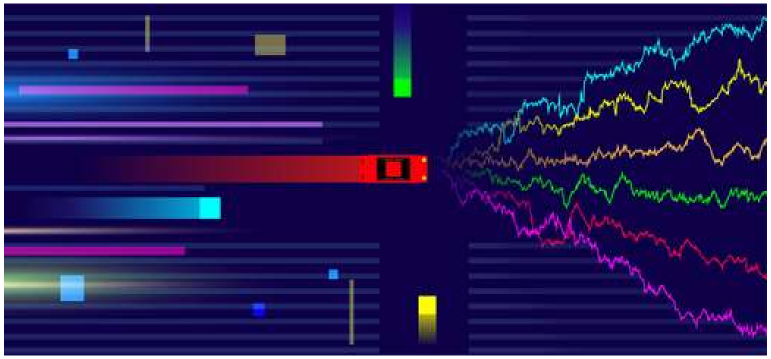
RL for Operations
Day 1: MDP Basics, VI+PI, Deep RL

Sean Sinclair, Sid Banerjee, Christina Yu
Cornell University



Plan for Today

MDP Basics

- Basic framework for Markov Decision Processes
- Tabular RL Algorithms with policy iteration + value iteration
- DeepRL algorithms (and their “tabular” counterparts)

Simulation Packages

- OpenAI Framework for simulation design
- Existing packages and code-bases for RL algorithm development

Simulation Implementation

- Develop simulator for problem using OpenAI Gym API

Tabular RL Algorithms

- Implement basic tabular RL algorithms to understand key algorithmic design aspects of *value estimates + value iteration*, *policy iteration*

Plan for Today

MDP Basics

- Basic framework for Markov Decision Processes
- Tabular RL Algorithms with policy iteration + value iteration
- DeepRL algorithms (and their "tabular" counterparts)

Simulation Packages

- OpenAI Framework for simulation design
- Existing packages and code-bases for RL algorithm development

Simulation Implementation

- Develop simulator for problem using OpenAI Gym API

Tabular RL Algorithms

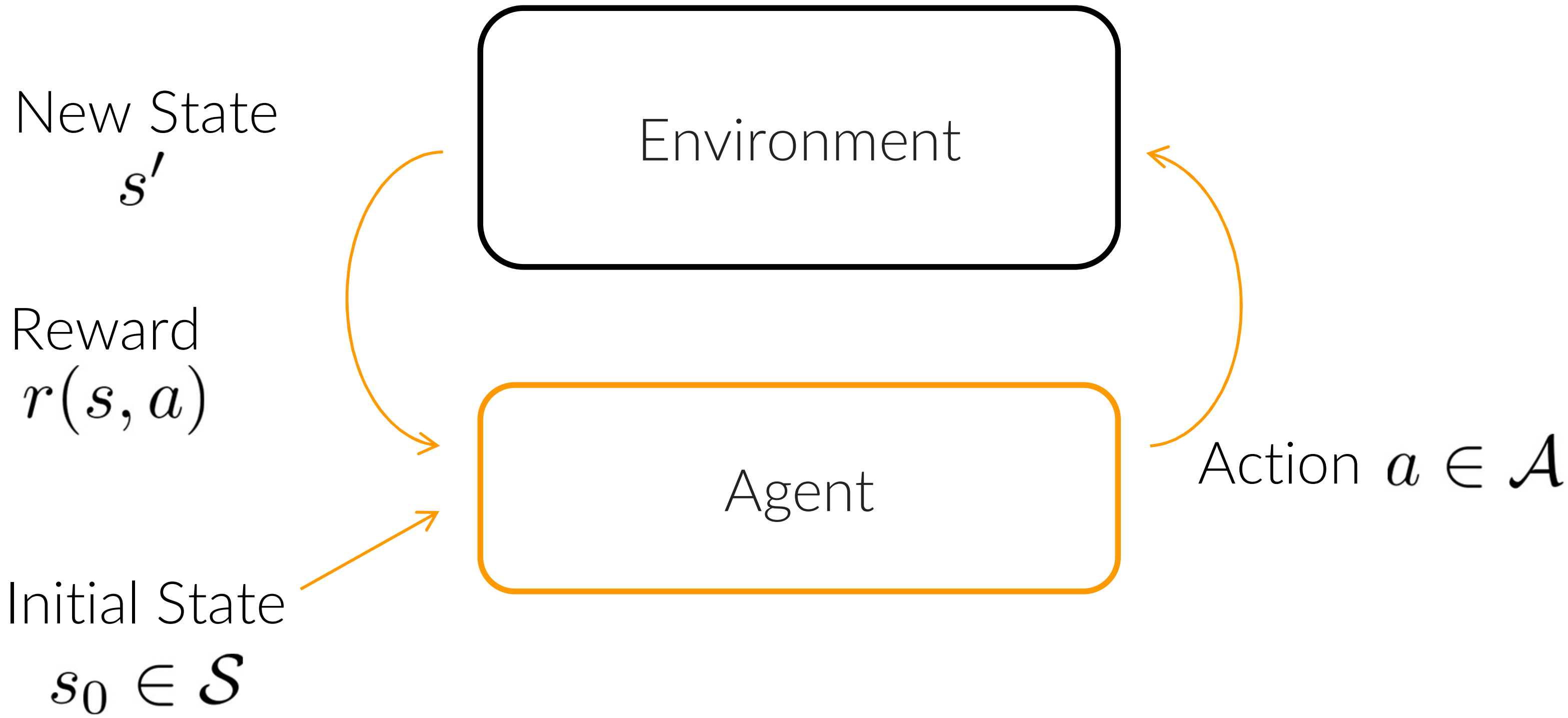
- Implement basic tabular RL algorithms to understand key algorithmic design aspects of *value estimates + value iteration*, *policy iteration*

