



Verification of neural network-controlled bioinspired drones

UK Robotics Summer School 2025

Colin Kessler
Heriot-Watt University, University of Edinburgh

Verification of neural network-controlled bioinspired drones



Background

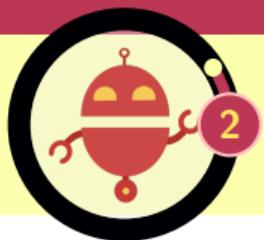
Verification Setup

Vehicle/Marabou Implementation

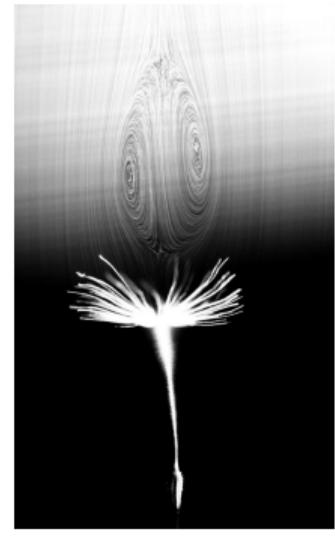
CORA Implementation

Conclusions

Motivation - gliding seeds



- ▶ Certain plant species (e.g. *Taraxacum* and *Alsomitra*) use aerodynamics to disperse seeds over large distances [1,2]
- ▶ Flying robots with similar characteristics could be very useful for environmental monitoring and meteorology

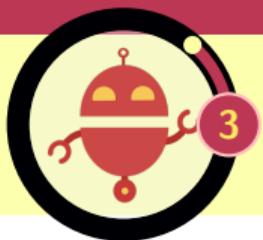


[1]



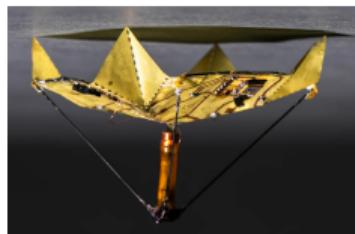
[2]

1. Cummins et al, 2018. *A separated vortex ring underlies the flight of the dandelion*. Nature
2. Certini, 2023. *The flight of Alsomitra macrocarpa*. Phd thesis, University of Edinburgh

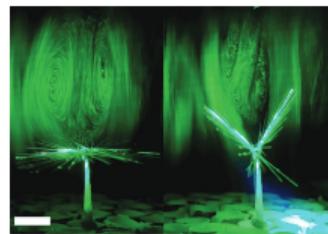


Gliding seed inspired robots

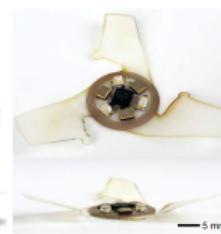
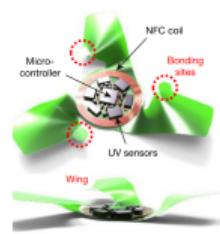
Actuation, sensing, and control methods are the focus of other research:



[3]



[4]

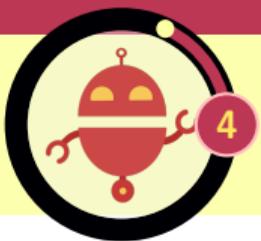


[5]

We will focus on modelling as a basis for verification

3. Johnson et al, 2023. *Solar-powered shape-changing origami microfliers*. Science Robotics
4. Yang et al, 2022. *Dandelion-Inspired, Wind-Dispersed Polymer-Assembly Controlled by Light*. Advanced Science
5. Kim et al, 2021. *Three-dimensional electronic microfliers inspired by wind-dispersed seeds*. Nature

