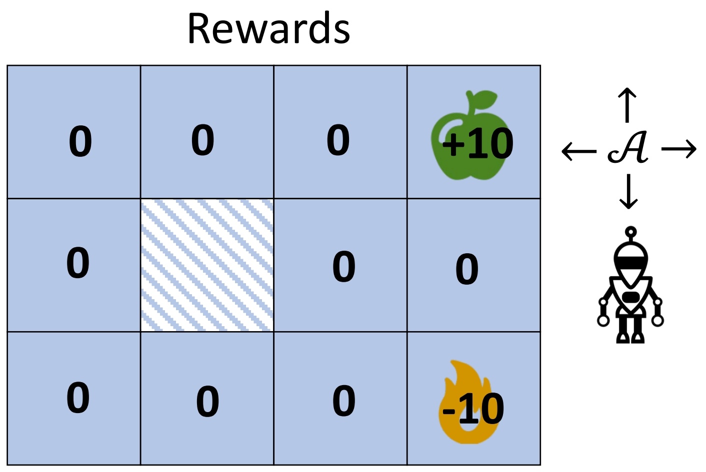
CSE 574 Planning and Learning Methods in AI

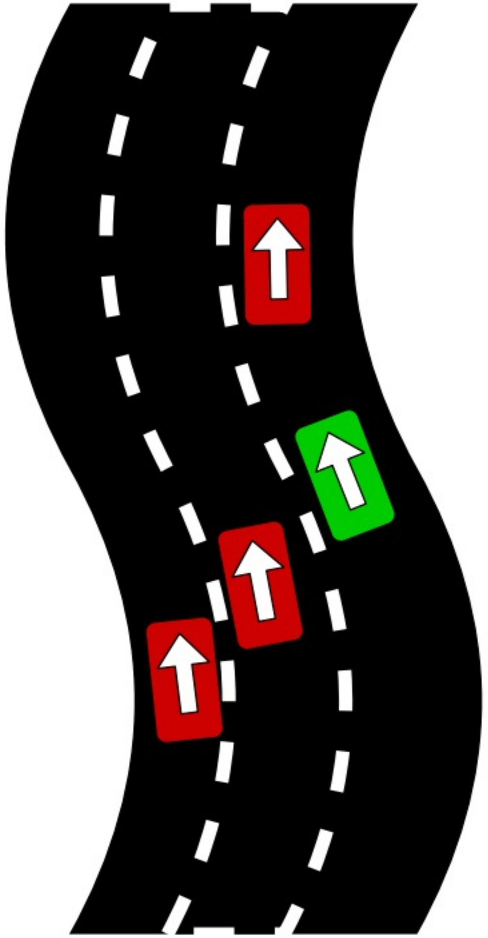
Ransalu Senanayake

Week 6

We have to define States, Actions, Transition Models, and Rewards



Lane Keeping and Changing



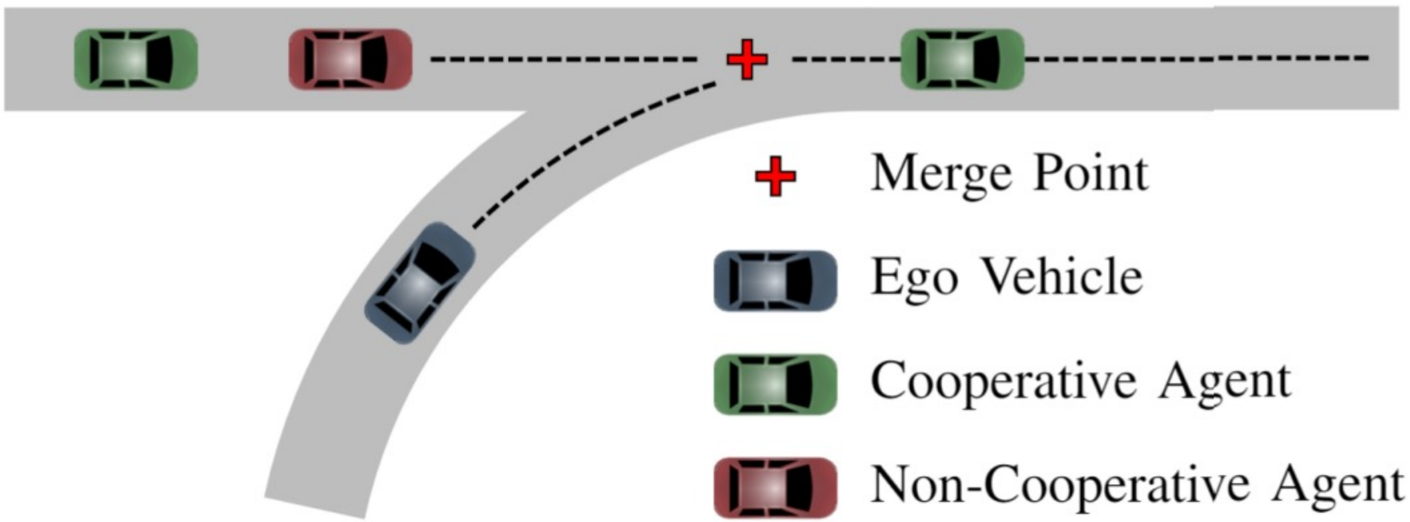
States: Position and velocity of all vehicles

