Pixels in, Controls out: How to do End-To-End control using neural networks

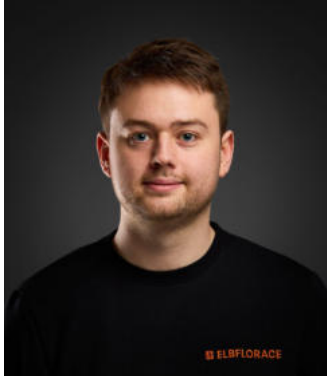
ARWo 2026

Alexander Phieler

Google slides link (for videos)

ELBFLORACE
FORMULA STUDENT TEAM TU DRESDEN

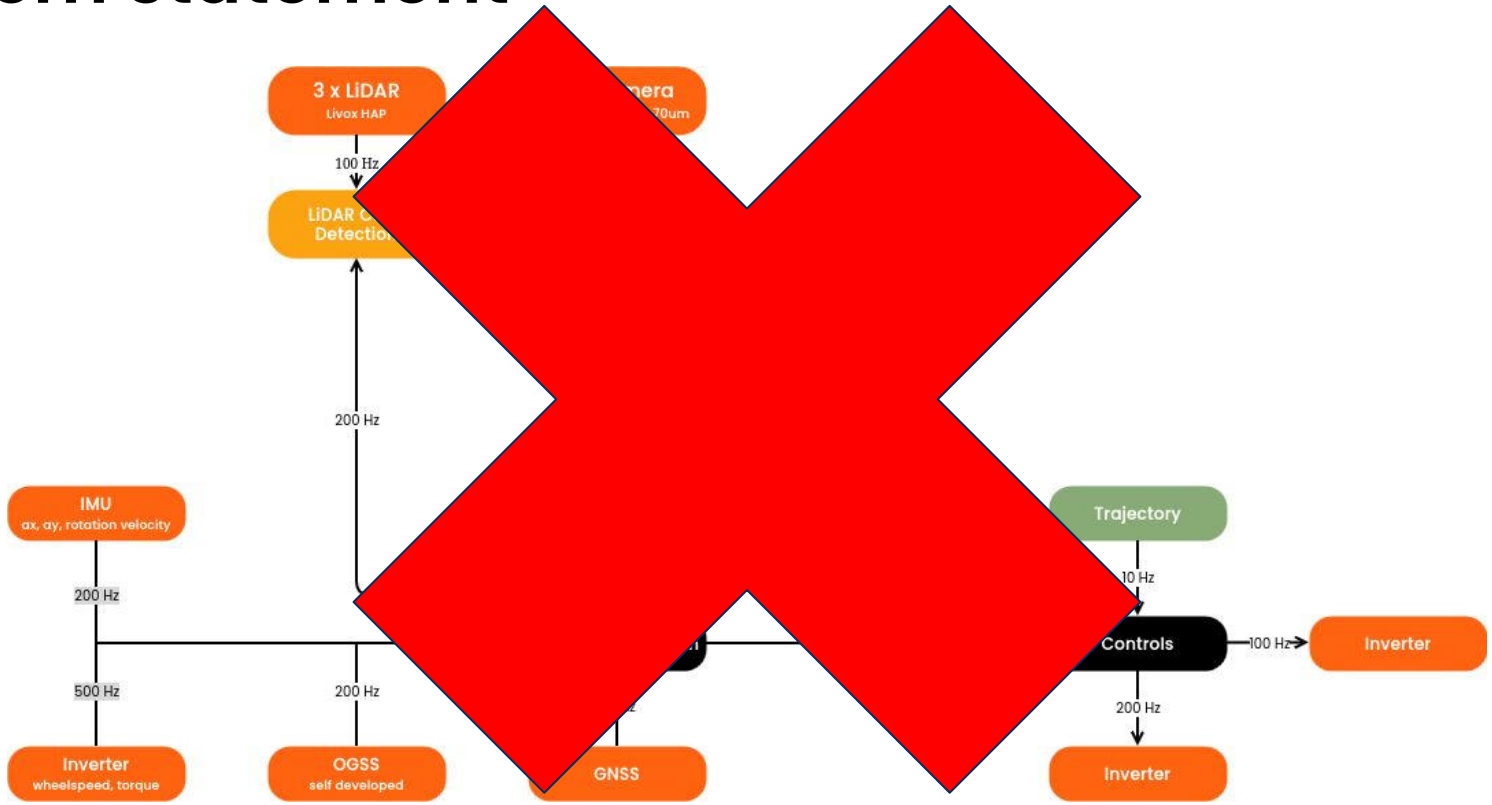
About me



- Alexander Phieler (alexander.phieler@elbflorace.de) https://www.linkedin.com/in/alexander-phieler/
- TU Dresden Diplom Informationssystemtechnik
- EFR13 (21): Head of AS
- EFR14 (22): DV Controls, (...)
- EFR15 (23): SLAM, (...)
- EFR16 (24): PacSim, (...)
- EFR17 (25): Camera perception, (...)
- EFR18 (26): ~~Hopefully Alumnus,~~ Some stuff
- EFR19 (27): Hopefully Graduated

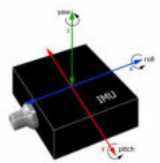
https://www.linkedin.com/in/alexander-phieler/

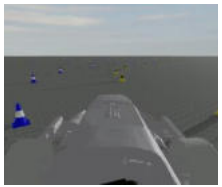
Problem statement

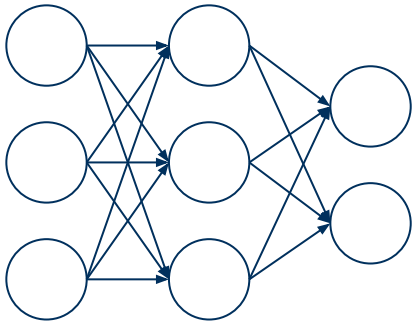


Problem statement

Sensor data



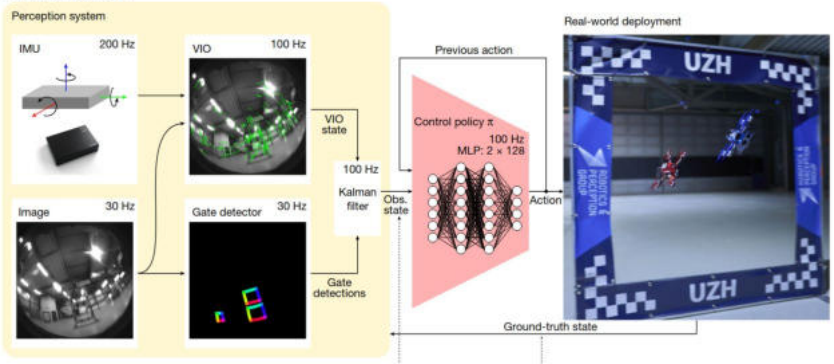




Actions

Related works

- Fully End-To-End
  - Map pixels directly to controls
- Hybrid approaches
  - Separate vision from control
  - Either learned vision->trajectory (conventional controller) or trajectory->controls (conventional perception)
- Training pure Reinforcement Learning (RL) policies directly from pixels is notoriously difficult; methods often suffer from extreme sample inefficiency and convergence issues



