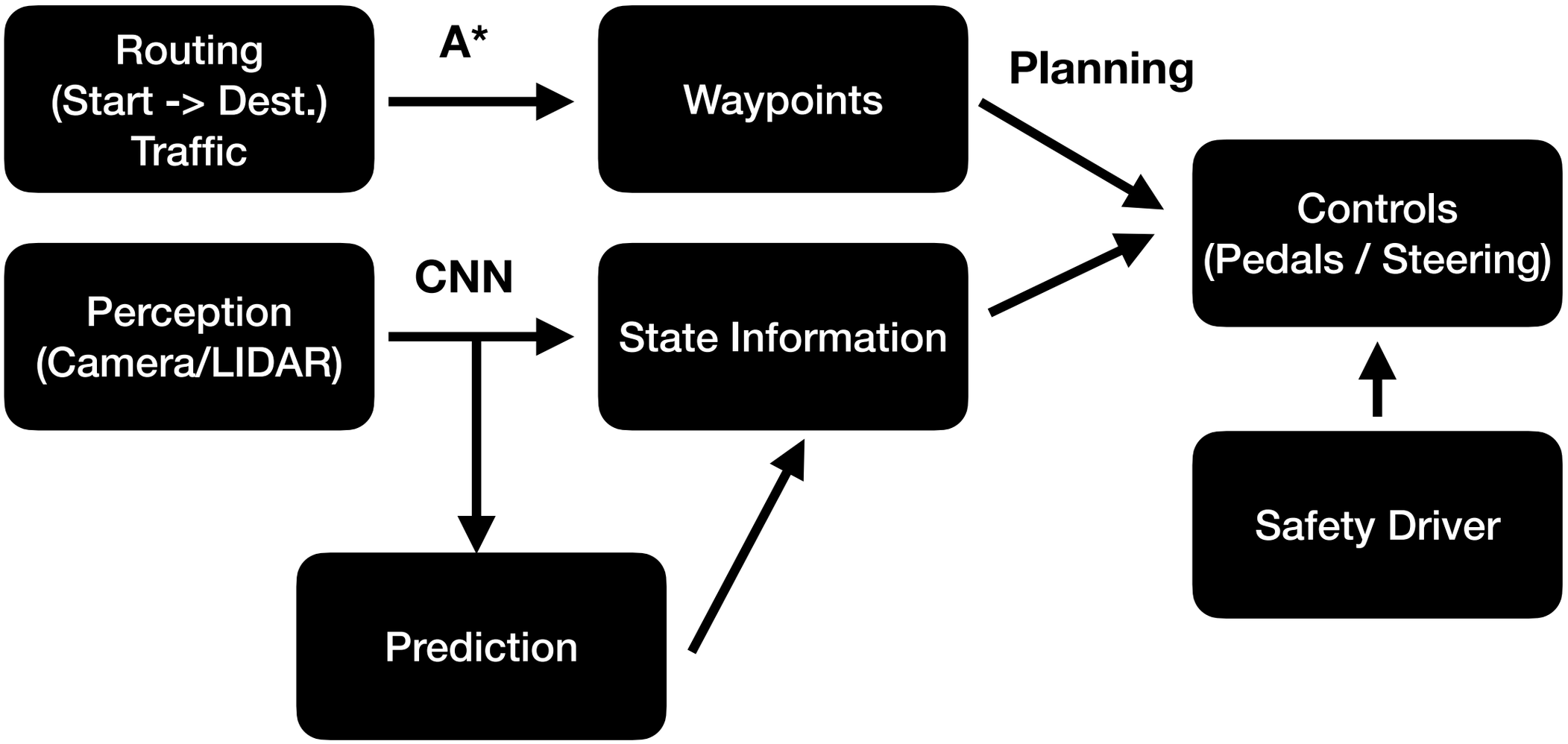
Leveraging Human Input for Training Self-Driving Cars

Gokul Swamy
gswamy@cmu.edu

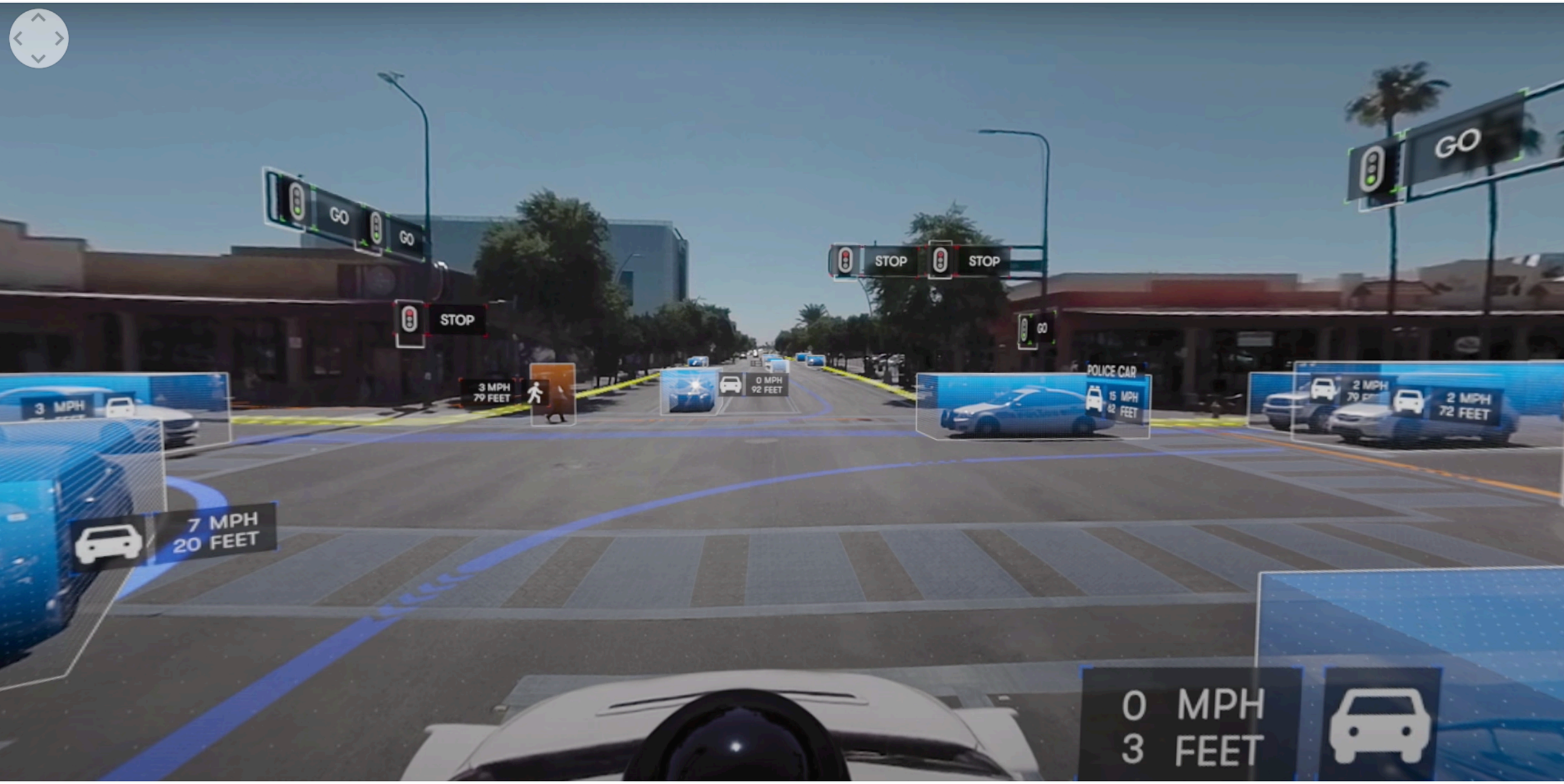
Key Idea: People in and around cars make *purposeful decisions*, which allows us to design efficient algorithms that take advantage of this structure.


