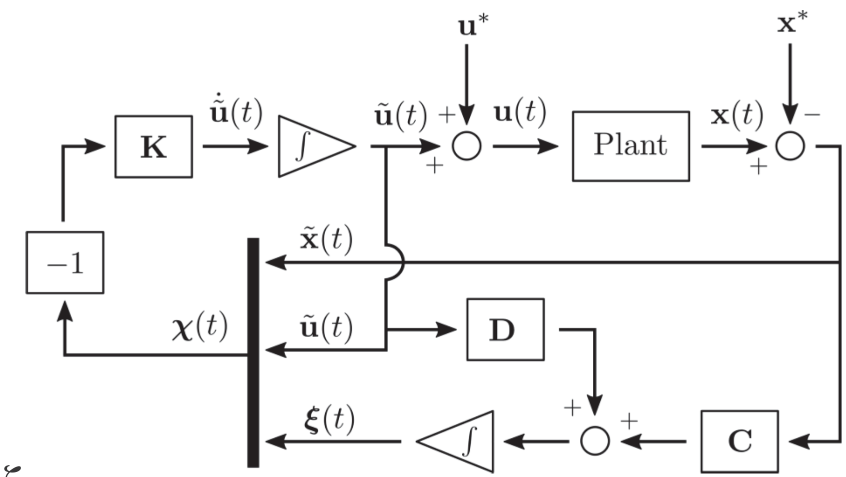
δ	Dirac delta
t_k	Time of kth spike
M	Spike count
τ_s	Synaptic time constant

PIF Compensator

- PIF control law is used as the target function for training an SNN offline

- Optimal linear controller guaranteed to stabilize linear system

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$
$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)$$



- Control law proportional to error in state $\mathbf{x}$, control $\mathbf{u}$, and integral of output ξ

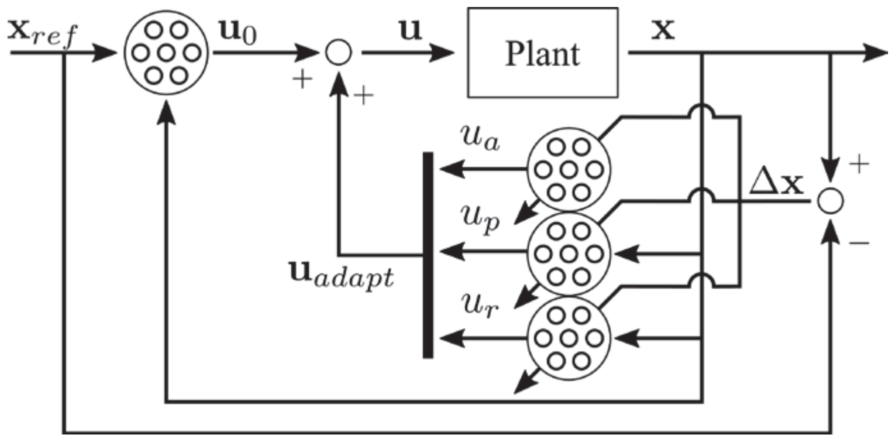
$$\mathbf{v}(t) \triangleq \dot{\tilde{\mathbf{u}}}(t) = -\mathbf{K}\chi(t)$$
$$= -\mathbf{K}_1\tilde{\mathbf{x}}(t) - \mathbf{K}_2\tilde{\mathbf{u}}(t) - \mathbf{K}_3\xi(t)$$

$\mathbf{x}^*$	State set point
$\mathbf{u}^*$	Control set point
χ	Augmented state vector
$\mathbf{v}$	Control rate of change

- The control law minimizes the quadratic cost J

$$J = \lim_{t_f \to \infty} \frac{1}{2t_f} \int_0^{t_f} \{\chi^T(t)\mathbf{Q}'\chi(t) + \mathbf{v}^T(t)\mathbf{R}'\mathbf{v}(t)\} dt$$

Adaptive SNN Controller (Hovering Only)



- First step towards full flight envelope control
- Control signal $\mathbf{u}(t)$ provided entirely by spiking neural networks

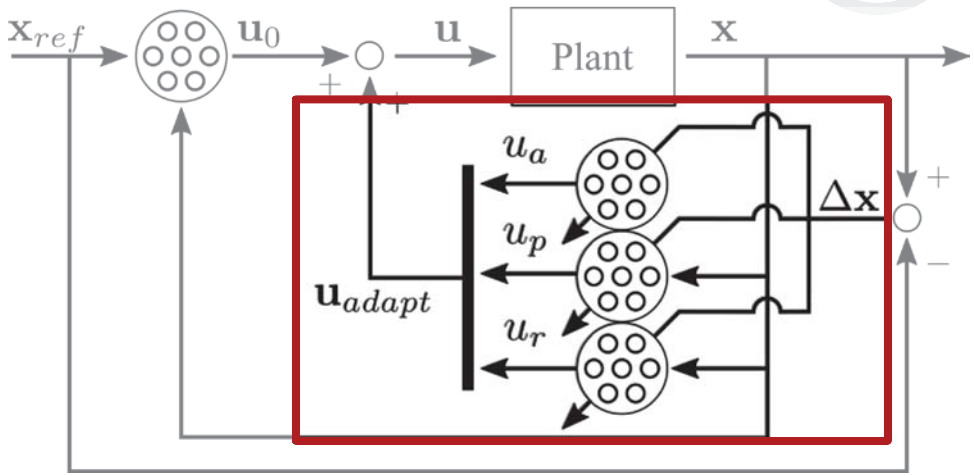
$$\mathbf{u}(t) = \mathbf{u}_0(t) + \mathbf{u}_{adapt}(t)$$

- Non-adaptive term $\mathbf{u}_0(t)$ trained offline by supervised learning to approximate PIF control law
- Adaptive term $\mathbf{u}_{adapt}(t)$ adapts online to minimize output error

$\mathbf{x}_{ref}$	Reference state
$\mathbf{u}_0$	Non-adaptive control input
$\mathbf{u}_{adapt}$	Adaptive control input
$\Delta\mathbf{x}$	State error
u_a	Amplitude input
u_p	Pitch input
u_r	Roll input

[T. S. Clawson, T. C. Stewart, C. Eliasmith, S. Ferrari "An Adaptive Spiking Neural Controller for Flapping Insect-scale Robots," *IEEE Symposium Series on Computational Intelligence (SSCI),* Honolulu, HI, December 2017]

Adaptive SNN Controller (Hovering Only)



- Adaptive term $\mathbf{u}_{adapt}$ comprised of inputs for flapping amplitude u_a, pitch u_p, roll u_r

$$\mathbf{u}_{adapt}(t) = \begin{bmatrix} u_a(t) & u_p(t) & u_r(t) \end{bmatrix}^T$$

- Output weights adapt online to minimize output error

$$E(t) = \mathbf{\Lambda}^T(\Delta x(t) + \alpha \Delta \dot{\mathbf{x}}(t))$$

- Every element of $\mathbf{u}_{adapt}$ computed from a single network of 100 neurons, e.g.