Alsomitra Macrocarpa

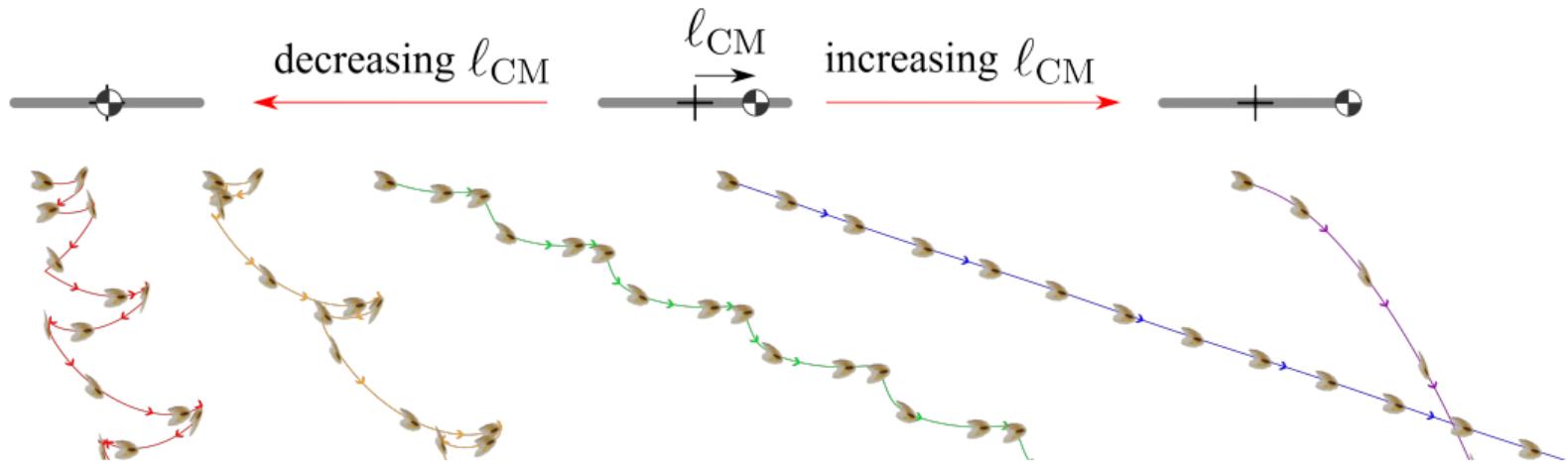


[2]



Modelling *Alsomitra*

Alsomitra seed flight can be considered 2D, and is affected by the **CoM position**:



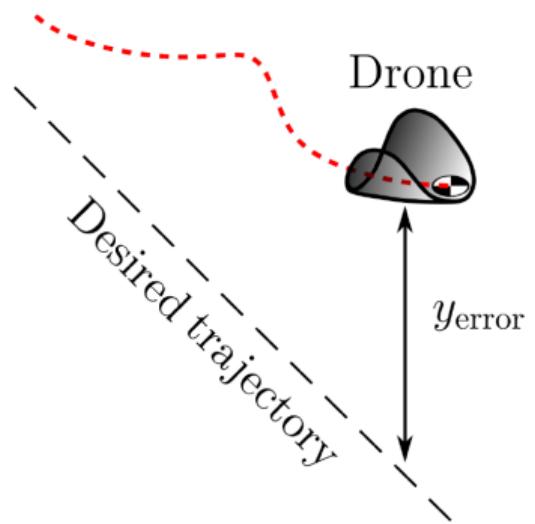
We can therefore employ a **quasi-steady low-order model [6]**, using a controller to actuate the CoM position

6. Li et al, 2022. *Centre of mass location, flight modes, stability and dynamic modelling of gliders*. Journal of Fluid Mechanics



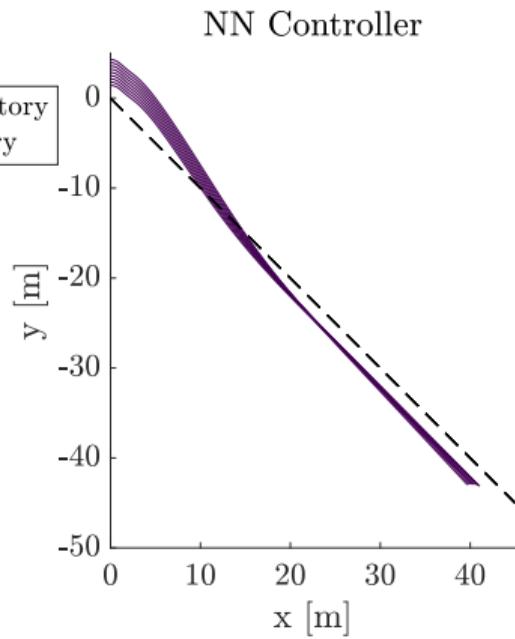
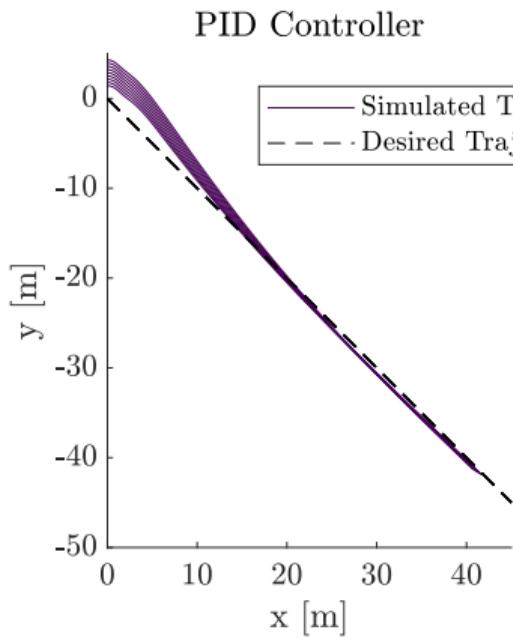
Control Problem

