

# Combining Reinforcement Learning and Search for Cooperative Trajectory Planning

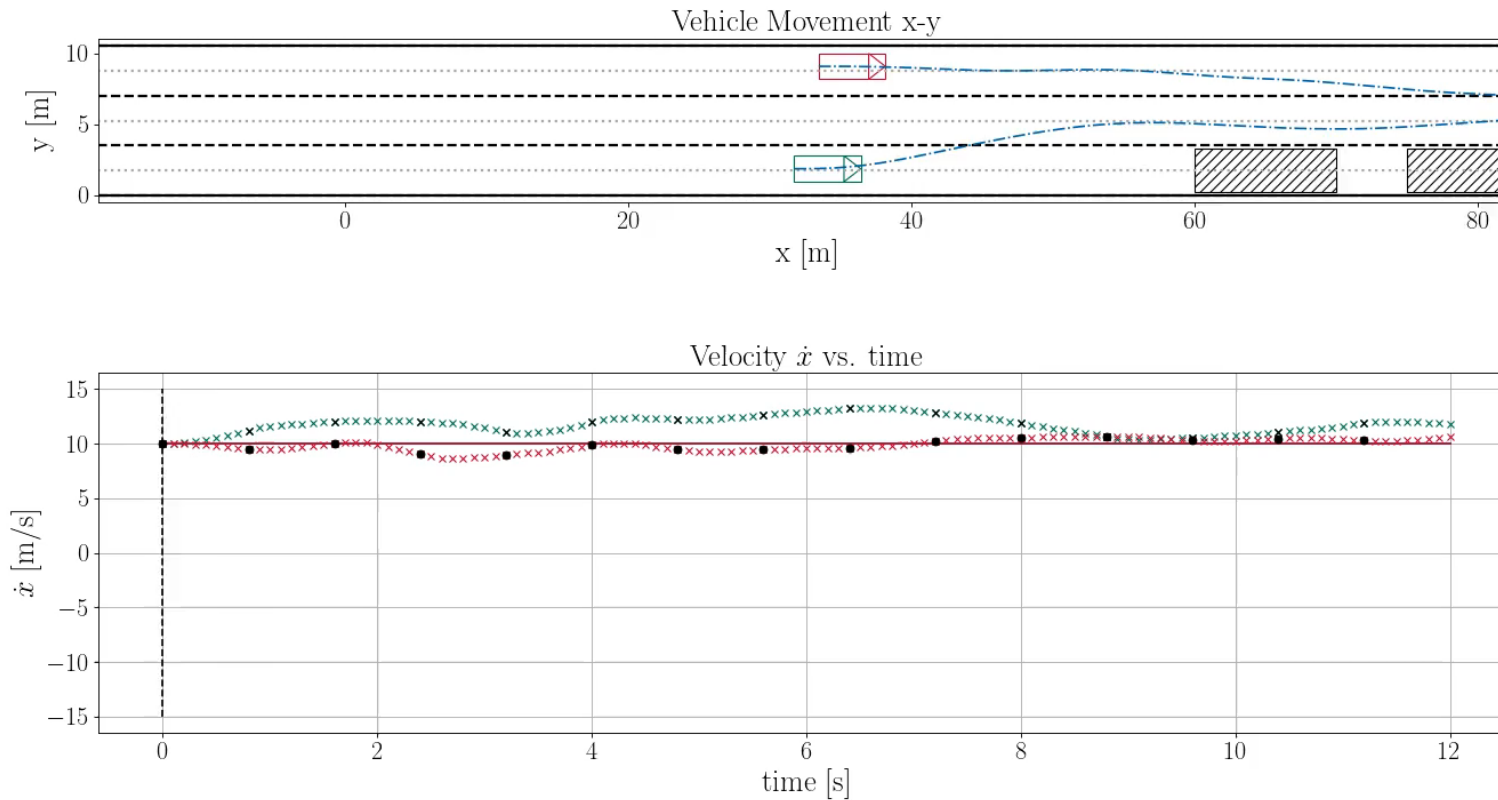
Timo Klein

ANGEWANDTE TECHNISCH-KOGNITIVE SYSTEME

Institut für angewandte Informatik und formale Beschreibungsverfahren, Forschungszentrum Informatik



# Cooperative Trajectory Planning



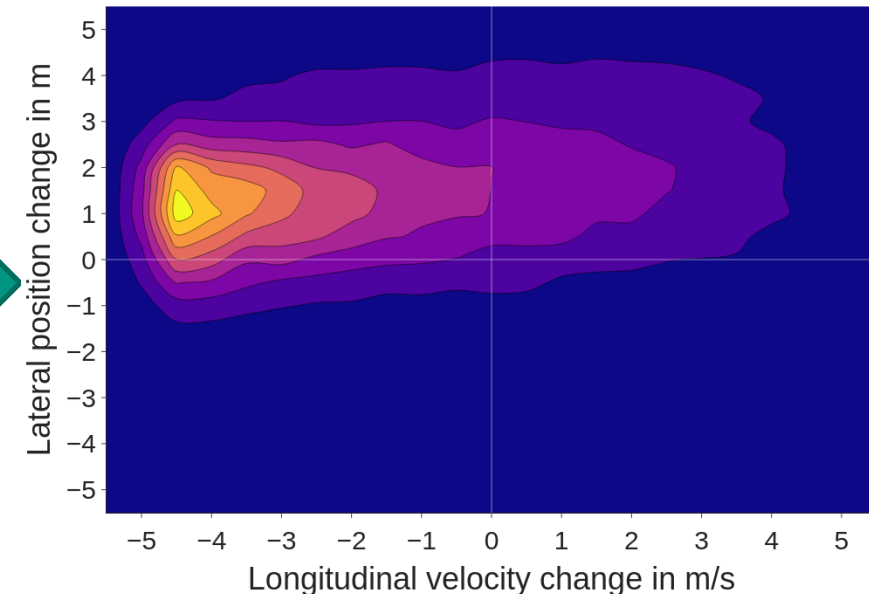
# Research Question

- Uniform sampling in 2D continuous action space is inefficient
- **Goal:** Increasing sample efficiency through focused sampling
- **Method:** Integrate learned knowledge into the search

Visual input



Sample space



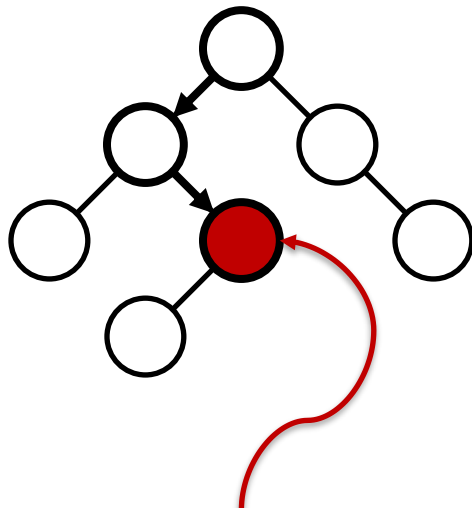
# Reinforcement Learning and Search

		Action space	
		Discrete	Continuous
Number of agents	Single agent	<ul style="list-style-type: none"> <li>▪ <i>AlphaGo</i> (Silver et al.)</li> <li>▪ <i>AlphaGo Zero</i> (Silver et al.)</li> <li>▪ <i>AlphaZero</i> (Silver et al.)</li> <li>▪ <i>MuZero</i> (Schrittwieser et al.)</li> <li>▪ <i>SAVE</i> (Hamrick et al.)</li> <li>▪ <i>Tactical Decision</i> (Hoel et al.)</li> </ul>	<ul style="list-style-type: none"> <li>▪ <i>A0C</i> (Moerland et al.)</li> <li>▪ <i>Continuous MuZero</i> (Yang et al.)</li> <li>▪ <i>Sampled MuZero</i> (Hubert et al.)</li> </ul>
	Multi agent	<ul style="list-style-type: none"> <li>▪ <i>Multiplayer AlphaZero</i> (Petosa et al.)</li> </ul>	<ul style="list-style-type: none"> <li>▪ <b><i>This work</i></b></li> </ul>