Aurora

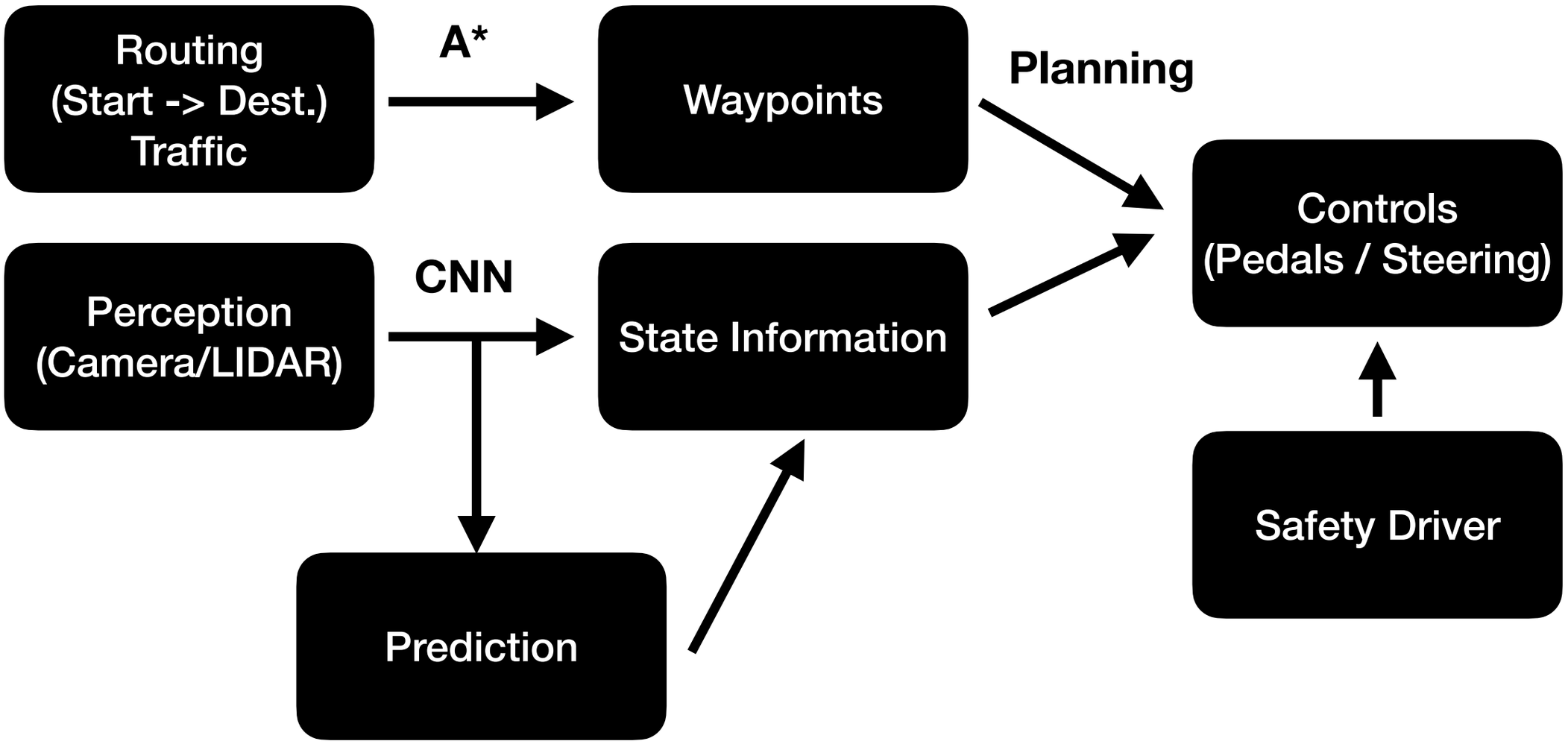
How do SDC work?



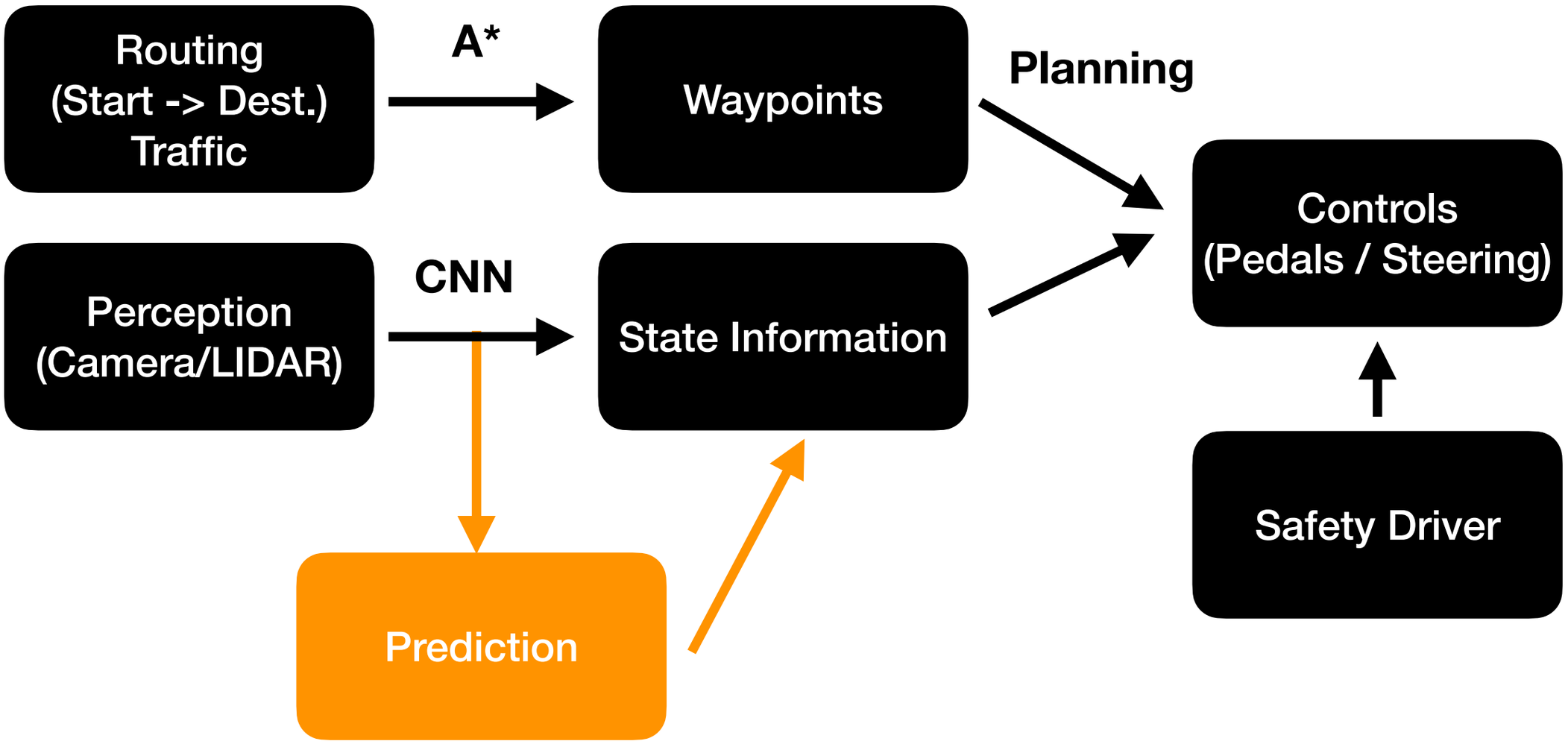

GO
GO
STOP
STOP
STOP
GO
GO
0 MPH
7 FEET
0 MPH
0 FEE
POLICE
8 MPH
31 FEET
0 MPH
6 FEET
1 MPH
7 MPH
20 FEET
0 MPH
3 FEET


GO
GO
STOP
STOP
STOP
GO
3 MPH
79 FEET
0 MPH
92 FEET
POLICE CAR
15 MPH
42 FEET
2 MPH
72 FEET
7 MPH
20 FEET
0 MPH
3 FEET

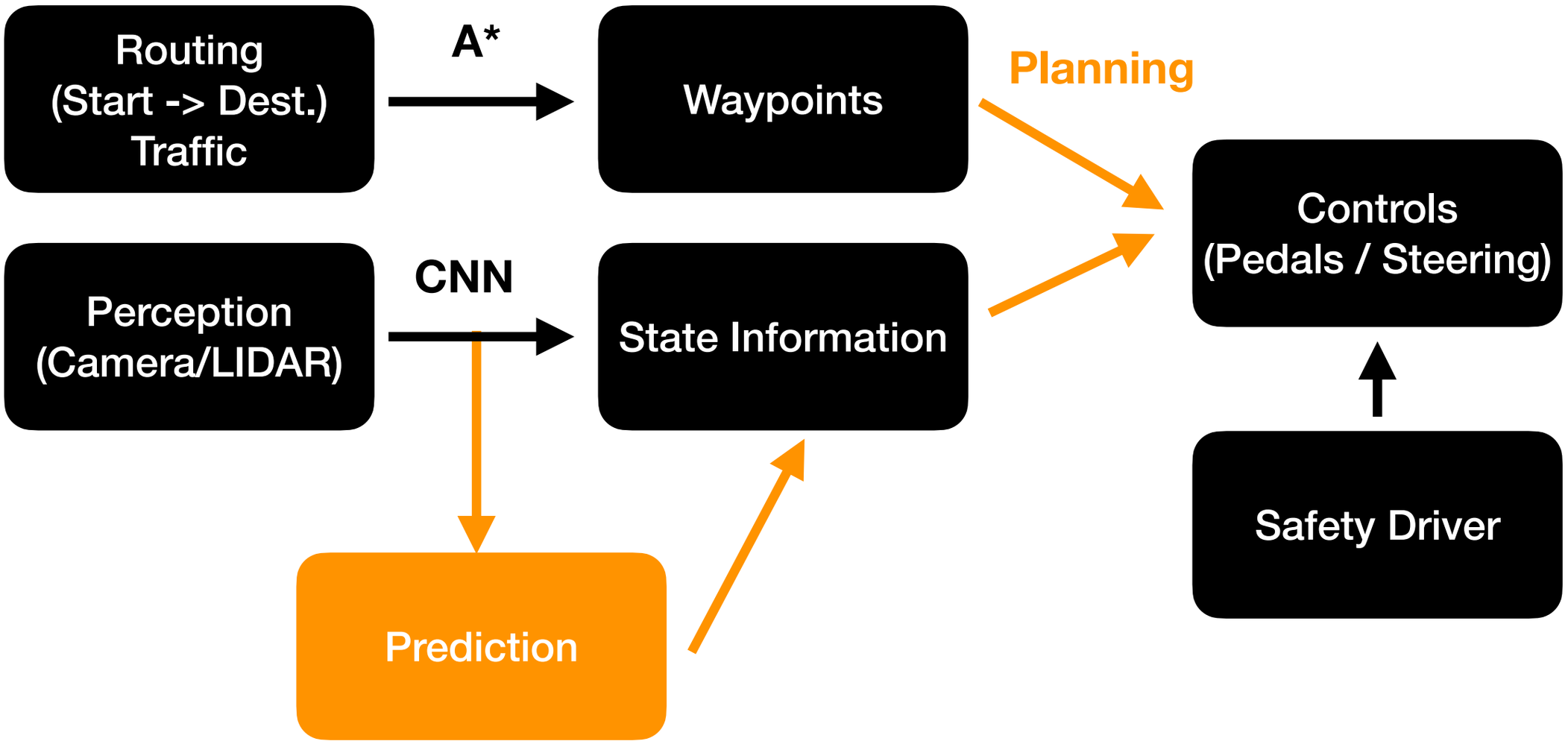
Where is human input required?