Our 2D *Alsomitra* model is used as the basis of a feedback control system. For verification, we consider a basic trajectory-following control problem:





Controlling Alsomitra



Verification of neural network-controlled bioinspired drones



Background

Verification Setup

Vehicle/Marabou Implementation

CORA Implementation

Conclusions

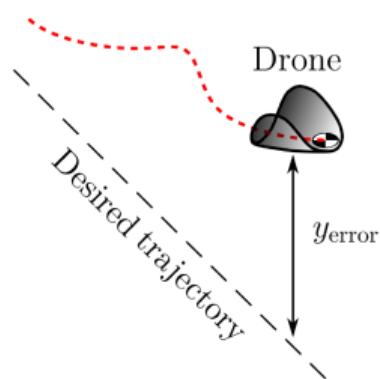


Ideal Verification Goal

For any starting state $x_1, \dots, x_6(0)$, after some time t^* the trajectory of the drone will always be within some small distance y^* of the target trajectory ($x_6 = -x_5$):

$$\forall t \geq t^*, \forall x_1, \dots, x_6(0) \in \mathbb{R} : |x_6(t) + x_5(t)| \leq y^* \quad (1)$$

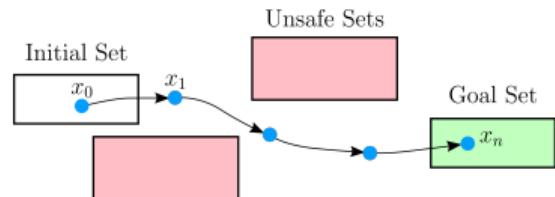
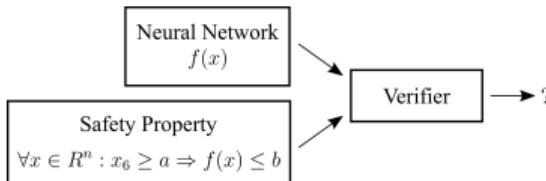
However, this is too complex for current tools to handle





Verification Approaches

	Neural Network Verification	Reachability Verification
Tools	Marabou, $\alpha\beta$ -CROWN, PyRAT	CORA, JuliaReach, POLAR-Express
Benchmarks	VNN-COMP	ARCH-COMP AINNCS
Advantages	can interrogate large parameter space	dynamics considered
Disadvantages	dynamics not considered	finite time, limited complexity





Lipschitz Robustness

We focus on two notion of robustness, *standard* and *Lipschitz* robustness. Given $x^* \in \mathcal{D}$ and constants $\epsilon, \delta, L \in \mathbb{R}$,

$$\forall x \in R^n : \|x - x^*\| \leq \epsilon \implies \|f(x) - f(x^*)\| \leq \delta \quad (2)$$

$$\forall x \in R^n : \|x - x^*\| \leq \epsilon \implies \|f(x) - f(x^*)\| \leq L \|x - x^*\| \quad (3)$$

Since the latter has been proven to be strictly stronger in [7], we implemented a form of PGD training with a Lipschitz loss function, and later verify a version of standard robustness.