$$u_a = \mathbf{W}_a \mathbf{s}_a(t)$$

- The connection weights are updated online to minimize output error

$$\dot{\mathbf{W}}_a(t) = \gamma \mathbf{s}_a(t) E(t)$$

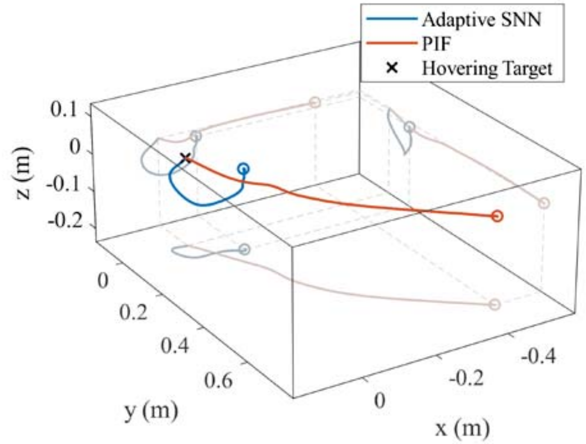
$\mathbf{\Lambda}$	Error weight matrix
Δx	State error
$\mathbf{W}_a$	Output connection weights
$\mathbf{s}_a$	Post-synaptic current
γ	Learning rate

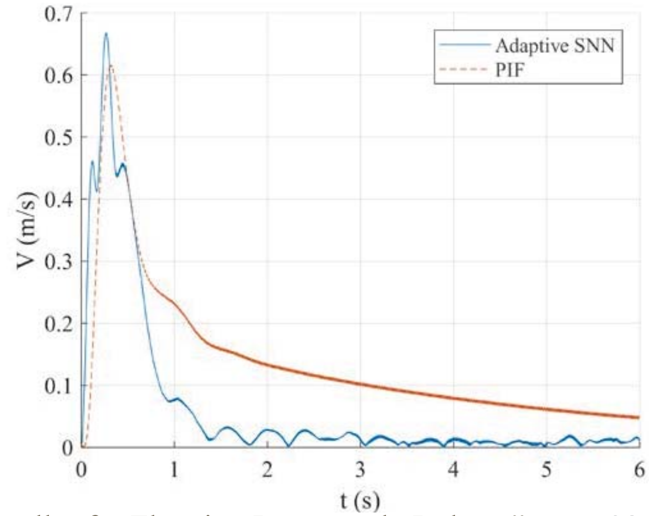
[T. S. Clawson, T. C. Stewart, C. Eliasmith, S. Ferrari "An Adaptive Spiking Neural Controller for Flapping Insect-scale Robots," *IEEE Symposium Series on Computational Intelligence (SSCI)*, Honolulu, HI, December 2017]

Adaptive SNN Controller (Hovering Only)

Comparison:

- SNN initialized with PIF
- SNN controller quickly adapts to asymmetries in the wings to stabilize hovering flight
- PIF compensator maintains stability, but drifts significantly from the origin





[T. S. Clawson, T. C. Stewart, C. Eliasmith, S. Ferrari "An Adaptive Spiking Neural Controller for Flapping Insect-scale Robots," *IEEE Symposium Series on Computational Intelligence (SSCI),* Honolulu, HI, December 2017]

SNN Controller – Full Flight Envelope

- SNN trained to approximate steady-state gain of gain-scheduled PIF
- PIF Gain matrices dependent on scheduling variables **a**

$$\dot{\tilde{\mathbf{u}}}(t) = -\mathbf{K}_1(\mathbf{a})\tilde{\mathbf{x}}(t) - \mathbf{K}_2(\mathbf{a})\tilde{\mathbf{u}}(t) - \mathbf{K}_3(\mathbf{a})\boldsymbol{\xi}(t)$$

- Steady-state gain computed using transfer function and final value theorem

$$\mathbf{G}(s) \triangleq -(s\mathbf{I} + \mathbf{K}_2(\mathbf{a}))^{-1}\mathbf{K}_1(\mathbf{a})$$

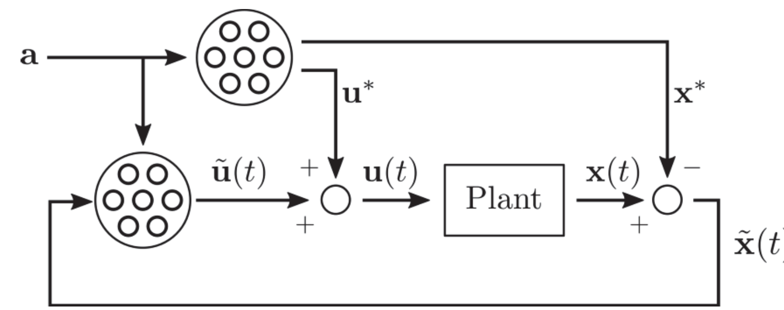
$$\mathbf{G}(0) = -\mathbf{K}(\mathbf{a})_2^{-1}\mathbf{K}_1(\mathbf{a}) \triangleq \mathbf{K}_{ss}(\mathbf{a})$$

- Network output weights computed to approximate steady-state gain matrix $\mathbf{K}_{ss}$

$$\mathbf{W} = \underset{\mathbf{V}}{\operatorname{argmin}} \sum_j \left\| \mathbf{K}_{ss}(\mathbf{a}) - \mathbf{V}F(\mathbf{M}\mathbf{a}_j + \mathbf{b}) \right\|^2$$

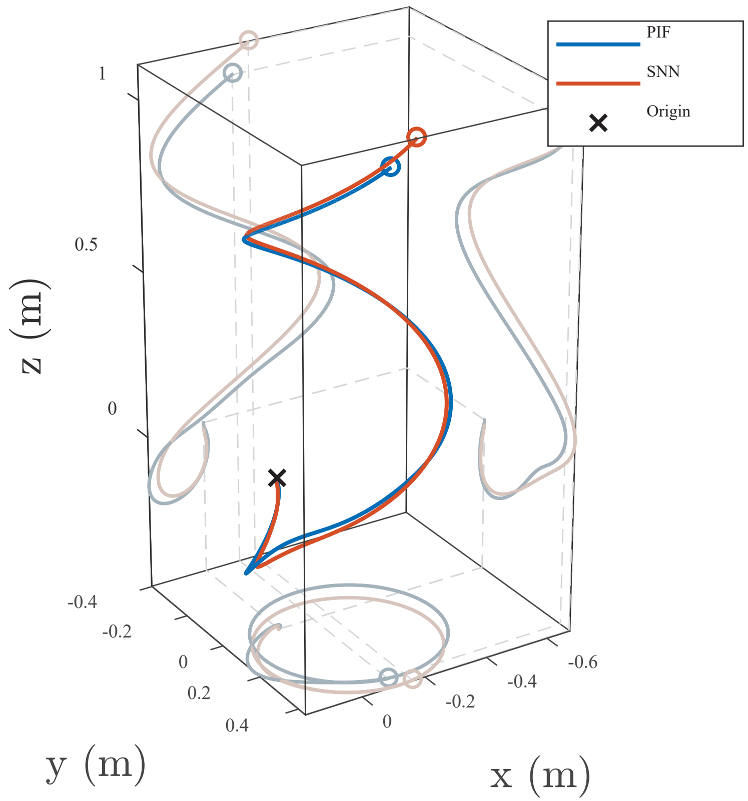
- SNN Control input is a linear transformation of post-synaptic current