Actions: Acceleration, lane change decision of the ego vehicle

Transition model: Based on position, velocity, and acceleration

Reward: Go as fast as possible keeping a safer distance

Merging into a Highway: States



The state of a given driving scene,

$$s = (V_E,\ V_T,\ c_T,\ V_L) \in \mathcal{S},$$

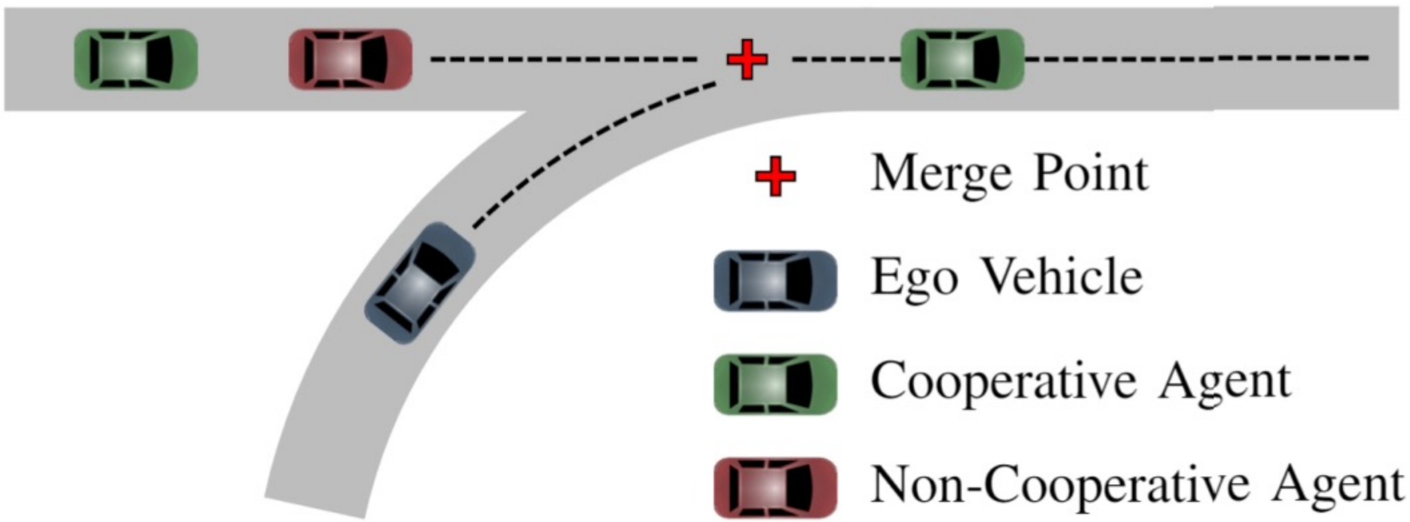
consists of the observable states V_E, V_T, V_L of the ego, trailing agent, and lead agent, respectively, and the latent cooperation level c_T which governs the C-IDM for the trailing agent. The observable vehicle state

$$V_i = (x_i,\ y_i,\ v_i,\ \dot{v}_i,\ \theta_i)$$

consists of the ith vehicle's position $(x_i,\ y_i)$ on the simulator map, longitudinal velocity v_i, longitudinal acceleration $\dot{v}_i$, and heading angle θ_i.