7. Casadio et al, 2022. *Neural network robustness as a verification property: A principled case study*. Computer Aided Verification



Verification of neural network-controlled bioinspired drones

Background

Verification Setup

Vehicle/Marabou Implementation

CORA Implementation

Conclusions



Vehicle setup

► Global safety properties

1. If the drone is above the line by some threshold y^* , the network output will always make the drone pitch down

$$x_6 \geq -x_5 + y^* \Rightarrow f(x) \geq 0.187 \quad (4)$$

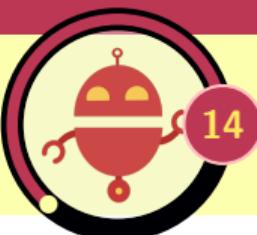
2. If the drone is below the line by some threshold y^* , the network output will always make the drone pitch up

$$x_6 \leq -x_5 - y^* \Rightarrow f(x) \leq 0.187 \quad (5)$$

► Local (pseudo Lipschitz) robustness

5. For any given input point x , the network output $f(x^*)$ of any perturbed point x^* within an ϵ -ball around x , will have a distance less than or equal to $L^*\epsilon$ to $f(x)$

$$\forall x \in R^n : \|x - x^*\| \leq \epsilon \implies \|f(x) - f(x^*)\| \leq L^*\epsilon \quad (6)$$



Vehicle Implementation - Property 1

1. If the drone is above the line by some threshold y^* , the network output will always make the drone pitch down

$$x_6 \geq -x_5 + y^* \Rightarrow f(x) \geq 0.187$$

```
droneFarAboveLine : UnnormalisedInputVector -> Bool
droneFarAboveLine x =
    x ! d_y >= - x ! d_x + ystar

@property
property1 : Bool
property1 = forall x . validInput x and droneFarAboveLine x =>
    alsomitra x ! e_x >= 0.187
```



Vehicle Results - Global Properties

Critical y^* values for Properties 1&2, for naive and adversarially trained networks. A lower y^* indicates a controller that better adheres to the target trajectory:

Property	Naive	Adversarial
1	46	30
2	42	42

$$x_6 \geq -x_5 + y^* \Rightarrow f(x) \geq 0.187$$

$$x_6 \leq -x_5 - y^* \Rightarrow f(x) \leq 0.187$$

Network performance is generally poor, although adversarial training improves P1 result, which could be further improved with PDT.



Vehicle Implementation - Local Robustness

5. For any given input point x , the network output $f(x^*)$ of any perturbed point x^* within an ϵ -ball around x , will have a distance less than or equal to $L^*\epsilon$ to $f(x)$

$$\forall x \in R^n : \|x - x^*\| \leq \epsilon \implies \|f(x) - f(x^*)\| \leq L^*\epsilon$$

```

boundedByEpsilon : InputVector -> Bool
boundedByEpsilon x = forall i in myList . -epsilon <= x ! i - x ! i + 6 <= epsilon

standardRobustness : InputVector -> OutputVector -> Bool
standardRobustness input output = forall perturbation .
    let perturbedInput = input - perturbation in validPerturbation perturbation and
    validInput perturbedInput and boundedByEpsilon perturbedInput =>
    (output ! e_x - alsomitra perturbedInput ! e_x2) <= Lipschitz * epsilon and
    alsomitra perturbedInput ! e_x2 - output ! e_x <= Lipschitz * epsilon

@property
property4 : Vector Bool n
property4 = foreach i . standardRobustness (trainingInputs ! i) (trainingOutputs ! i)

```

*requires onnx workaround to double network.



Vehicle Results - Local Robustness

Verification success rates (%) of Property 5 for naive and adversarial networks, per L^* values and ϵ , with respect to the training dataset:

		$L^*\epsilon$						$L^*\epsilon$			
		10^{-3}	10^{-2}	10^{-1}	10^0			10^{-3}	10^{-2}	10^{-1}	10^0
Naive						Adversarial					
ϵ	10^{-5}				100	ϵ	10^{-5}				100
	10^{-4}			96.1	100		10^{-4}			99.7	100
	10^{-3}		17.0	98.5	100		10^{-3}		23.9	99.7	100
	10^{-2}	-	-	-	100		10^{-2}	0	13.6	92.5	100

$$\forall x \in R^n : \|x - x^*\| \leq \epsilon \implies \|f(x) - f(x^*)\| \leq L^*\epsilon$$

Network performance mostly degrades with decreasing L^* (stricter gradient threshold) and increasing ϵ (larger region to find perturbations). **Adversarial training improves robustness.**