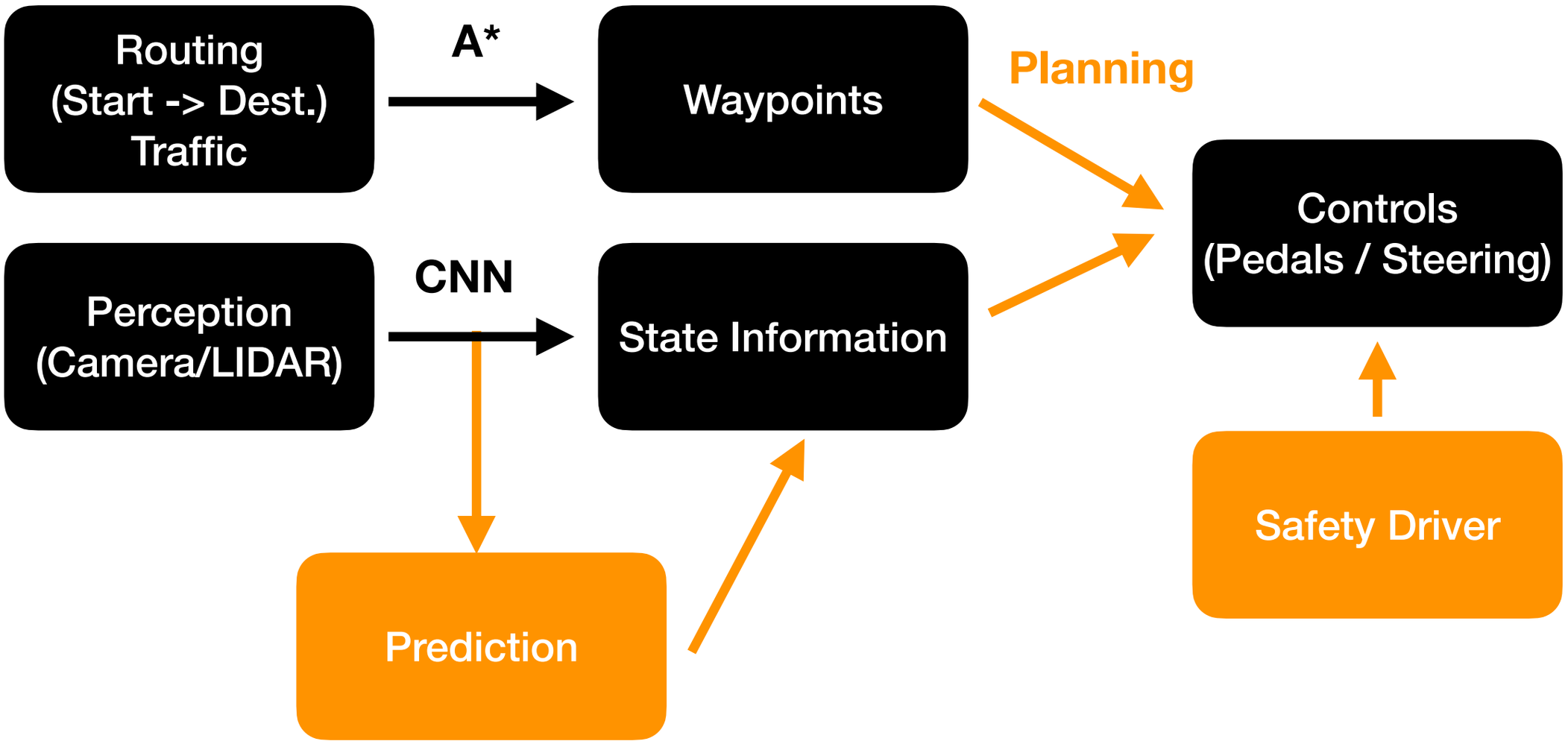
Where is human input required?



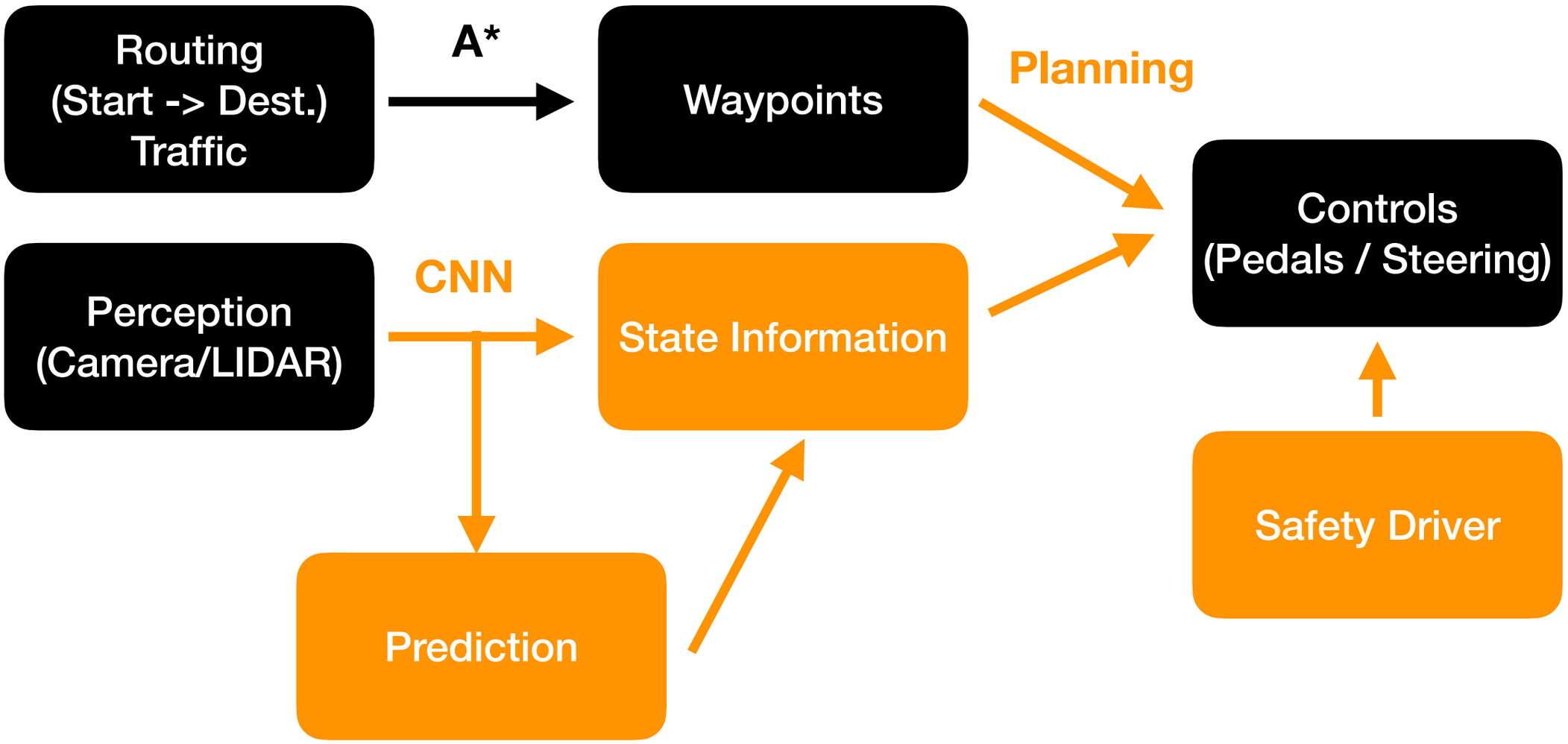
Where is human input required?



