ADLR Final Presentation: Development of a reinforcement learning based controller for a VTOL drone.

Christopher Narr & Oliver Hausdörfer

Supervisor: Finn Süberkrüb

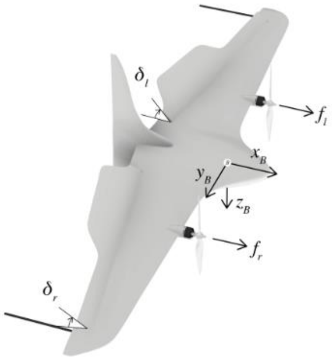
TUM / DLR Institute of Robotics and Mechatronics

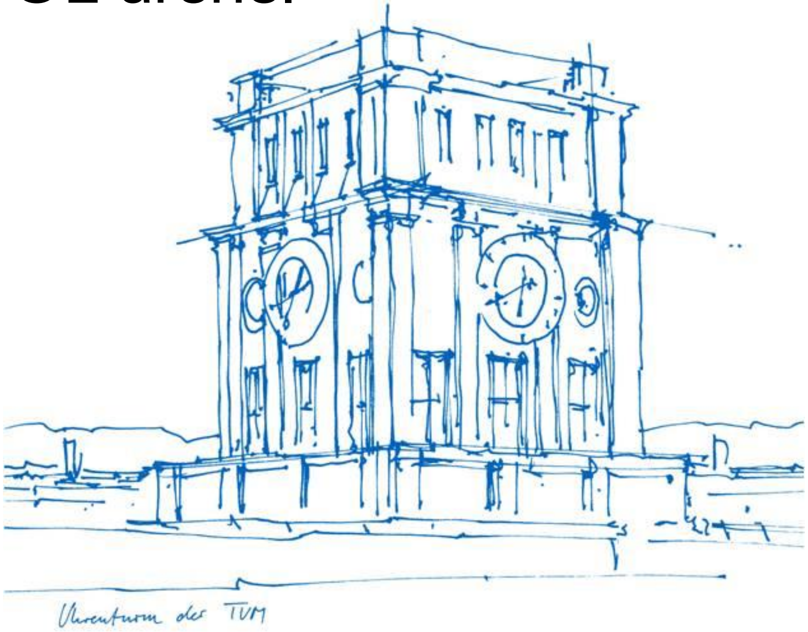
Advanced Deep Learning for Robotics

23rd February 2023

c.narr@tum.de

oliver.hausdoerfer@tum.de





Agenda

Topic

Main results

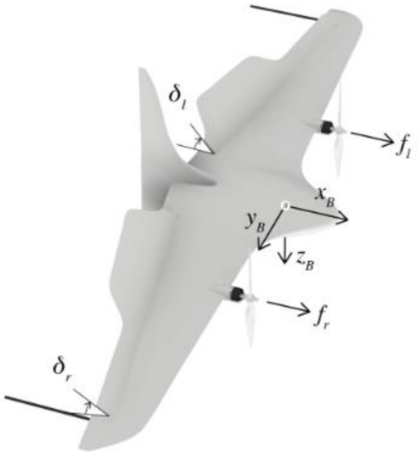
Deep dives

1. Topic

Developing a RL-based velocity-tracking controller for VTOL (*Vertical Take Off and Landing*) **drones**

Our contributions: + Model time-dependencies
+ Use active wing flaps
+ Incorporate gusty conditions

Midterm status:
- Good tracking of simple trajectories
- Using FNN for actor network
- **ToDo**: model time dependencies



