From Simulation to Flight: An Ecosystem to Program, Test, and Tune Advanced Control Systems for UAVs

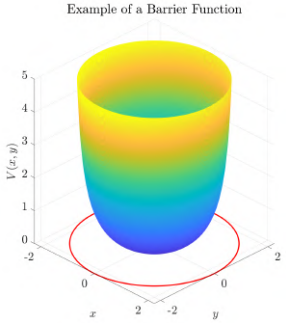
Mattia Gramuglia[1] Andrea L'Afflitto[2]

[1]Department of Mechanical Engineering, Virginia Tech

[2]Grado Department of Industrial and Systems Engineering, Virginia Tech
a.lafflitto@vt.edu

Workshop on Control Barrier Functions in Aerospace Applications
EuroGNC - Conference on Guidance, Navigation, and Control
Madrid, Spain - May 5, 2026

- Classical **Model Reference Adaptive Control (MRAC)** approaches cannot simultaneously enforce
  - **Hard constraints** on the tracking error, and
  - **Soft constraints** on quantities such as *actuator saturation*
- Existing adaptive control systems enforce hard constraints through **barrier Lyapunov functions (BLFs)**
- BLFs used to **enforce a user-defined rate of convergence** of the trajectory tracking error may lead to aggressive control effort
- Hard and soft constraints may **compete** with each other



Objectives

Develop an MRAC framework able to **simultaneously**

- enforce **hard constraints** on the tracking error
- enforce **soft constraints** on user-defined parameters
- mediate between **competing hard and soft constraints** through an adaptive mechanism
- be robust to **unmatched uncertainties**
- enforce user-defined **rate of convergence** without leveraging BLFs
- **decouple** the tuning of the *convergence rate* from the *reference model*

Plant model

$$\dot{x}(t) = Ax(t) + B\Lambda\left[u(t) + \Theta^{\mathrm{T}}\Phi(t, x(t))\right]$$

Matching conditions

$$A_{\mathrm{ref}} = A + B\Lambda K_x^{\mathrm{T}}, \quad B_{\mathrm{ref}} = B\Lambda K_r^{\mathrm{T}}$$

Algebraic Lyapunov equation

$$A_{\mathrm{ref}}^{\mathrm{T}}P + PA_{\mathrm{ref}} = -Q$$

Adaptive laws

$$\dot{\hat{K}}_x(t) = -\Gamma_x x(t)e^{\mathrm{T}}(t)PB$$
$$\dot{\hat{K}}_r(t) = -\Gamma_r r_{\mathrm{cmd}}(t)e^{\mathrm{T}}(t)PB$$
$$\dot{\hat{\Theta}}(t) = \Gamma_\Theta \Phi(t, x(t))e^{\mathrm{T}}(t)PB$$

Reference model

$$\dot{x}_{\mathrm{ref}}(t) = A_{\mathrm{ref}}x_{\mathrm{ref}}(t) + B_{\mathrm{ref}}r_{\mathrm{cmd}}(t)$$

Trajectory tracking error

$$e(t) = x(t) - x_{\mathrm{ref}}(t)$$

Control law

$$u(t) = \hat{K}_x^{\mathrm{T}}(t)x(t) + \hat{K}_r^{\mathrm{T}}(t)r_{\mathrm{cmd}}(t) - \hat{\Theta}^{\mathrm{T}}(t)\Phi(t, x(t))$$

Overview of Two-Layer MRAC

Plant model

$$\dot{x}(t) = Ax(t) + B\Lambda\left[u(t) + \Theta^{\mathrm{T}}\Phi(t, x(t))\right]$$

Auxiliary reference model

$$\dot{e}_{\mathrm{tran}}(t) = A_{\mathrm{tran}} e_{\mathrm{tran}}(t)$$

Matching conditions

$$A_{\mathrm{ref}} = A + B\Lambda K_x^{\mathrm{T}}, \quad B_{\mathrm{ref}} = B\Lambda K_r^{\mathrm{T}},$$
$$A_{\mathrm{tran}} = A_{\mathrm{ref}} + B\Lambda K_g^{\mathrm{T}}$$

Adaptive laws

$$\dot{\hat{K}}_x(t) = -\Gamma_x x(t)\varepsilon^{\mathrm{T}}(t)PB$$
$$\dot{\hat{K}}_r(t) = -\Gamma_r r_{\mathrm{cmd}}(t)\varepsilon^{\mathrm{T}}(t)PB$$
$$\dot{\hat{\Theta}}(t) = \Gamma_\Theta \Phi(t, x(t))\varepsilon^{\mathrm{T}}(t)PB$$
$$\dot{\hat{K}}_g(t) = -\Gamma_g e(t)\varepsilon^{\mathrm{T}}(t)PB$$

Reference model

$$\dot{x}_{\mathrm{ref}}(t) = A_{\mathrm{ref}} x_{\mathrm{ref}}(t) + B_{\mathrm{ref}} r_{\mathrm{cmd}}(t)$$

Trajectory tracking error and ε

$$e(t) \triangleq x(t) - x_{\mathrm{ref}}(t), \qquad \varepsilon(t) \triangleq e(t) - e_{\mathrm{tran}}(t)$$