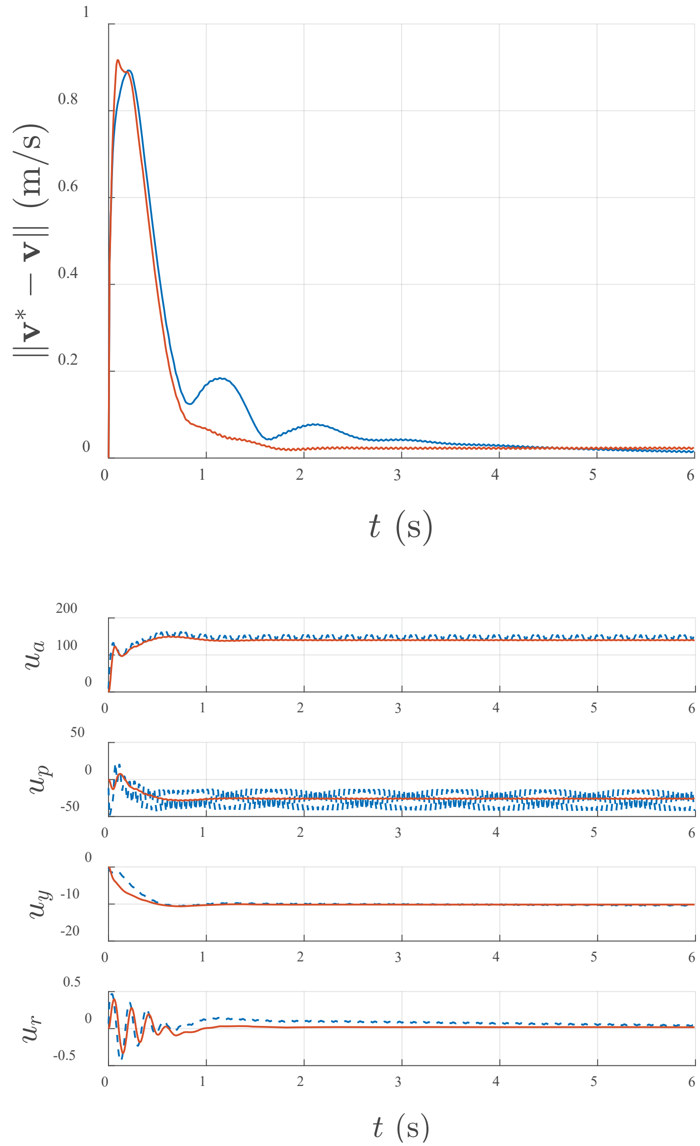
$$\tilde{\mathbf{u}}(t) = \mathbf{W}(\mathbf{a})\mathbf{s}(t)$$



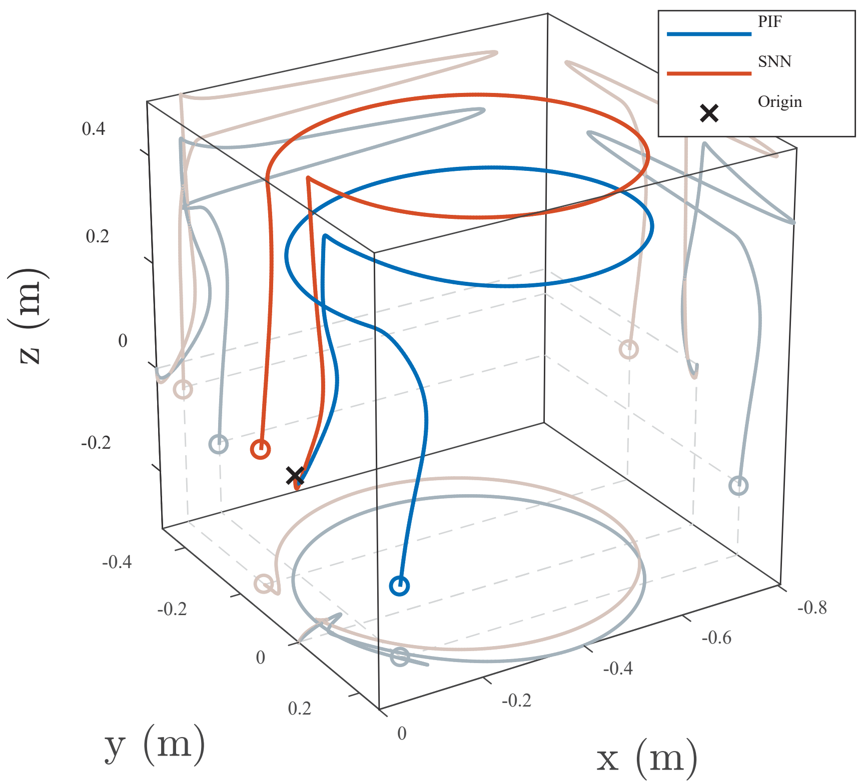
$\mathbf{K}_i$	PIF gain matrices	$\tilde{\mathbf{x}}$	State deviation
$\tilde{\mathbf{u}}$	Control deviation	ξ	Integral of output error
$\mathbf{a}$	Scheduling variables	$\mathbf{G}(s)$	Transfer function
s	Laplace variable	$\mathbf{K}_{ss}$	Steady-state gain matrix
$\mathbf{W}$	Output connection weights	$\mathbf{s}$	Post-synaptic current