Champion-level drone racing using deep reinforcement learning, Kaufmann et. al

ELBFLORACE
FORMULA STUDENT TEAM TU DRESDEN

Gaps and novelty

- Formula Student (FS) has special challenges:
  - Sparse visual features
  - Narrow lane (low margin for errors)
  - Push vehicle to its absolute physical limits
- Applications specific for FS are underdeveloped
- Current works work on simplified problem definitions
  - Steering only (safe but slow constant velocity)
  - Preprocessed perception input (e.g. cone positions)

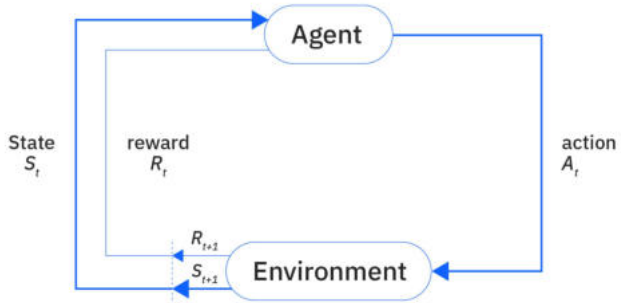
➔ There exist no works which address the full scale problem of mapping raw pixels to holistic actuation (steering and torques) for FS

The aim is not to provide a super polished and optimized system but to get a proof-of-concept for this architecture

ELBFLORACE
FORMULA STUDENT TEAM TU DRESDEN

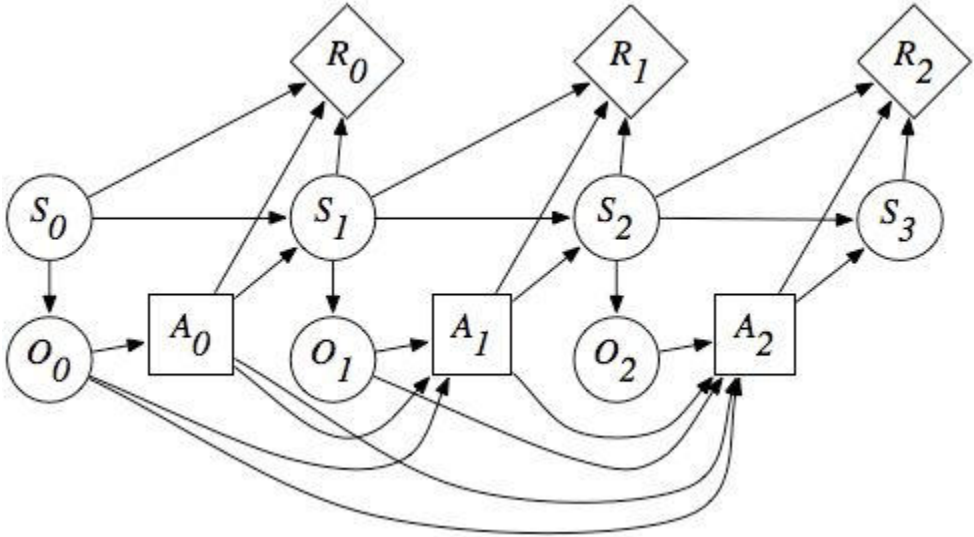
Intermezzo: RL

- Reinforcement learning (RL) is a method to train an “agent” by interacting with an environment
- Agent receives a reward for it’s actions, indicating how good they were
  - High reward: good
  - Small reward: bad
- Reward signal is used to optimize the agent



Intermezzo: POMDP

- **Partially observable markov decision process** is a mathematical framework for sequential decision making
- System is in (hidden) state $\mathbf{s_t}$
- Policy π applies action $\mathbf{a_t}$ based on observation $\mathbf{o_t}$ it gets from environment
- The state-action pair leads to reward $\mathbf{r_t}$ and transition to state $\mathbf{s_{t+1}}$

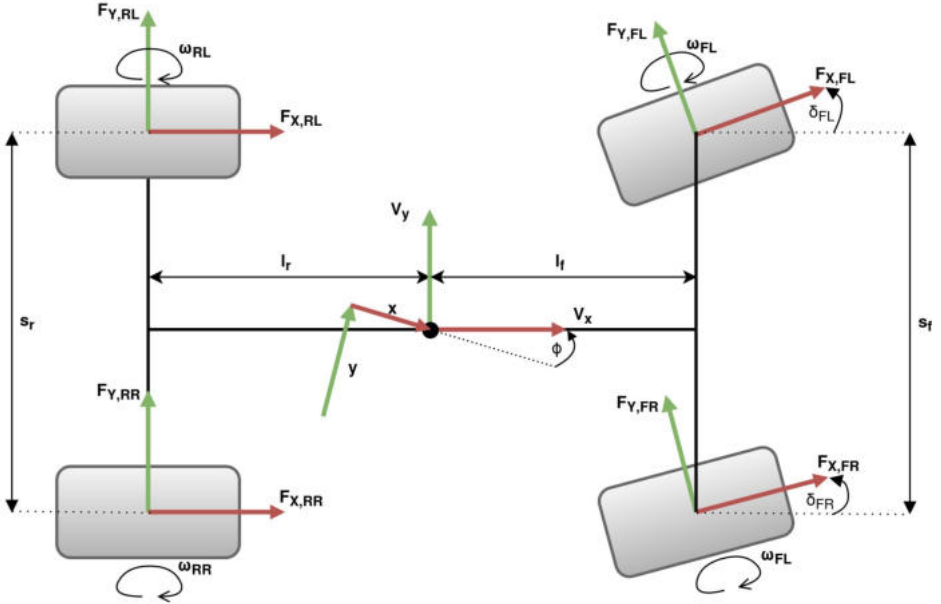


Methodology: Environment

- Model system as a partially observable decision process
- Implementation follows gymnasium api convention
- Physics / complex functions are implemented in C++ and exposed as pybind11 bindings
- Rendering using Panda3d library