Control law

$$u(t) = \hat{K}_x^{\mathrm{T}}(t)x(t) + \hat{K}_r^{\mathrm{T}}(t)r_{\mathrm{cmd}}(t) - \hat{\Theta}^{\mathrm{T}}(t)\Phi(t, x(t)) + \hat{K}_g^{\mathrm{T}}(t)e(t)$$

Algebraic Lyapunov equation

$$A_{\mathrm{tran}}^{\mathrm{T}} P + PA_{\mathrm{tran}} = -Q$$

Overview of Robust MRAC – Projection Operator

Plant model

$$\dot{x}(t) = Ax(t) + B\Lambda\left[u(t) + \Theta^{\mathrm{T}}\Phi(t, x(t))\right] + \xi_{\mathrm{d}}(t)$$

Matching conditions

$$A_{\mathrm{ref}} = A + B\Lambda K_x^{\mathrm{T}}, \quad B_{\mathrm{ref}} = B\Lambda K_r^{\mathrm{T}}$$

Algebraic Lyapunov equation

$$A_{\mathrm{ref}}^{\mathrm{T}}P + PA_{\mathrm{ref}} = -Q$$

Adaptive laws

$$\dot{\hat{K}}_x(t) = \mathrm{Proj}\left(\hat{K}_x(t), -\Gamma_x x(t)e^{\mathrm{T}}(t)PB\right)$$
$$\dot{\hat{K}}_r(t) = \mathrm{Proj}\left(\hat{K}_r(t), -\Gamma_r r_{\mathrm{cmd}}(t)e^{\mathrm{T}}(t)PB\right)$$
$$\dot{\hat{\Theta}}(t) = \mathrm{Proj}\left(\hat{\Theta}(t), \Gamma_\Theta \Phi(t, x(t))e^{\mathrm{T}}(t)PB\right)$$

Reference model

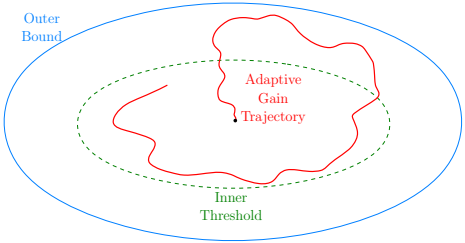
$$\dot{x}_{\mathrm{ref}}(t) = A_{\mathrm{ref}}x_{\mathrm{ref}}(t) + B_{\mathrm{ref}}r_{\mathrm{cmd}}(t)$$

Trajectory tracking error

$$e(t) = x(t) - x_{\mathrm{ref}}(t)$$

Control law

$$u(t) = \hat{K}_x^{\mathrm{T}}(t)x(t) + \hat{K}_r^{\mathrm{T}}(t)r_{\mathrm{cmd}}(t) - \hat{\Theta}^{\mathrm{T}}(t)\Phi(t, x(t))$$



Plant model

$$\dot{x}(t) = Ax(t) + B\Lambda\left[u(t) + \Theta^{\mathrm{T}}\Phi(t, x(t))\right] + \xi_{\mathrm{d}}(t)$$

Auxiliary reference model

$$\dot{e}_{\mathrm{tran}}(t) = A_{\mathrm{tran}} e_{\mathrm{tran}}(t)$$

Matching conditions

$$A_{\mathrm{ref}} = A + B\Lambda K_x^{\mathrm{T}}, \quad B_{\mathrm{ref}} = B\Lambda K_r^{\mathrm{T}},$$
$$A_{\mathrm{tran}} = A_{\mathrm{ref}} + B\Lambda K_g^{\mathrm{T}}$$

Adaptive law

$$\dot{\hat{K}}(t) = \mathrm{Proj}\Big(\hat{K}(t), -\Gamma\mu(\|e(t)\|)H^{-1}(t, \varepsilon(t)) \cdot \left[\pi(t)\varepsilon^{\mathrm{T}}(t)\left(P + V_e(t, \varepsilon(t))M\right)B\right]\Big)$$

Reference model

$$\dot{x}_{\mathrm{ref}}(t) = A_{\mathrm{ref}} x_{\mathrm{ref}}(t) + B_{\mathrm{ref}} r_{\mathrm{cmd}}(t)$$

Trajectory tracking error and ε

$$e(t) \triangleq x(t) - x_{\mathrm{ref}}(t), \qquad \varepsilon(t) \triangleq e(t) - e_{\mathrm{tran}}(t)$$

Control law

$$\pi(t) = \left[x^{\mathrm{T}}(t), r_{\mathrm{cmd}}^{\mathrm{T}}(t), -\Phi^{\mathrm{T}}(t, x(t)), e^{\mathrm{T}}(t)\right]^{\mathrm{T}}$$
$$\hat{K}(t) = \left[\hat{K}_x^{\mathrm{T}}(t), \hat{K}_r^{\mathrm{T}}(t), \hat{\Theta}^{\mathrm{T}}(t), \hat{K}_g^{\mathrm{T}}(t)\right]^{\mathrm{T}}$$
$$u(t) = \hat{K}^{\mathrm{T}}(t)\pi(t)$$