1. Topic

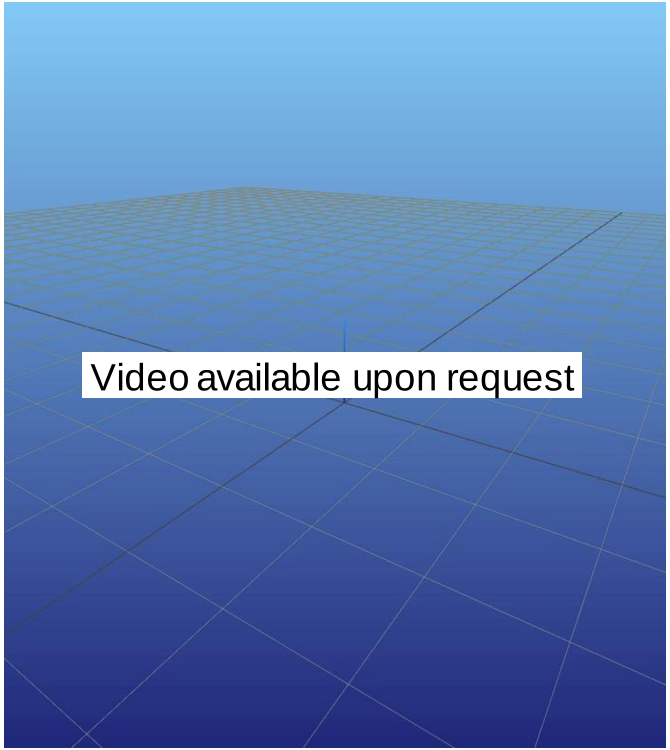
Midterm status

RL-Algorithm: PPO

State space (9D): rotation [x,y,z]
rotational velocity [x,y,z]
linear velocity error [x,y,z]

Action space (4D): 2x thrusts, 2x active flaps

Reward: $r = \underbrace{4.0}_{\text{stay alive}} - 0.5 * \underbrace{||\boldsymbol{v_d} - \boldsymbol{v_t}||_1}_{\text{tracking error}} - 0.01 * \underbrace{||\boldsymbol{\omega_B}||_1}_{\text{rotational velocity}} - 0.01 * \underbrace{||\boldsymbol{a}||_1}_{\text{actuation}}$

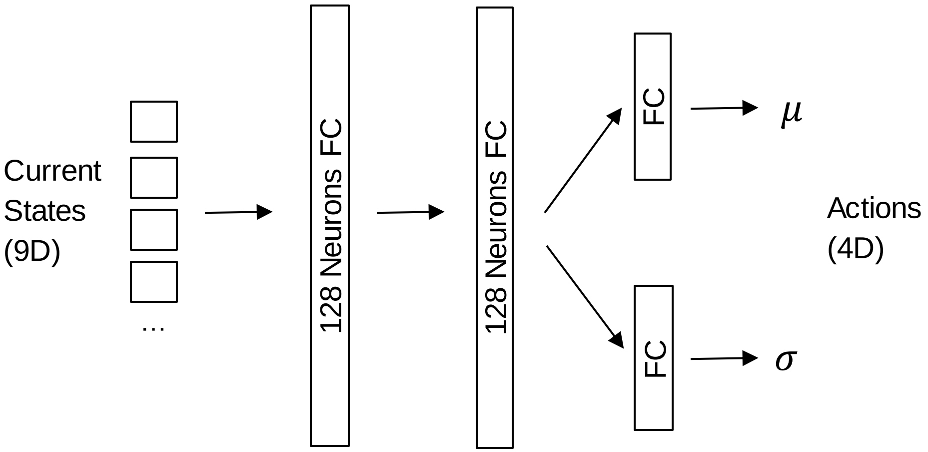


Blue: target velocity | Yellow: actual velocity | Red: thrusts

1. Topic

Midterm status: Actor Network

FNN (Baseline)