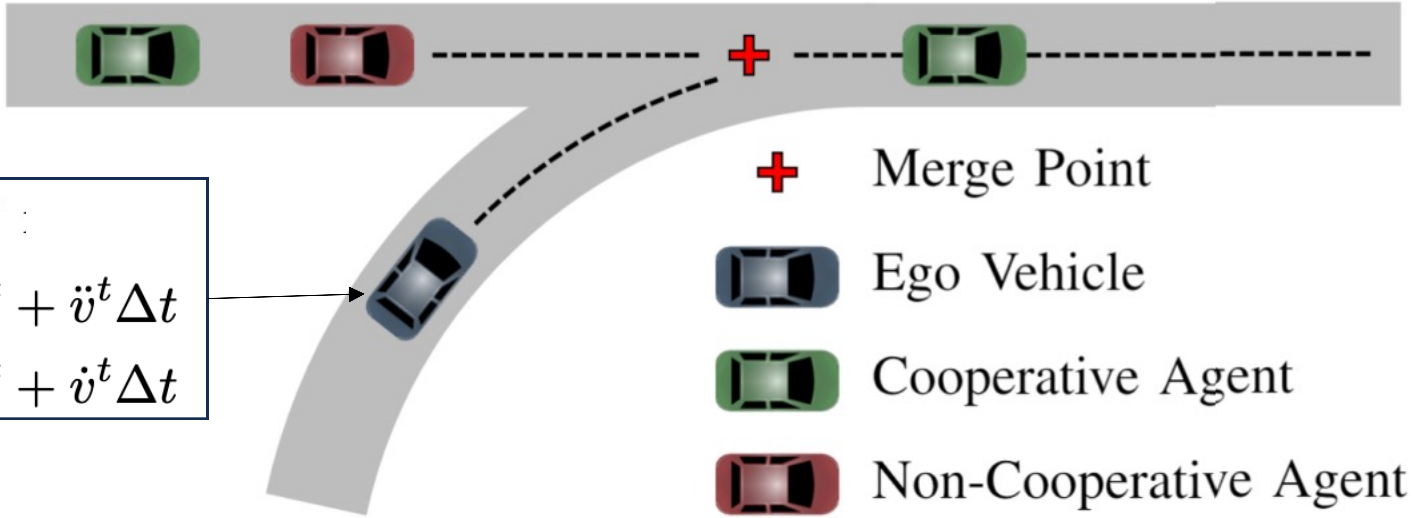
L. Kruse, E. Yel, R. Senanayake, and M. Kochenderfer, Uncertainty-Aware Online Merge Planning with Learned Driver Behavior

Merging into a Highway: Actions



At each time step, the ego takes an action $a \in \mathcal{A}$ which consists of a longitudinal jerk value. Planning occurs in jerk space to produce smooth velocity profiles. Jerk values are restricted to the range $-0.6\,\mathrm{m/s^3} \leq a \leq 0.6\,\mathrm{m/s^3}$ to prevent unrealistic or unsafe motion.