Algebraic Lyapunov equation

$$A_{\mathrm{tran}}^{\mathrm{T}} P + P A_{\mathrm{tran}} = -Q$$

Constraint set

$$\overline{\mathcal{E}}_t \triangleq \{\varepsilon \in \mathbb{R}^n : H(t,\varepsilon) \geq 0\}$$

Constraint function (*hard* constraint)

$$H(t,\varepsilon) \triangleq \underbrace{\eta_{\max} - \eta^2(t)}_{\text{Funnel diameter}} - \varepsilon^{\mathrm{T}} M \varepsilon, \qquad -Q_M = A_{\text{tran}}^{\mathrm{T}} M + M A_{\text{tran}}$$

Barrier Lyapunov function

$$V_e(t,\varepsilon) \triangleq \frac{\varepsilon^{\mathrm{T}} P \varepsilon}{H(t,\varepsilon)}$$

***Example* of function for regulating funnel diameter** (*soft* constraint)

$$\xi(t) = \frac{u_{\max} - \|u(t)\|}{\max\{\|u(t)\| - u_{\min}, \Delta u_{\min}\}}$$

Adaptive law funnel diameter

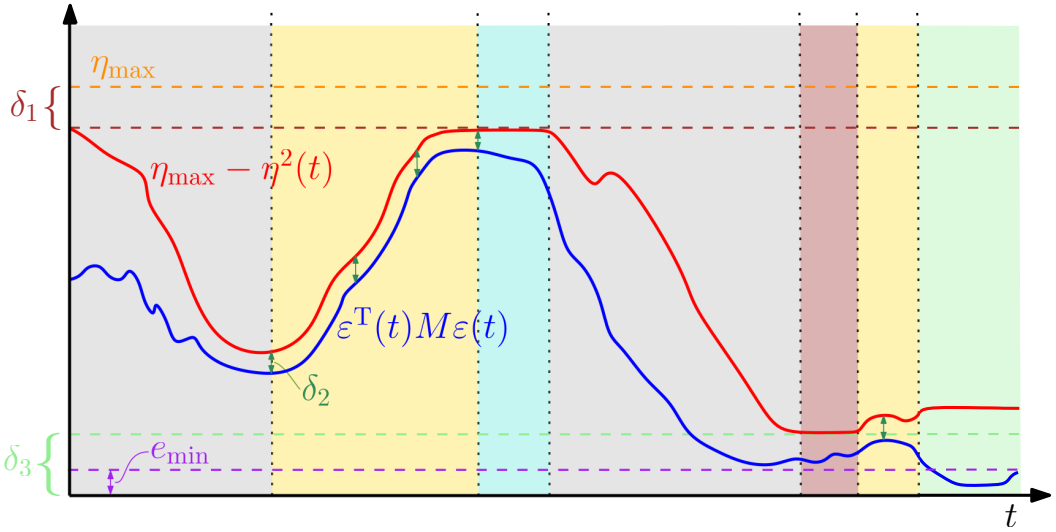
$$\dot{\eta}(t) \triangleq \begin{cases} \min\left\{0,\ -\dfrac{\varepsilon^{\top}(t)\, M\, \dot{\varepsilon}(t)}{\eta(t)}\right\}, & \text{if } \|\varepsilon(t)\| > e_{\min},\ H(t,\varepsilon(t)) = \delta_2,\ \eta(t) \in \left(\sqrt{\delta_1}, \sqrt{\eta_{\max} - \delta_3}\right), \\ \sigma_{\text{nom}}(t), & \text{if } \|\varepsilon(t)\| > e_{\min},\ H(t,\varepsilon(t)) > \delta_2,\ \eta(t) \in \left(\sqrt{\delta_1}, \sqrt{\eta_{\max} - \delta_3}\right), \\ \max\{0,\ \sigma_{\text{nom}}(t)\}, & \text{if } \|\varepsilon(t)\| > e_{\min},\ \eta(t) = \sqrt{\delta_1}, \\ \min\{0,\ \sigma_{\text{nom}}(t)\}, & \text{if } \|\varepsilon(t)\| > e_{\min},\ \eta(t) = \sqrt{\eta_{\max} - \delta_3}, \\ 0, & \text{if } \|\varepsilon(t)\| \leq e_{\min}, \end{cases} \qquad \eta(t_0) = \eta_0$$

where

$$\sigma_{\text{nom}}(t) \triangleq \min\left\{\eta(t)\xi(t), \lambda_{\text{sat}}(t,\varepsilon(t))\right\},$$

$$\lambda_{\text{sat}}(t,\varepsilon) \triangleq \max\left\{0, \frac{H(t,\varepsilon)}{2\lambda_{\max}(P)\eta(t)}\left[\lambda_{\min}\left(Q + V_e(t,\varepsilon)Q_M - \frac{2\bar{\xi}_{\text{d}}}{\|\varepsilon\|}\left(P + V_e(t,\varepsilon)M\right)\right) - \nu\right]\right\}$$