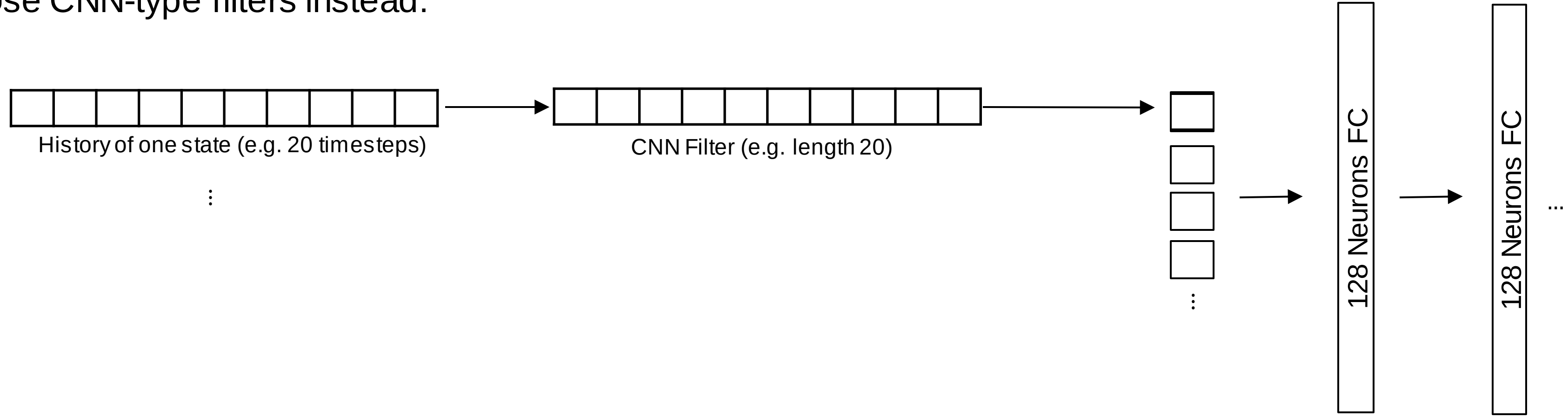


1. Topic

Initial Idea: Use RNN in actor network for modelling time-dependency

But: RNN for RL not supported in Julia “ReinforcementLearning” Library GitHub Issue #144

Use CNN-type filters instead:



Agenda

Topic

Main results

Deep dives

2. Main results

Performance of network architectures on test set of 100 envs

Network	**Reward**	**Tracking error (m/s)**	**# early term. envs**	**# train-able params**
FNN (Baseline)	2696	1.214	4	36.7k
Shared filters (classical CNN)				
1 Filter $\rightarrow$ FNN	2664	1.231	2	36.7k
3 Filters $\rightarrow$ FNN	2531	1.586	4	41.1k
5 Filters $\rightarrow$ FNN	2691	1.158	3	46.1k
9 Filters $\rightarrow$ FNN	2714	1.138	2	55.5k
18 Filters $\rightarrow$ FNN	2820	0.899	1	76.6k

Result

State history significantly improves performance

2. Main results

Performance of network architectures on test set of 100 envs

Network	**Reward**	**Tracking error (m/s)**	**# early term. envs**	**# train-able params**
FNN (Baseline)	2696	1.214	4	36.7k
Shared filters (classical CNN)				
1 Filter → FNN	2664	1.231	2	36.7k
3 Filters → FNN	2531	1.586	4	41.1k
5 Filters → FNN	2691	1.158	3	46.1k
9 Filters → FNN	2714	1.138	2	55.5k
18 Filters → FNN	2820	0.899	1	76.6k
State dependent filters				
2*9 Filters → FNN	2924	0.62	0	39.1k
4*9 Filters → FNN	2951	0.56	0	45.1k