Verification of neural network-controlled bioinspired drones



Background

Verification Setup

Vehicle/Marabou Implementation

CORA Implementation

Conclusions



Reachability Goal

Ideal Verification Goal:

$$\forall t \geq t^*, \forall x_1, \dots, x_6(0) \in \mathbb{R} : |x_6(t) + x_5(t)| \leq y^*$$

With a reachability tool like CORA, we can verify the entire system - but **only for a small initial region and limited time horizon**. We define the initial set identically to the training simulations:

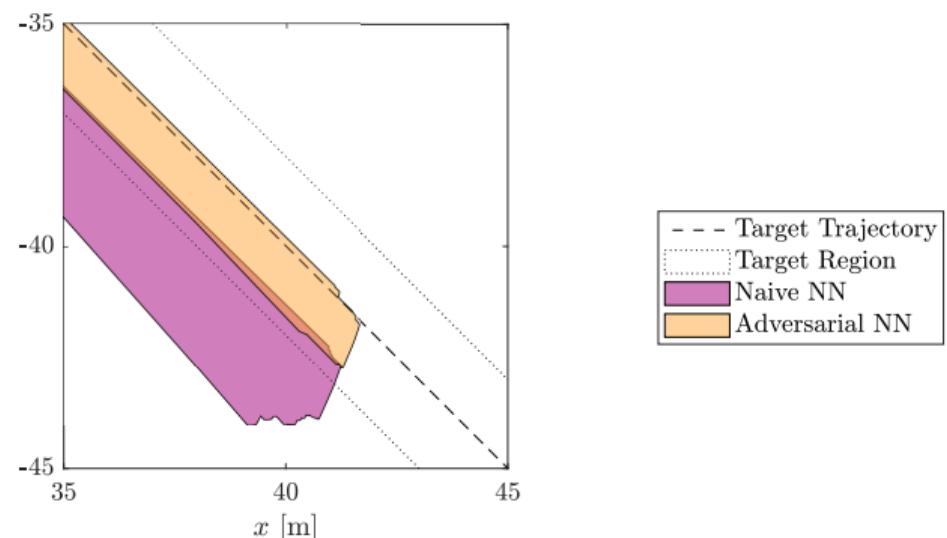
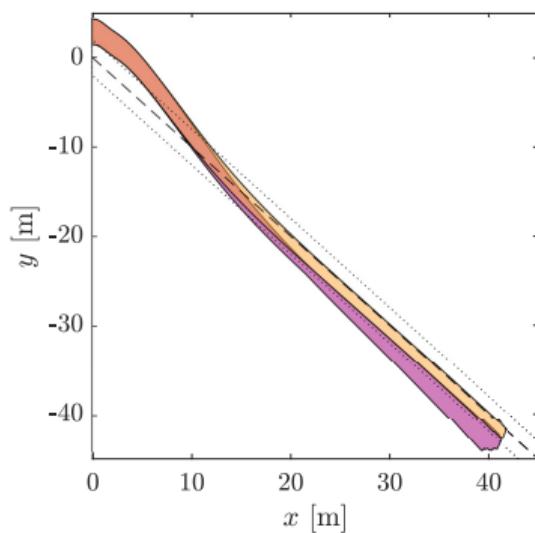
$$x_1 = 1, x_2 = 0, x_3 = 0, x_4 = 0, x_5 = 0, x_6 \in [1.43, 4.29] \quad (7)$$

The goal is for the drone to always be within a distance $y^* = 2$ of the target trajectory after 20 s.



Reachability Result

Reachable regions in x_5 and x_6 for naive and adversarial networks:



*requires simplified equations, divided initial set, and onnx normalisation workaround.



Verification of neural network-controlled bioinspired drones

Background

Verification Setup

Vehicle/Marabou Implementation

CORA Implementation

Conclusions



Conclusions

Key takeaways:

- ▶ Gliding seed-inspired robots are a novel application for verification
- ▶ Verification of such neural-network controlled robots has been demonstrated using two approaches:
 1. Neural network (controller) verification with [Vehicle/Marabou](#)
 2. Full system reachability with [CORA](#)
- ▶ Both approaches are limited in terms of [complexity, scalability, and ease of use](#)
- ▶ Our results give insights into [controller safety, robustness performance, and the effects of adversarial training](#)
- ▶ Developing new technologies with safety in mind is a worthwhile endeavour