Tabular RL Algorithms: Q-Learning and UCBVI

Sean Sinclair,
Cornell University

Markov Decision Process (MDP)

Environment: Determine reward and new state



Policy: Determine action based on state

Model Based

Estimate reward and transition via empirical:

$\overline{r}_h^k(s,a) = \frac{1}{n_h(s,a)} \sum_{(s,a)\in\mathcal{D}^k} R_h^k$ $\overline{T}_h^k(\cdot \mid s,a) = \frac{1}{n_h(x,a)} \sum_{(s,a,S_{h+1}^{k'})\in\mathcal{D}^k} \delta_{S_{h+1}^{k'}}$

$n_h(s,a)$ Number of times (s,a) visited

Plug estimates into Bellman Optimality Equations

$$\overline{V}_h^k(s) = \max_{a\in\mathcal{A}} \overline{Q}_h^k(s,a)$$

$$\overline{Q}_h^k(s,a) = \overline{r}_h^k(s,a) + \mathbb{E}_{S'\sim\overline{T}_h^k(\cdot|s,a)}[\overline{V}_{h+1}^k(S')] + \iota \frac{1}{\sqrt{n_h^k(s,a)}}$$

$$\pi_h^k(s) = \operatorname*{argmax}_{a\in\mathcal{A}} \overline{Q}_h^k(s,a)$$

Empirical value iteration with reward and transition estimates

Model Free

Results in following update procedure:

$$\overline{V}_h^k(s) = \max_{a \in \mathcal{A}} \overline{Q}_h^k(s, a)$$

$$\overline{Q}_h^{k+1}(S_h^k, A_h^k) = (1 - \alpha_t)\overline{Q}_h^k(S_h^k, A_h^k) + \alpha_t(R_h^k + \overline{V}_h^k(S_{h+1}^k) + \iota\frac{1}{\sqrt{t}})$$

$$\pi_h^k(s) = \operatorname*{argmax}_{a \in \mathcal{A}} \overline{Q}_h^k(s, a)$$