# Monte Carlo Tree Search (MCTS)

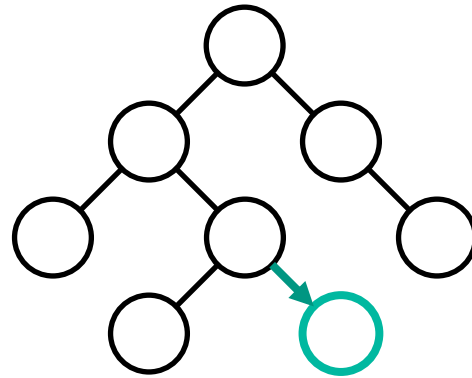
Selection



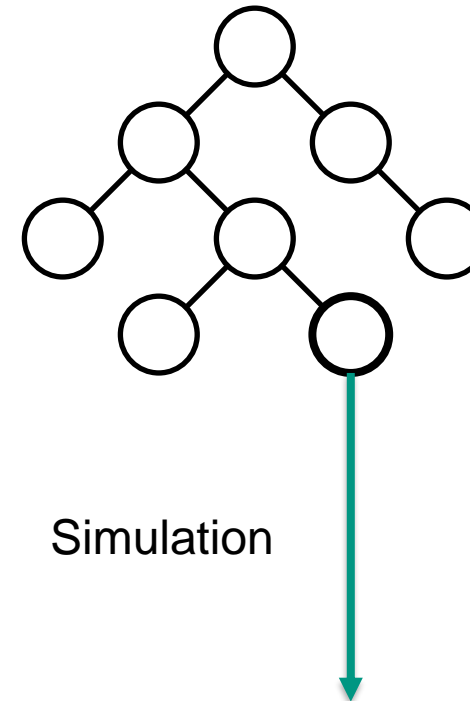
**Leaf node**

- Expandable
- Non-terminal

Expansion

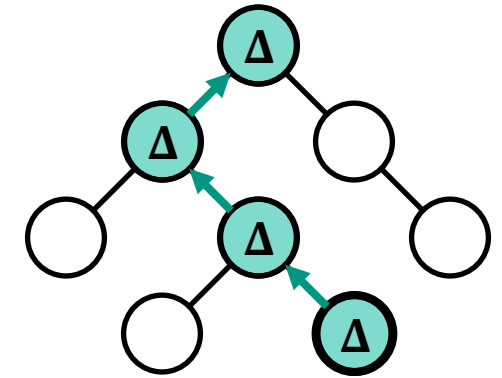


Simulation



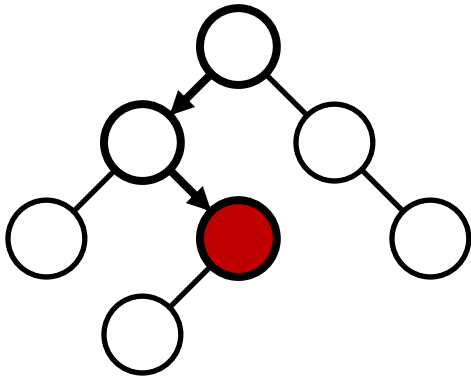
**Outcome  $\Delta$**

Backpropagation

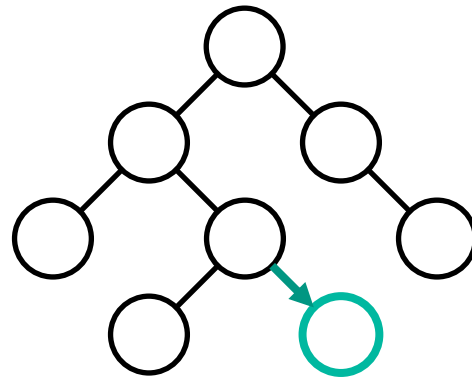


# Concept: Guided MCTS

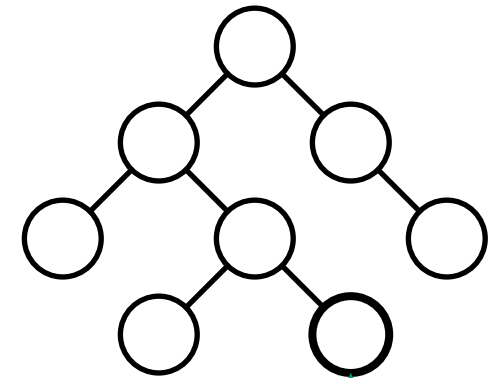
**Selection**



**Expansion**

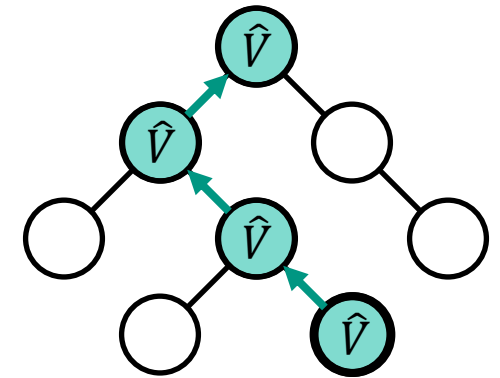


**Simulation**



$$f_{\theta}(s) = (\mu, \sigma, \hat{V})$$

**Backpropagation**



# Problem 1: Input Representation

## Visual map

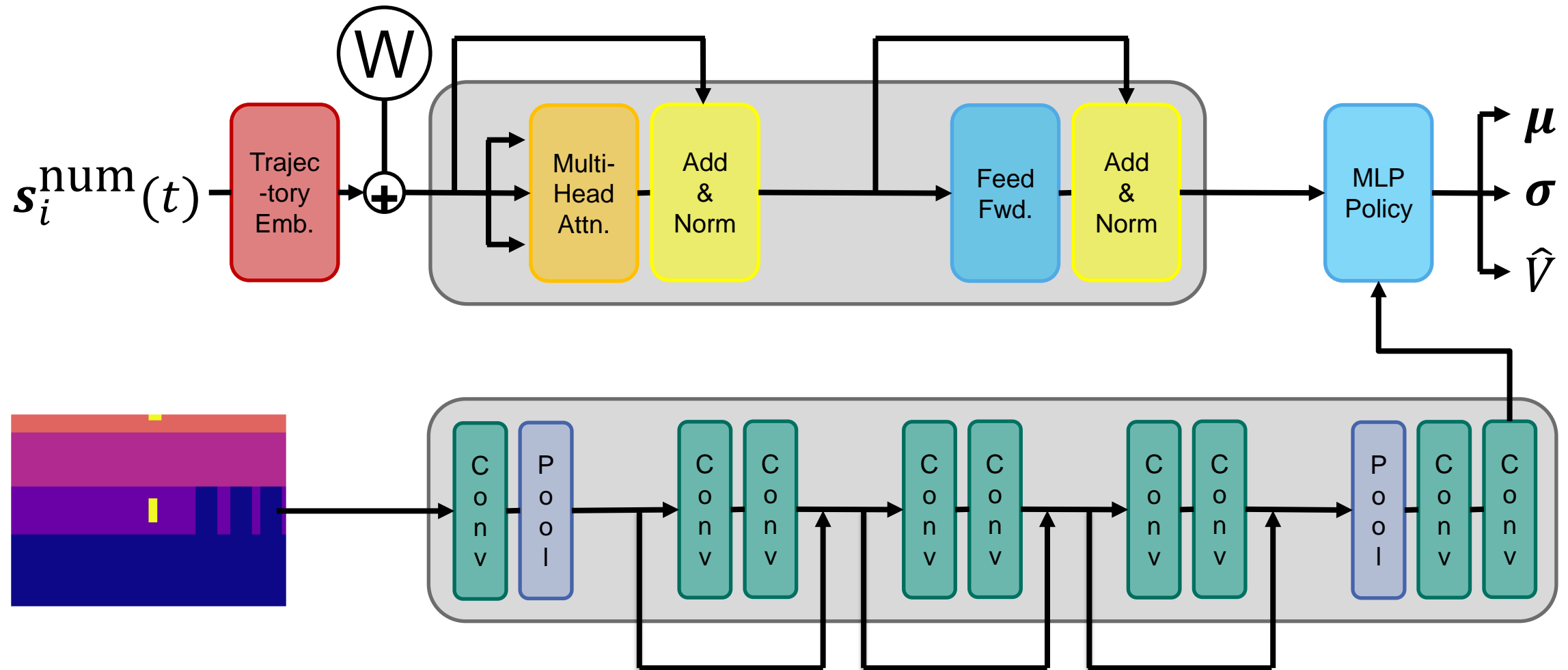


## Agent trajectories

$$\mathbf{s}_i^{\text{num}}(t) = \begin{pmatrix} \mathbf{n}_0(t) \\ \mathbf{n}_1(t) \\ \vdots \\ \mathbf{n}_i(t) \\ \vdots \\ \mathbf{n}_Y(t) \end{pmatrix}$$