Result

State history significantly improves performance

Agenda

Topic

Main results

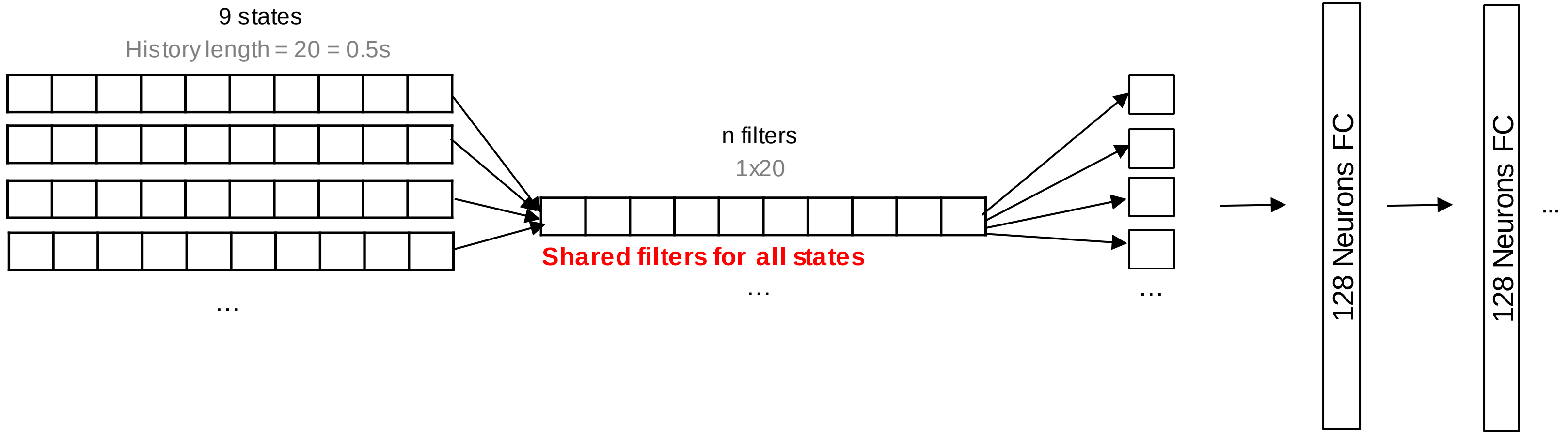
Deep dives

3. Deep Dive

Network	Reward	Tracking error (m/s)	# early term. envs	# trainable params
FNN (Baseline)	2696	1.214	4	36.7k
Shared filters (classical CNN)				
1 Filter → FNN	2664	1.231	2	36.7k
3 Filters → FNN	2531	1.586	4	41.1k
5 Filters → FNN	2691	1.158	3	46.1k
9 Filters → FNN	2714	1.138	2	55.5k
18 Filters → FNN	2820	0.899	1	76.6k

Modelling time dependencies: Shared filters (classical CNN)

Implementation

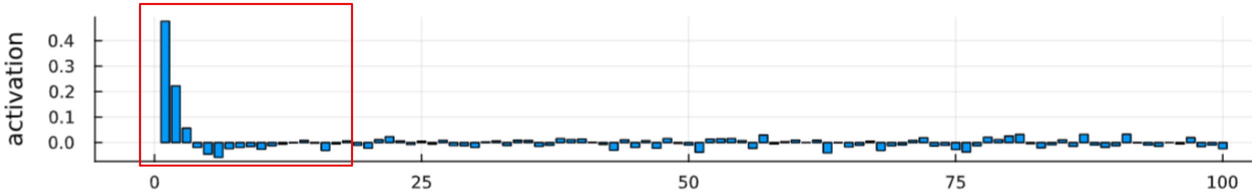


3. Deep Dive

Modelling time dependencies: Shared filters (classical CNN)

Analyzing filter length

Network	**Reward**	**Tracking error (m/s)**	**# early term. envs**	**# train-able params**
FNN (Baseline)	2696	1.214	4	36.7k
Shared filters (classical CNN)				
1 Filter → FNN	2664	1.231	2	36.7k
3 Filters → FNN	2531	1.586	4	41.1k
5 Filters → FNN	2691	1.158	3	46.1k
9 Filters → FNN	2714	1.138	2	55.5k
18 Filters → FNN	2820	0.899	1	76.6k



Activation of weights of trained CNN filter with length 100 (=2.5s)

Reward : 2415 | Tracking Error: 1.7m/s

Result

Only short-term dependencies relevant

3. Deep Dive

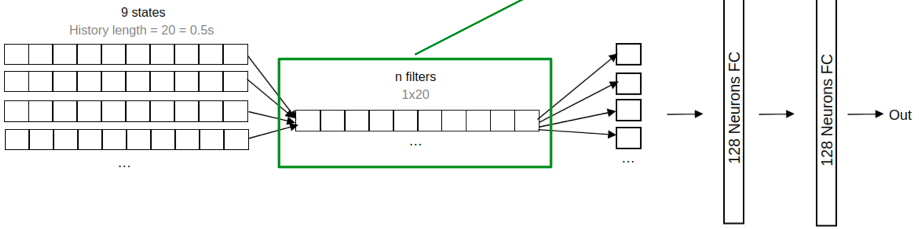
Modelling time dependencies: Shared filters (classical CNN)

Analyzing filter number

Network	Reward	Tracking error (m/s)	# early term. envs	# train-able params
FNN (Baseline)	2696	1.214	4	36.7k
Shared filters (classical CNN)				
1 Filter $\rightarrow$ FNN	2664	1.231	2	36.7k
3 Filters $\rightarrow$ FNN	2531	1.586	4	41.1k
5 Filters $\rightarrow$ FNN	2691	1.158	3	46.1k
9 Filters $\rightarrow$ FNN	2714	1.138	2	55.5k
18 Filters $\rightarrow$ FNN	2820	0.899	1	76.6k