Empirical fixed point iteration with exploration bonuses

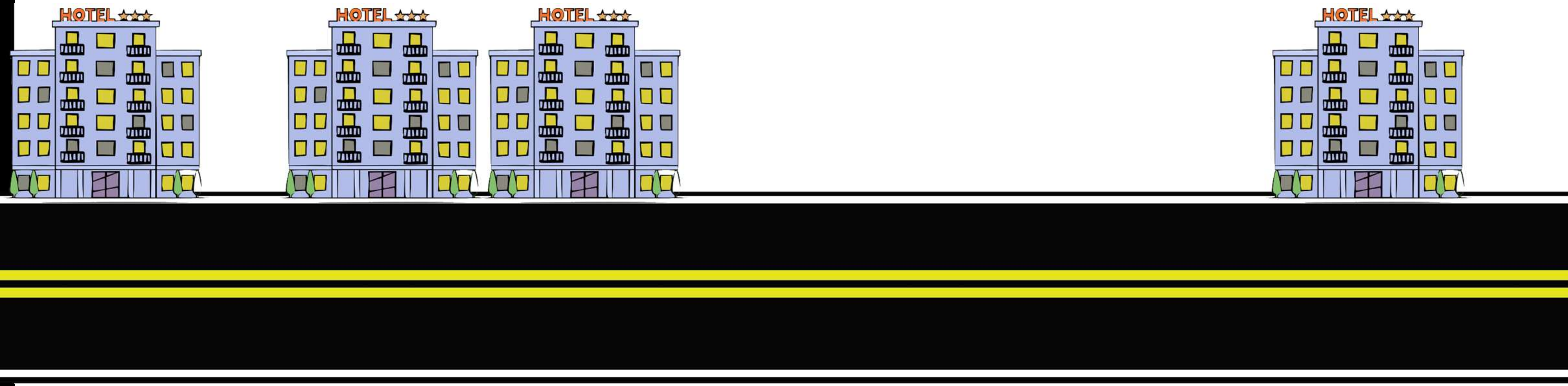
This Code Demo

- Implement basic tabular RL algorithms to understand key algorithmic design aspects of *value estimates + value iteration*, *policy iteration*

- Run experiments on 'WindyGridWorld' and the ambulance problem on a graph (i.e. metrical task systems)

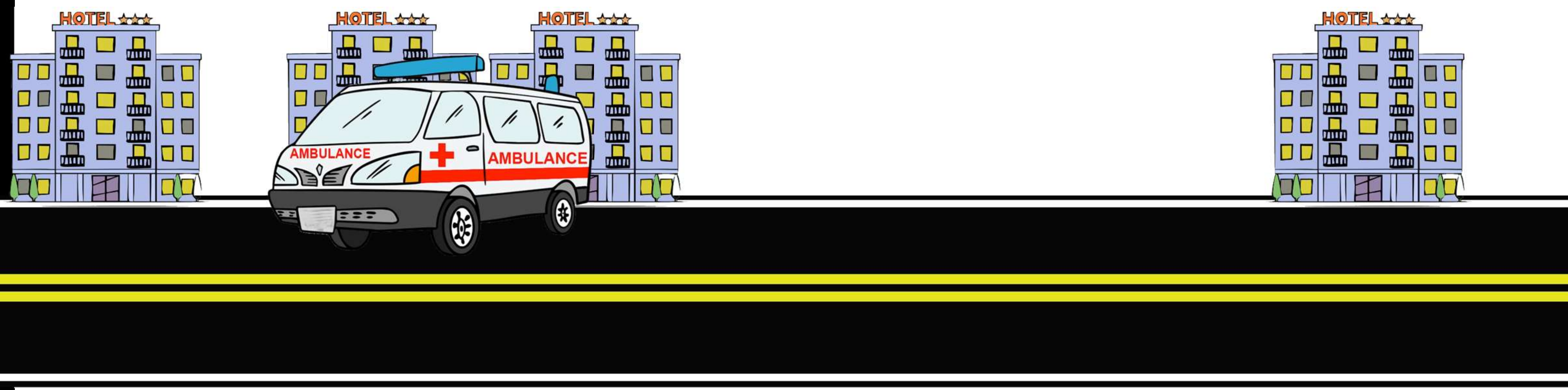
Ambulance Routing

- Operator decides location to station ambulance, paying a transportation cost
- Random request realized, ambulance pays cost for travel delay to serve patient
- Goal: learn policy which minimizes costs w/o knowledge of arrival distribution