Trajectory of agent 1 relative to the ego agent  $i$

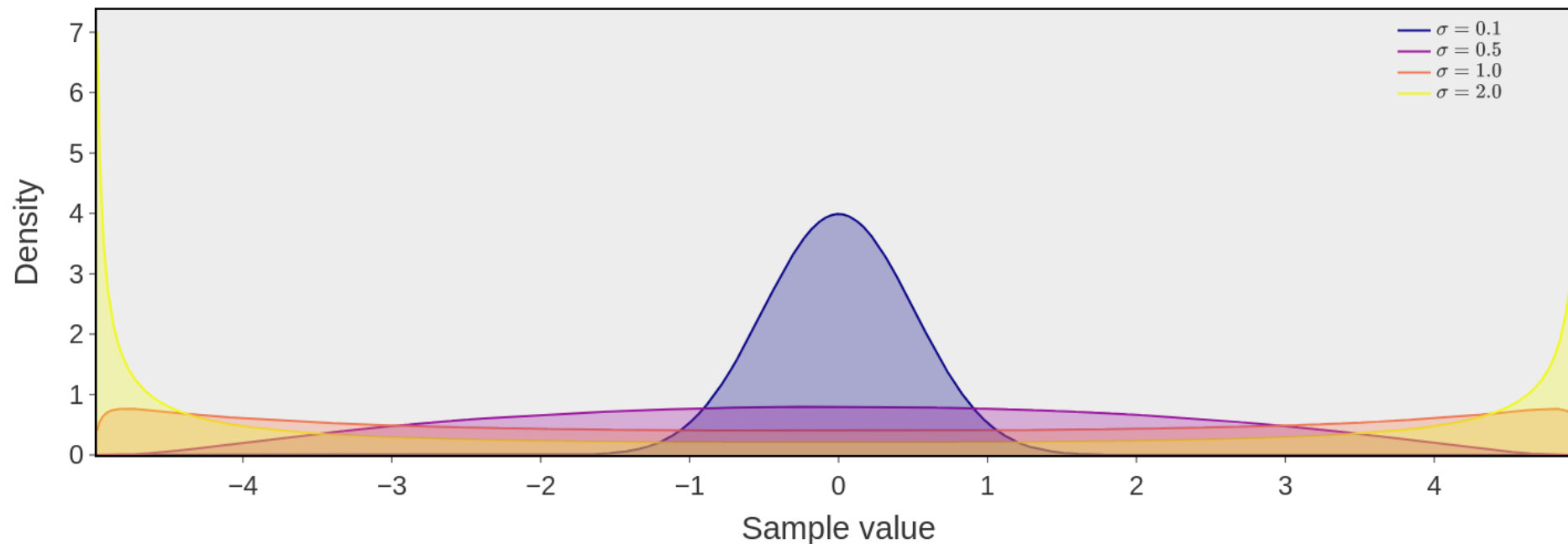
# Problem 2: Flexible Number of Agents



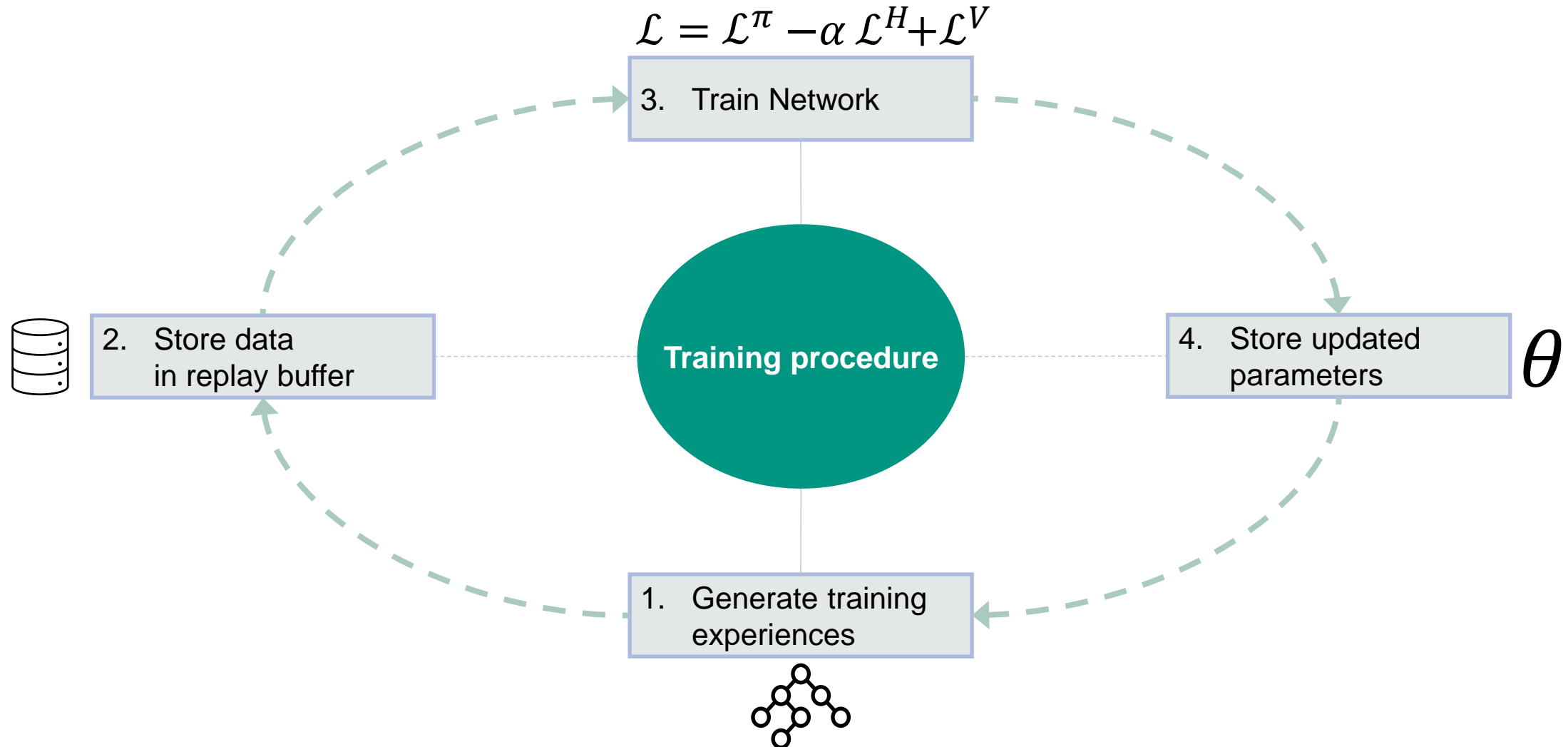


# Problem 3: Constraints of Vehicle

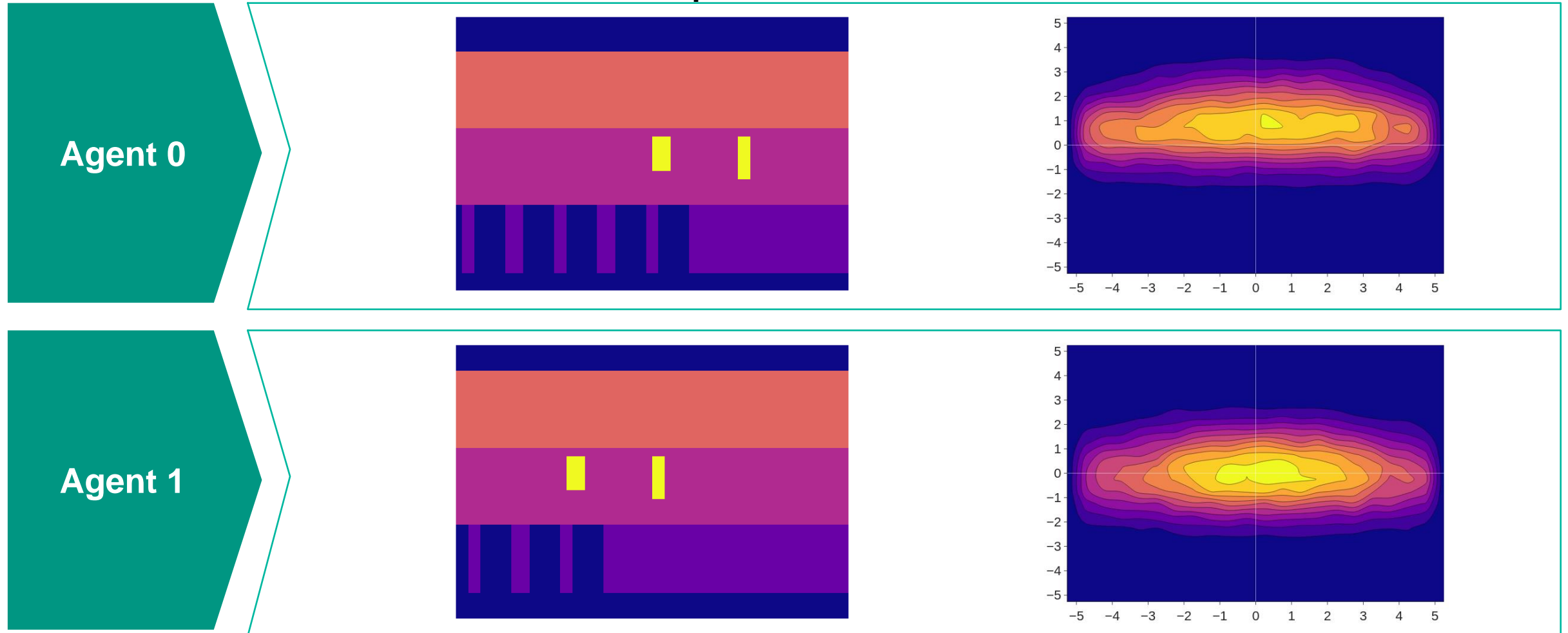
- The support of a normal distribution is unbounded
- Sampled actions  $\mathbf{u} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$  can theoretically take any value
- Transformed distribution  $\mathbf{a} = c \cdot \tanh(\mathbf{u})$  enforces action bounds