Adaptive Law Funnel Diameter



Numerical Simulations

Choice of the Physics Engine/Simulator

Simulators for UAVs

- AirSim
- FlightGear
- RotorS
- Flightmare
- FlightGoggles
- PyChrono*

Main requirements for choosing a simulator

- Accurately capture dynamics of UAV & payload
- Capture effects such as contact forces and moments, deformable components, and fluid-solid interaction
- Account for geometric and inertia properties of UAV from CAD model
- Real-time rendering
- Open-source

Simulator	Open source	Rendering	Dynamics	Multiple vehicles	FEM	Contact	Collision	Sling payload	UAV oriented	Documentation available
PyChrono	Yes	Irrlicht/ POVray/ OpenGL	Chrono::engine	Yes	Yes	Yes	Yes	Yes	No	Yes
Flightmare	Yes	Unity	Gazebo-based/ Real-world dynamics	Yes	No	No	Yes	No	Yes	No
FlightGoggles	Yes	Unity	multicopterDynamicsSim/ Real-world dynamics	Yes	No	No	Yes	No	Yes	Yes

Simulation Wrapper

Purpose:

- Automates running multiple simulations in sequence
- Enables rapid testing and performance evaluation over a wide envelope of operating conditions

Parameter Variation:

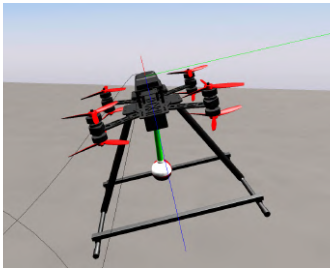
- Control algorithm
- Simulation environment

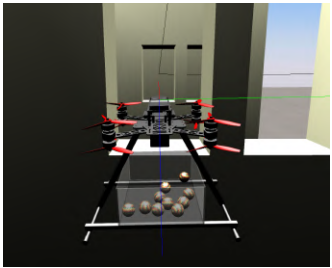
Implementation:

- Built using the PyChrono physics engine
- Can run multiple simulations in parallel, up to the number of threads

Example Hardware:

- CPU: 13th Gen Intel® Core™ i9-13900H
- Cores: 14, Threads: 20

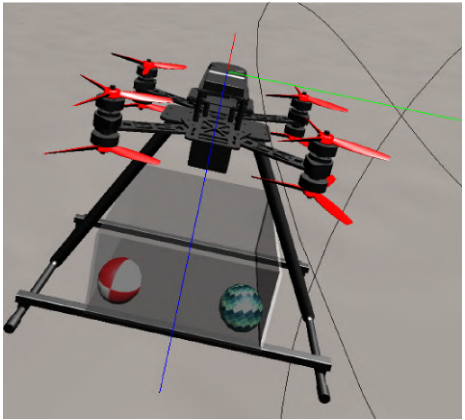




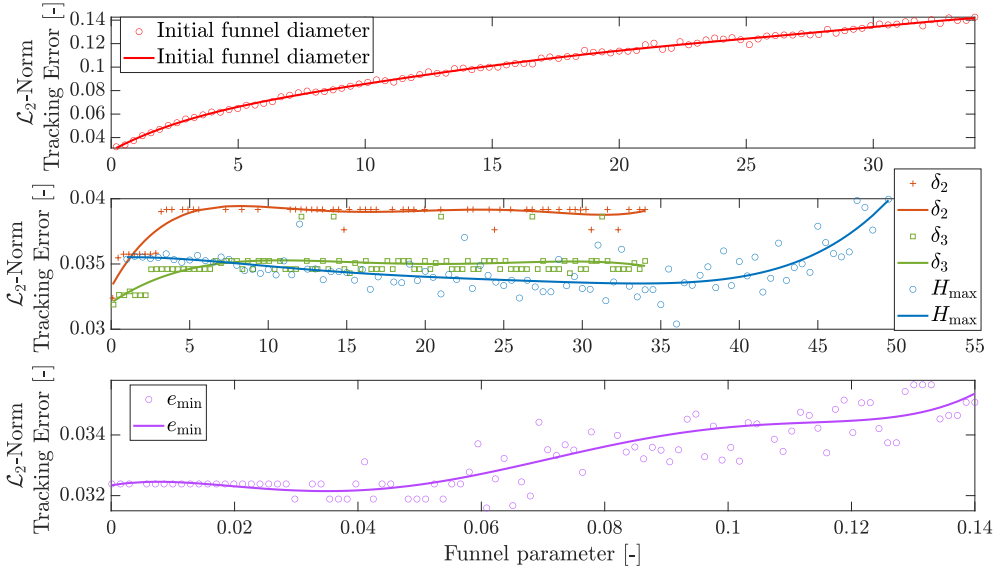
UAV_Sim_PyChrono

- High-fidelity simulations
- PyChrono-based dynamics

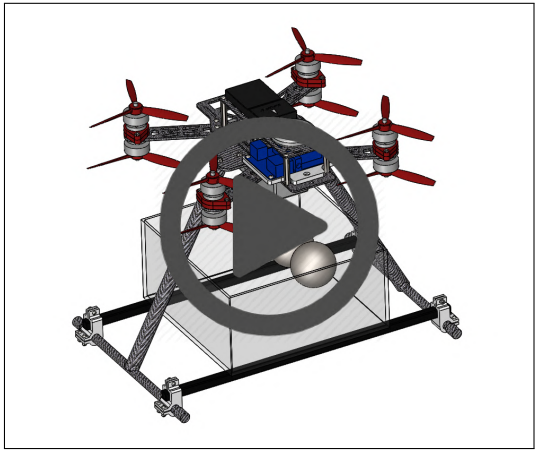
github.com/andrealaffly/UAV_Sim_PyChrono