Merging into a Highway: Transition Model



$$v^{t+1} = v^t + \dot{v}^t \Delta t$$

$$x^{t+1} = x^t + v^t \Delta t + \frac{1}{2} \dot{v}^t \Delta t^2$$

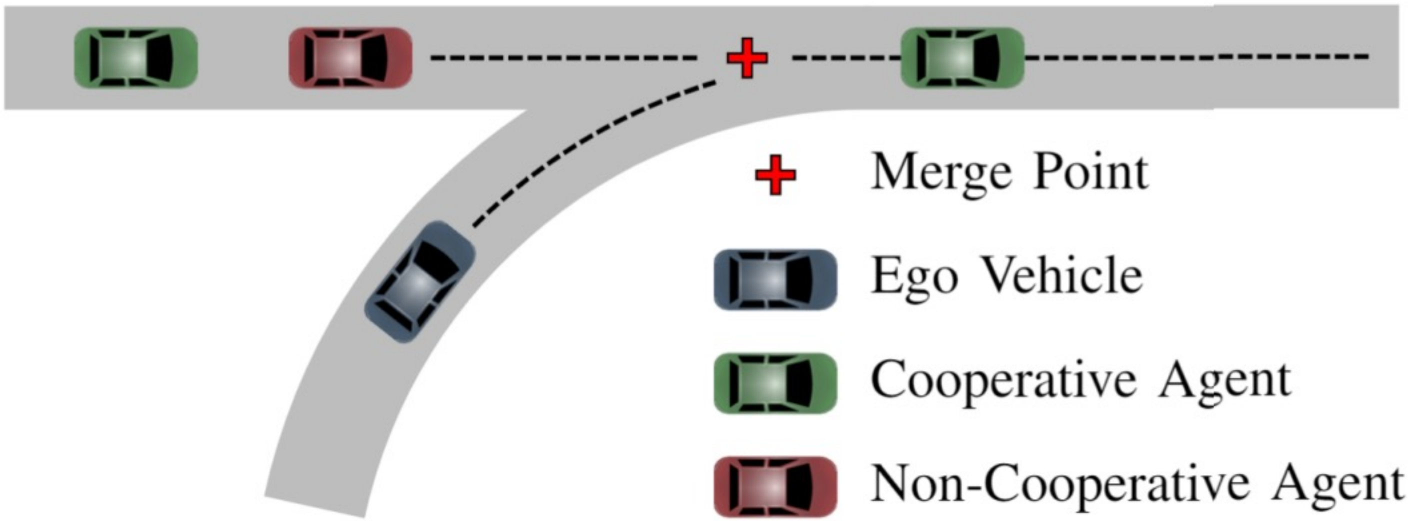
$$a = \ddot{v}^t$$

$$\dot{v}^{t+1} = \dot{v}^t + \ddot{v}^t \Delta t$$

$$v^{t+1} = v^t + \dot{v}^t \Delta t$$

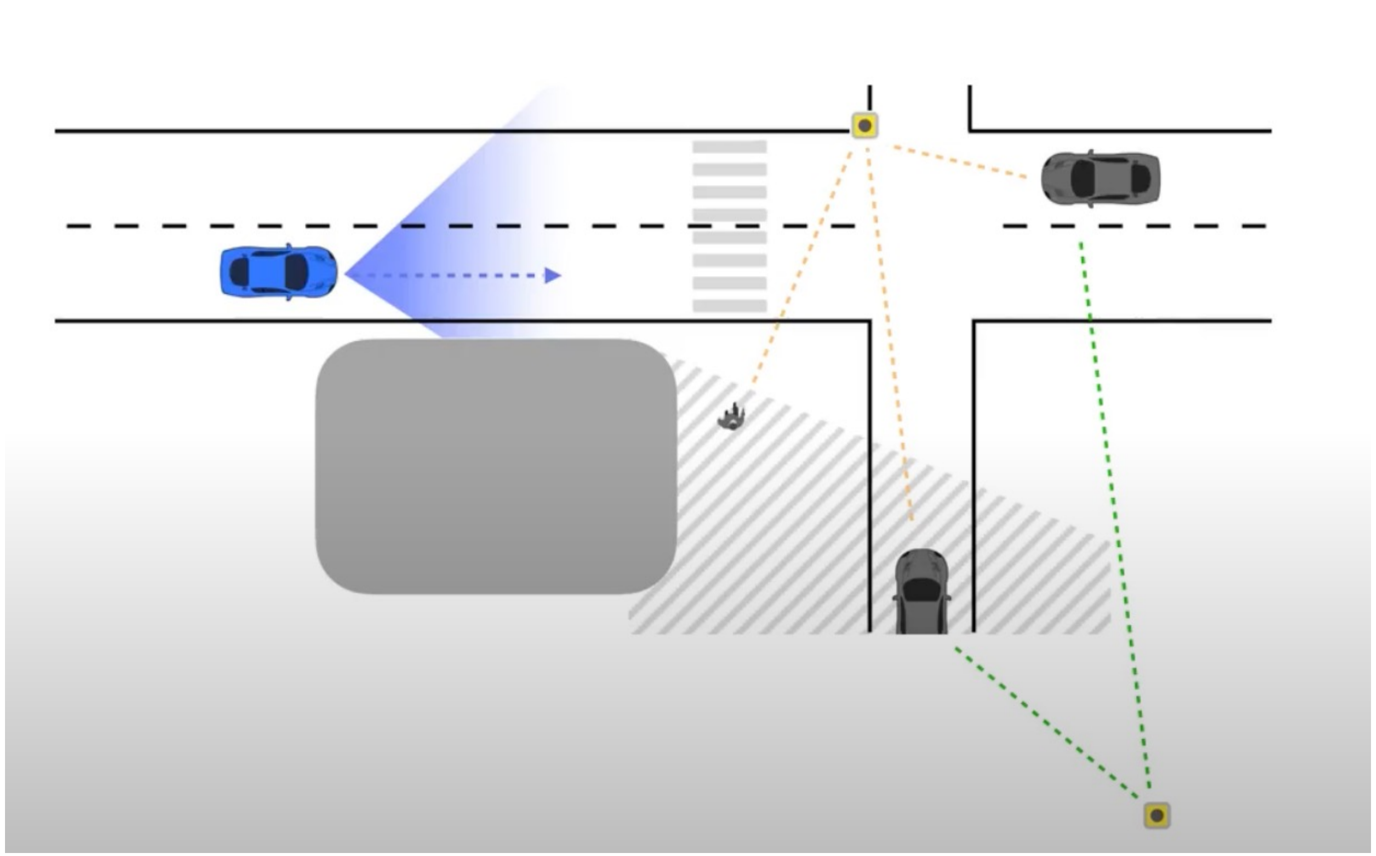
$$\Pr(s' \mid s, a)$$

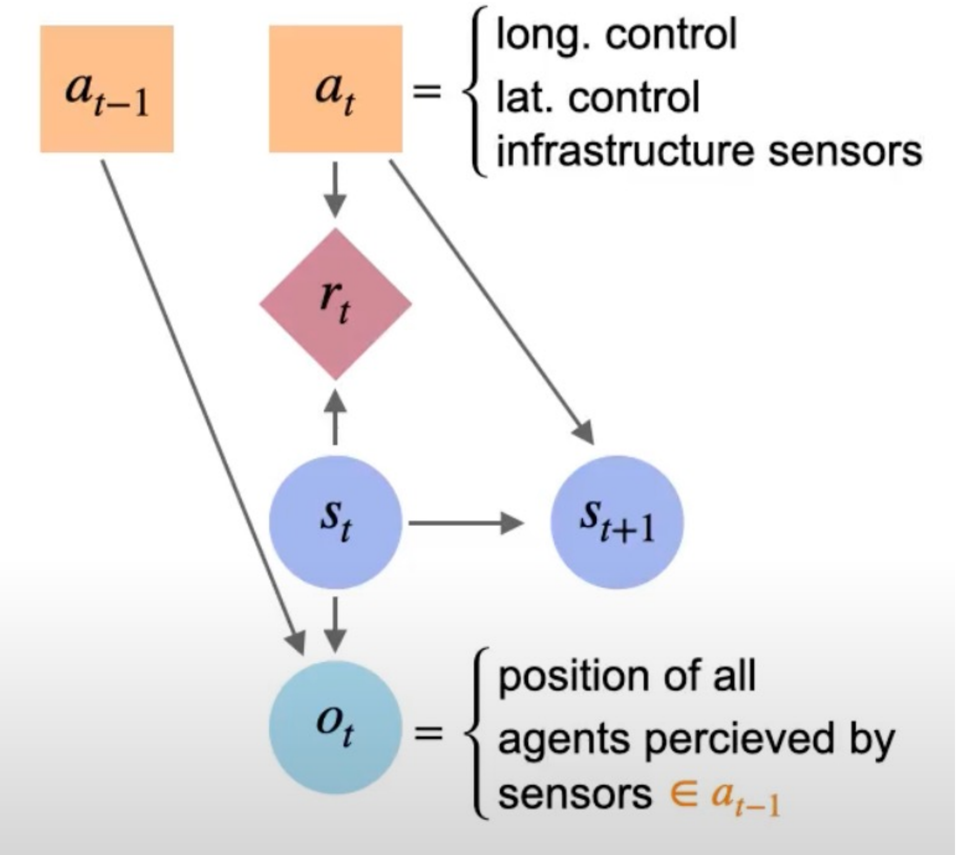
Merging into a Highway: Rewards



$$R(s) = -\lambda_1 \|v_E - v_{\text{ref}}\| - \lambda_2 \|\dot{v}_E\| - \lambda_3 \mathbf{1}_{b_{\text{hard}}}$$