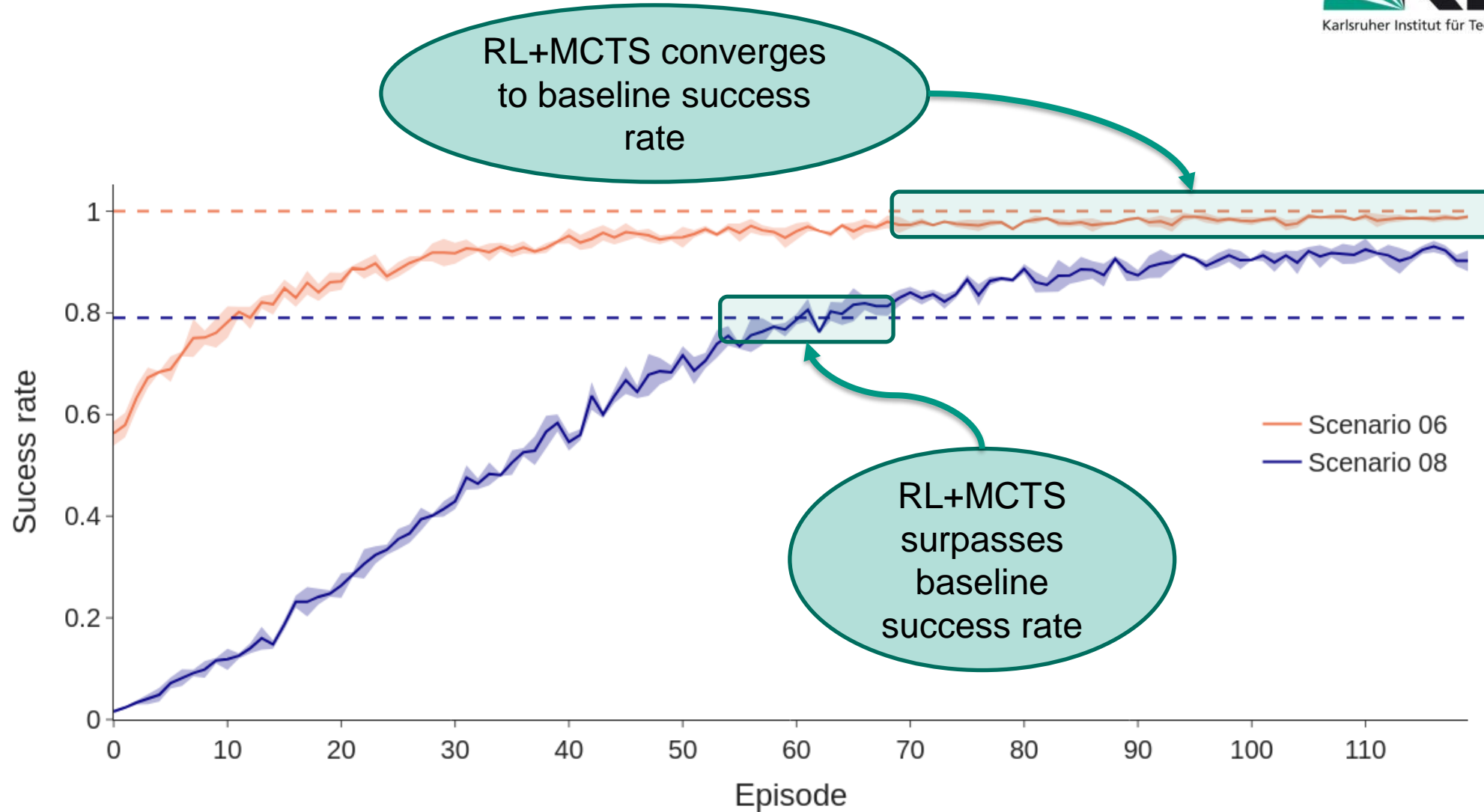
# Training



# Qualitative Evaluation (Scenario 06)



# RL+MCTS vs Baseline



# Wall Clock Time

## Scenario 06

Model	Iterations	Success	Wall clock
RL+MCTS	100	0.99	2m43s
Baseline	100	0.99	4s

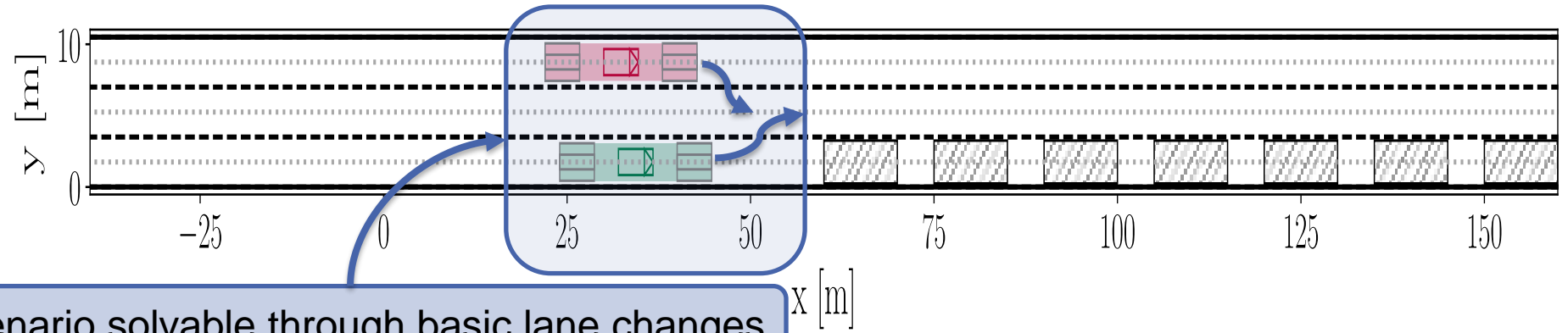
## Scenario 08

Model	Iterations	Success	Wall clock
RL+MCTS	50	0.84	49s
Baseline	400	0.87	57s