Methodology: Vehicle model

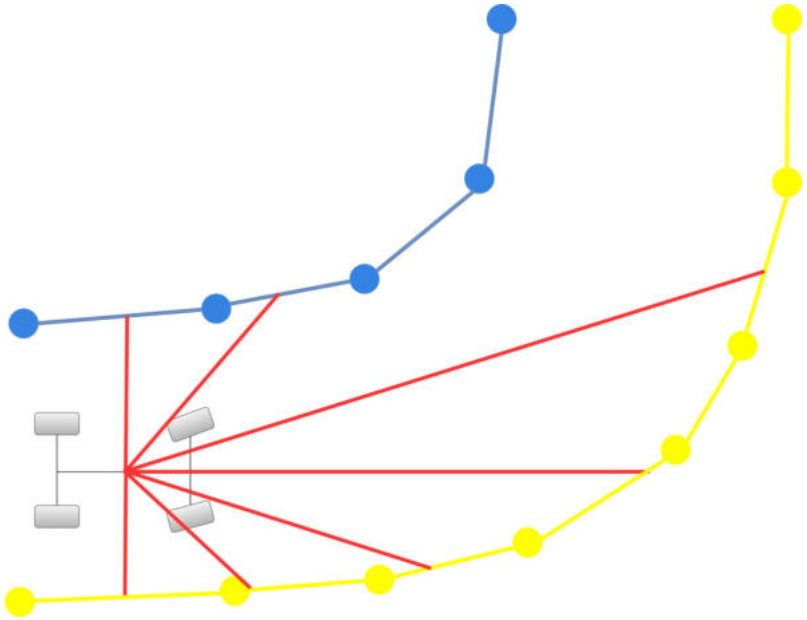
- 7dof vehicle model
  - Planar motion, 4 individual wheelspeeds
- 2nd order steering actuator model
- Input: Steering angle setpoint, 4 wheel torques



ELBFLORACE
FORMULA STUDENT TEAM TU DRESDEN

Methodology: Observations

- Privileged state observations o_{priv}
- Contains privileged data only obtainable in simulation
- More compact and “direct”, allows for simplified problem



Wheelspeeds

Velocity vector

$$o_{priv} = [\omega_{FL}, \omega_{FR}, \omega_{RL}, \omega_{RR}, \delta_{steer}, a_x, a_y, r, o_{vel}, o_{rangefinder}]$$

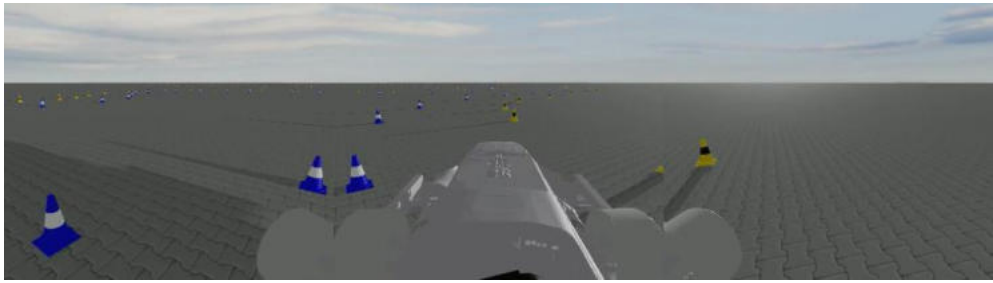
Steering angle

Accelerations, yaw rate

ELBFLORACE
FORMULA STUDENT TEAM TU DRESDEN

Methodology: Observations

- Data obtainable on real vehicle
- Higher dimensional, less “direct”



3 RGB cameras with
306x256 px. ~180° hFov

Wheelspeeds

Accelerations, yaw rate

$$o_{vision} = [\omega_{FL}, \omega_{FR}, \omega_{RL}, \omega_{RR}, \delta_{steer}, a_x, a_y, r, o_{camera}]$$

Steering angle

ELBFLORACE
FORMULA STUDENT TEAM TU DRESDEN

Methodology: Environment behavior

- Episode terminates under following conditions
  - Lap finished
  - Crash (collide with cone)
  - Timeout (120 seconds)

Methodology: Rewards

- Sum of multiple reward terms
- r_finish: positive reward when finishing
- r_collision: negative reward when crashing
- r_progress, r_stand, r_time: encourage fast driving
- others: discourage implausibilities or overly aggressive driving
- 2 presets: Aggressive and Conservative

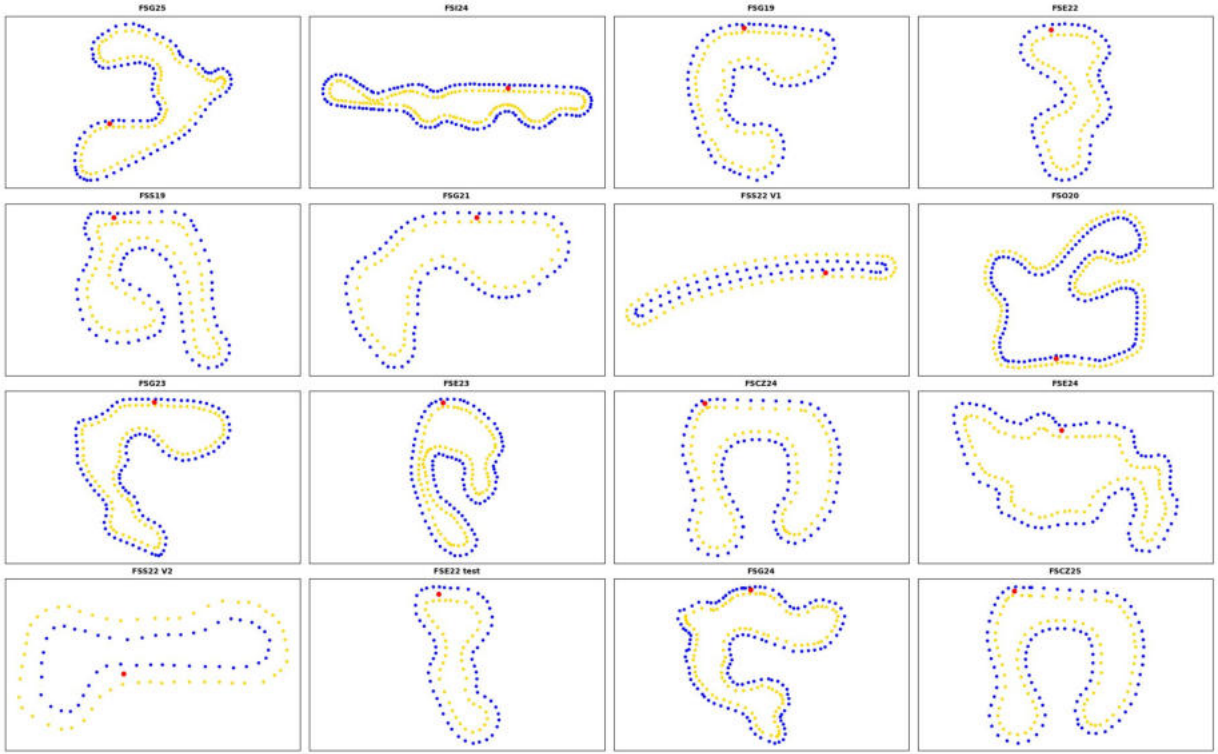
Component	Symbol	Aggressive	Conservative	Type
Track Progress	$\lambda_{progress}$	0.02	0.007	Reward
Tracking	$\lambda_{tracking}$	0.003	0.005	Penalty
Finish	λ_{finish}	10.0	10.0	Reward
Collision	$\lambda_{collision}$	10.0	10.0	Penalty
Stand Still	λ_{stand}	0.5	0.5	Penalty
Constant (Time)	$\lambda_{constant}$	0.1	0.03	Penalty
Slip Angle	$\lambda_{slipAngle}$	0.005	0.005	Penalty
Slip Ratio	$\lambda_{slipRatio}$	0.05	0.05	Penalty
Action rate	$\lambda_{actionRate}$	0.002	0.002	Penalty
Cross Axle Consistency	$\lambda_{crossAxle}$	0.001	0.001	Penalty
Long. Consistency	$\lambda_{longConsistency}$	0.0002	0.0002	Penalty