Numerical Simulations: Sensitivity Analysis



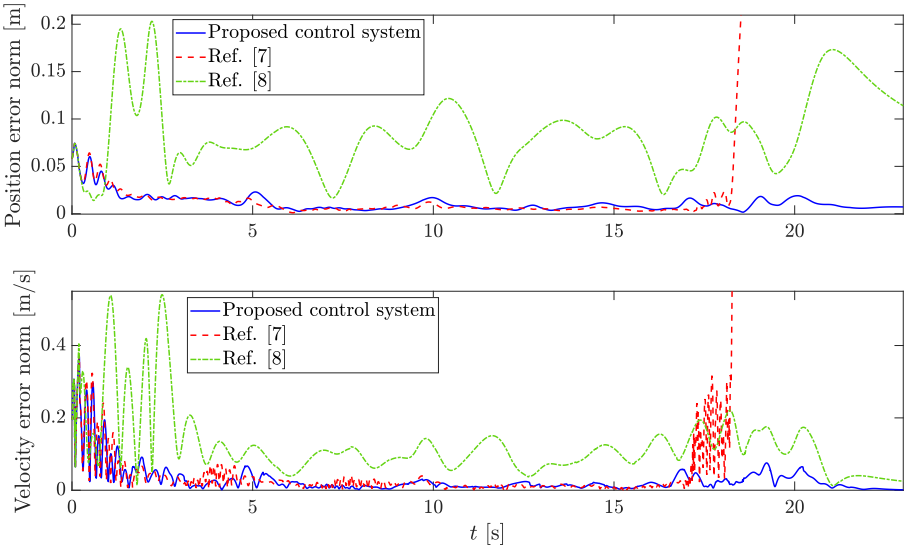
Proposed Control System (Left) – Ref. [7] (Center) – Ref. [8] (Right)



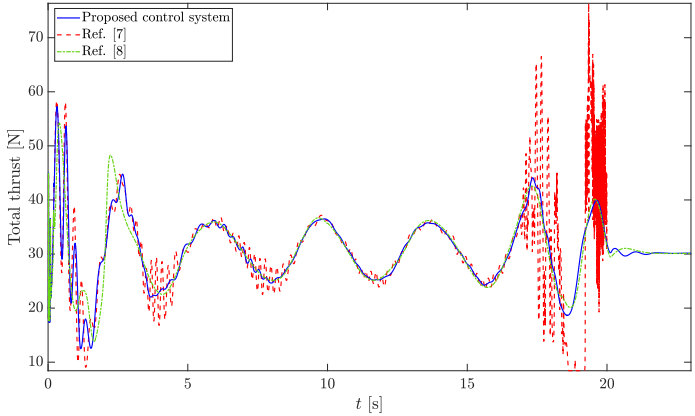
[7] M. Gramuglia, G. M. Kumar, and A. L'Afflitto. *Two-layer Adaptive Funnel MRAC with Applications to the Control of Multi-Rotor UAVs*. 13th IEEE International Workshop on Robot Motion and Control (RoMoCo), Poznań, Poland, July 2024.

[8] M. Gramuglia and A. L'Afflitto. *A Robust Multi-Constraint Funnel MRAC System with Applications to Autonomous Multi-Rotor UAVs Transporting Payloads Connected by Ropes*. AIAA Science and Technology Forum and Exposition (SciTech), Orlando, FL, USA, January 2026.