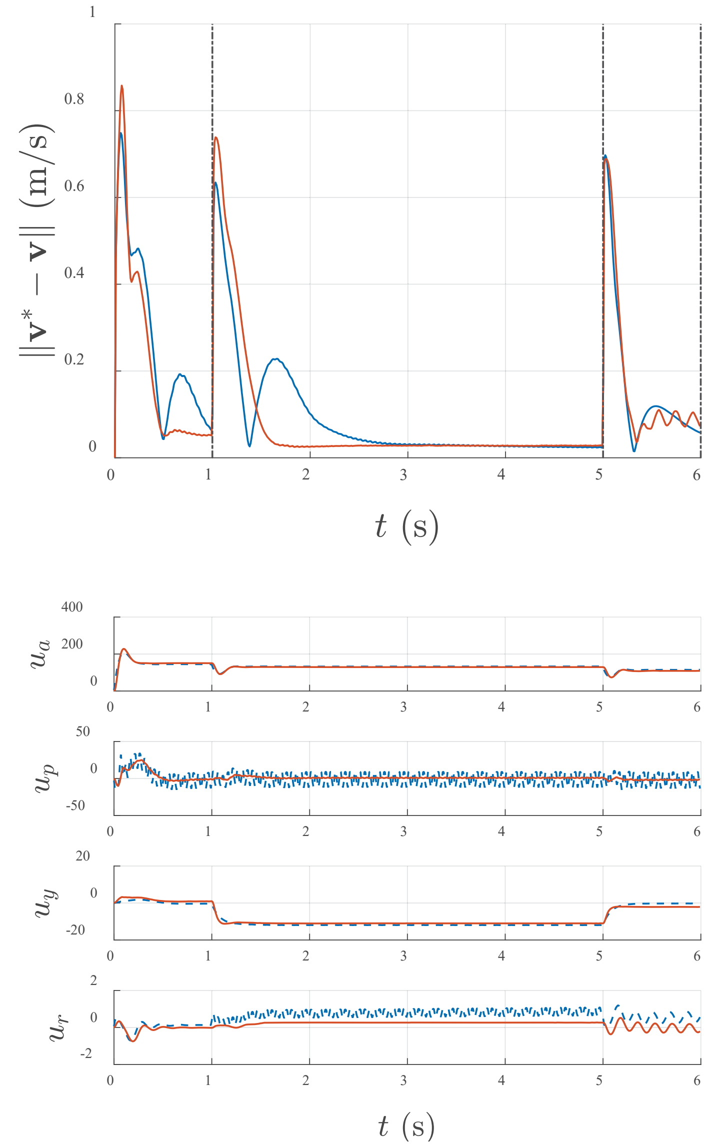
SNN Control – Climbing Turn





SNN Control – Complete Turn



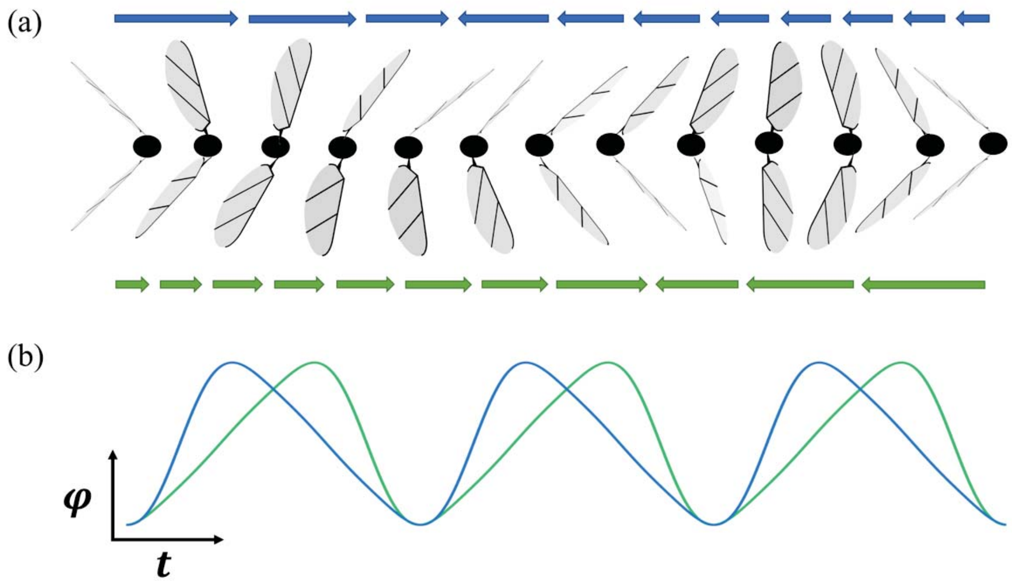


Hardware/control Developments to Enable Aggressive Yaw Maneuvers

Novel yaw generation for flapping-wing MAVs

Motivation:

- The RoboBee nominally achieves yaw control via modulation of the ratio of upstroke to downstroke speed for each wing (“split-cycling”)

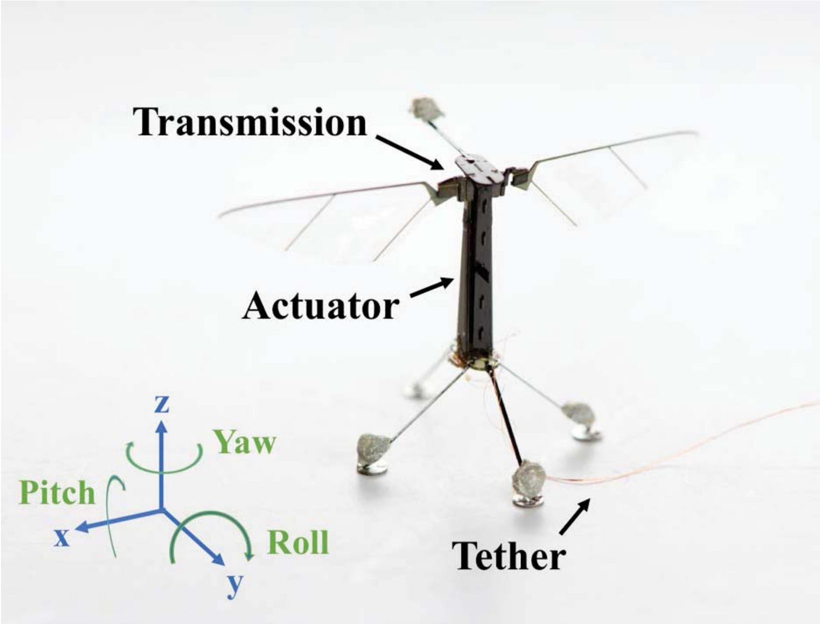


(a) depiction of “split-cycle” motion. (b) drive signals that achieve split cycle motion through the addition of higher harmonics onto the fundamental drive frequency.

Novel yaw generation for flapping-wing MAVs

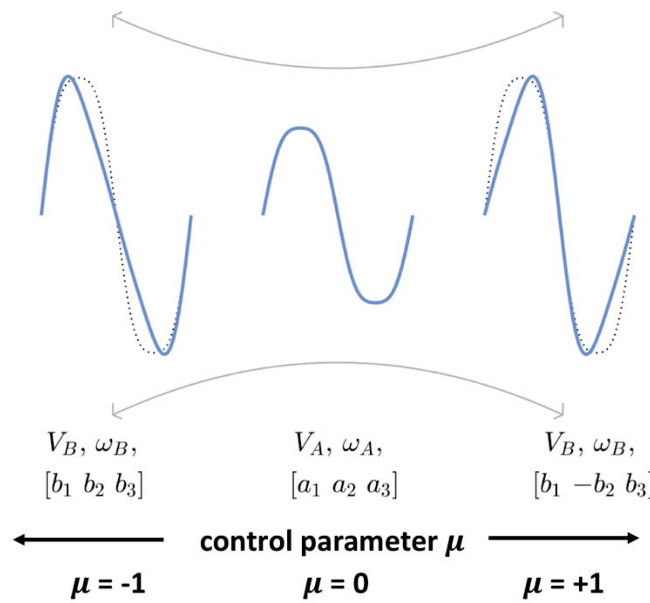
Motivation:

- RoboBee nominally achieves yaw control via modulation of ratio of upstroke to downstroke speed for each wing (“split-cycling”)
- Split-cycling is filtered out by the transmission during high-frequency flapping



Novel yaw generation for flapping-wing MAVs

- Next objective: Demonstrate yaw control in-flight
  – Confirm that (as according to model and basic kinematic tests) flightworthy lift should be maintained during yaw
  – Implement in-flight yaw control
    - Improved basic hovering
    - Yaw maneuvers in flight
- Exploring non-resonant flapping regimes opens new family of control parameters conducive to event-based architectures



- Paper accepted to ICRA 2019:
  - R. Steinmeyer, E.F. Helbling, and R.J. Wood, “Yaw Torque Authority for a Flapping-Wing Micro-Aerial Vehicle,” to appear: IEEE Int. Conf. on Robotics and Automation, Montreal, Canada, May, 2019.