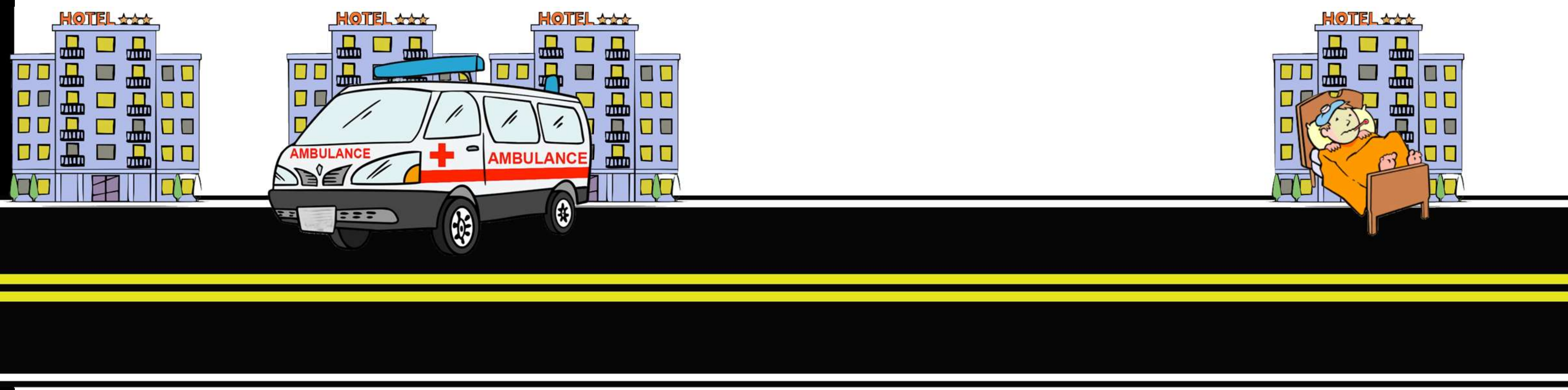
Ambulance Routing

- Operator decides location to station ambulance, paying a transportation cost
- Random request realized, ambulance pays cost for travel delay to serve patient
- Goal: learn policy which minimizes costs w/o knowledge of arrival distribution



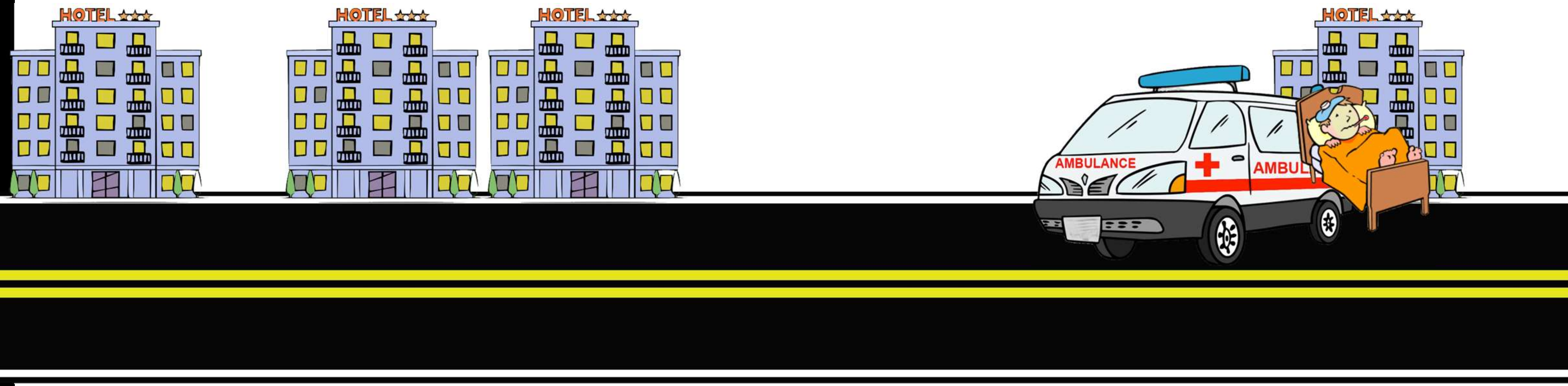
Ambulance Routing

- Operator decides location to station ambulance, paying a transportation cost
- Random request realized, ambulance pays cost for travel delay to serve patient
- Goal: learn policy which minimizes costs w/o knowledge of arrival distribution