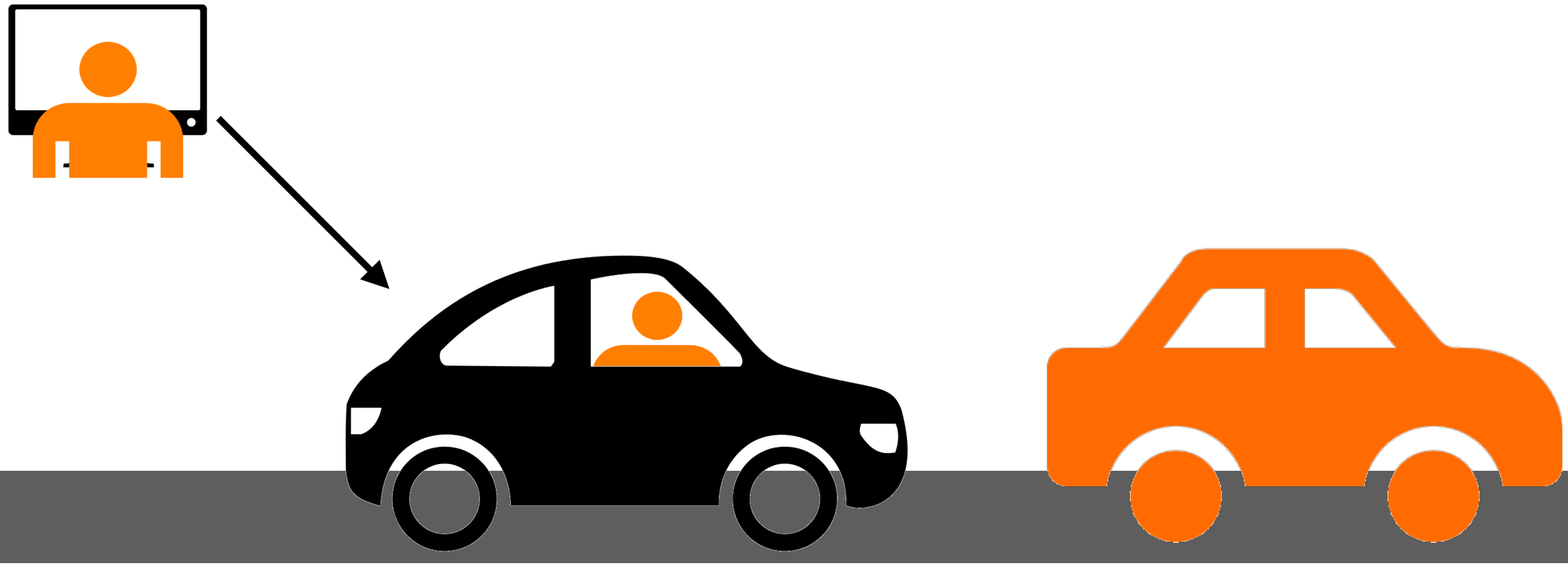
Where is human input required?



Where is human input required?



What we're going to talk about



Learning to Drive via Imitation

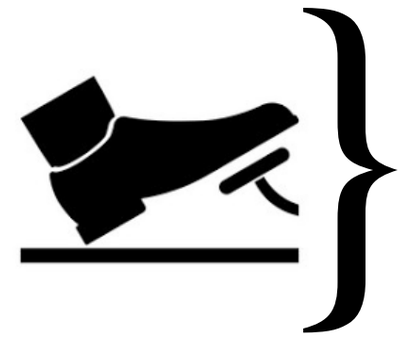
Why Imitation Learning for SDC?

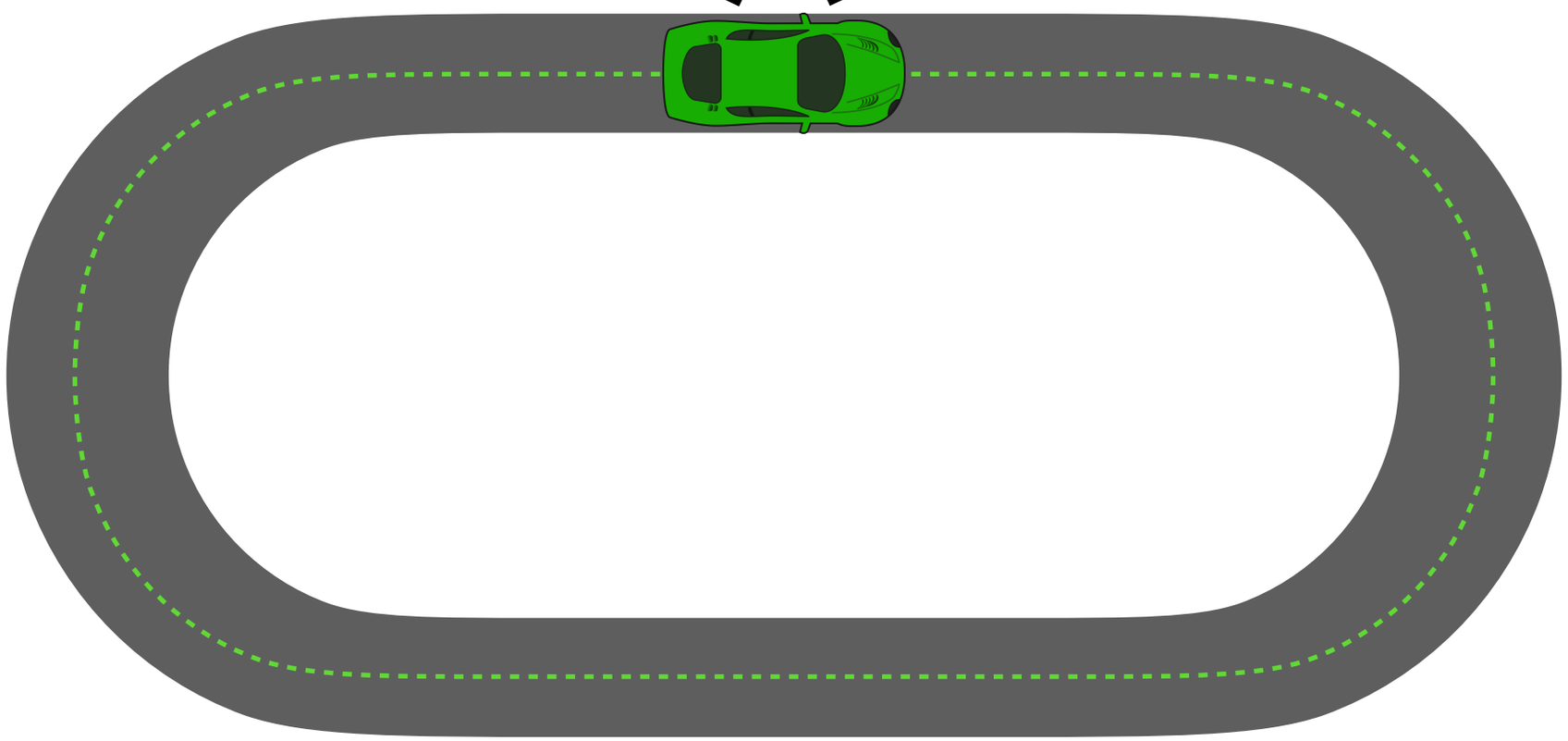
- Generally speaking, people are good drivers.
  - Collected data will be of a high quality. Can filter out bad examples.
- We can identify what they care about.
  - This makes designing a state space not too bad.

Why Imitation Learning for SDC?

- Reward design is very hard.
  - How would you write down a reward function for good driving?