Exteroceptive Sensing

Exteroceptive Sensing Motivation

- Onboard exteroceptive sensors required for full flight autonomy
- Fast dominant time scales of insect-scale flight require high sensing rate and low latency
  - Traditional sensors consume large amounts of power for high sensing rate (e.g. ~100 watts for high speed camera)
  - High data rate requires additional data processing
- Neuromorphic vision sensors have 1μs temporal resolution and require at most a few milliwatts of power [Lichtsteiner, ’08], [Brandli, ’14]

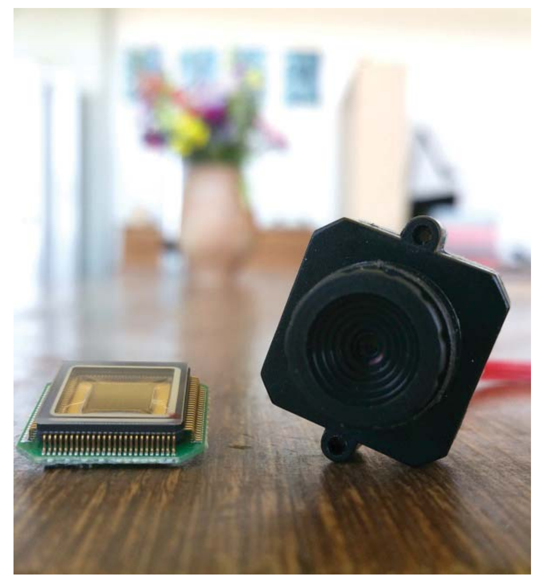


Image Credit: inivation (https://inivation.com)

Neuromorphic Vision Sensors

- Neuromorphic cameras generate asynchronous events instead of frames
- An event at (x, y) is generated at time t_i, with polarity

$$p_i = \begin{cases} 1, & \text{if } \ln(I(x,y,t_{i-1})) - \ln(I(x,y,t_i)) = -\theta \\ -1, & \text{if } \ln(I(x,y,t_{i-1})) - \ln(I(x,y,t_i)) = \theta \end{cases}$$

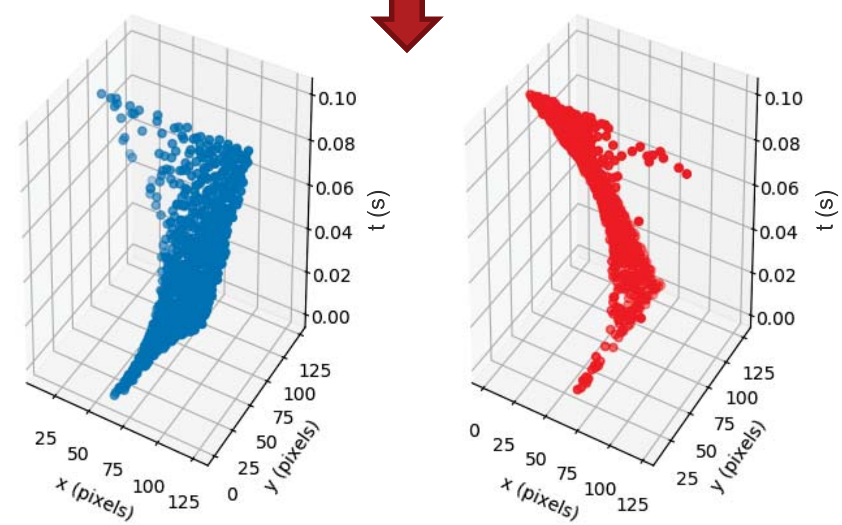
- "On" events when $p_i = 1$
- "Off" events when $p_i = -1$
- The ith event $\mathbf{e}_i$ is described by the tuple $\mathbf{e}_i = (x, y, t, p)_i$

$$x, y \in \mathbb{N}^+ \qquad t \in \mathbb{R}^+ \qquad p \in \{-1, 1\}$$

- The set of all events is

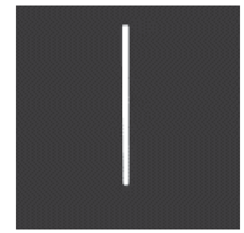
$$\mathcal{E} = \{\mathbf{e}_i \,|\, i = 1, \ldots, N\}$$

Source



"On" Events "Off" Events

The Optical Flow Problem

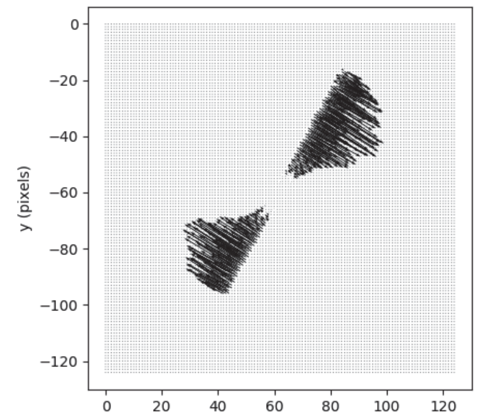


Standard Optical Flow Problem

- Assume: $$\frac{dI(x,y,t)}{dt}=0$$

- Determine horizontal and vertical flow (v_x, v_y) from

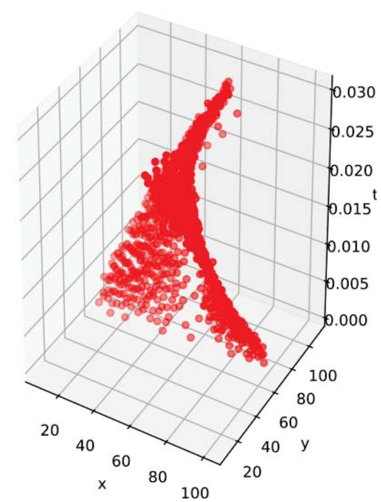
$$\frac{dI(x,y,t)}{dt}=\begin{bmatrix} I_x(x,y,t) & I_y(x,y,t) \end{bmatrix}\begin{bmatrix} v_x(x,y,t) \\ v_y(x,y,t) \end{bmatrix}+I_t(x,y,t)=0$$



Neuromorphic Optical Flow

- Coordinates of some point $\mathbf{r}=\begin{bmatrix} r_x & r_y \end{bmatrix}^T$ in the image plane determined by optical flow

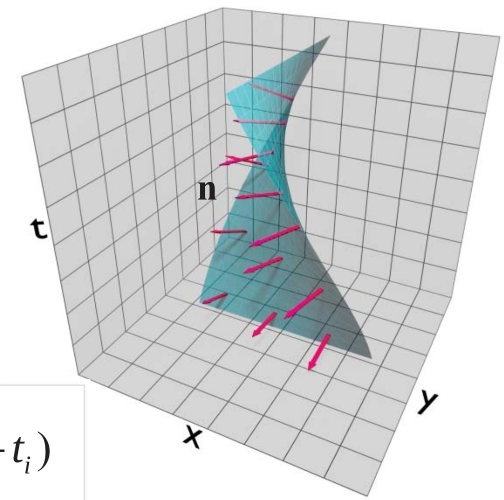
$$\begin{bmatrix} r_x(t_2)-r_x(t_1) \\ r_y(t_2)-r_y(t_1) \end{bmatrix}=\int_{t_1}^{t_2}\mathbf{v}(\tau)d\tau \approx \begin{bmatrix} v_x dt \\ v_y dt \end{bmatrix}, \qquad \mathbf{v}(\tau)=\begin{bmatrix} v_x(\tau) \\ v_y(\tau) \end{bmatrix}$$