$$r = \lambda_{progress} * r_{progress} + \lambda_{tracking} * r_{tracking} + \lambda_{finish} * r_{finish} + \lambda_{collision} * r_{collision} +$$
$$\lambda_{stand} * r_{stand} + \lambda_{constant} * r_{constant} + \lambda_{slipAngle} * r_{slipAngle} + \lambda_{slipRatio} * r_{slipRatio} +$$
$$\lambda_{actionRate} * r_{actionRate} + \lambda_{crossAxle} * r_{crossAxle} + \lambda_{longConsistency} * r_{longConsistency}$$

Methodology: Scenarios

- 16 past competition tracks in total
- 4 are not used for training, only for evaluation
- Additional random augmentations
  - Flip X
  - Flip Y
  - Random starting position

Name	Competition	Length	Train/Eval
FSG19	FS Germany 2019	$303m$	Train
FSS19	FS Spain 2019	$230m$	Train
FSG21	FS Germany 2021	$220m$	Train
FSO20	FS Online 2020	$383m$	Train
FSS22 V1	FS Spain 2022 (First track)	$267m$	Train
FSS22 V2	FS Spain 2021 (Second track)	$127m$	Train
FSE22	FS East 2022	$155m$	Train
FSE22 test	FS East 2022 (test track)	$129m$	Train
FSE23	FS East 2023	$241m$	Train
FSG23	FS Germany 2023	$320m$	Train
FSCZ24	FS Czech 2024	$287m$	Train
FSI24	FS ATA (Italy) 2024	$382m$	Train
FSE24	FS East 2024	$230m$	Eval
FSG24	FS Germany 2024	$343m$	Eval
FSCZ25	FS Czech 2025	$286m$	Eval
FSG25	FS Germany 2024 (Trackdrive)	$239m$	Eval



ELBFLORACE
FORMULA STUDENT TEAM TU DRESDEN

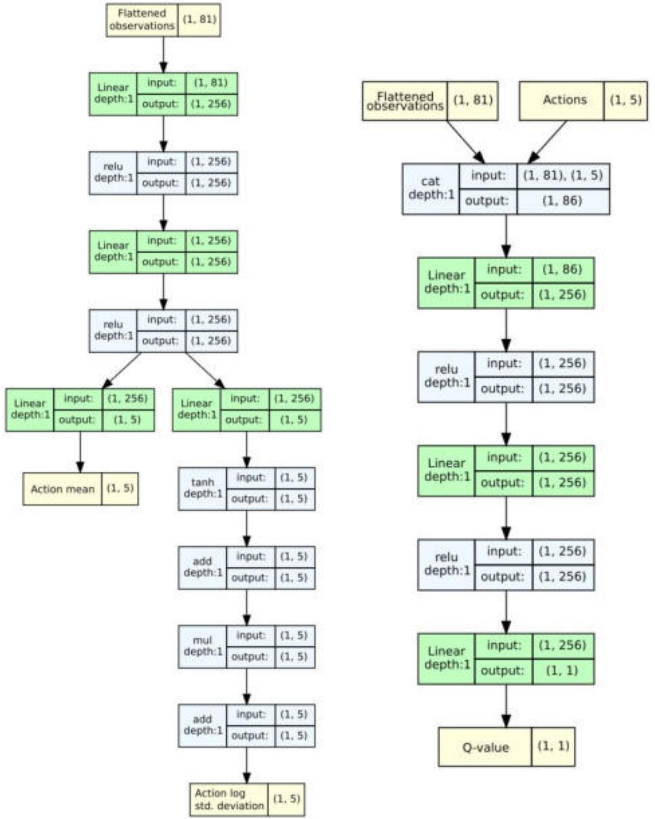
Methodology: Policy

- To mitigate the issues of pure vision-based E2E RL, some state-of-the-art works utilize a two-stage "Teacher-Student" training process.
- **Stage 1 (The Teacher):** A policy is trained via RL using privileged, ground-truth state information (e.g., exact position and velocity).
- **Stage 2 (The Student):** A vision-based policy is then trained via Imitation Learning to mimic the Teacher's actions.

- The Advantage: This hybrid method combines the sample efficiency of Imitation Learning with the performance maximization capabilities of RL.

Methodology: State-based teacher

- Observations stacked over 3 frames and flattened
- Trained using RL with Soft-Actor-Critic (SAC) algorithm over 4 million iterations
  - Used implementation from [CleanRL repo]
- MLP architecture



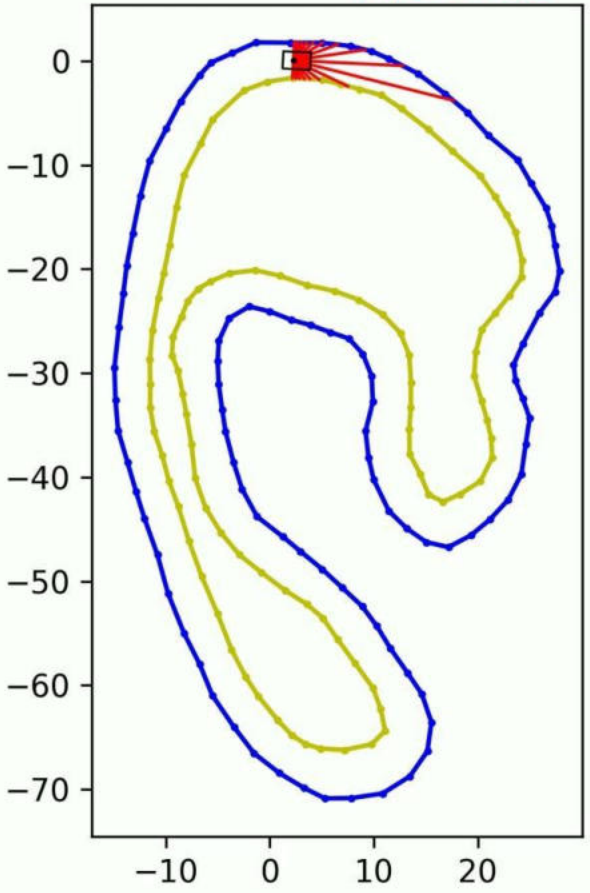
ELBFLORACE
FORMULA STUDENT TEAM TU DRESDEN

t=0.2
v=[2.03, −0.0]
yawrate=−0.004

trackNumber=753505
acc=[10.23, 0.01]

