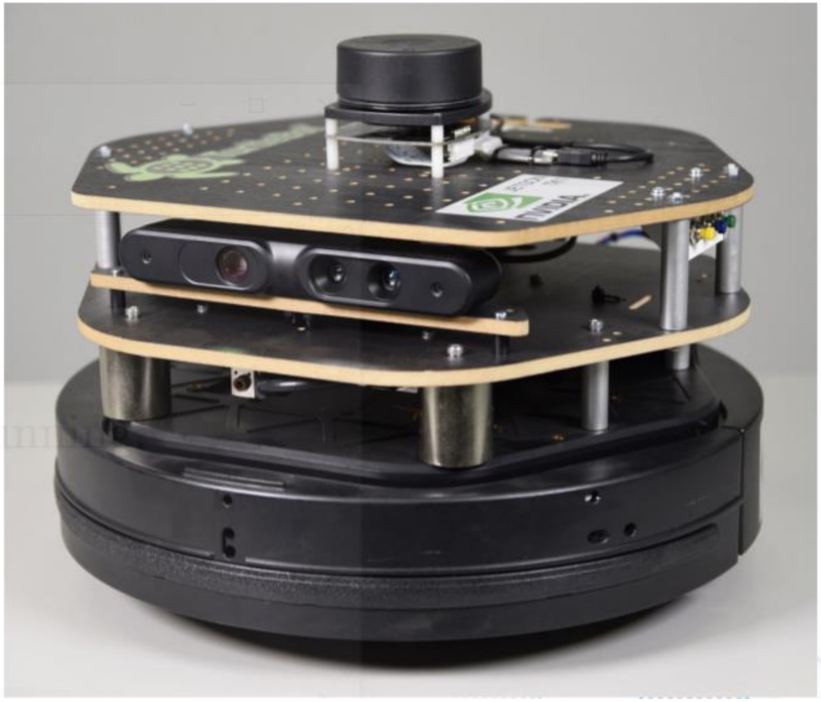
# Applied Reinforcement Learning

# Obstacle Avoidance

Alperen Gündogan

Rachid Ellouze

Uzair Akbar



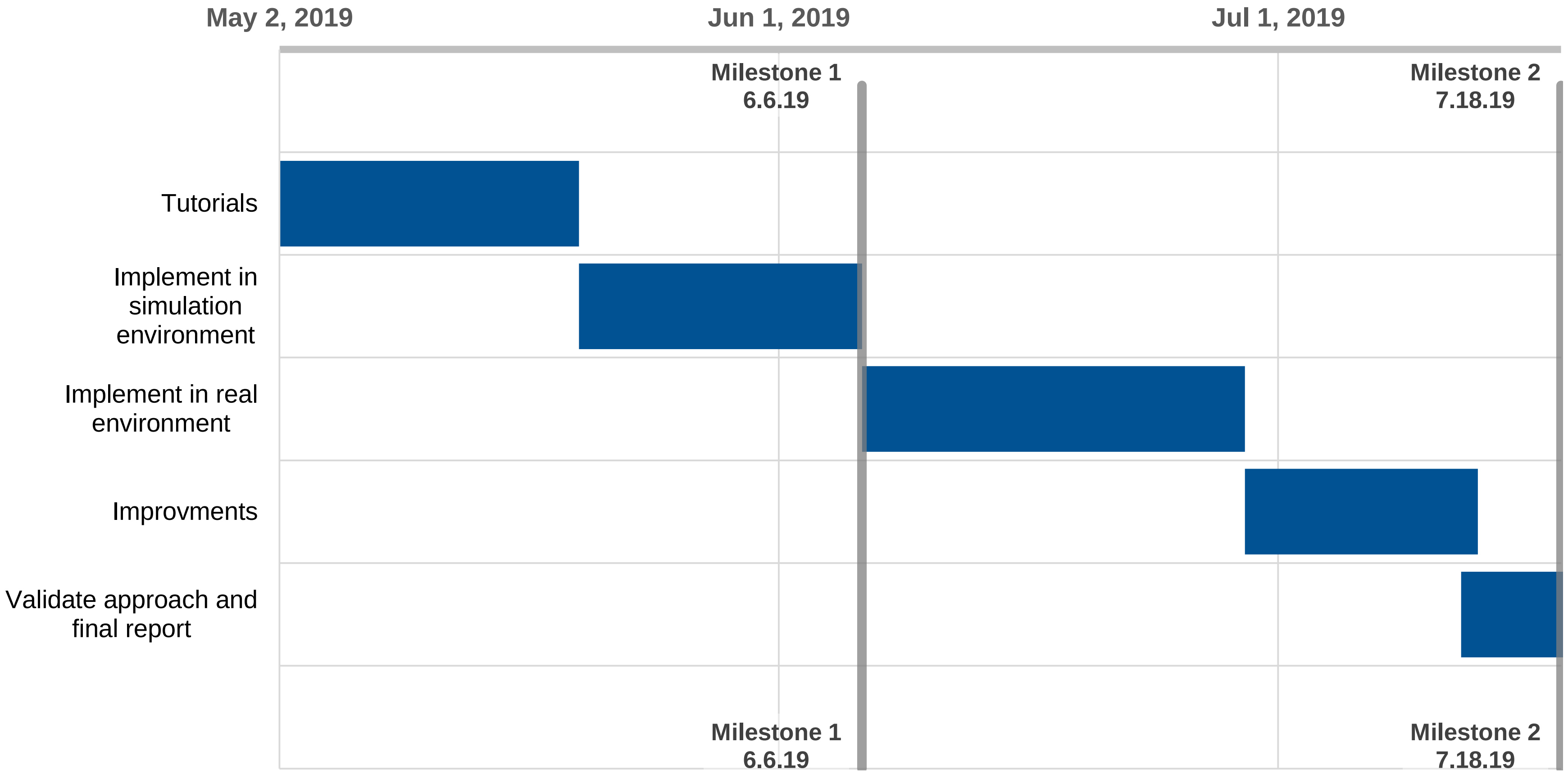
# Table of contents

1. Goals / Time Line Review
2. Ongoing Work:
   - Envirement Setup
   - Approaches
   - Results
3. Future Work

# Goal Review

- The turtlebot avoids collision.
- The turtlebot covers max ground in an episode

# Time Line Review

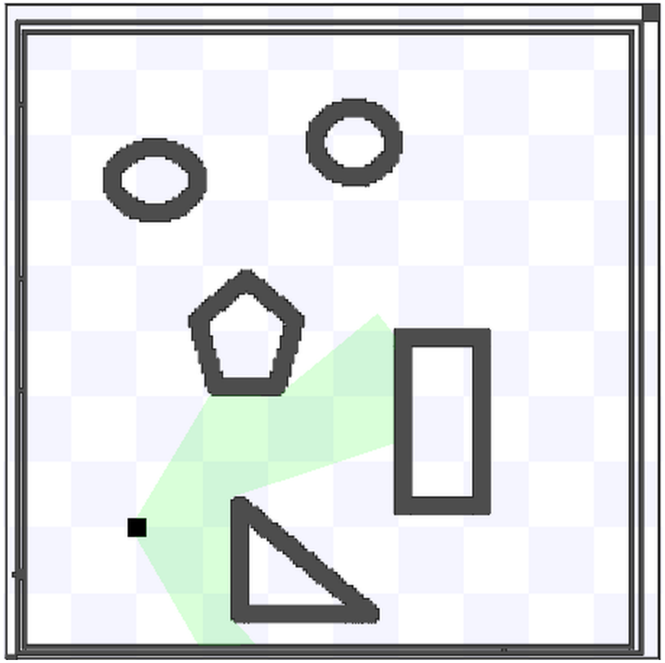


# Ongoing Work

- Environment setup
- Approaches
- Results

# Environment setup

- Familization with ROS: modules, topics, nodes, messages
- Stage simulator + Turtlebot
- Custom maps in Stage
- Integration with Pycharm: real-time debug possible



# States space

- State = (lidar, linear velocity, anglular velocity)

- Lidar: range + angles
- Ranges: 5 Values
  High resolution for small ranges. E.g. (0, 0.5), (0.5, 1.0)
  Low resolution for large ranges. E.g. (1.5, 2,5) (2,5, 6,5)

- Angles: (-60°, 60°) => 6 Values
  Each value the min range of an Interval.