Infrastructure-Aware Vehicle Control





https://drive.google.com/file/d/1MHuOMwdDGiIpVGaIFngcqBlW8wohiAlB/view?usp=sharing

V.M. Dax, M. Kochenderfer, R. Senanayake, and U. Ibrahim, Infrastructure-Enabled Autonomy: An Attention Mechanism for Occlusion Handling

Infrastructure-Aware Vehicle Control

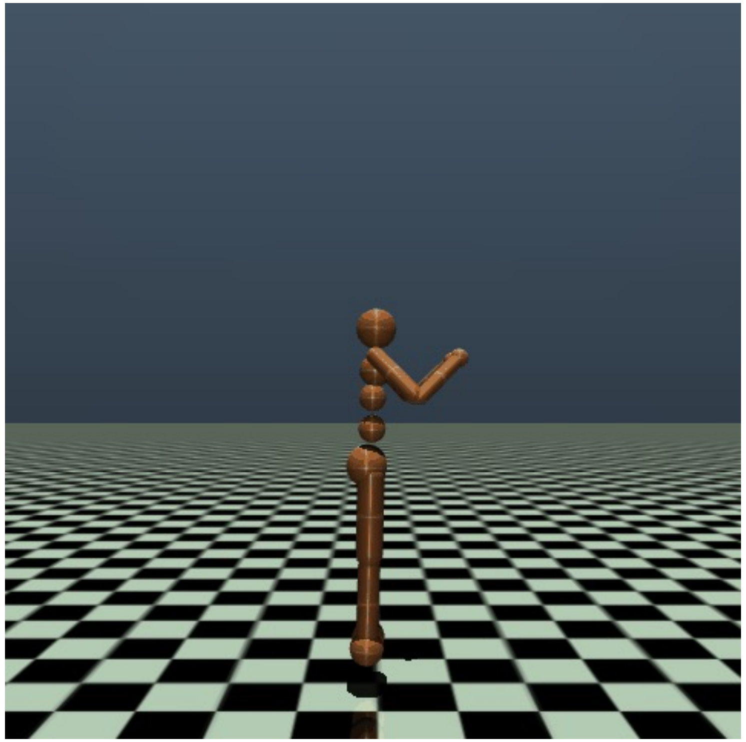
States: [x, y, speed v, heading θ, acceleration a, object type (car, bike, or pedestrian) with spatial dimensions].