Numerical Simulations

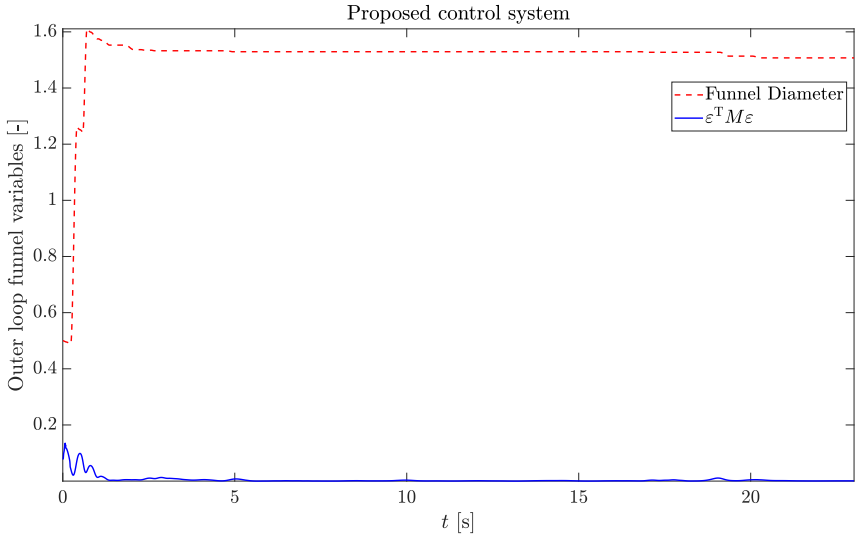


Numerical Simulations: Unknown Unsteady Payload



Controller	$\mathcal{L}_\infty$-norm Tracking Error		$\mathcal{L}_2$-norm Tracking Error		$\mathcal{L}_\infty$-norm Thrust [N$\sqrt{\text{s}}$]	$\mathcal{L}_2$-norm Thrust [N$\sqrt{\text{s}}$]
	Position [m]	Velocity [m/s]	Position [m$\sqrt{\text{s}}$]	Velocity [m/$\sqrt{\text{s}}$]		
Proposed control system	**0.0745**	**0.3728**	**0.0699**	**0.2371**	57.28	149.82
Ref. [7]	∞	∞	∞	∞	∞	∞
Ref. [8]	0.2031	0.5405	0.4466	0.7227	**54.21**	**149.55**

Numerical Simulations: Unknown Unsteady Payload



Experimental Flight Tests

Features:

- Modular
- Easy to implement new control algorithms
- Option to use either RTK-GPS, MoCap, or VIO fusion

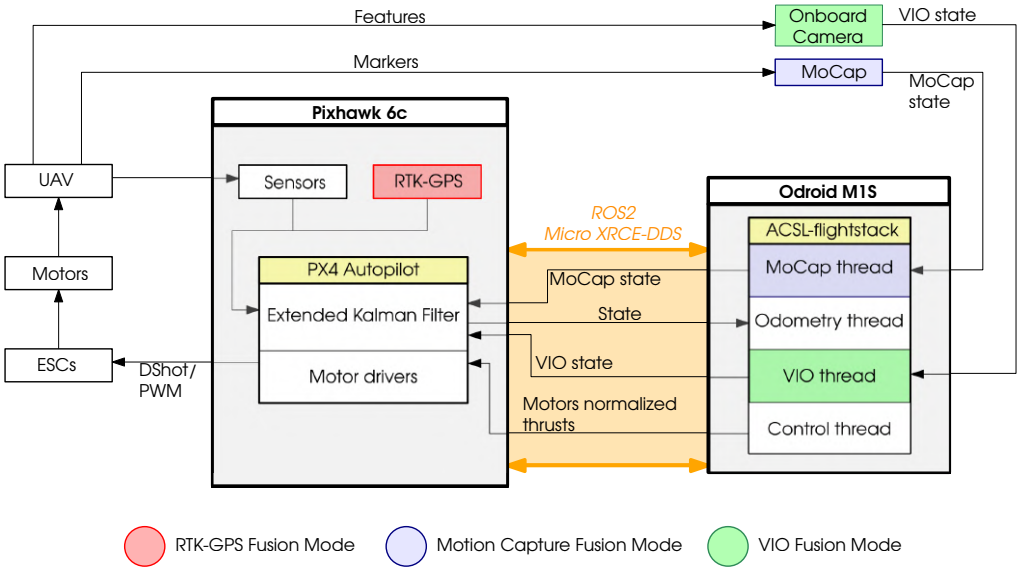
Tested with:

- Odroid M1S (8GB RAM, 4 Core 4 Thread processor)
- Pixhawk 6c
- PX4 Autopilot v1.15
- VICON and RTK-GPS H-RTK ZED-F9P
- Ubuntu 20.04
- ROS2 Galactic

Figure: Odroid M1S

Figure: Pixhawk 6c

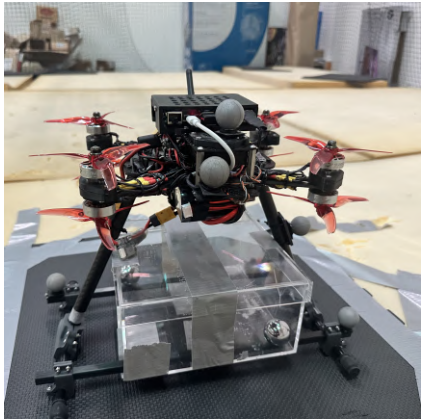
Flight Stack Architecture & Interface with PX4