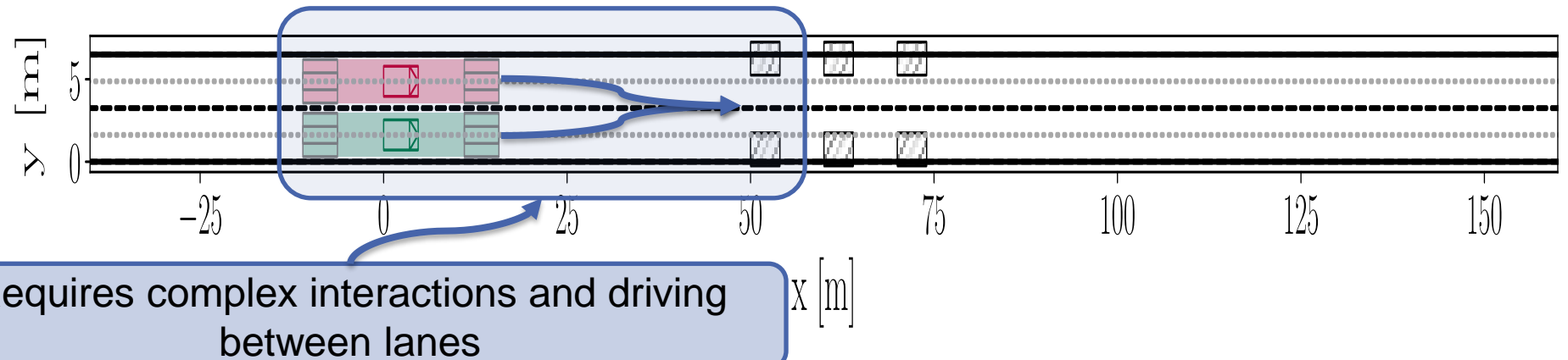


# Scenario 08 vs Scenario 06

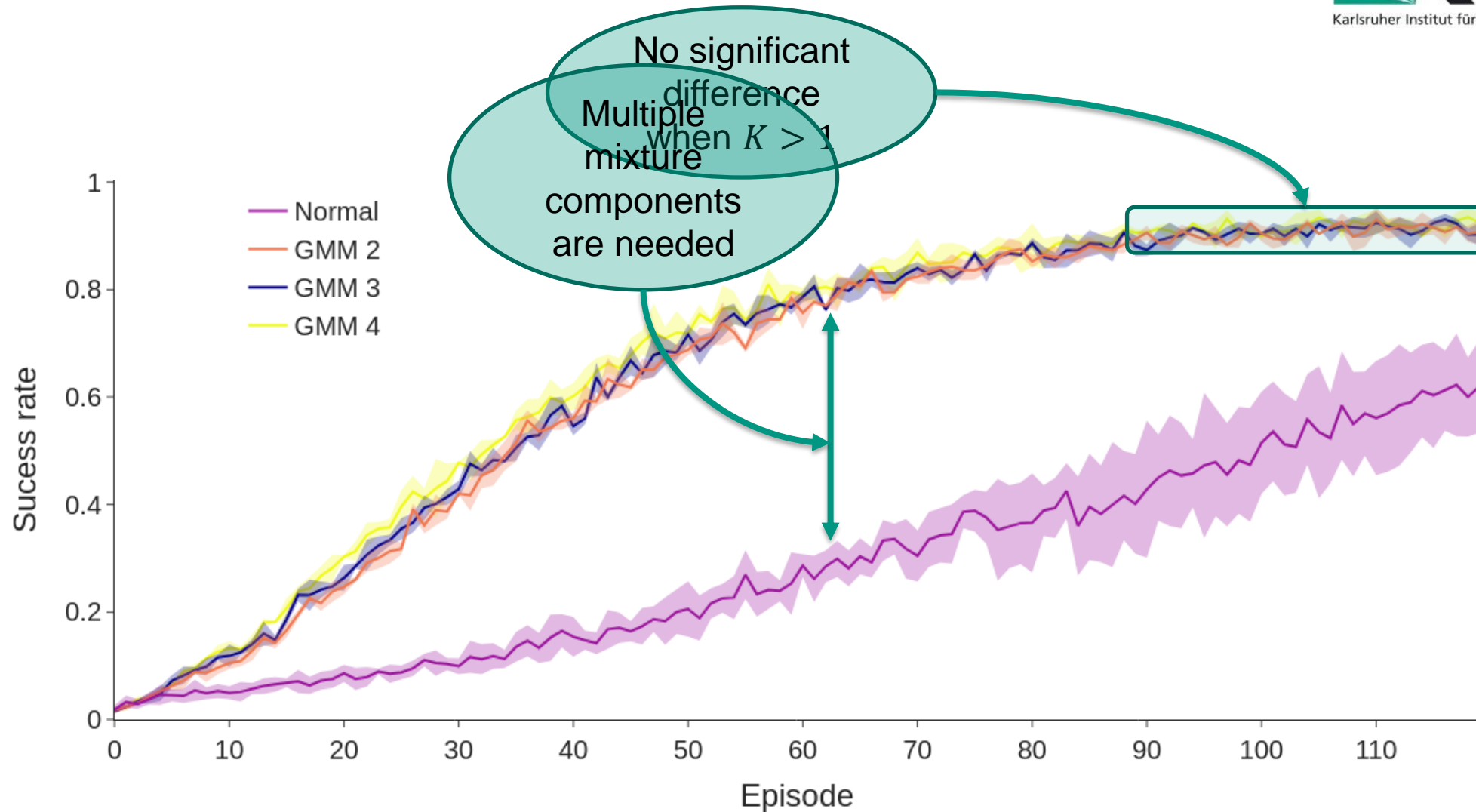
## Scenario 06



## Scenario 08



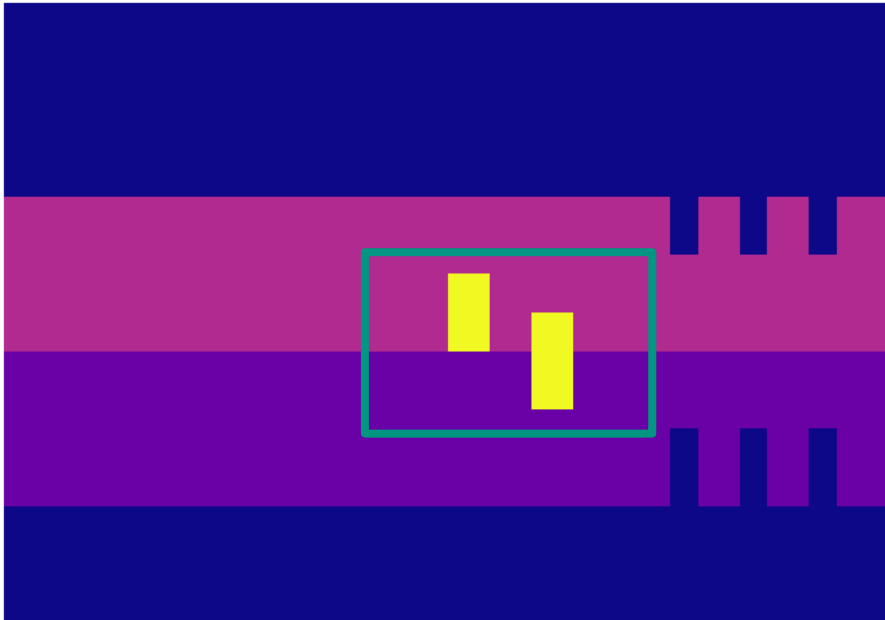
# Mixture Components (Scenario 08)



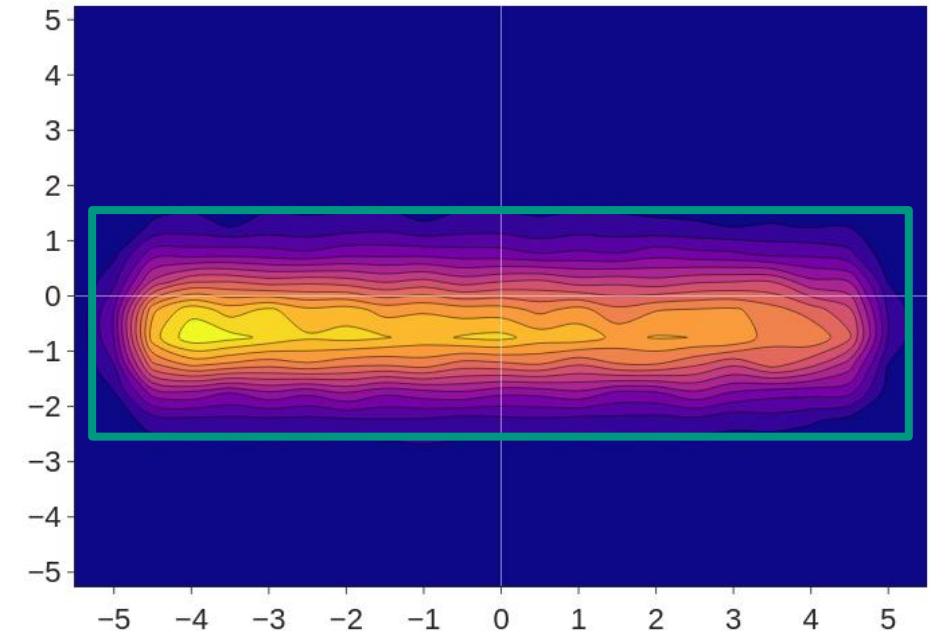
# Mixture Components

- Where do multiple components help?
- Sampling over whole longitudinal range helps avoid collisions!