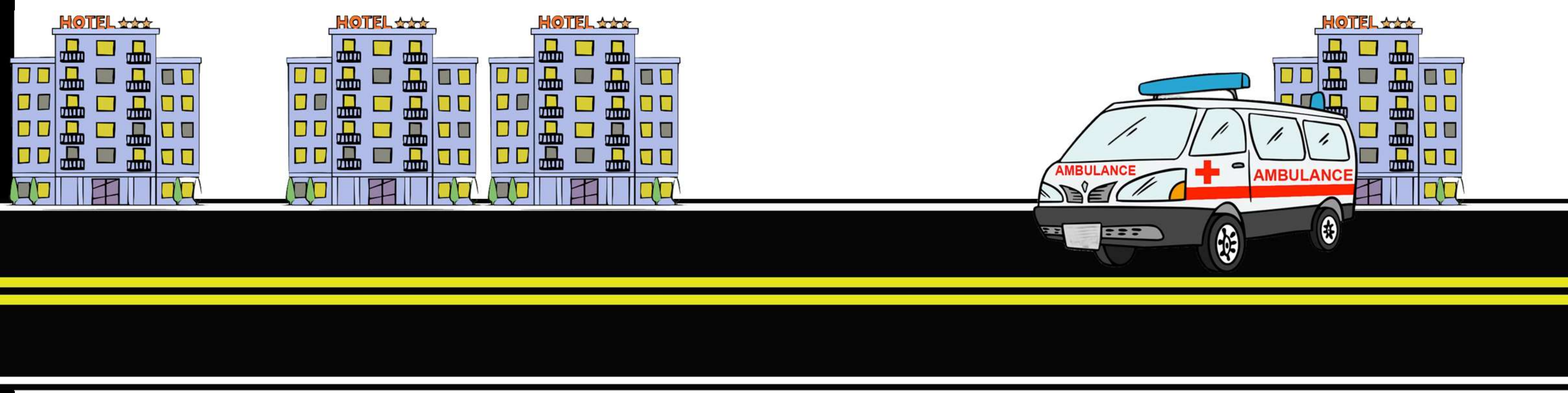
Ambulance Routing

- Operator decides location to station ambulance, paying a transportation cost
- Random request realized, ambulance pays cost for travel delay to serve patient
- Goal: learn policy which minimizes costs w/o knowledge of arrival distribution