Actions: [pick a lane on the current road segment to travel along, pick a maximum of k infrastructure sensors to query for additional perception data]

$$
\begin{aligned}
R = \; & w_{proximity} \cdot \min(0, d_{\min} - d_{safety}) \; + \\
& w_{time} \cdot v \; + \; w_{is} \cdot n_{\text{sensors}} \; + \\
& w_{collision} \cdot \mathbb{1}(d_{\min} < d_{collision}) \cdot v \; + \\
& w_{jerk} \cdot j \\
\text{with} \quad & d_{\min} = \min_i \|s_{ego} - s_i\|
\end{aligned}
$$

while rewarding speed. In simulation, we used $w_{time} = 1.0$, $w_{proximity} = -2.0$, $w_{is} = -0.25$, $w_{jerk} = -1.0$. Algorithm 2

Humanoid

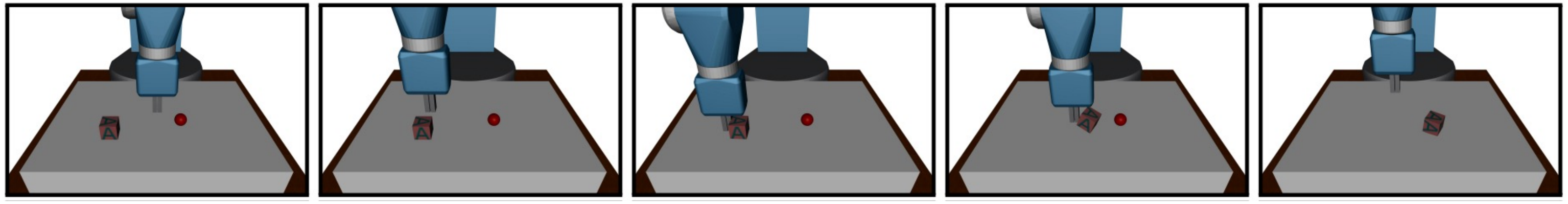


$$R = \underbrace{x_t}_{\text{position}} - 0.2\underbrace{||\dot{x_t}||_2^2}_{\text{energy expenditure}} - 2\underbrace{(z_t - 0.7)}_{\text{center of gravity}}$$

https://gymnasium.farama.org/environments/mujoco/humanoid/

Sparse rewards

We can only collect a meaningful reward when the agent successfully completes the task.



Sparse reward: If we push the box into the red spot, reward = 1 and 0, otherwise.

Dense reward: If we push the box into the red spot, reward = 1 and reward=1/distance(box,red), otherwise.

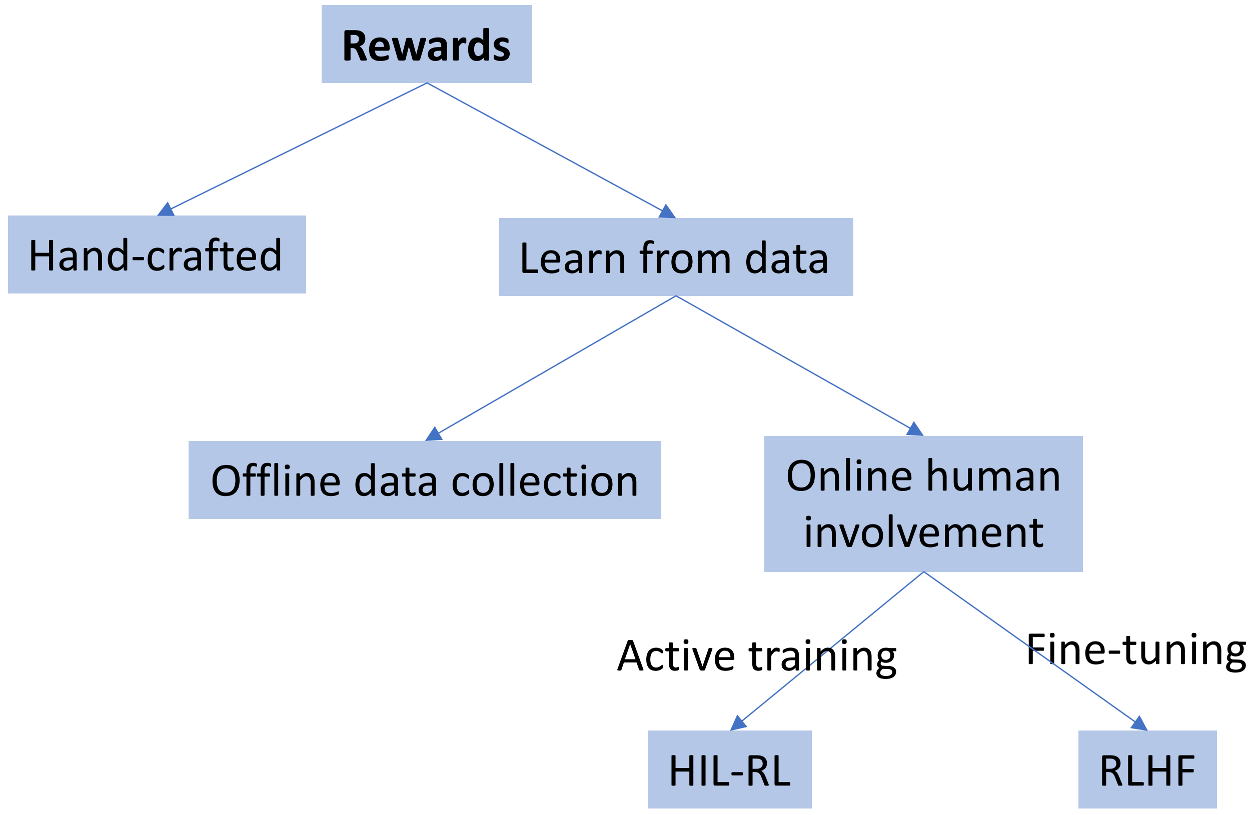
Intrinsic reward: Providing reward for *curiosity-driven exploration* (e.g., finding something novel) or information gain.
We might need extra systems/equipment to do this. Therefore, providing dense intrinsic reward is not always possible.