ACSL-flightstack

- Flight tests
- Real UAV deployment

github.com/andrealaffly/ACSL-flightstack

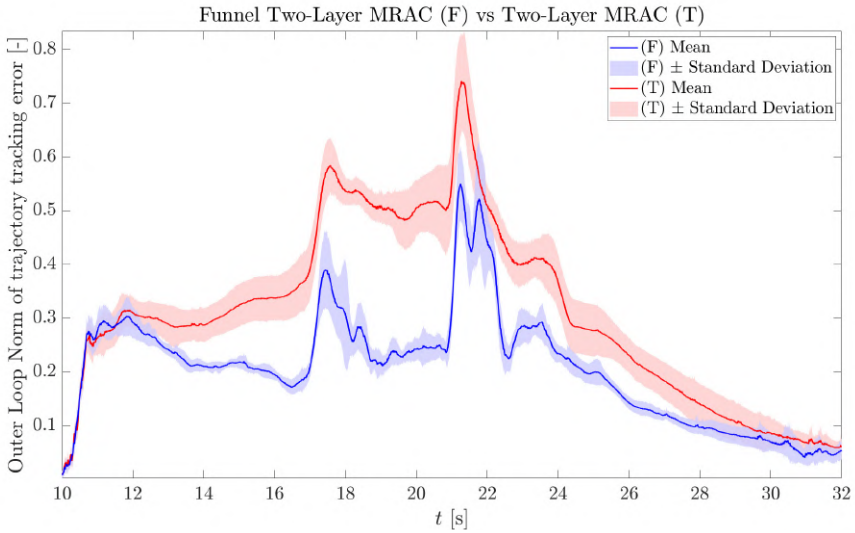


Experimental Flight Tests: Nominal vs Off-Nominal Conditions

Nominal Conditions – Funnel Two-Layer MRAC (Selected Flight Tests)

Nominal Conditions – Classical Two-Layer MRAC (Selected Flight Tests)

Experimental Flight Tests: Nominal Conditions

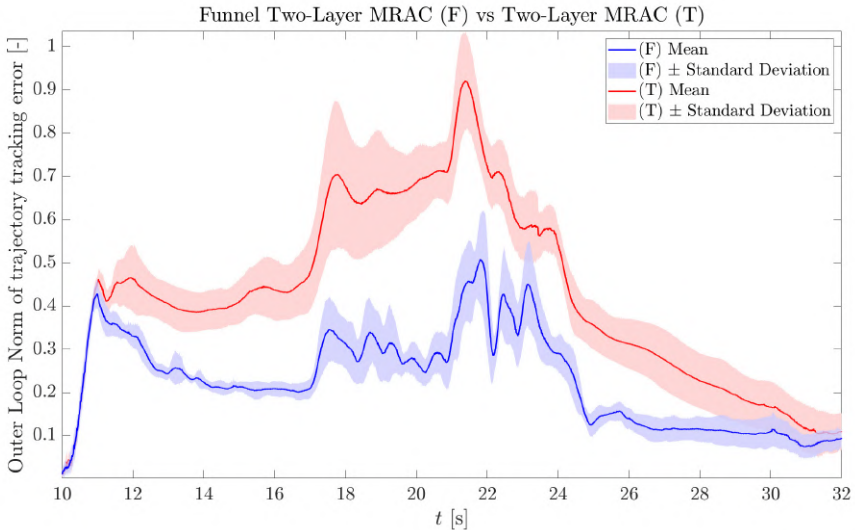


Performance computed over all 20 flights executed for the nominal conditions mission.

Off-Nominal Conditions – Funnel Two-Layer MRAC (Selected Flight Tests)

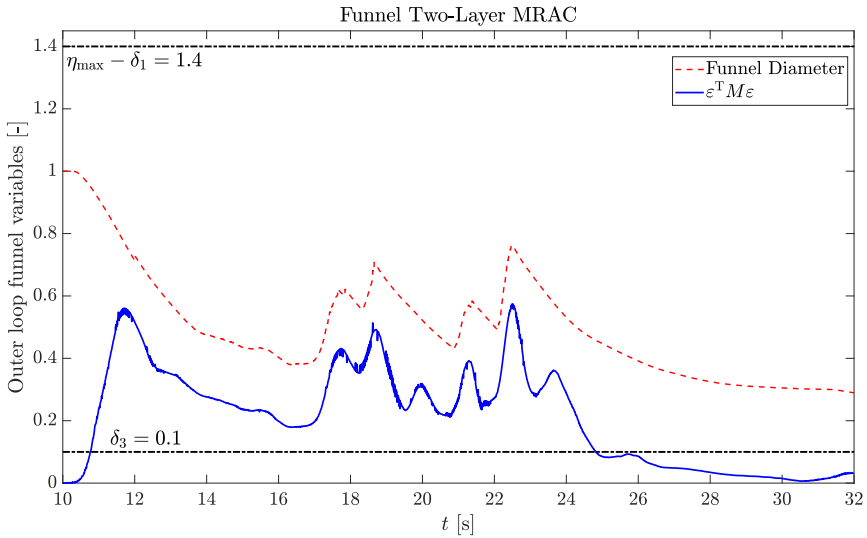
Off-Nominal Conditions – Classical Two-Layer MRAC (Selected Flight Tests)