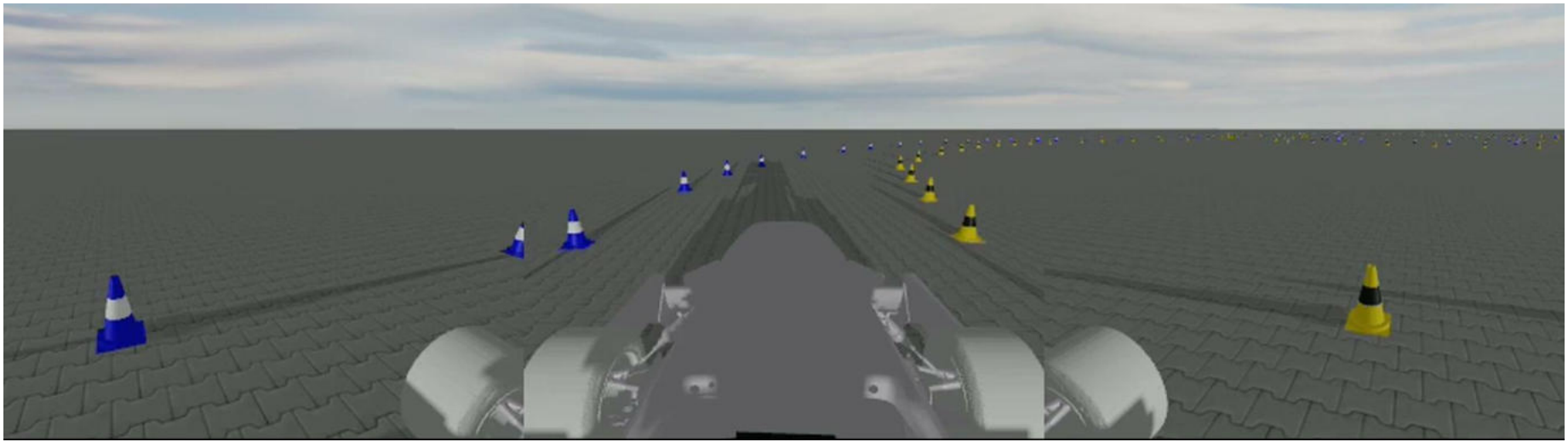
steering=−0.001
torque FL=21.34
torque RL=21.83

torque FR=21.13
torque RR=21.87



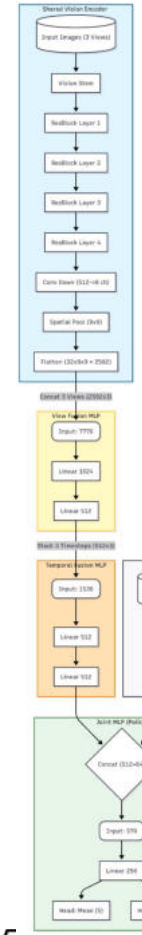
.ORACE
ENT TEAM TU DRESDEN

State based policy: video



Vision based policy (ResNet)

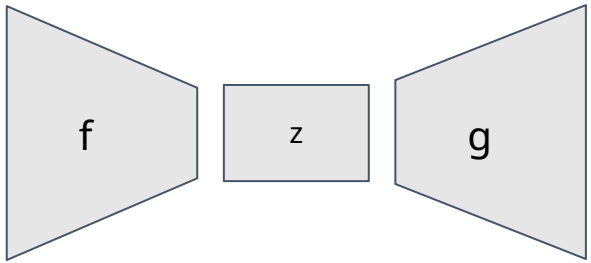
- Stack over 3 frames
- Shared feature extractor, flatten features for each image
- Multi view fusion using shared MLP
- Temporal fusion using MLP
- MLP for proprioceptive sensors feature extraction
- Combine vision and proprioceptive features using MLP, output actions



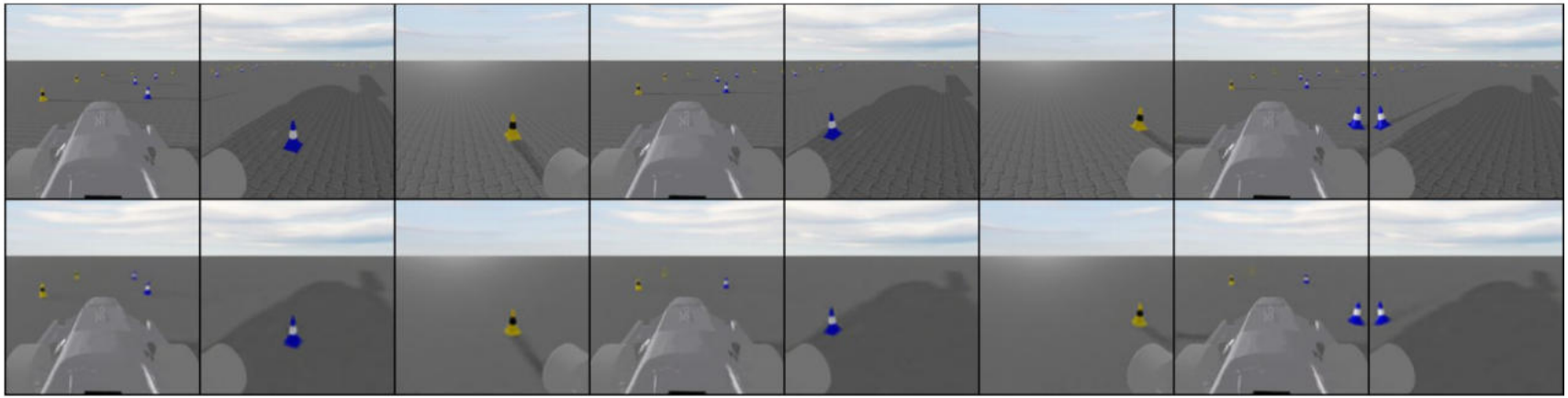
ELBFLORACE
FORMULA STUDENT TEAM TU DRESDEN

Vision based policy (Autoencoder)

- Encoder “f(x) = z”
  - Compress input to low dimension latent code z
- Decoder “g(z) = x”
  - Decompress z to input again
- Learn g(f(x)) = x in unsupervised manner
- Use z=f(x) as compact input to policy network
- Otherwise use same architecture as ResNet approach