Low filter number reduces performance

Result: Increase filter number



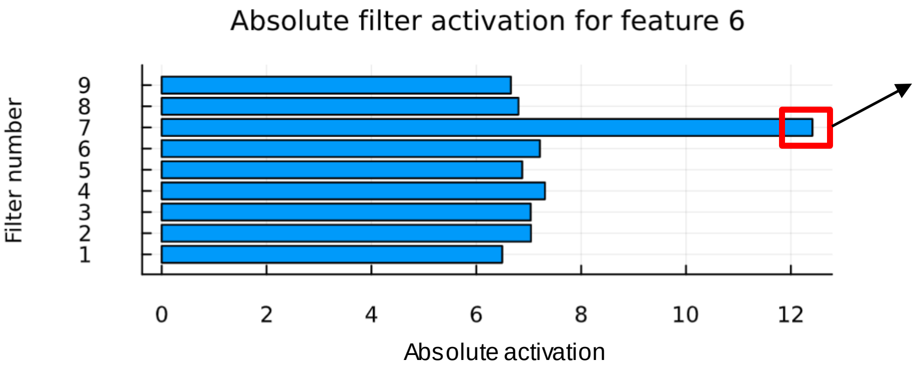
3. Deep Dive

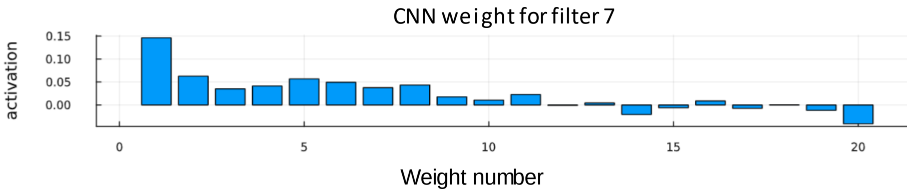
Network	Reward	Tracking error (m/s)	# early term. envs	# train-able params
FNN (Baseline)	2696	1.214	4	36.7k
Shared filters (classical CNN)				
1 Filter → FNN	2664	1.231	2	36.7k
3 Filters → FNN	2531	1.586	4	41.1k
5 Filters → FNN	2691	1.158	3	46.1k
9 Filters → FNN	2714	1.138	2	55.5k
18 Filters → FNN	2820	0.899	1	76.6k

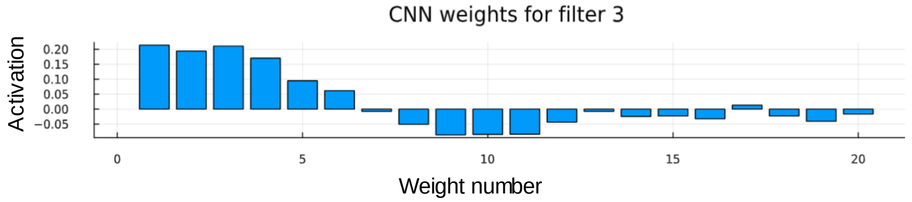
Modelling time dependencies: Shared filters (classical CNN)

Analyzing filter activation (Take with grain of salt!)

Example feature: rotational velocity around z







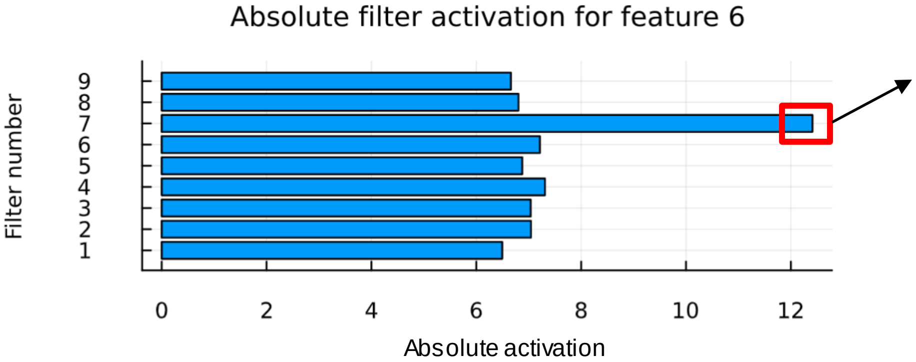
3. Deep Dive

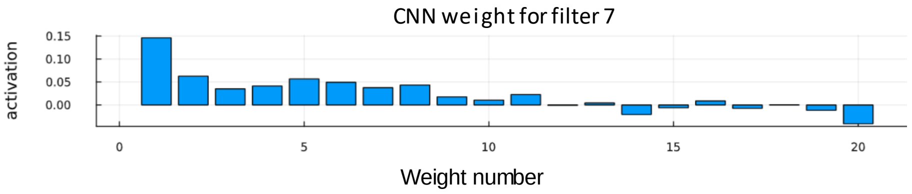
Network	Reward	Tracking error (m/s)	# early term. envs	# train-able params
FNN (Baseline)	2696	1.214	4	36.7k
Shared filters (classical CNN)				
1 Filter → FNN	2664	1.231	2	36.7k
3 Filters → FNN	2531	1.586	4	41.1k
5 Filters → FNN	2691	1.158	3	46.1k
9 Filters → FNN	2714	1.138	2	55.5k
18 Filters → FNN	2820	0.899	1	76.6k

Modelling time dependencies: Shared filters (classical CNN)

Analyzing filter activation (Take with grain of salt!)