- Scattered events are generated by motion of the point
- Determine optical flow by estimating the motion of points in the scene using the scattered events

Neuromorphic Optical Flow

- Existing neuromorphic optical flow methods rely on optimization [Benosman, '14], [Rueckauer, '16]
- Estimate continuous motion from discrete events
- Introduce continuous event rate *f* through convolution of events with continuous kernel *K*



$$f(x,y,t) = K(x,y,t) * E(x,y,t)$$

$$E(x,y,t) = \sum_{i=1}^{N} \delta(x - x_i, y - y_i, t - t_i)$$

- Assume gradient **n** of event rate is normal to the motion of points in the scene
- Speed of the motion is inversely proportional to magnitude of gradient
- Optical flow is written directly in terms of the event rate gradient

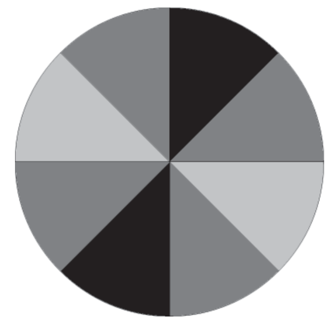
$$\mathbf{n} = \begin{bmatrix} a & b & c \end{bmatrix}^T$$

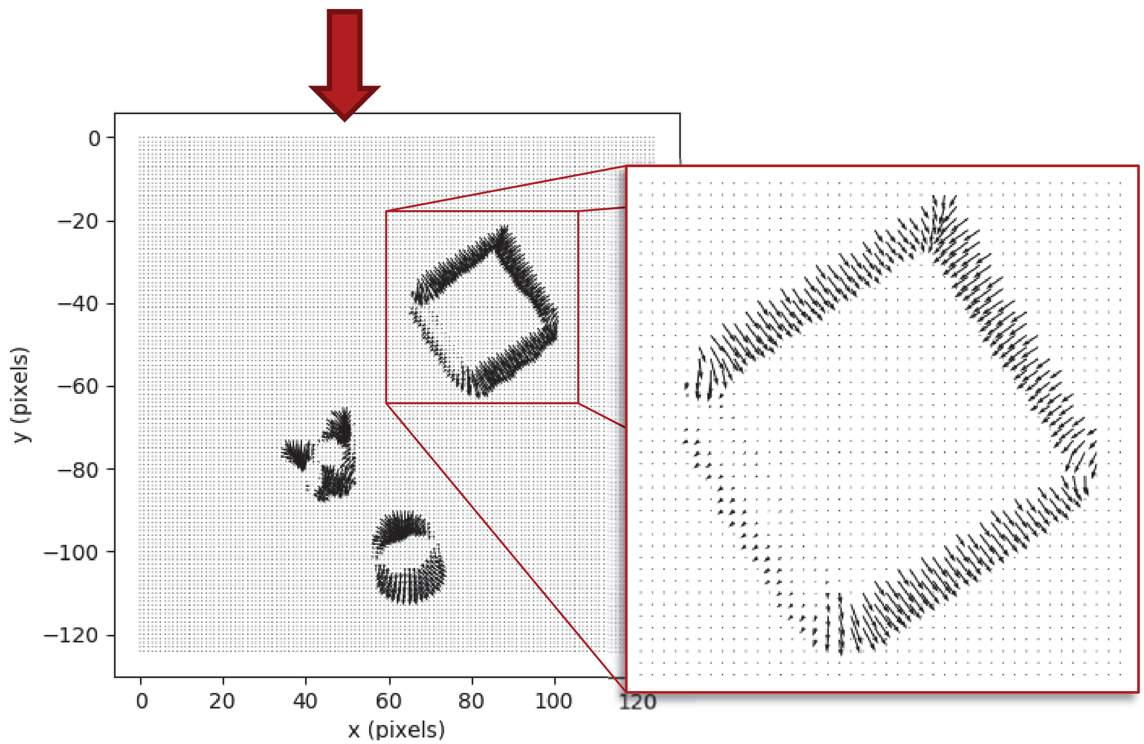
$$\mathbf{w} = \begin{bmatrix} \frac{\partial t}{\partial x} & \frac{\partial t}{\partial y} \end{bmatrix}^T = \begin{bmatrix} -\frac{a}{c} & -\frac{b}{c} \end{bmatrix}^T$$

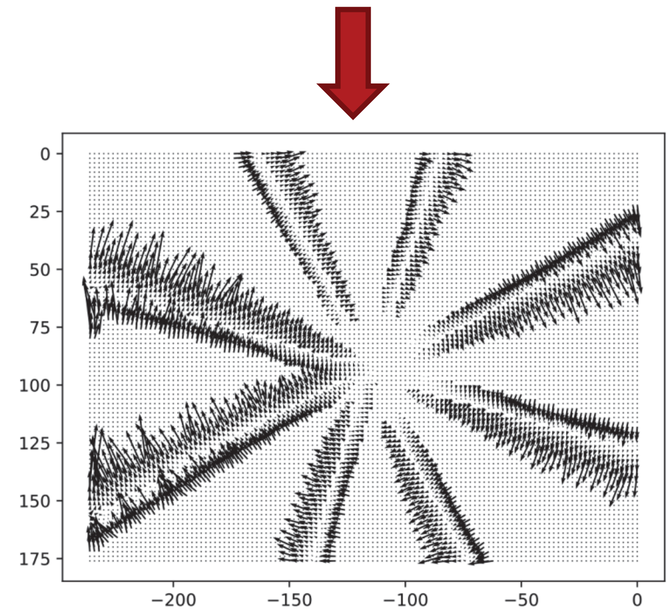
$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \left(\frac{1}{\|\mathbf{w}\|} \right) \frac{\mathbf{w}}{\|\mathbf{w}\|} = -\frac{c}{a^2 + b^2} \begin{bmatrix} a \\ b \end{bmatrix}$$

Neuromorphic Optical Flow Results

Mean processing time per event: 0.7 µs







Neuromorphic Motion Detection

Detect motion relative to the environment
using a rotating neuromorphic camera

Assumptions

- Known camera motion
- Camera motion dominated by rotation
- Total derivative of pixel intensity is zero