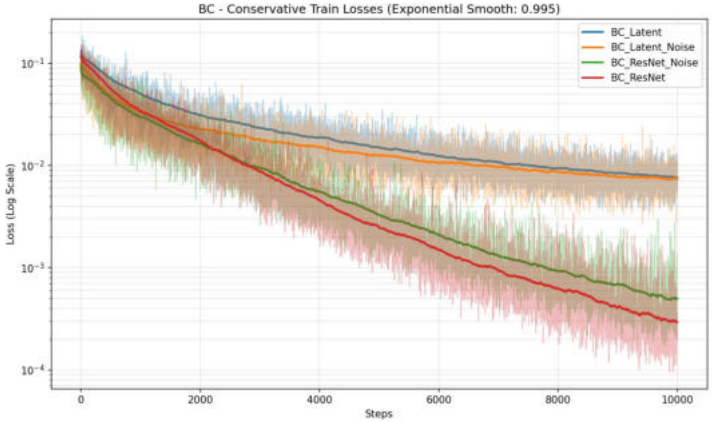
Originals (Top Row) vs. Reconstructions (Bottom Row)

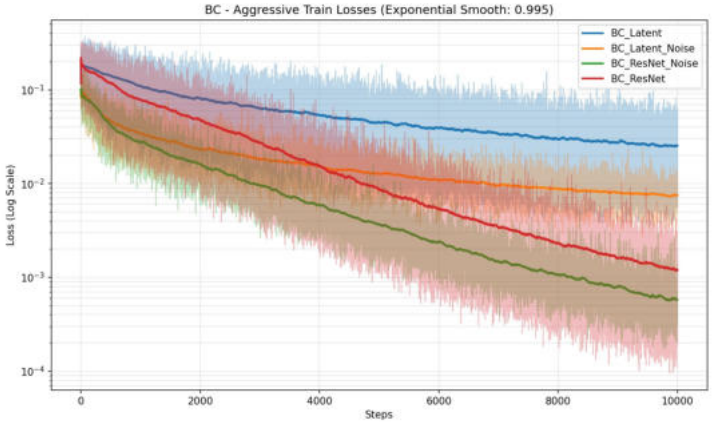


Vision based policy: Training

- Step 1: Behavior cloning (BC) (l2 loss) with 8000 observations -> action correspondences from teacher policy. Run 10000 train steps.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2$$

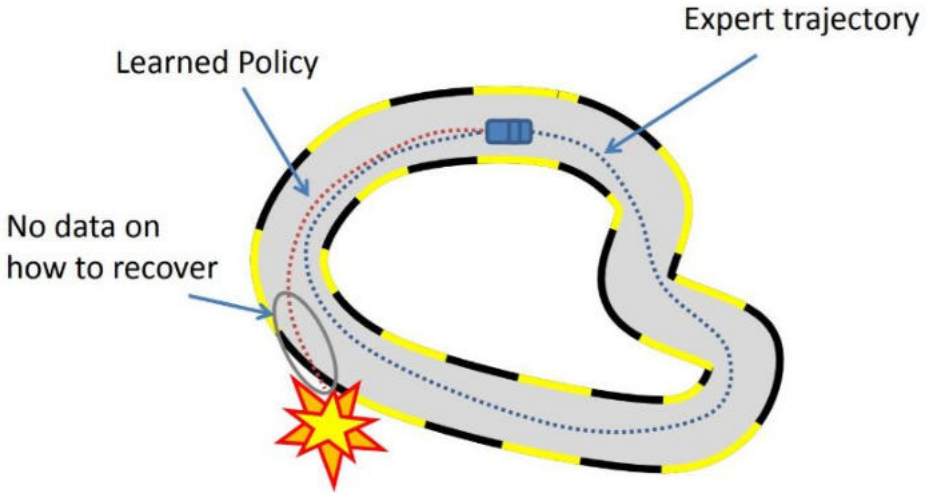




ELBFLORACE
FORMULA STUDENT TEAM TU DRESDEN

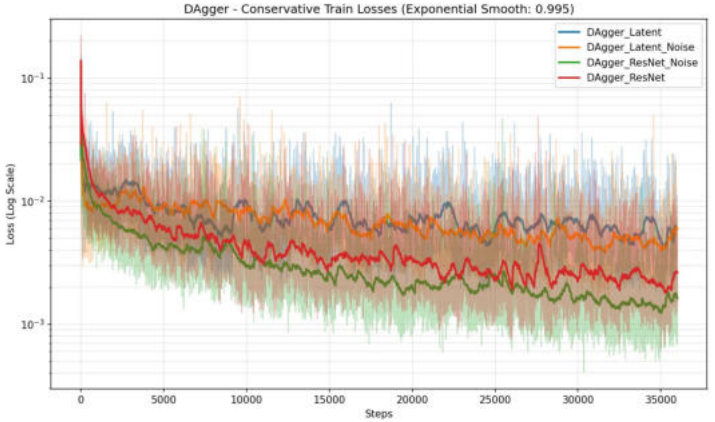
Dataset Aggregation (DAgger)

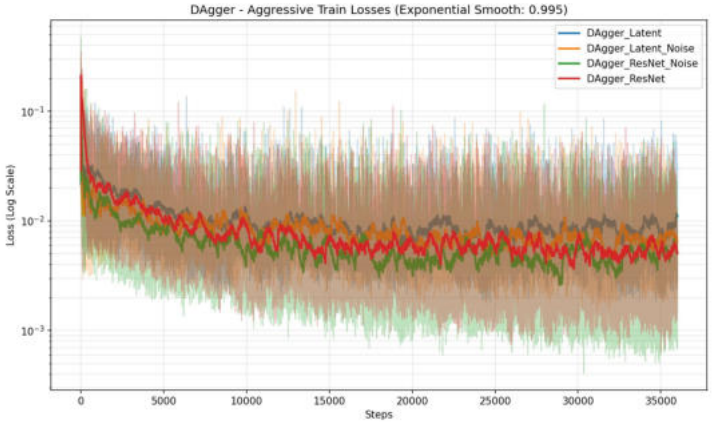
- Problem: In basic BC, robots only learn from "perfect" demonstrations. If the robot makes a tiny mistake, it ends up in a situation it has never seen before
- Solution
  - Let the student policy drive
  - Record action expert would take
  - Extend IL dataset with these trajectories
  - Retrain
- Advantage: Student learns to recover from it’s own mistakes