Hindsight Experience Replay (HER)
If the box is moved to a different position, rather than considering it as a failure, we use that experience by pretending that we indeed intended to move to that position. This way, we don't have to wait until many attempts to get a useful signal to learn a policy.

Reward Hacking

https://openai.com/research/faulty-reward-functions

Rewards



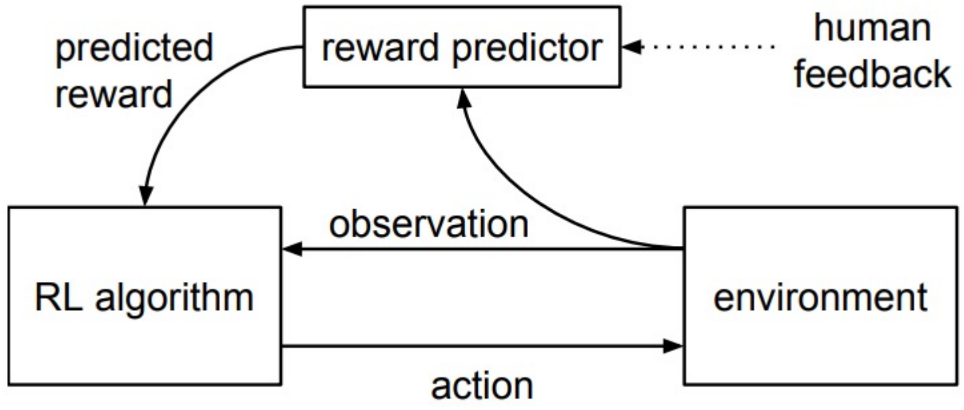
Human feedback modalities:

- Demonstrations (e.g., behavioral cloning)
- Interventions (e.g., DAgger)
- Preference elicitation (e.g., InstructGPT)

DeepRL from Human Preferences

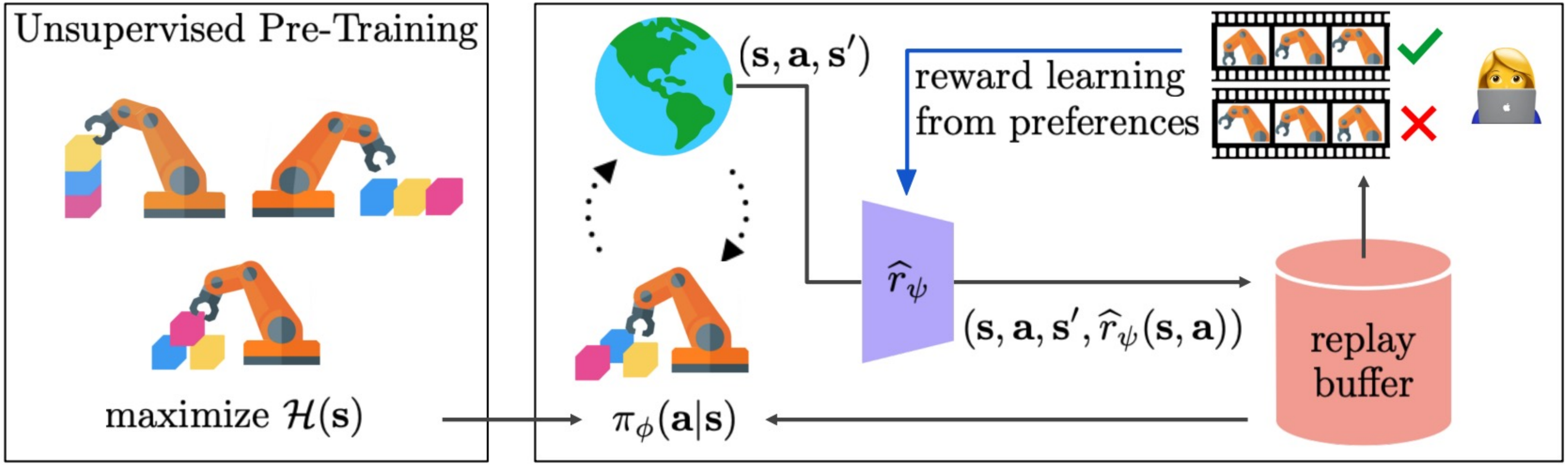
1. The policy π interacts with the environment to produce a set of trajectories $\{\tau^1, \ldots, \tau^i\}$. The parameters of π are updated by a traditional reinforcement learning algorithm, in order to maximize the sum of the predicted rewards $r_t = \hat{r}(o_t, a_t)$.
2. We select pairs of segments (σ^1, σ^2) from the trajectories $\{\tau^1, \ldots, \tau^i\}$ produced in step 1, and send them to a human for comparison.
3. The parameters of the mapping $\hat{r}$ are optimized via supervised learning to fit the comparisons collected from the human so far.