Experimental Flight Tests: Off-Nominal Conditions

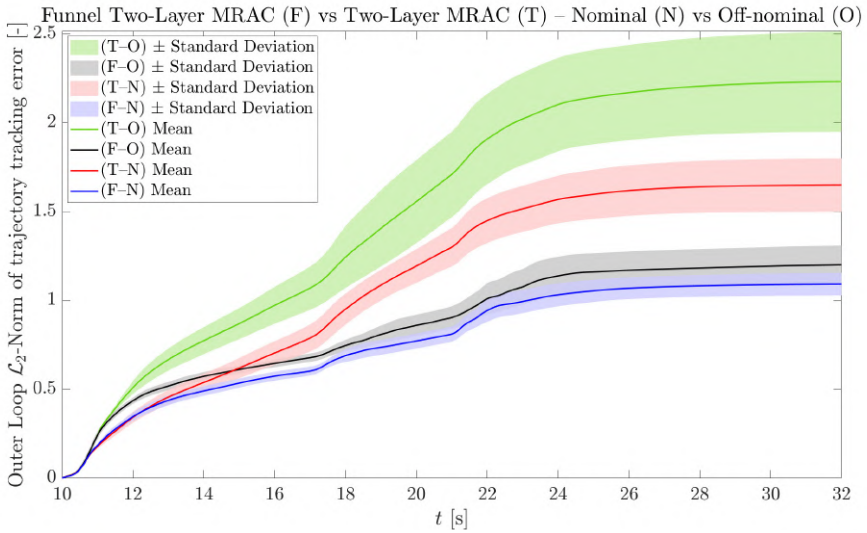


Performance computed over all 20 flights executed for the off-nominal conditions mission.

Experimental Flight Tests: Off-Nominal Conditions



Experimental Flight Tests: Nominal vs Off-Nominal Conditions



Performance computed over all 40 flights executed for both the nominal and off-nominal conditions missions.

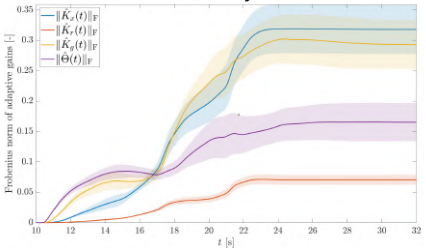
Experimental Flight Tests: Nominal vs Off-Nominal Conditions

Mission	Controller	$\mathcal{L}_2$-norm		Average		Maximum	
		Mean($\|e(\cdot)\|_{\mathcal{L}_2}$)	Std($\|e(\cdot)\|_{\mathcal{L}_2}$)	Mean($\|e(\cdot)\|$)	Std($\|e(\cdot)\|$)	Mean($\|e(\cdot)\|$)	Std($\|e(\cdot)\|$)
1	Funnel Two-Layer MRAC	**1.1044**	**0.0612**	**0.1936**	**0.0273**	**0.5501**	0.2321
	Classical Two-Layer MRAC	1.6624	0.1455	0.2944	0.0419	0.7408	**0.1984**
Improvement (vs Classical) [%]		**33.6%**	**57.9%**	**34.2%**	**34.8%**	**25.7%**	**-17.0%**
2	Funnel Two-Layer MRAC	**1.2199**	**0.1027**	**0.2136**	**0.0421**	**0.5077**	**0.2372**
	Classical Two-Layer MRAC	2.2471	0.2825	0.4070	0.0686	0.9215	0.2410
Improvement (vs Classical) [%]		**45.7%**	**63.6%**	**47.5%**	**38.6%**	**44.9%**	**1.6%**

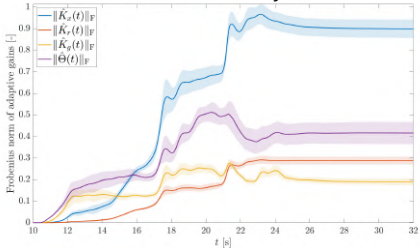
Performance computed over all 40 flights executed for both the nominal and off-nominal conditions missions.

Frobenius Norm of the Outer-Loop Adaptive Gains

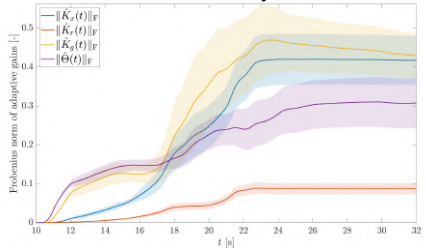
Nominal: Two-Layer MRAC



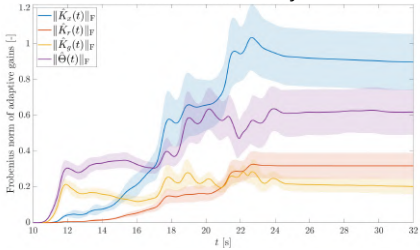
Nominal: Funnel Two-Layer MRAC



Off-Nominal: Two-Layer MRAC



Off-Nominal: Funnel Two-Layer MRAC



Thank you for your attention

For more information, please
contact Andrea L'Afflitto (a.lafflitto@vt.edu)
or visit **https://lafflitto.com**

ACKNOWLEDGMENTS

This work was partly supported through the US Department of the Navy through the grant no. N004212520003.

Software Repositories

ACSL-flightstack

- Flight tests
- Real UAV deployment

`github.com/andrealaffly/ACSL-flightstack`

UAV_Sim_PyChrono

- High-fidelity simulations
- PyChrono-based dynamics

`github.com/andrealaffly/UAV_Sim_PyChrono`