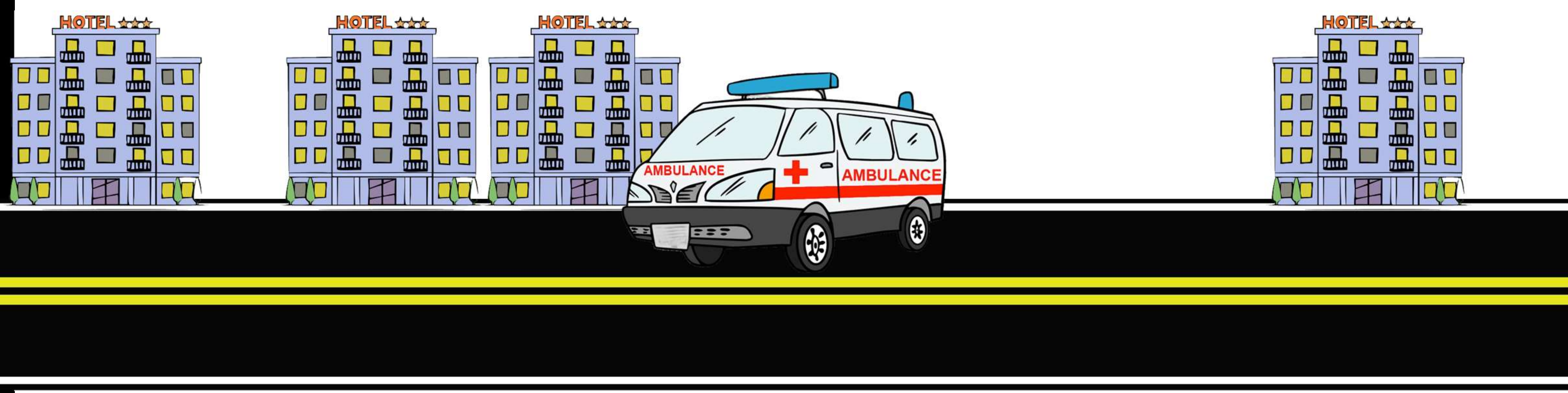
Ambulance Routing

- Operator decides location to station ambulance, paying a transportation cost
- Random request realized, ambulance pays cost for travel delay to serve patient
- Goal: learn policy which minimizes costs w/o knowledge of arrival distribution



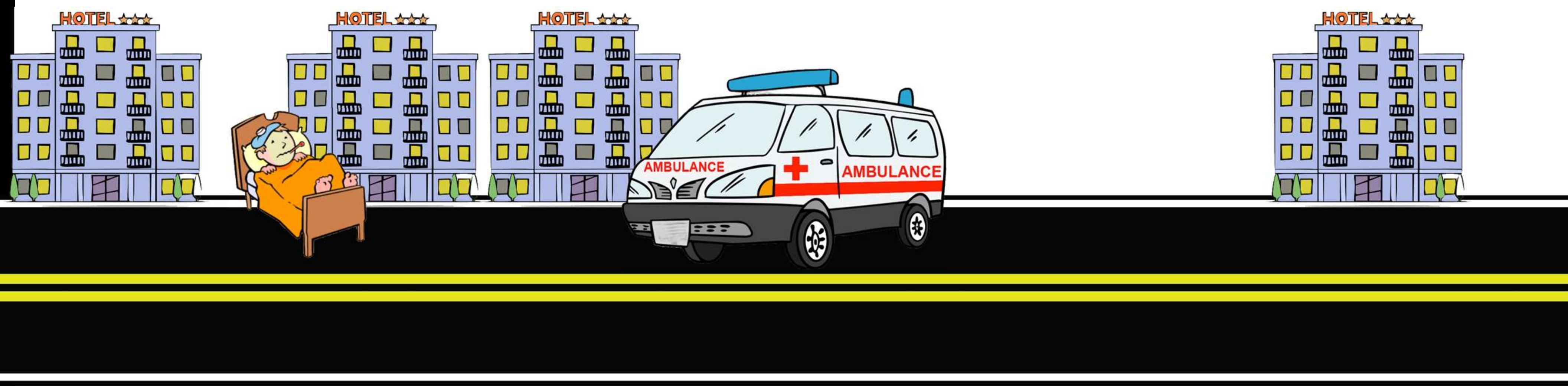
Ambulance Routing

- Operator decides location to station ambulance, paying a transportation cost
- Random request realized, ambulance pays cost for travel delay to serve patient
- Goal: learn policy which minimizes costs w/o knowledge of arrival distribution