- Lin. Velocity: (0.2, 0.4 m/s): updated from the taken action (not from the environment)
- Anglular velocity: (-30, -15, 0, 15 , 30) updated from the taken action (not from the environment)

**5^6 * 2 * 5 = 156250 States**

# Actions space

- (Linear velocity, angular velocity)
- Linear velocity: 2 values (0.2, 0.4 m/s)
- Angular velocity: 5 values (-30, -15, 0, 15 , 30)

- 2 approaches for taking an action.
- Combination of linear and angular velocity:  10 actions
- Either change linear velocity or angular velocity: 7 actions

# Reward function

$$\mathrm{r}(s) = \begin{cases} v.\cos w & if \quad episode\ running \\ -10 & if \quad collision \end{cases}$$

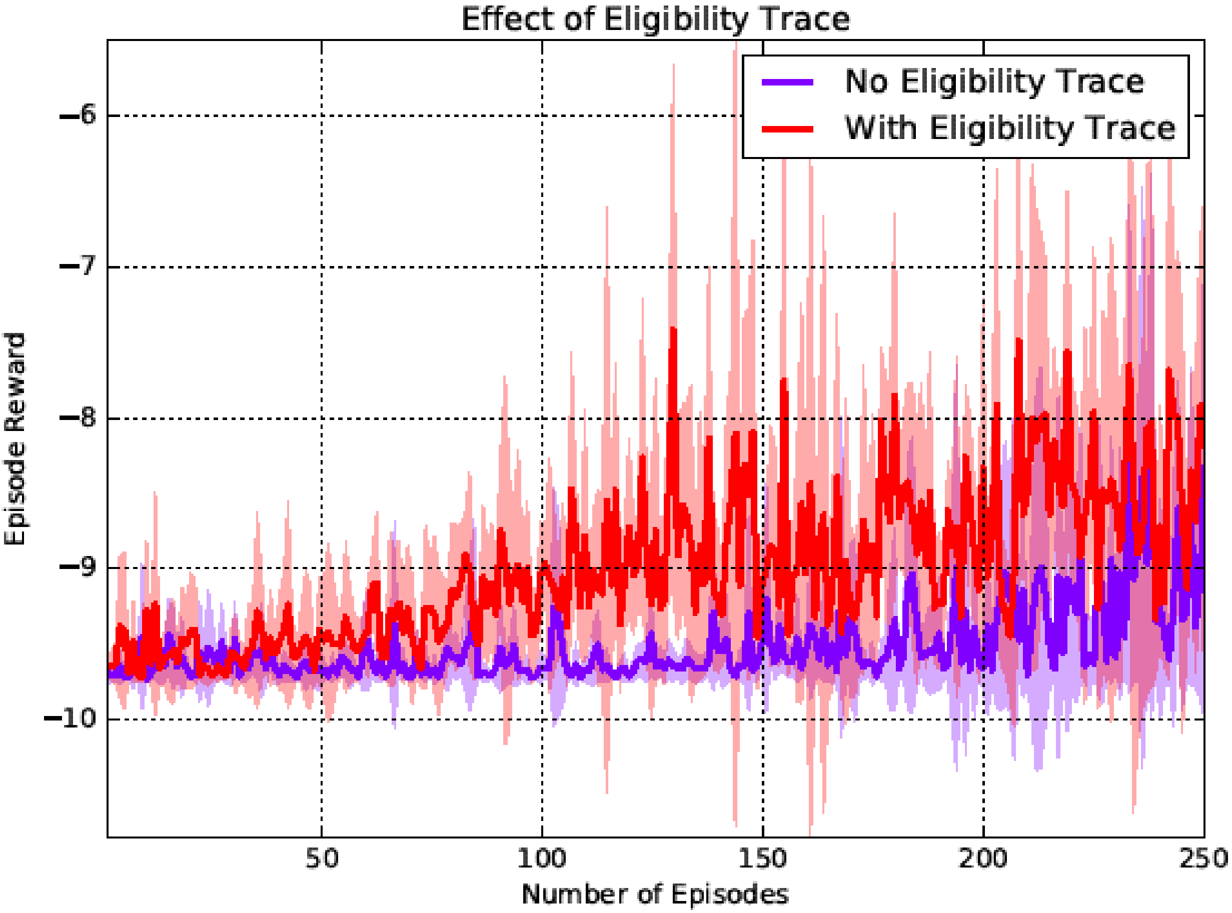
- Navigate as fast as possible without any collition
- Cosine term penalizes meaningless rotation