Example feature: rotational velocity around z





Results

Filters are selected

Filters might be state dependent

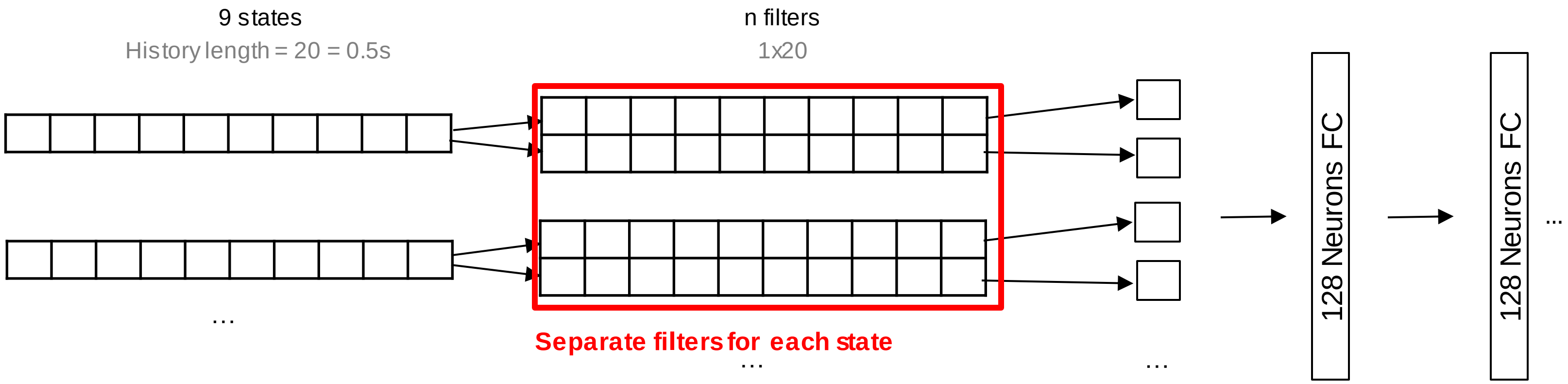
Filters are 'meaningful'

3. Deep Dive

State dependent filters				
2*9 Filters → FNN	2924	0.62	0	39.1k
4*9 Filters → FNN	2951	0.56	0	45.1k

Modelling time dependencies: State dependent filters

Implementation



3. Deep Dive

Modelling time dependencies: State dependent filters

Network	Reward	Tracking error (m/s)	# early term. envs	# train-able params
FNN (Baseline)	2696	1.214	4	36.7k
Shared filters (classical CNN)				
1 Filter → FNN	2664	1.231	2	36.7k
3 Filters → FNN	2531	1.586	4	41.1k
5 Filters → FNN	2691	1.158	3	46.1k
9 Filters → FNN	2714	1.138	2	55.5k
18 Filters → FNN	2820	0.899	1	76.6k
State dependent filters				
2*9 Filters → FNN	2924	0.62	0	39.1k
4*9 Filters → FNN	2951	0.56	0	45.1k

Results

Better performance

Less parameters

3. Deep Dive

Modelling time dependencies: State dependent filters

Network	Reward	Tracking error (m/s)	# early term. envs	# trainable params
FNN (Baseline)	2696	1.214	4	36.7k
Shared filters (classical CNN)				
1 Filter → FNN	2664	1.231	2	36.7k
3 Filters → FNN	2531	1.586	4	41.1k
5 Filters → FNN	2691	1.158	3	46.1k
9 Filters → FNN	2714	1.138	2	55.5k
18 Filters → FNN	2820	0.899	1	76.6k
State dependent filters				
2*9 Filters → FNN	2924	0.62	0	39.1k
4*9 Filters → FNN	2951	0.56	0	45.1k

Total energy consumption on test trajectories:

1.965

1.767

1.701

Less energy consumption indicate more stable flights

Results

Better performance

Less parameters

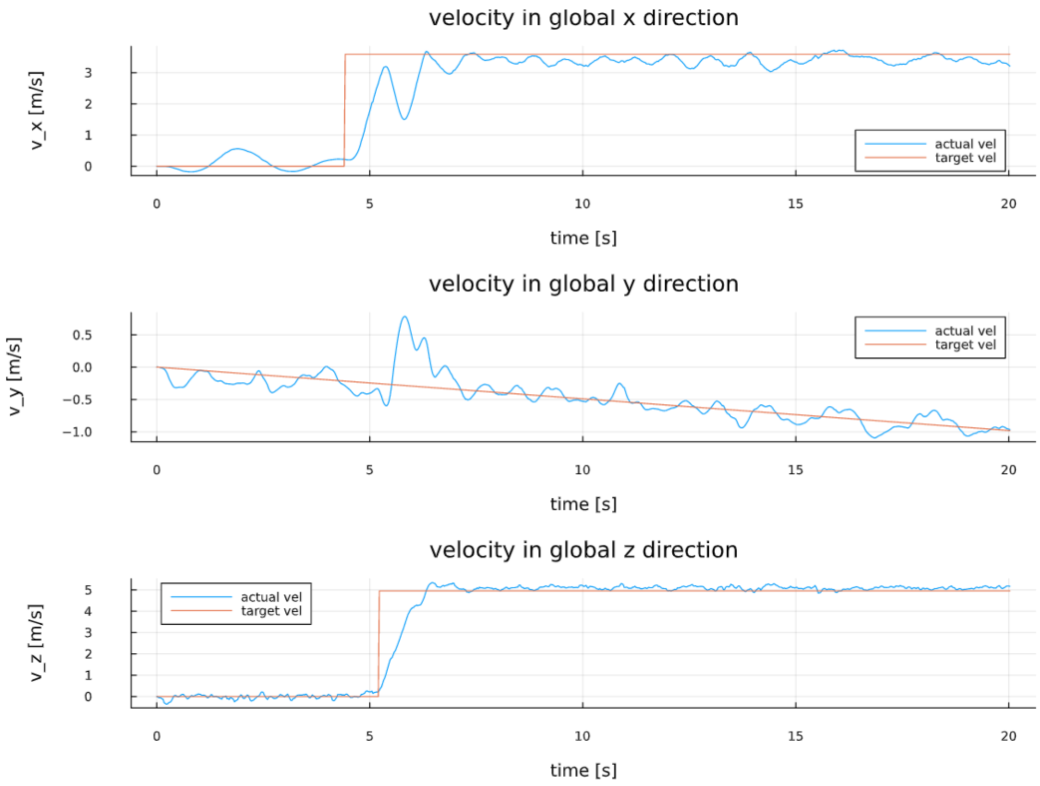
More stable flights

3. Deep Dive

One example trajectory

Reward: 2897

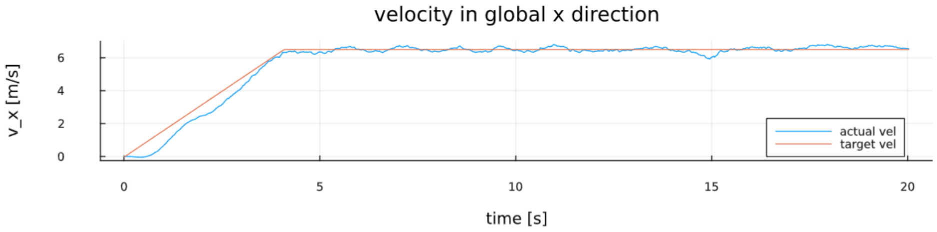
Tracking error: 0.68 m/s

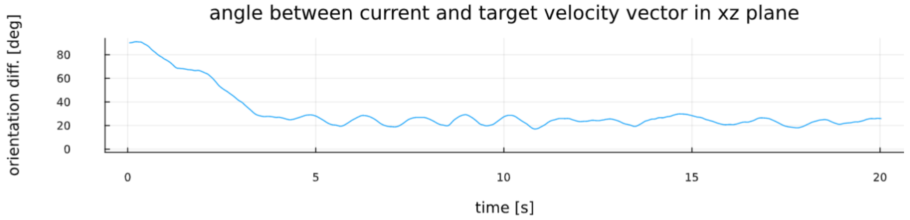


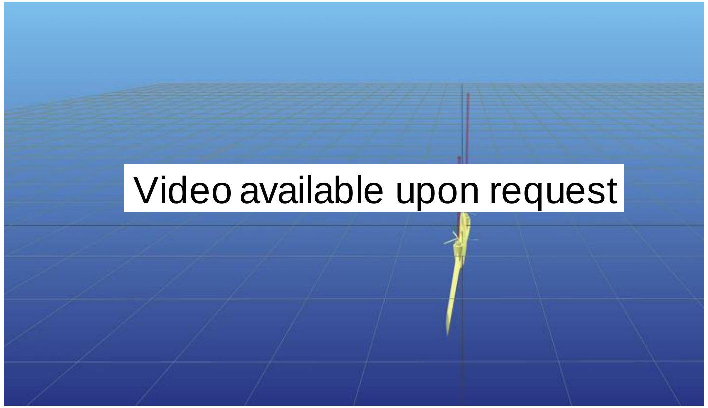
Bonus

Drone to fixed-wing mode

Current velocity as input didn't improve performance







Green: target velocity | Yellow: actual velocity | Red: thrusts

Bonus

Other results

- Wind speed as input didn't improve performance
  - Maybe used too low wind speeds (5m/s max)

- Two step training is a double edge sword
  - Updated reward function (less stay alive reward)

Network	**Reward**	**Tracking error (m/s)**	**# early terminated envs**
18 filters $\rightarrow$ FNN (Baseline)	2820	0.899	1
Add wind speed [x,y,z] as input	2816	0.879	2
Two-step training	-	0.667	13

Summary & Outlook