Use policy gradient methods because they are better at handling non-stationary rewards: TRPO, A2C



Christiano et al, 2017

PEBBLE: unsupervised PrE-training and preference-Based learning via relaBeLing Experience



Substantially reduce the amount of human effort required for HiL learning

https://sites.google.com/view/icml21pebble

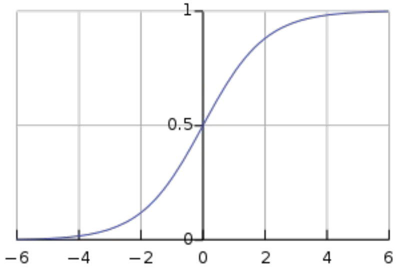
Lee et al, 2021

PEBBLE

Step 1 (reward learning): Learn a reward function that can lead to the desired behavior by getting feedback from a teacher

Bradley-Terry Model

$$f(x) = \frac{1}{1+e^{-x}}.\qquad f(x_1 - x_2) = \frac{1}{1+e^{-(x_1-x_2)}} = \frac{e^{x_1}}{e^{x_1}+e^{x_2}}$$



$$p(\sigma^1 \succ \sigma^0) = \frac{\exp \sum_t r(s_t^1, a_t^1)}{\exp \sum_t r(s_t^0, a_t^0) + \exp \sum_t r(s_t^1, a_t^1)}$$

$$\mathcal{L}^{\texttt{Reward}} = - \mathop{\mathbb{E}}_{(\sigma^0,\sigma^1,y)\sim\mathcal{D}} \Big[y(0)\log P_\psi[\sigma^0 \succ \sigma^1] + y(1)\log P_\psi[\sigma^1 \succ \sigma^0] \Big]$$

Learn a NN

Step 2 (agent learning): Update the policy and Q-function using an off-policy RL algorithm with relabeling to mitigate the effects of a *non-stationary* (i.e., changes during training) reward function

PEBBLE

Step 0 (unsupervised pre-training): We pre-train the policy only using intrinsic motivation to explore and collect diverse experiences

Algorithm 1 EXPLORE: Unsupervised exploration

Initialize parameters of Q_θ and π_ϕ and a replay buffer $\mathcal{B} \leftarrow \emptyset$
for each iteration **do**
  for each timestep t **do**
    Collect $\mathbf{s}_{t+1}$ by taking $\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t | \mathbf{s}_t)$
    Compute intrinsic reward $r_t^{\texttt{int}} \leftarrow r^{\texttt{int}}(\mathbf{s}_t)$ as in (5)
    Store transitions $\mathcal{B} \leftarrow \mathcal{B} \cup \{(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}, r_t^{\texttt{int}})\}$
  end for
  for each gradient step **do**
    Sample minibatch $\{(\mathbf{s}_j, \mathbf{a}_j, \mathbf{s}_{j+1}, r_j^{\texttt{int}})\}_{j=1}^{B} \sim \mathcal{B}$
    Optimize $\mathcal{L}_{\texttt{critic}}^{\texttt{SAC}}$ in (1) and $\mathcal{L}_{\texttt{act}}^{\texttt{SAC}}$ in (2) with respect to θ and ϕ
  end for
end for
return $\mathcal{B}, \pi_\phi$

$$r^{\texttt{int}}(\mathbf{s}_t) = \log(||\mathbf{s}_t - \mathbf{s}_t^k||)$$

Closest of k in the buffer