# Reinforcemnt Learning appraoches

- Q-Learning
- SARSA: eps-greedy polocy
- Start with high eps (0.9) and decrease during training

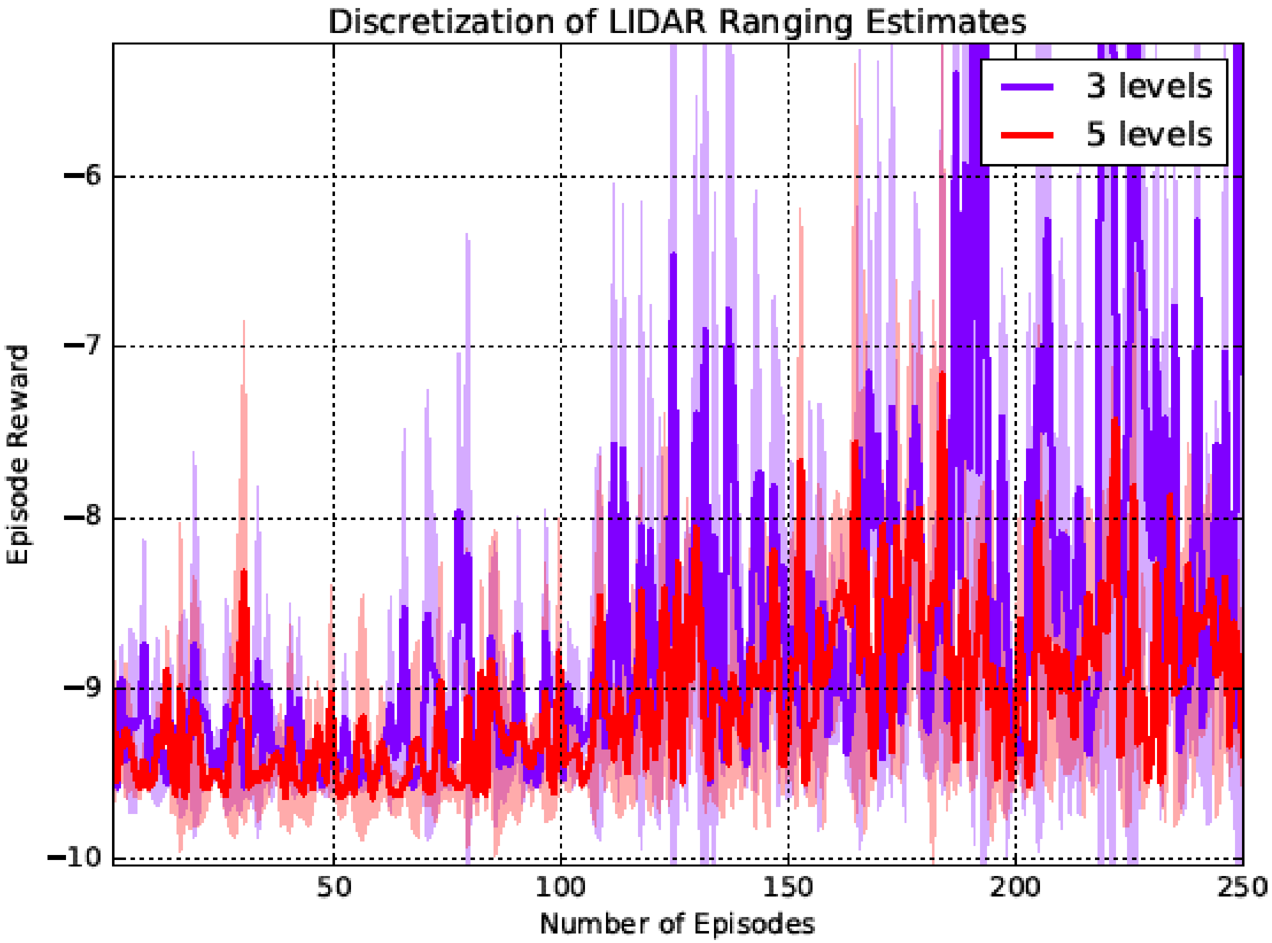
# Results

Eligibility trace vs no eligibility trace



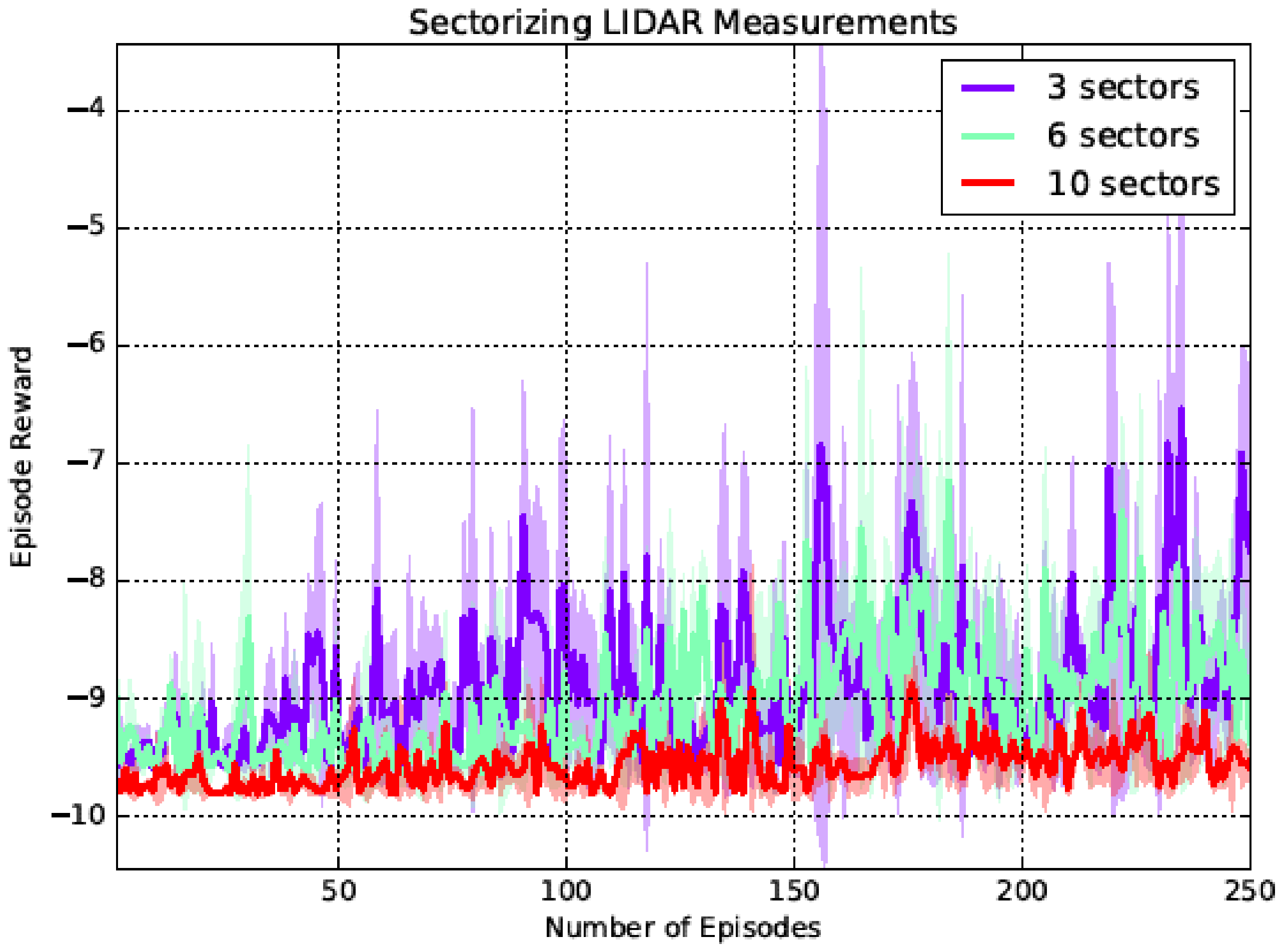