Vision based policy: Training

- Step 2: DAgger, 300 episodes with 120 train steps each





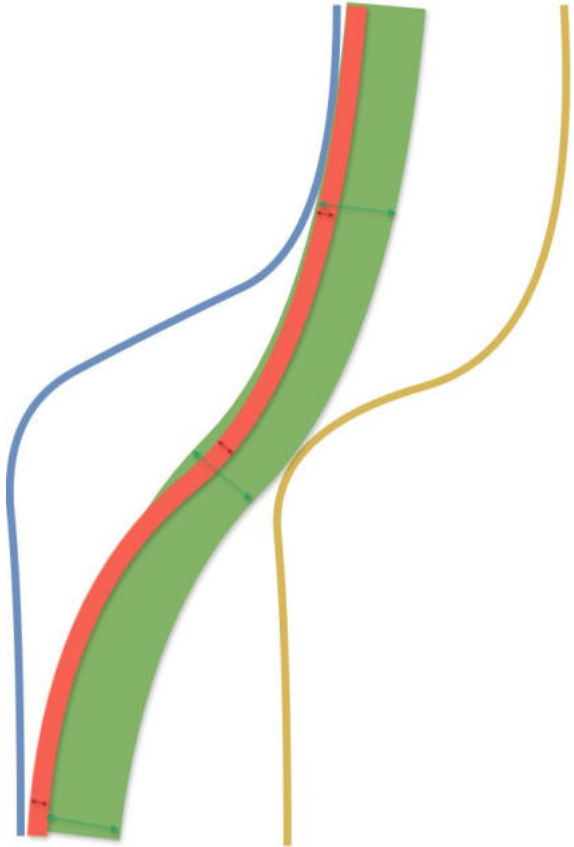
ELBFLORACE
FORMULA STUDENT TEAM TU DRESDEN

Brittle policy problem

- State based policy is trained for “flawless” driving
- It pushes to the absolute limits
- utilize maximum tire friction
- drive very close to track boundaries
- Even small deviations (inevitable in IL) can lead to states which cannot be saved by the teacher (e.g. slide), leading to a crash.

Brittle policy problem

- Solution: Train teacher with these deviations in mind by injecting noise to the actions during training
- The environment behaves stochastically
- The teacher policy doesn’t just learn to drive an optimal trajectory, but to drive on a “tube” around it.



Results

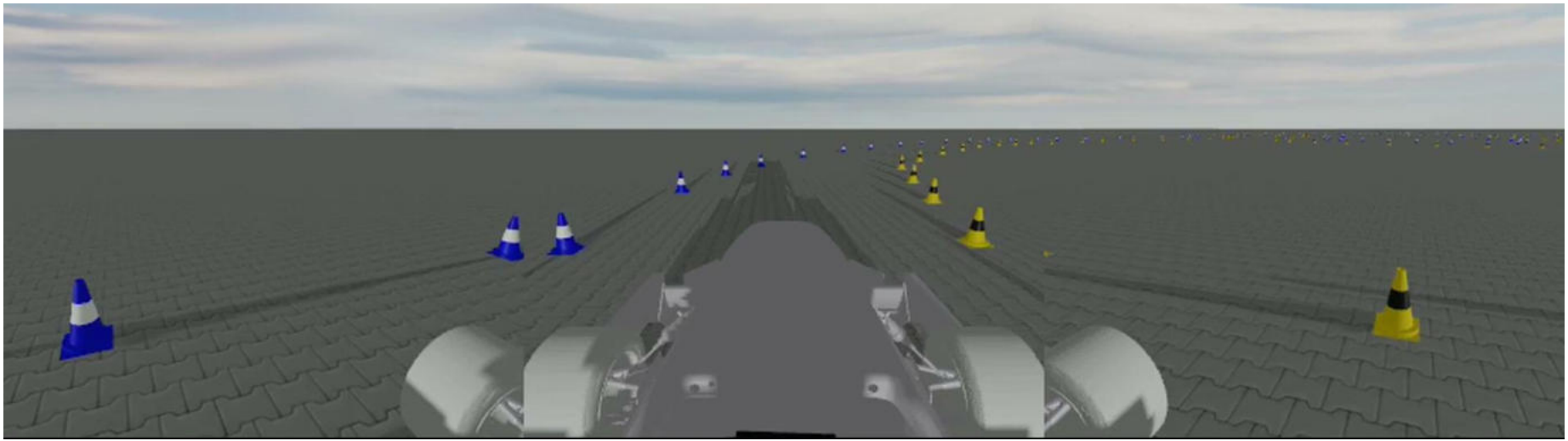
- Two metrics: Success rate and geometric mean of normalized lap times

$$\text{Geom. Mean (Norm.)} = \left(\prod_{i=1}^{N} \frac{t_i}{t_{best,i}} \right)^{\frac{1}{N}}$$

- Use rules based system as baseline

Algorithm	Geom. Mean (Norm.)	Success Rate	Success Rate (Unseen)
Baseline	1.417	100.0%	100.0%
State_Aggr_Noise	1.002	100.0%	100.0%
State_Aggr	1.023	93.8%	100.0%
State_Cons_Noise	1.737	93.8%	75.0%
State_Cons	1.486	87.5%	75.0%
DAgger_ResNet_Cons_Noise	1.722	93.8%	75.0%
DAgger_Latent_Cons	1.573	68.8%	25.0%
DAgger_ResNet_Cons	1.504	62.5%	25.0%
DAgger_Latent_Cons_Noise	1.789	43.8%	25.0%
DAgger_ResNet_Aggr	1.045	37.5%	25.0%
DAgger_ResNet_Aggr_Noise	1.059	31.2%	0.0%
DAgger_Latent_Aggr	1.139	6.2%	0.0%
DAgger_Latent_Aggr_Noise	1.189	6.2%	0.0%
BC_Latent_Aggr	-	0.0%	0.0%
BC_ResNet_Aggr_Noise	-	0.0%	0.0%
BC_Latent_Aggr_Noise	-	0.0%	0.0%
BC_ResNet_Aggr	-	0.0%	0.0%
BC_Latent_Cons_Noise	-	0.0%	0.0%
BC_ResNet_Cons_Noise	-	0.0%	0.0%
BC_ResNet_Cons	-	0.0%	0.0%
BC_Latent_Cons	-	0.0%	0.0%

Video: Best vision based policy



Future work

- Many aspect can be improved
- **Environment throughput** (optimization, parallelization)
- **Sim-to-real**
- **Architectural improvements**
  - Temporal fusion with e.g. RNN or Transformer
- **Vision based RL**
  - Imitation loss is a proxy metric. Small imitation errors can still give big difference in end-to-end performance
- **Image and feature scale**
  - 306x256 image size (FS teams usually use camera with multiple megapixels)
  - 9x9 feature scale

Conclusion and learnings

- Proof of concept for novel problem statement (Full E2E control in FS)
- Design training workflow which is also feasible with more constrained compute resources

→ E2E approaches are a viable alternative to modular pipelines in Formula Student
→ Current approach still has limitations in terms of robustness and reliability

- Debugging RL sucks, feedback loops of up to multiple hours
- RL frameworks are not worth it, just use CleanRL

Next steps

- Code on Github: https://github.com/alexphieler/fs-e2e-project
- Scientific Publication
- Optimize the performance of the environment
    - New renderer
- Run policies in ROS loop
- I think I will continue this project a bit, hope someone will take over next season

- Slides and written thesis will be available on ARWo website and elbflorace.de/publications