- Why has no one done this?
- CNN might be better than RNN for our case
- Good bias improves performance
- State-history control?
- Benchmarks for comparison?

Network	**Reward**	**Tracking error (m/s)**	**# early term. envs**	**# train-able params**
FNN (Baseline)	2696	1.214	4	36.7k
Shared filters (classical CNN)				
1 Filter → FNN	2664	1.231	2	36.7k
3 Filters → FNN	2531	1.586	4	41.1k
5 Filters → FNN	2691	1.158	3	46.1k
9 Filters → FNN	2714	1.138	2	55.5k
18 Filters → FNN	2820	0.899	1	76.6k
State dependent filters				
2*9 Filters → FNN	2924	0.62	0	39.1k
4*9 Filters → FNN	2951	0.56	0	45.1k

ADLR Final Presentation: Development of a reinforcement learning based controller for a VTOL drone.

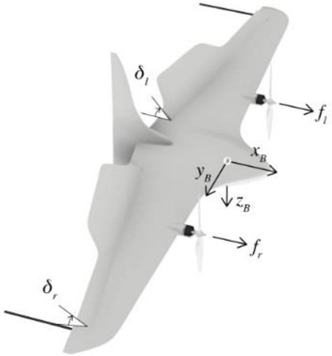
Christopher Narr & Oliver Hausdörfer

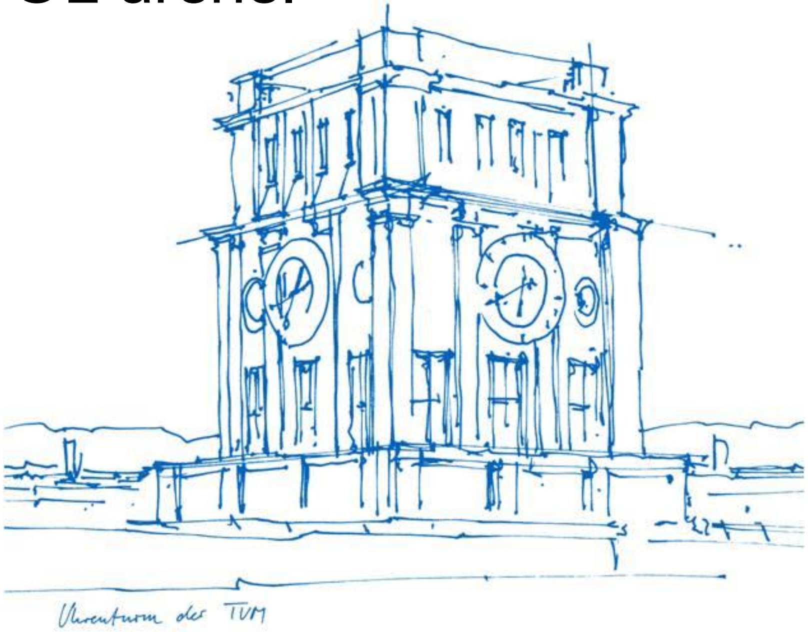
Supervisor: Finn Süberkrüb

TUM / DLR Institute of Robotics and Mechatronics

Advanced Deep Learning for Robotics

23rd February 2023





Thank you!

Q&A