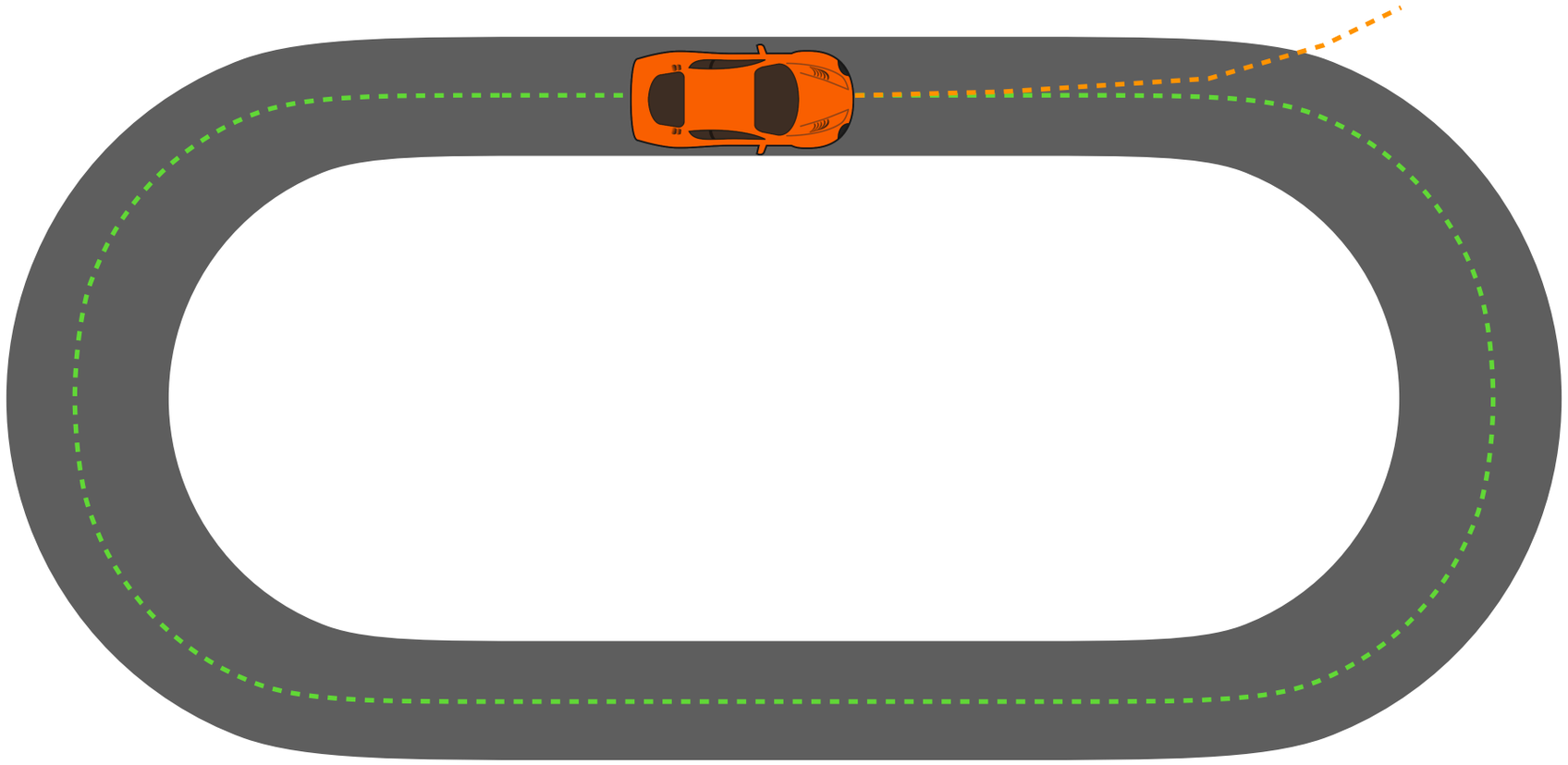
Imitation Learning: Behavioral Cloning

$s = \{x, y\}$ $a = \{$  $\}$



$$L(\pi) = \frac{1}{N} \sum_{i}^{N} (\pi(s_i) - a_i)^2$$

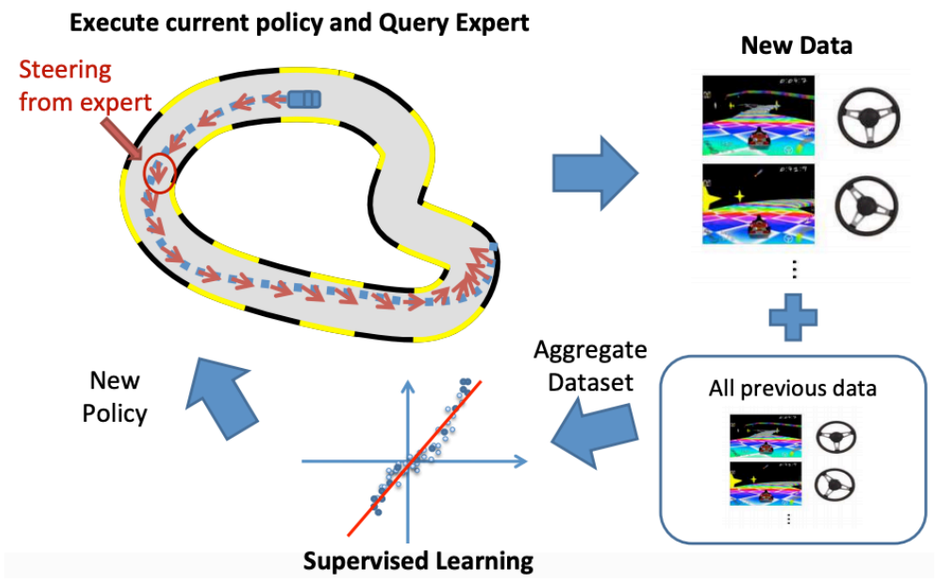
Imitation Learning: Behavioral Cloning



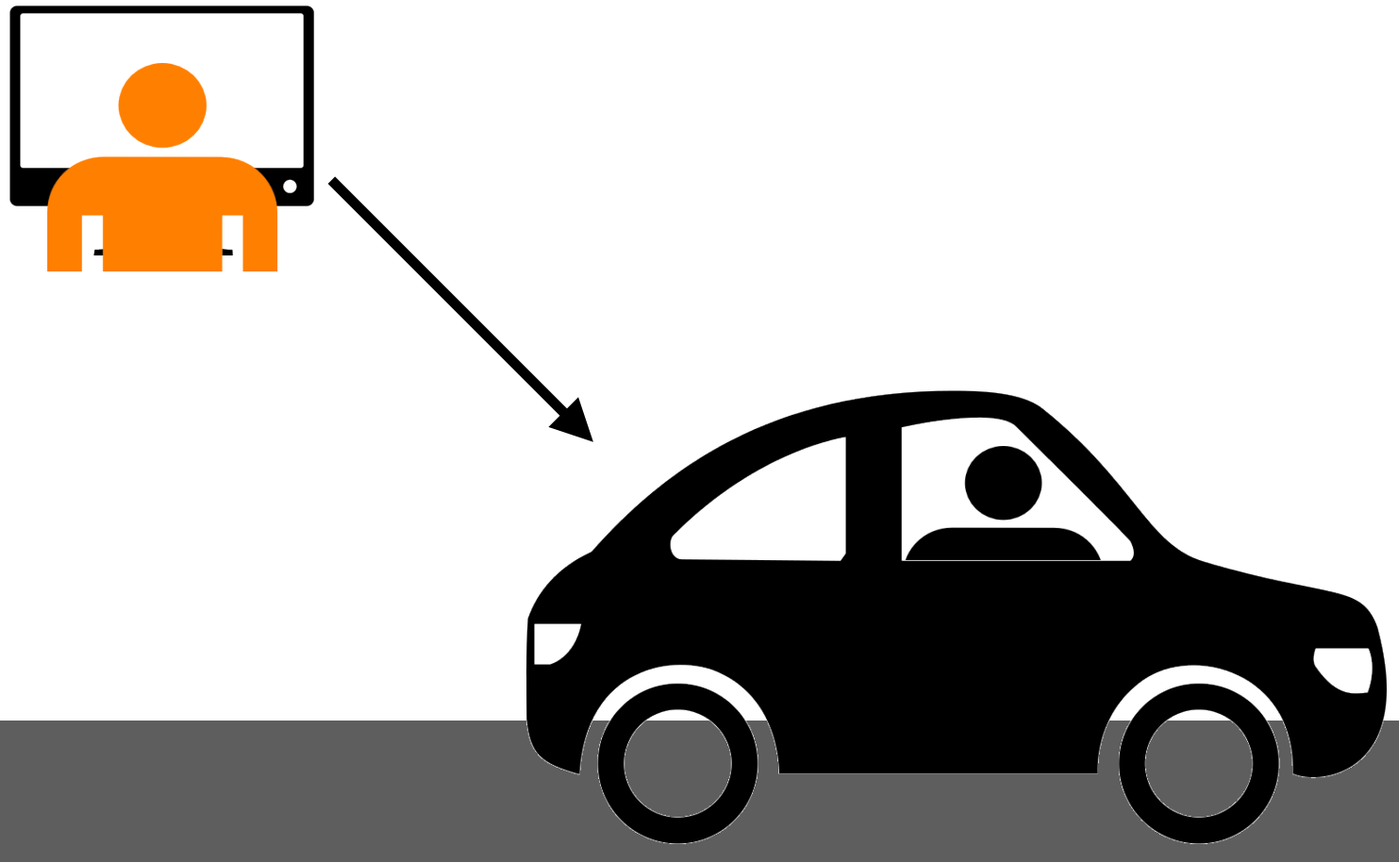
Imitation Learning: DAgger

1. train $\pi_\theta(u_t|o_t)$ from human data $\mathcal{D}_{\pi^*} = \{o_1, u_1, ..., o_N, u_N\}$
2. run $\pi_\theta(u_t|o_t)$ to get dataset $\mathcal{D}_\pi = \{o_1, ..., o_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions u_t
4. Aggregate: $\mathcal{D}_{\pi^*} \leftarrow \mathcal{D}_{\pi^*} \cup \mathcal{D}_\pi$
5. GOTO step 1.