Camera View

Neuromorphic Camera

World View

Neuromorphic Motion Detection

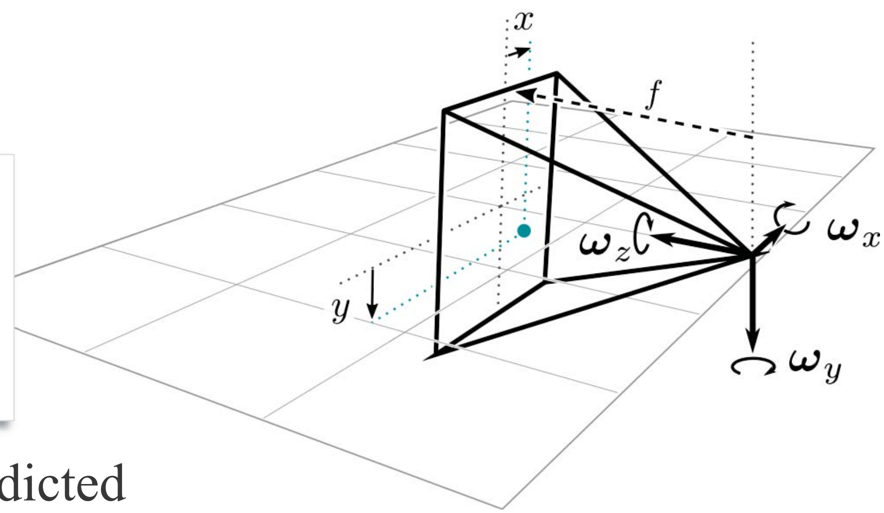
- Pixel intensity can be recovered by integrating the event rate

$$I(x,y,t) = \exp\left(\theta \int_0^t f(x,y,\tau)\,d\tau\right) + I(x,y,0)$$

- By previous assumptions, future pixel intensity can be predicted if image-plane motion field (v_x, v_y) is known

- Motion field due to camera rotation is

$$v_x(x,y) = -\frac{x^2}{f}\omega_y(t) + \frac{xy}{f}\omega_x(t) - f\,\omega_y(t) + y\omega_z(t)$$
$$v_y(x,y) = -\frac{xy}{f}\omega_y(t) + \frac{y^2}{f}\omega_x(t) - x\omega_z(t) + f\,\omega_x(t)$$



- Pixel intensity after a short time Δt is predicted from motion field:

$$\tilde{I}(x,y,t) = I(x - v_x\Delta t,\, y - v_y\Delta t,\, t - \Delta t)$$

Neuromorphic Motion Detection Results

1. Compute difference between predicted and measured intensity

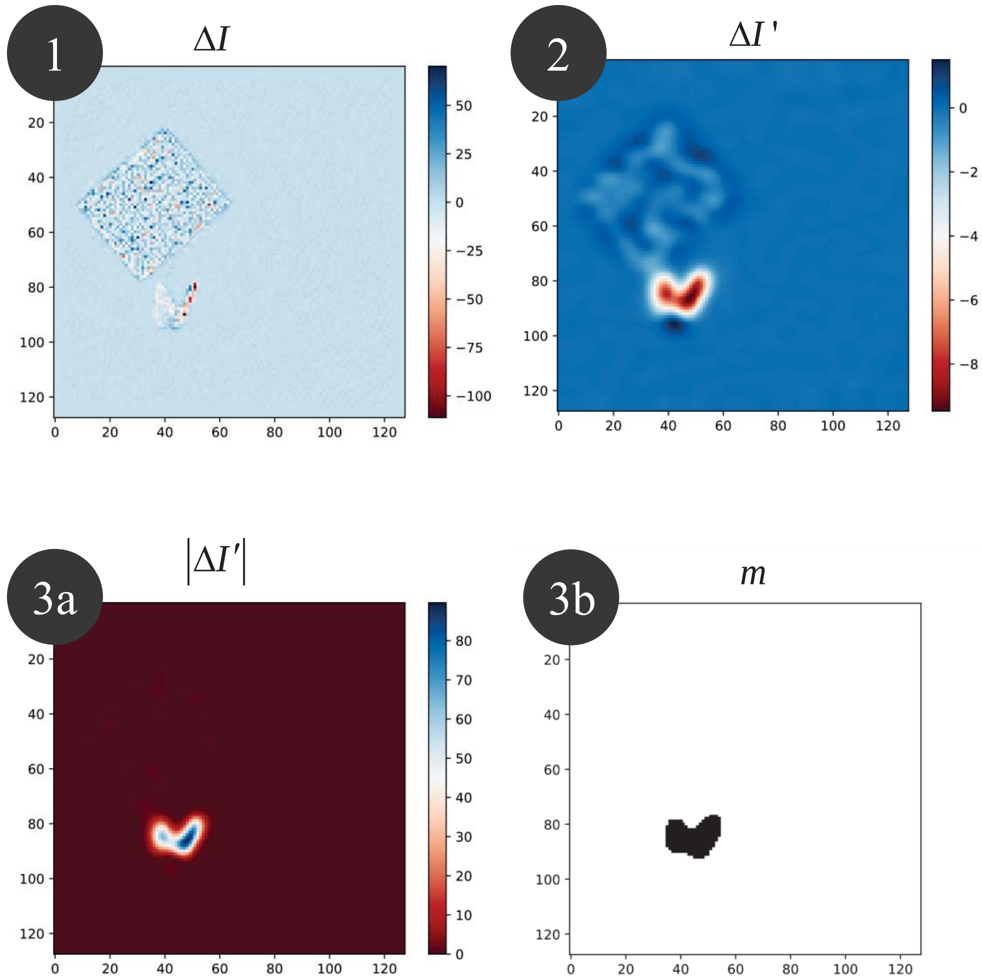
$$\Delta I(x,y,t) = I(x,y,t) - \tilde{I}(x,y,t)$$

2. Denoise by convolving with a multivariate Gaussian kernel K_Σ with covariance Σ

$$\Delta I'(x,y,t) = K_\Sigma(x,y,t) * I(x,y,t)$$

3. Detect motion by comparing smoothed intensity difference with a threshold γ

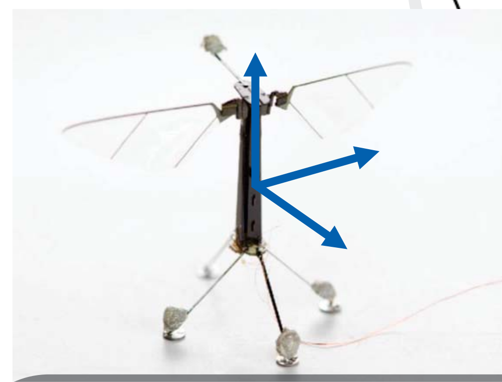
$$m(x,y,t) = \begin{cases} 1, & \text{if } |\Delta I'(x,y,t)| > \gamma. \\ 0, & \text{otherwise.} \end{cases}$$

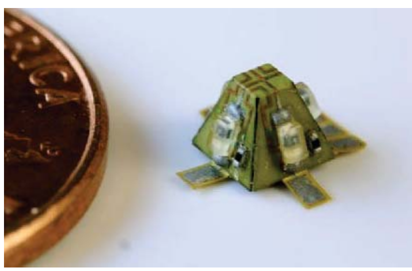


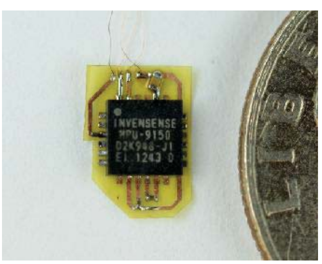
Sensor Hardware Integration for Event-driven Control Experiments