Visual input



Sample space





# Conclusion & Outlook

## Conclusion

- Efficacy shown
- Moderate generalization
- Scenario dependent

## Limitations

- Not efficient
- Hard to scale & tune
- No model learning

## Future work

- Scale-up
- Variance scaling
- Integrating heuristics



# Reference list

- Electrek.co (2021). Watch Tesla Full Self-Driving Beta navigate through the eye of the system in impressive video. <https://electrek.co/2021/03/29/tesla-full-self-driving-beta-navigate-video/>. Accessed 2021-08-17.
- Kurzer, K., Engelhorn, F., & Zöllner, J. M. (2018, November). Decentralized cooperative planning for automated vehicles with continuous monte carlo tree search. In 2018 21st International Conference on Intelligent Transportation Systems (ITSC) (pp. 452-459). IEEE.
- Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... & Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *nature*, 529(7587), 484-489.
- Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., ... & Lillicrap, T. (2017). Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*.
- Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., ... & Hassabis, D. (2017). Mastering the game of go without human knowledge. *nature*, 550(7676), 354-359.
- Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., ... & Silver, D. (2020). Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839), 604-609.
- Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Pfaff, T., Weber, T., Buesing, L., & Battaglia, P. W. (2019). Combining q-learning and search with amortized value estimates. *arXiv preprint arXiv:1912.02807*.
- Moerland, T. M., Broekens, J., Plaat, A., & Jonker, C. M. (2018). A0C: Alpha zero in continuous action space. *arXiv preprint arXiv:1805.09613*.
- Yang, X., Duvaud, W., & Wei, P. (2020). Continuous Control for Searching and Planning with a Learned Model. *arXiv preprint arXiv:2006.07430*.
- Hubert, T., Schrittwieser, J., Antonoglou, I., Barekatin, M., Schmitt, S., & Silver, D. (2021). Learning and Planning in Complex Action Spaces. *arXiv preprint arXiv:2104.06303*.
- Petosa, N., & Balch, T. (2019). Multiplayer AlphaZero. *arXiv preprint arXiv:1910.13012*.

# Reference list

- Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., ... & Colton, S. (2012). A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1), 1-43.
- Kurzer, K., Fechner, M., & Zöllner, J. M. (2020). Accelerating Cooperative Planning for Automated Vehicles with Learned Heuristics and Monte Carlo Tree Search. *arXiv preprint arXiv:2002.00497*.
- Wang, Q., Wu, B., Zhu, P., Li, P., Zuo, W., & Hu, Q. (2020, June). ECA-Net: efficient channel attention for deep convolutional neural networks, 2020 IEEE. In *CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.
- Bacchiani, G., Molinari, D., & Patander, M. (2019). Microscopic traffic simulation by cooperative multi-agent deep reinforcement learning. *arXiv preprint arXiv:1903.01365*.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998-6008).
- Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018, July). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning* (pp. 1861-1870). PMLR.
- Wang, H., Emmerich, M., Preuss, M., & Plaat, A. (2019, December). Alternative loss functions in alphazero-like self-play. In *2019 IEEE Symposium Series on Computational Intelligence (SSCI)* (pp. 155-162). IEEE.

# Generalization Performance

Model Scenario	SC06	SC08	Reg. SC08	SC06 & SC08	SC06 & SC08 + Exp.	Random	Random + Exp.	Baseline
SC01	0.44	0.98	0.93	0.98	1.00	0.01	0.99	1.00
SC02	0.27	0.72	0.83	0.78	0.95	0.01	0.90	0.97
SC03	0.48	0.99	0.98	0.96	1.00	0.00	1.00	1.00
SC04	0.10	0.82	0.87	0.60	1.00	0.00	0.99	0.99
SC05	0.70	0.21	0.17	0.61	0.96	0.34	0.95	1.00
SC06	0.99	0.39	0.32	0.90	0.98	0.31	0.98	0.99
SC07	0.00	0.00	0.02	0.05	0.01	0.00	0.00	0.07
SC08	0.39	0.84	0.81	0.84	0.54	0.00	0.04	0.20
Mean 01-05, 07	0.3317	0.6200	0.6333	0.6630	0.8200	0.0600	0.8050	0.8383
Mean unseen	0.3400	0.5871	0.5886	0.6630	0.8200	0.0838	0.7313	0.7775
Mean	0.4212	0.6187	0.6162	0.7150	0.8050	0.0838	0.7313	0.7775

Trained model with heuristics is overall the strongest strong

# Success Rate

$$I_{success} = \max(1 - I_{collision} - I_{invalid} - I_{unableContinue}, 0)$$

$$P_{success} = \frac{1}{N} \sum_{n=1}^N I_{success}^n$$

- $I_{collision}$ : Indicator for occurred collision
- $I_{invalid}$ : Indicator for invalid state, e.g. driving off road
- $I_{unableContinue}$ : Situation where the algorithm was unable to find actions for an agent

# Numerical State in Detail

$$\mathbf{n}_i^{\text{dynamic}}(t) = (x_i(t), y_i(t), \dot{x}_i(t), \dot{y}_i(t), \ddot{x}_i(t), \ddot{y}_i(t), \phi_i(t))$$

$$\mathbf{n}_i^{\text{static}} = (\dot{x}_i^{\text{desire}}, l_i^{\text{desire}}, v_i^{\text{width}}, v_i^{\text{length}})$$

$$\mathbf{n}_i(t) = \mathbf{n}_i^{\text{dynamic}}(t - 7) \oplus \mathbf{n}_i^{\text{dynamic}}(t - 6) \oplus \dots \oplus \mathbf{n}_i^{\text{dynamic}}(t) \oplus \mathbf{n}_i^{\text{static}}$$

## Information at time step $t$

- ▶  $(x_i(t), y_i(t))$ : Position
- ▶  $(\dot{x}_i(t), \dot{y}_i(t))$ : Velocity
- ▶  $(\ddot{x}_i(t), \ddot{y}_i(t))$ : Acceleration
- ▶  $\phi_i(t)$ : Steering angle

## Static information

- ▶  $(\dot{x}_i^{\text{desire}}, l_i^{\text{desire}})$ : Target state
- ▶  $(v_i^{\text{width}}, v_i^{\text{length}})$ : Vehicle dimensions

## Agent state

- ▶ History of past 8 dynamic states
- ▶ Scalar state

# Training Wall Clock Time (Scenario 06)

- Results are averaged over three models

Iterations	50	100	200
Wall clock time	8h53m	14h28m	32h11m
Mean success rate	0.7686	0.7933	0.7867

- ▶ Fast training
- ▶ Decent success rate over tests with multiple iterations

- ▶ Trade-off between training time and success
- ▶ Best performing model

- ▶ 2.5x slower than 50 iterations
- ▶ Success rate moderately improved



# Resource Consumption

```
1 [|||||] 100.0% 13 [|||||] 100.0% 25 [|||||] 97.2% 37 [|||||] 100.0%
2 [|||||] 98.4% 14 [|||||] 100.0% 26 [|||||] 92.2% 38 [|||||] 100.0%
3 [|||||] 100.0% 15 [|||||] 100.0% 27 [|||||] 100.0% 39 [|||||] 100.0%
4 [|||||] 100.0% 16 [|||||] 100.0% 28 [|||||] 90.4% 40 [|||||] 100.0%
5 [|||||] 90.6% 17 [|||||] 100.0% 29 [|||||] 100.0% 41 [|||||] 100.0%
6 [|||||] 99.4% 18 [|||||] 100.0% 30 [|||||] 94.5% 42 [|||||] 100.0%
7 [|||||] 100.0% 19 [|||||] 100.0% 31 [|||||] 100.0% 43 [|||||] 100.0%
8 [|||||] 100.0% 20 [|||||] 100.0% 32 [|||||] 100.0% 44 [|||||] 98.9%
9 [|||||] 100.0% 21 [|||||] 100.0% 33 [|||||] 100.0% 45 [|||||] 100.0%
10 [|||||] 100.0% 22 [|||||] 100.0% 34 [|||||] 100.0% 46 [|||||] 99.5%
11 [|||||] 100.0% 23 [|||||] 100.0% 35 [|||||] 100.0% 47 [|||||] 100.0%
12 [|||||] 100.0% 24 [|||||] 100.0% 36 [|||||] 100.0% 48 [|||||] 100.0%
Mem[|||||] 43.7G/62.8G Tasks: 778, 3474 thr; 48 running
Swp[|||||] 1.49G/2.00G Load average: 67.82 41.64 23.05
Uptime: 10 days, 15:00:33
```