ELBFLORACE
FORMULA STUDENT TEAM TU DRESDEN

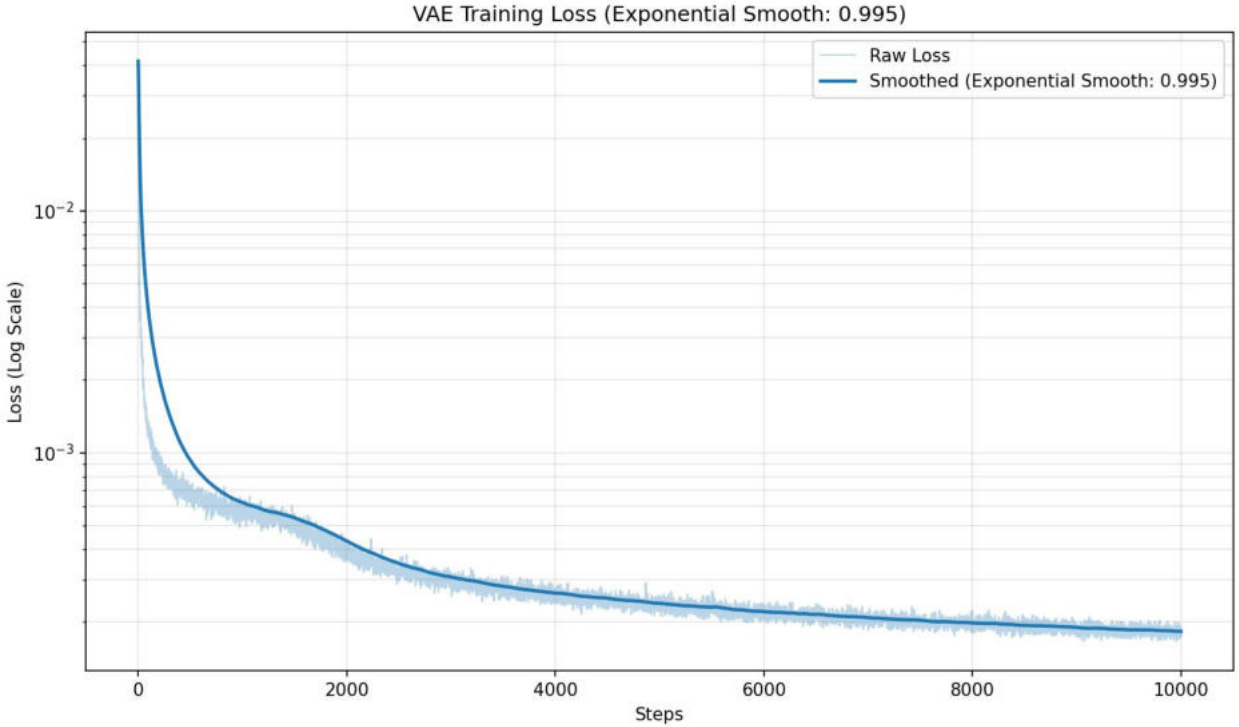
The end

Feedback link:

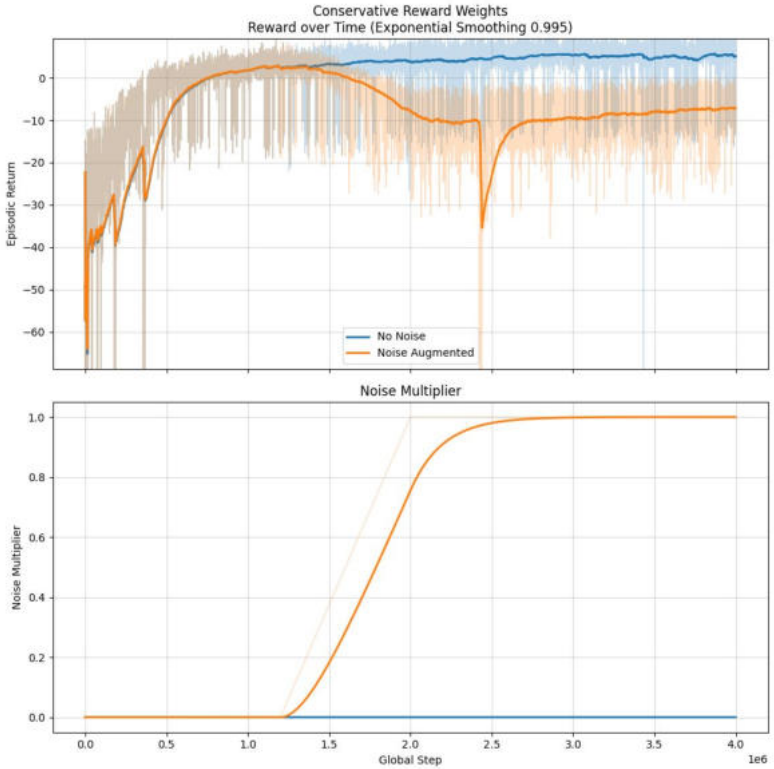
If you want to connect:

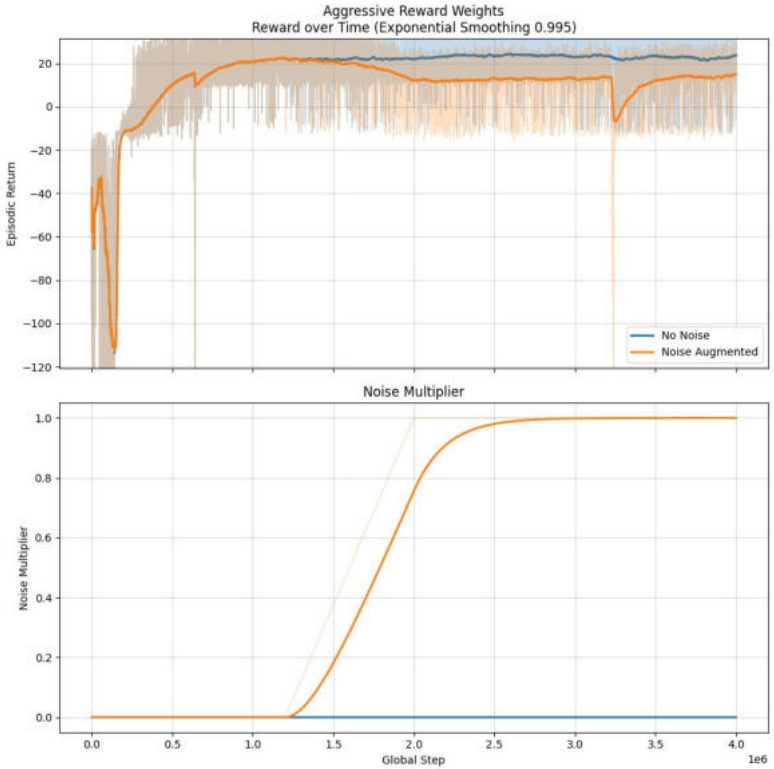
ELBFLORACE
FORMULA STUDENT TEAM TU DRESDEN

Extra: VAE loss



Extra: reward graphs





ELBFLORACE
FORMULA STUDENT TEAM TU DRESDEN