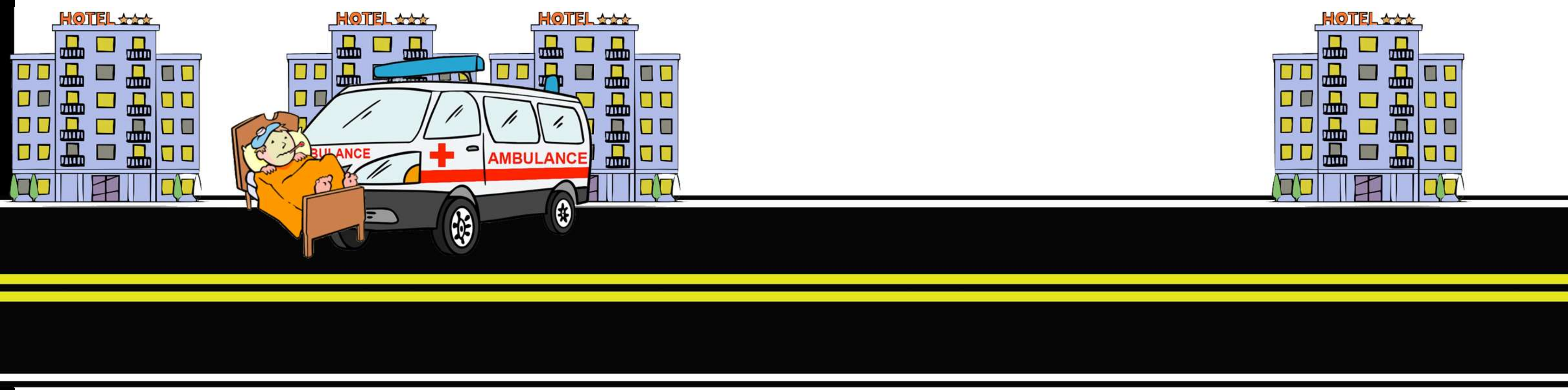
Ambulance Routing

- Operator decides location to station ambulance, paying a transportation cost
- Random request realized, ambulance pays cost for travel delay to serve patient
- Goal: learn policy which minimizes costs w/o knowledge of arrival distribution



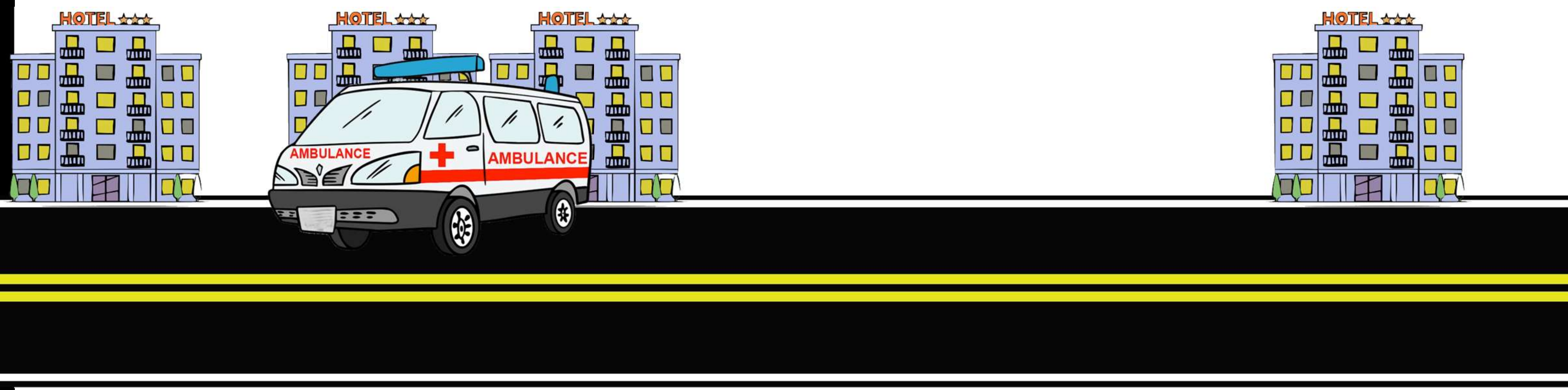
Ambulance Routing

- Operator decides location to station ambulance, paying a transportation cost
- Random request realized, ambulance pays cost for travel delay to serve patient
- Goal: learn policy which minimizes costs w/o knowledge of arrival distribution



Ambulance Routing

- Operator decides location to station ambulance, paying a transportation cost
- Random request realized, ambulance pays cost for travel delay to serve patient
- Goal: learn policy which minimizes costs w/o knowledge of arrival distribution



References

https://github.com/seanrsinclair/RLinOperations