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
166835	ufpzp	20	0	785M	273M	69240	R	99.2	0.4	0:28.89	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 801c34227a8f4a1fb
166477	ufpzp	20	0	684M	171M	69544	R	97.6	0.3	0:51.20	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node cc6a4154d188442a9
166766	ufpzp	20	0	878M	366M	69824	R	97.6	0.6	0:34.09	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 64c140560c7d4527b
166994	ufpzp	20	0	848M	336M	69788	R	97.6	0.5	0:19.27	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node e0e8a0d05103431f9
167070	ufpzp	20	0	674M	162M	69504	R	97.6	0.3	0:15.84	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node bb4968d30fb64d818
167209	ufpzp	20	0	675M	163M	69464	R	97.6	0.3	0:07.88	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node eaf4ec79f604459c8
167299	ufpzp	20	0	627M	115M	60600	R	97.6	0.2	0:02.48	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 5f2f27da76774a44a
166930	ufpzp	20	0	795M	283M	69636	R	94.8	0.4	0:23.37	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node f1d4560474f246389
166888	ufpzp	20	0	760M	248M	69708	R	93.7	0.4	0:27.68	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 88c712f7a7ac4655b
167227	ufpzp	20	0	671M	159M	69420	R	93.2	0.2	0:06.93	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 089b4a8edc434c009
167066	ufpzp	20	0	684M	172M	69376	R	92.6	0.3	0:17.39	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node d77ad4ad8b9054934b
167191	ufpzp	20	0	722M	209M	69700	R	92.1	0.3	0:08.96	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 7afc746b2c9942e98
167094	ufpzp	20	0	667M	154M	69604	R	92.1	0.2	0:14.31	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 99b14a7514ba40e79
167127	ufpzp	20	0	692M	180M	69464	R	92.1	0.3	0:11.29	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 96696da744e14a428
166831	ufpzp	20	0	808M	296M	69636	R	92.1	0.5	0:28.21	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 046629d571024b52b
167220	ufpzp	20	0	676M	164M	69496	R	91.5	0.3	0:07.46	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 99ce9793818346239
166934	ufpzp	20	0	789M	276M	69672	R	89.3	0.4	0:23.98	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node ae63970b48774e259
167135	ufpzp	20	0	700M	187M	69376	R	88.8	0.3	0:09.97	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node bb0b5cbd5fe042d4b
167003	ufpzp	20	0	793M	280M	69528	R	88.2	0.4	0:20.21	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node bb3c5f1b0f3d4ad49
167195	ufpzp	20	0	667M	155M	69404	R	88.2	0.2	0:07.72	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node baa3dd600f5d4b329
167043	ufpzp	20	0	935M	422M	69288	R	87.1	0.7	0:17.12	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node d21a2ba2e1bf4bbd8
167184	ufpzp	20	0	711M	198M	69048	R	85.5	0.3	0:08.47	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 68427a9c76f44f7ca
167180	ufpzp	20	0	762M	249M	69532	R	84.4	0.4	0:08.99	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 4151e226847f481ea
167272	ufpzp	20	0	648M	136M	69456	R	82.7	0.2	0:04.58	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 454d26b8c543480db
167239	ufpzp	20	0	670M	158M	69624	R	82.2	0.2	0:06.30	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node bb1ef957f3914732b
167129	ufpzp	20	0	858M	345M	69460	R	81.6	0.5	0:11.27	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node f358531a406a4f238
167019	ufpzp	20	0	804M	292M	69648	R	81.0	0.5	0:18.53	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 40def9e0507b4eb0b
166988	ufpzp	20	0	816M	304M	69500	R	79.9	0.5	0:20.70	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 7db08f5f3ca5540f0b
167170	ufpzp	20	0	668M	155M	69360	R	79.9	0.2	0:09.61	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node a6aa56d903c94417a
167297	ufpzp	20	0	622M	109M	60120	R	79.4	0.2	0:02.15	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node e085586c61804ce28
166932	ufpzp	20	0	676M	163M	69476	R	78.8	0.3	0:24.09	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node b4979adfa5e546ffb
167109	ufpzp	20	0	759M	247M	69564	R	78.3	0.4	0:12.07	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 577d418e34dc4f7fb
167131	ufpzp	20	0	666M	154M	69164	R	77.2	0.2	0:11.53	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 4f88f963f934442b
166926	ufpzp	20	0	756M	244M	69624	R	77.2	0.4	0:24.52	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node d68aac5142904dd28
166907	ufpzp	20	0	813M	301M	69520	R	77.2	0.5	0:24.47	/home/ws/ufpzp/catkin_ws/devel isolated/ros_proseco_planning/lib/ros_proseco_planning/proseco_planning_node 6cec186da01713a9

1 Help F2 Setup F3 Search F4 Filter F5 Tree F6 Sort By F7 Nice F8 Nice F9 Kill F10 Quit



# Loss Function in Detail

$$\mathcal{L}(\phi) = \mathcal{L}_i^\pi(\phi) - \alpha_i \mathcal{L}_i^H(\phi) + \mathcal{L}_i^V(\phi)$$

## ■ Policy loss

$$\mathcal{L}^\pi(\phi) = D_{KL} \left( \pi_\phi(\mathbf{a}|\mathbf{s}) \parallel \hat{\pi}(\mathbf{a}|\mathbf{s}) \right) = \mathbb{E}_{\mathbf{a} \sim \pi_\phi(\mathbf{a}|\mathbf{s})} [\log \pi_\phi(\mathbf{a}|\mathbf{s}) - \log \hat{\pi}_\phi(\mathbf{a}|\mathbf{s})]$$

## ■ Entropy loss

$$\mathcal{L}^H(\phi) = H \left( \pi_\phi(\mathbf{a}|\mathbf{s}) \right) = \mathbb{E}_{\mathbf{a} \sim \pi_\phi(\mathbf{a}|\mathbf{s})} [-\log \pi_\phi(\mathbf{a}|\mathbf{s})]$$

## ■ Value loss

$$\mathcal{L}^V(\phi) = \mathbb{E}_{\mathbf{s} \sim \mathcal{D}} \left[ \left( V_\phi(\mathbf{s}) - \hat{V}(\mathbf{s}) \right)^2 \right]$$