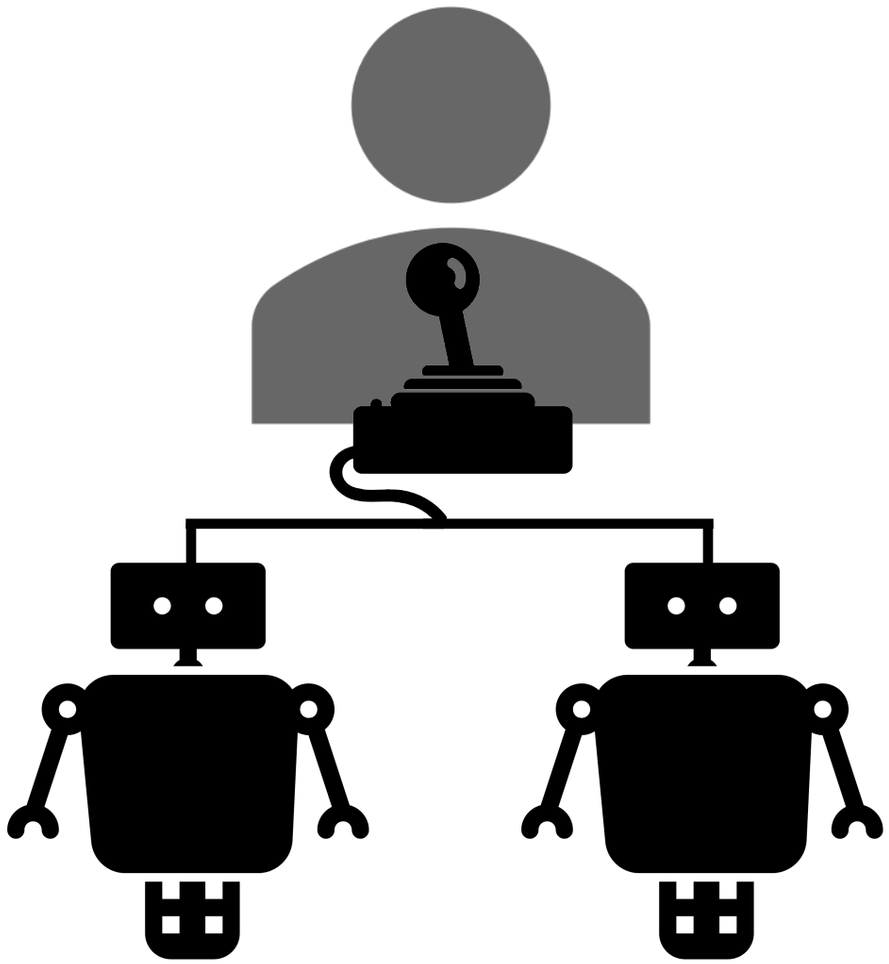
(For more info, read https://www.ri.cmu.edu/pub_files/2015/3/InvitationToImitation_3_1415.pdf)



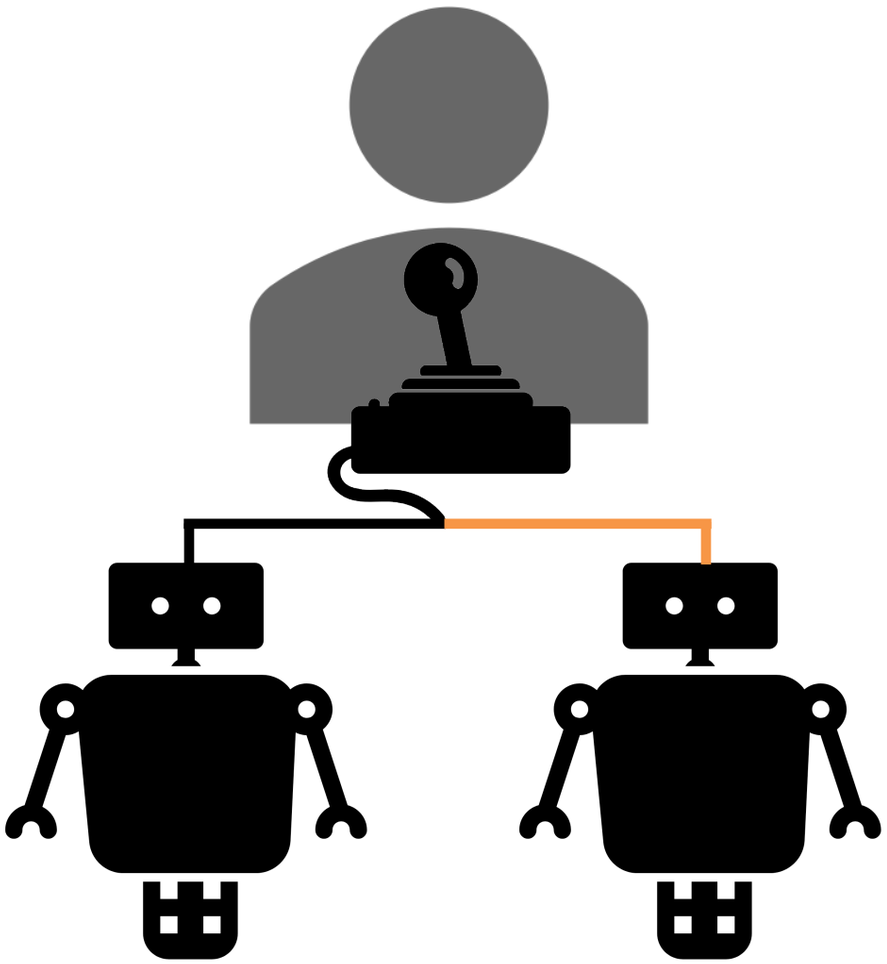
Learning to Intervene via Imitation



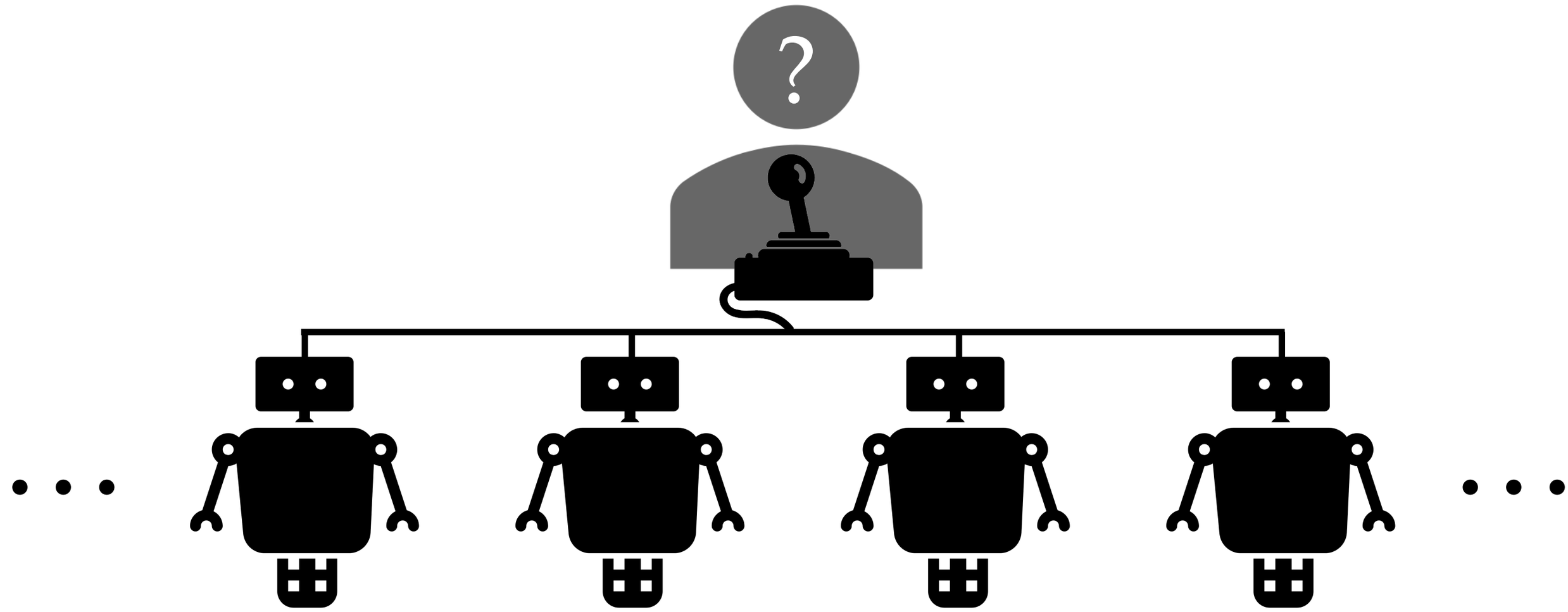
Scaling Teleoperation



Scaling Teleoperation



Scaling Teleoperation



Imitation learning is hard when the expert doesn't know what to do!