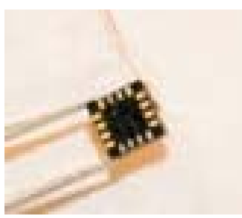
Background: prior sensor integration on RoboBees

- Past work on RoboBee sensors has focused on individual sensor integration and characterization

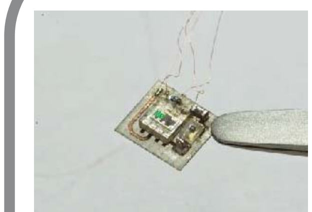


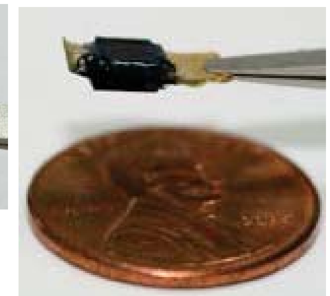


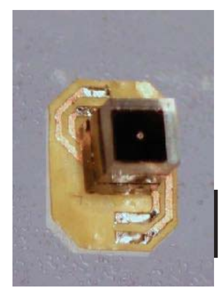


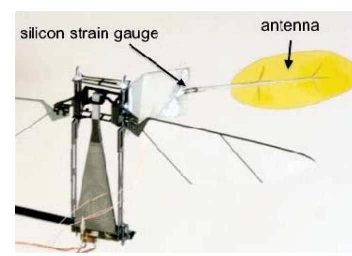


$$\begin{bmatrix} \theta_x \\ \theta_y \\ \theta_z \\ \dot{\theta}_x \\ \dot{\theta}_y \\ \dot{\theta}_z \end{bmatrix} \quad \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}$$









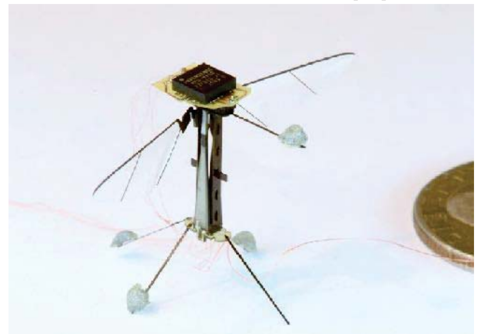
EF Helbling, et al., ICRA 2014
EF Helbling, et al., IMAV 2014
EF Helbling, et al., ISRR 2015

S Mange, EF Helbling, ICRA 2017
PE Duhamel, EF Helbling, et al., in preparation

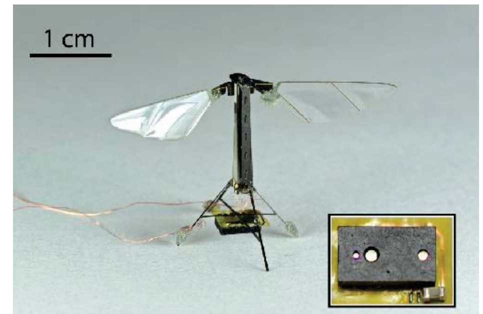
Current sensor integration on RoboBees

- Current work emphasizing two classes of sensors:

Attitude ($\dot{\theta}$)



Altitude (z)

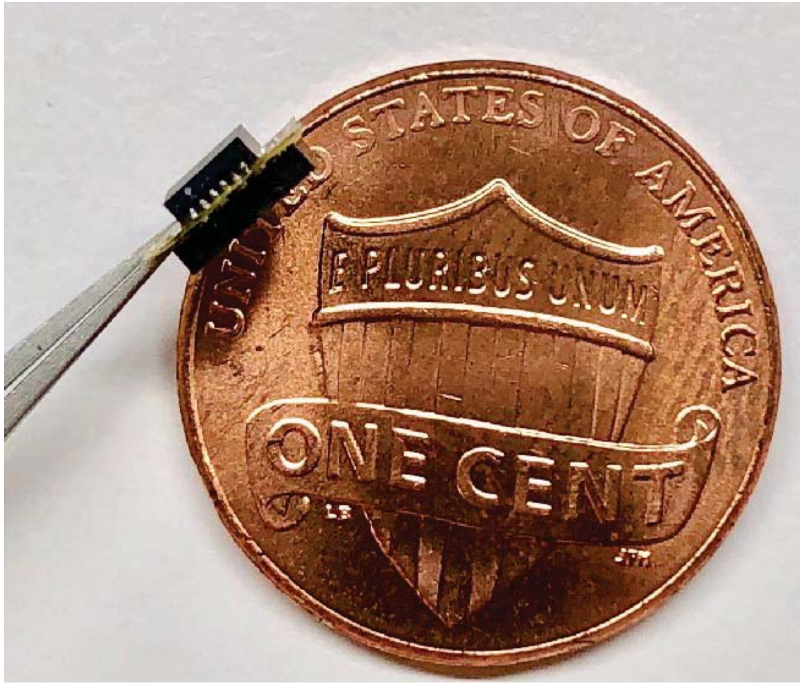


Mass	37 mg	21 mg
Size	4x4x1 mm	5x3x1 mm
Power	<3mW	4 mW
Maximum Range	2000 deg/s	14 cm
Maximum Data Rate	1kHz	50 Hz
Communication	I2C (4-wire)	I2C (4-wire)

EF Helbling, et al., IMAV 2014
EF Helbling, et al., ISRR 2015

Current sensor integration on RoboBees

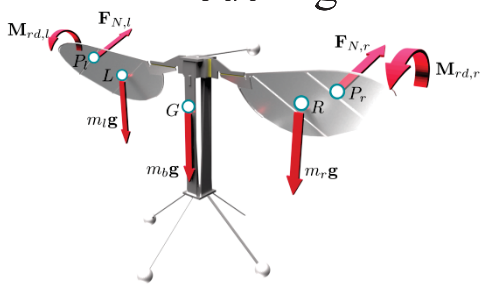
- Current work is combining these two into a single package as we gear up for integration and flight control experiments with Cornell
  - Combination of IMU and proximity sensor
  - Stabilize attitude with gyroscope
  - Incorporate onboard accelerometer measurements to compensate for integration drift
  - Current specifications:
    - Mass: 53mg
    - Power: 8mW
    - Dimension: 5x3x2 mm



Summary

1

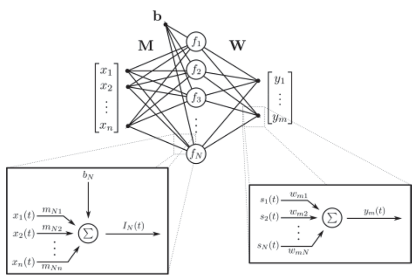
Modeling



- Flight model captures dominant modes
- Set points for steady maneuvers were computed
- Model predicts that forward flight becomes stable with increasing speed

2

Adaptive Flight Control



- Adaptive SNN Controller can adapt to unmodeled parameter variations
- SNN can provide control for full flight envelope
- Hardware developments: RoboBee aggressive yaw authority

3

Exteroceptive Sensing

- Optical flow can be efficiently computed from neuromorphic cameras
- Target motion can be detected from a rotating neuromorphic camera
- Hardware developments: integrated exteroceptive GNC RoboBee sensing