# Results

States spaces comparision



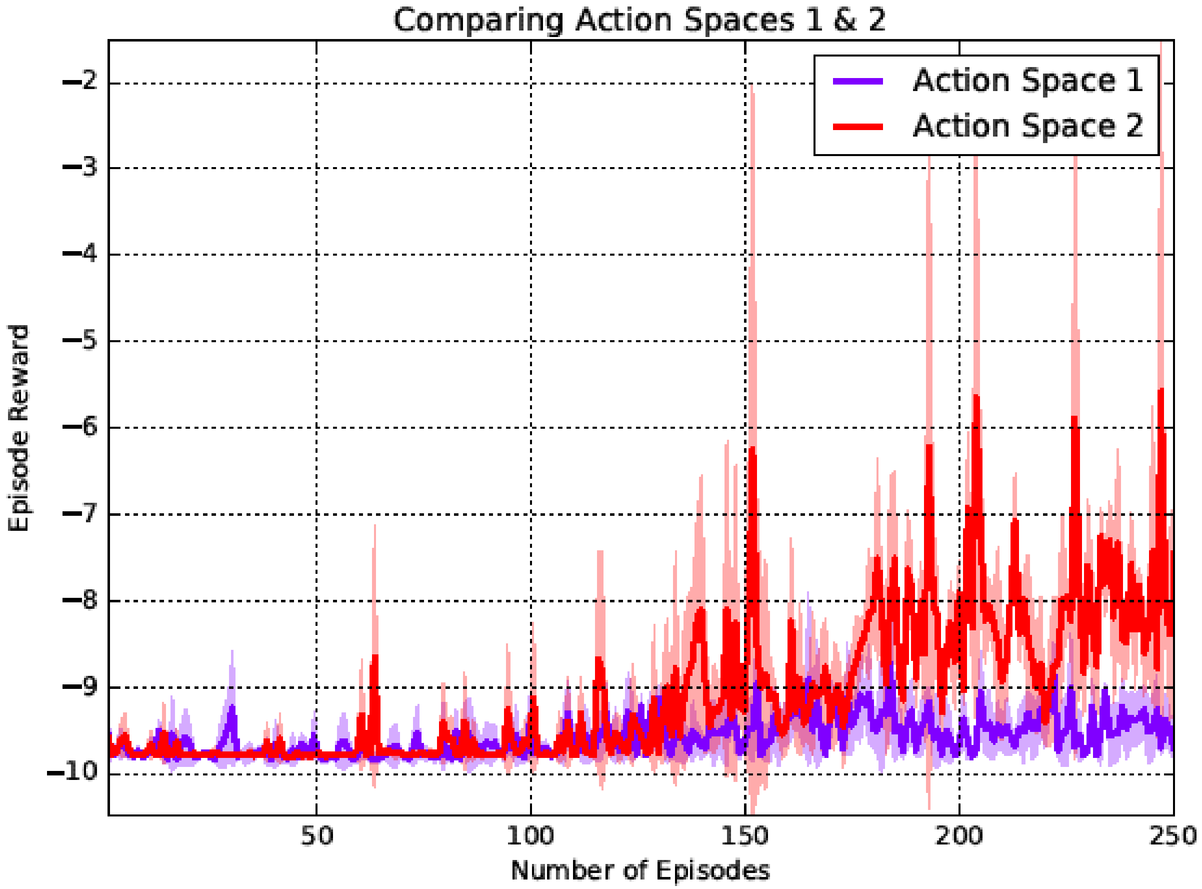
# Results

States spaces comparision



# Results

Action spaces comparision



# Demonstration

https://youtu.be/tGrZRYJZ-YE

# Future Work

- Port the algorithms to the real Turtlebot and real environment
- Other state space / action space / reward configurations
- SARSA with LVFA on eps-greedy policy
- discrete latent autoencoders for state space
- Softmax policy
- Validate a final approach

# Thank you for your attention !

Questions