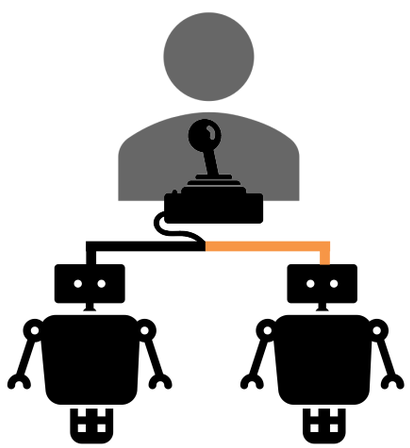
Key Insight

We can use decisions that the operator makes in easy settings with only a few robots to train a predictive model of user behavior that generalizes to challenging settings with many robots.

Key Insight

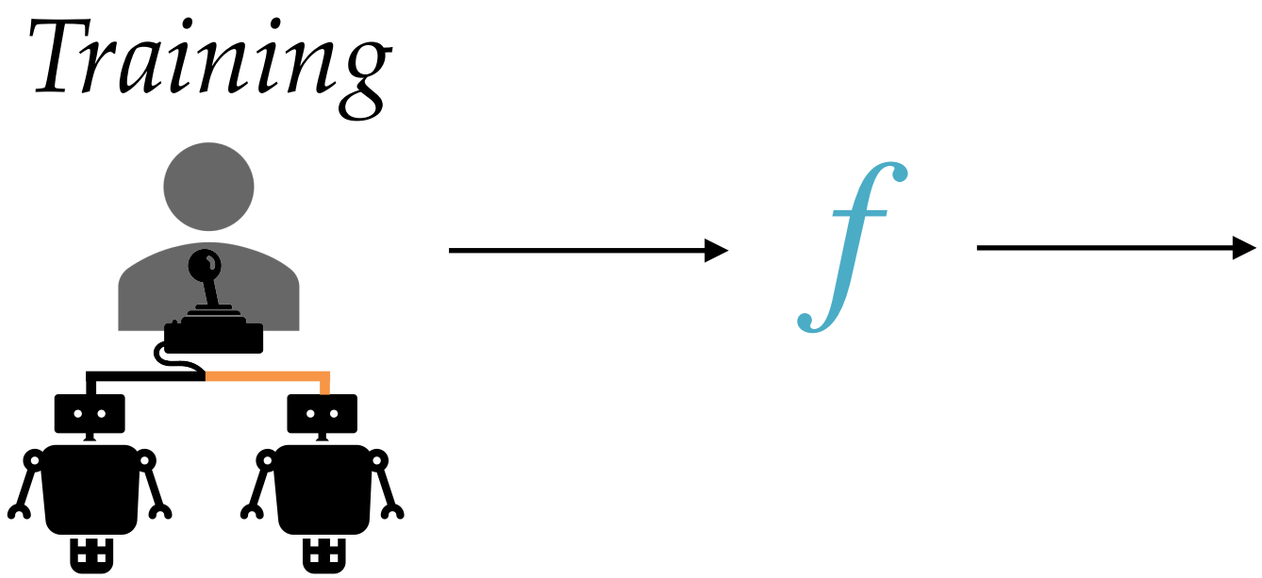
We can use decisions that the operator makes in easy settings with only a few robots to train a predictive model of user behavior that generalizes to challenging settings with many robots.

Training



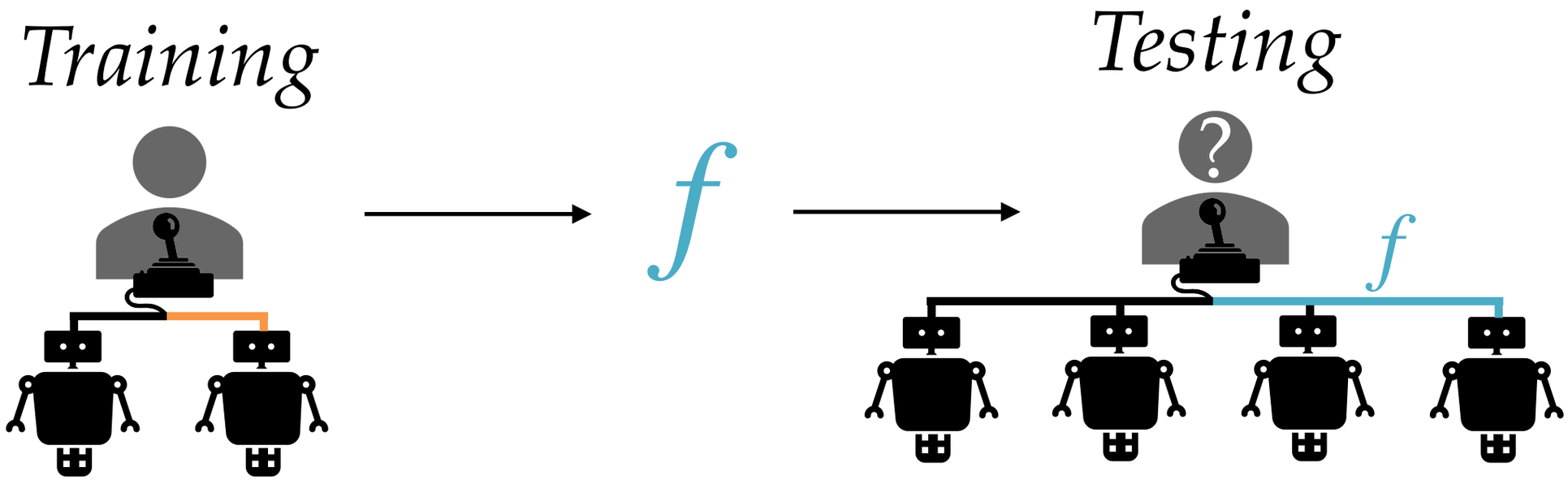
Key Insight

We can use decisions that the operator makes in easy settings with only a few robots to train a predictive model of user behavior that generalizes to challenging settings with many robots.

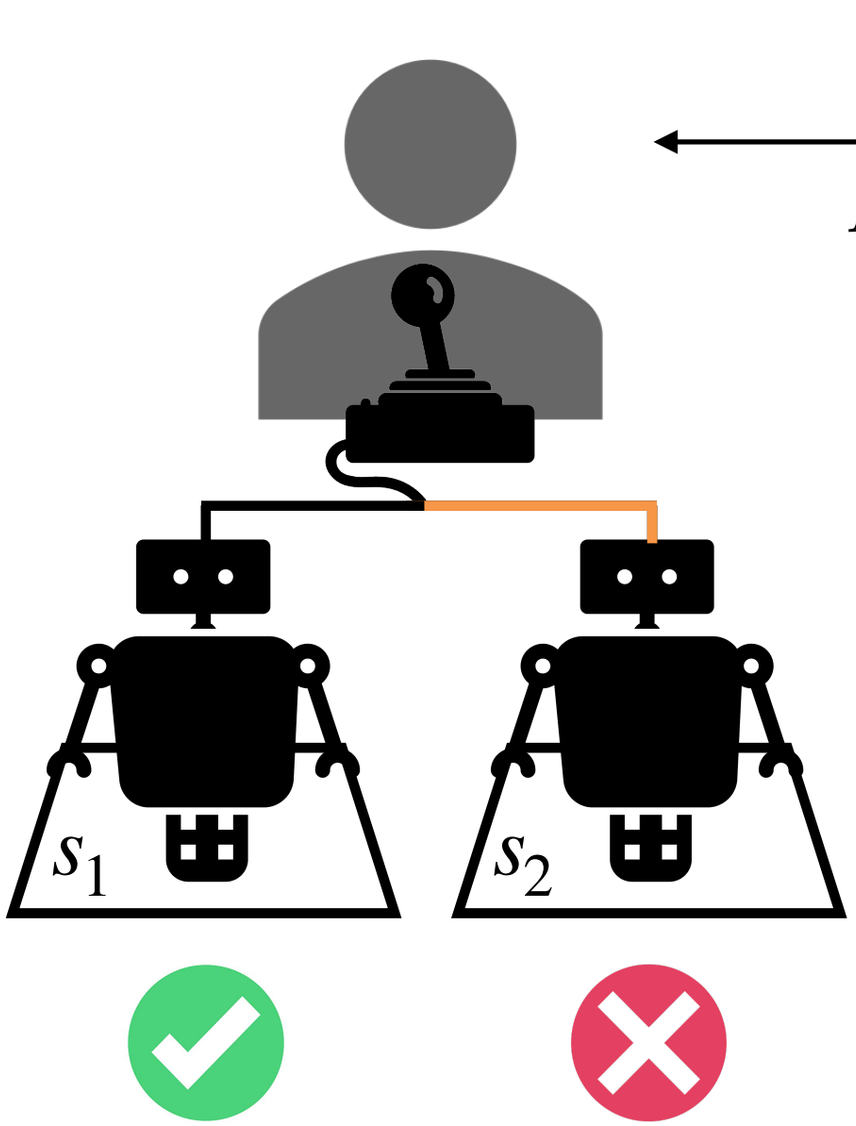


Key Insight

We can use decisions that the operator makes in easy settings with only a few robots to train a predictive model of user behavior that generalizes to challenging settings with many robots.