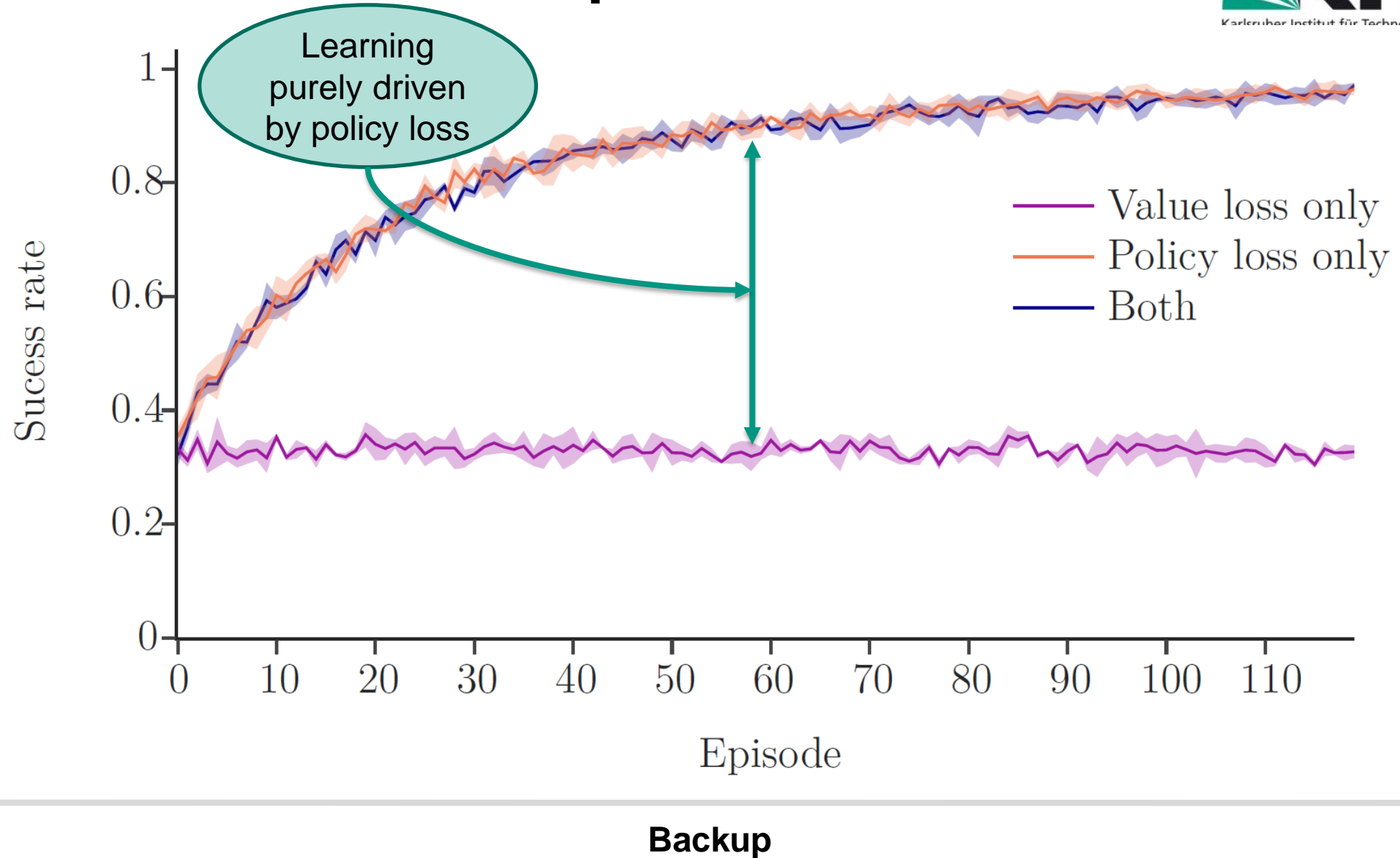
# Entropy Correction

- **Goal:** Express density of transformed distribution  $\pi(\mathbf{a}|\mathbf{s})$  in terms of untransformed distribution  $\mu(\mathbf{u}|\mathbf{s})$
- From the change of variables formula and the multivariable inverse function theorem, we know that

1. Plugging the derivative into  
 $\left| \det \frac{d\mathbf{a}}{d\mathbf{u}} \right|^{-1}$
2. Simplifying using the  
logarithm rules

$$\pi(\mathbf{a}|\mathbf{s}) = \mu(\mathbf{u}|\mathbf{s}) \left| \det \frac{d\mathbf{a}}{d\mathbf{u}} \right|^{-1}$$
$$= \log \mu(\mathbf{u}|\mathbf{s}) - \sum_{i=1}^D \log \left( 1 - \tanh^2(u_i) \right)$$

# Importance of Loss Components



# Reward for Scenario 08

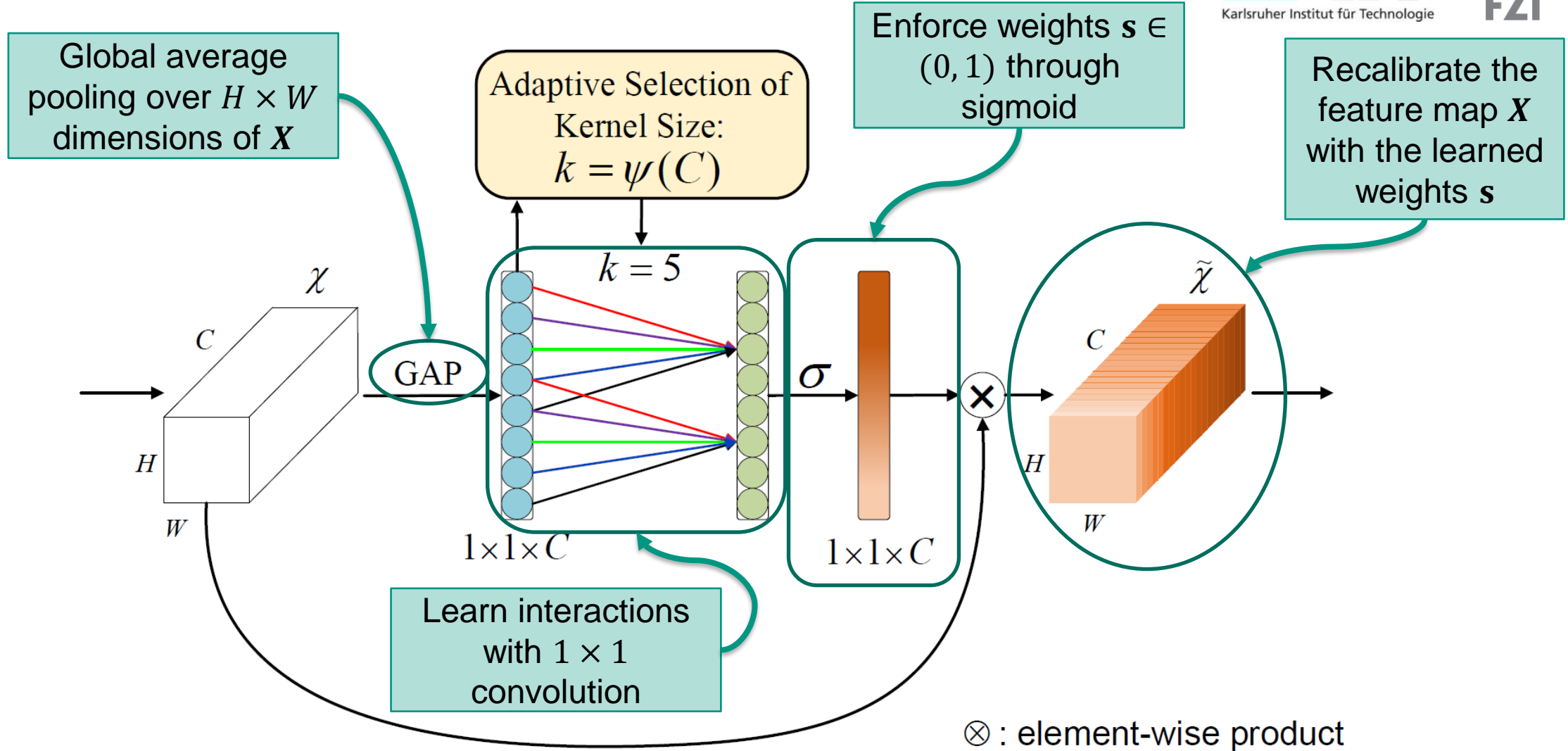
	Iterations	5	10	25	50	100	200	400	Mean
Model	Metric								
GMM 3	Success	0.3000	0.4967	0.6833	0.8267	0.8300	0.9433	0.9500	0.7186
	Reward	0.2779	0.2954	0.3119	0.3205	0.3276	0.3341	0.3367	0.3149
	Desire	0.0067	0.0200	0.0567	0.1767	0.1333	0.2900	0.3833	0.1524
Baseline	Success	0.0200	0.0100	0.0200	0.0900	0.2000	0.7900	0.8700	0.2857
	Reward	0.6738	0.6430	0.6404	0.6096	0.6145	0.6255	0.6238	0.6330
	Desire	0.0000	0.0100	0.0000	0.0000	0.0002	0.0002	0.1000	0.0214

- ▶ Baseline achieves highest reward for lowest success rate
- ▶ Reward is hand-crafted with possibly suboptimal parameters
- ▶ Other student worked on this via IRL

# Full Results for Scenario 06

	Iterations	5	10	25	50	100	200	400	Mean
Model	Metric								
GMM 3	Success	0.4367	0.5733	0.7133	0.8533	0.9467	0.9900	0.9933	<b>0.7867</b>
	Reward	0.2168	0.2345	0.2564	0.2755	0.2901	0.2986	0.3045	<b>0.2681</b>
	Desire	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	<b>0.0000</b>
Baseline	Success	0.8700	0.9600	1.0000	0.9900	0.9900	1.0000	1.0000	<b>0.9729</b>
	Reward	0.5798	0.4976	0.4943	0.4633	0.4791	0.5126	0.5157	<b>0.5060</b>
	Desire	0.0000	0.0200	0.0100	0.0300	0.0000	0.0000	0.0000	<b>0.0086</b>

# Efficient Channel Attention (ECA)



# Lookout & Pitfalls