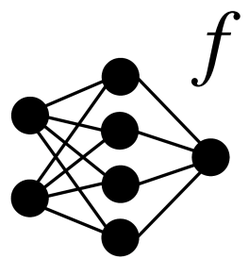


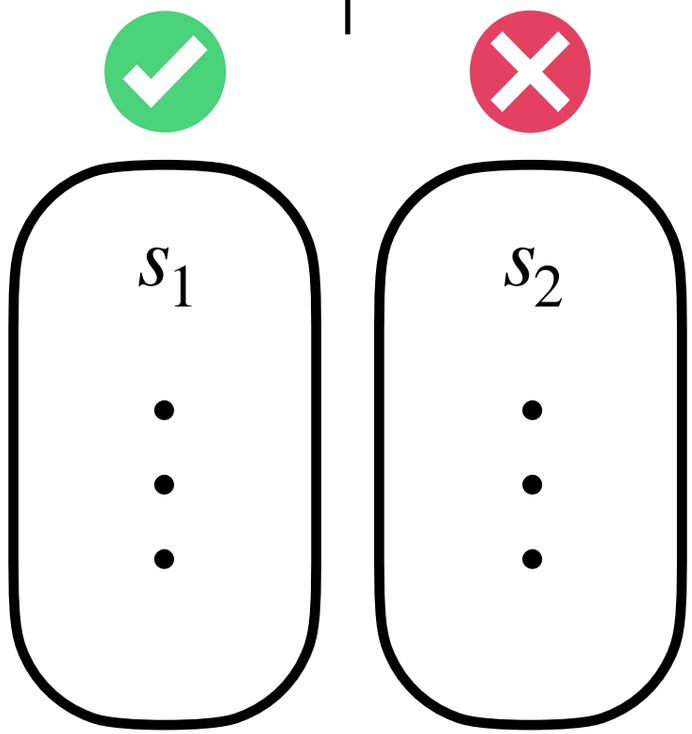
Naive Approach via Classification



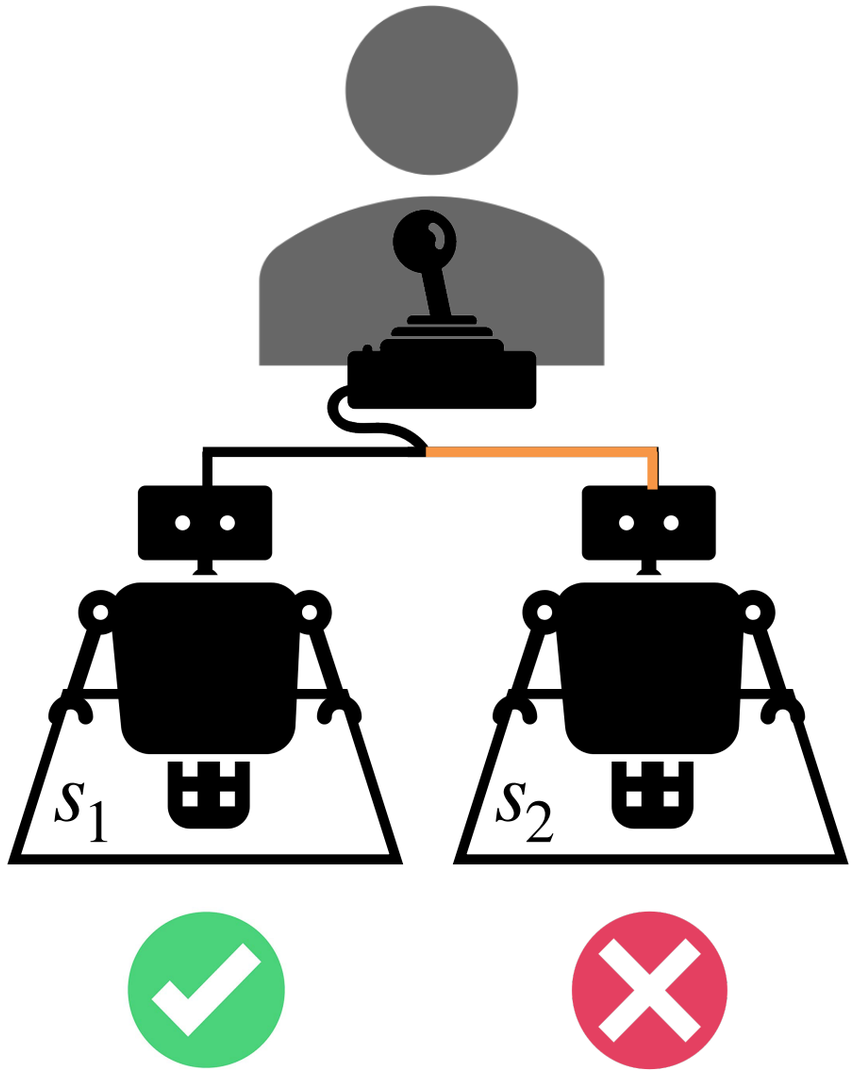
Make automated choices

$$R_t = \arg\max_i f(s_t^i)$$

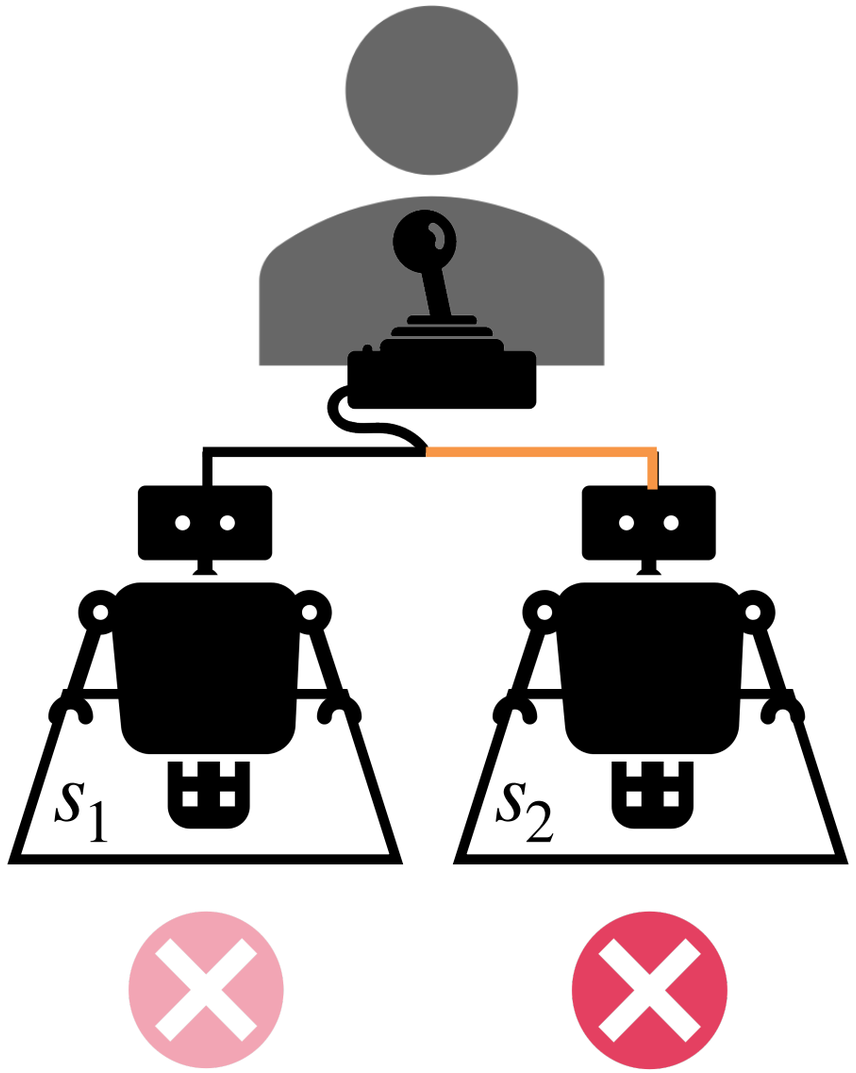




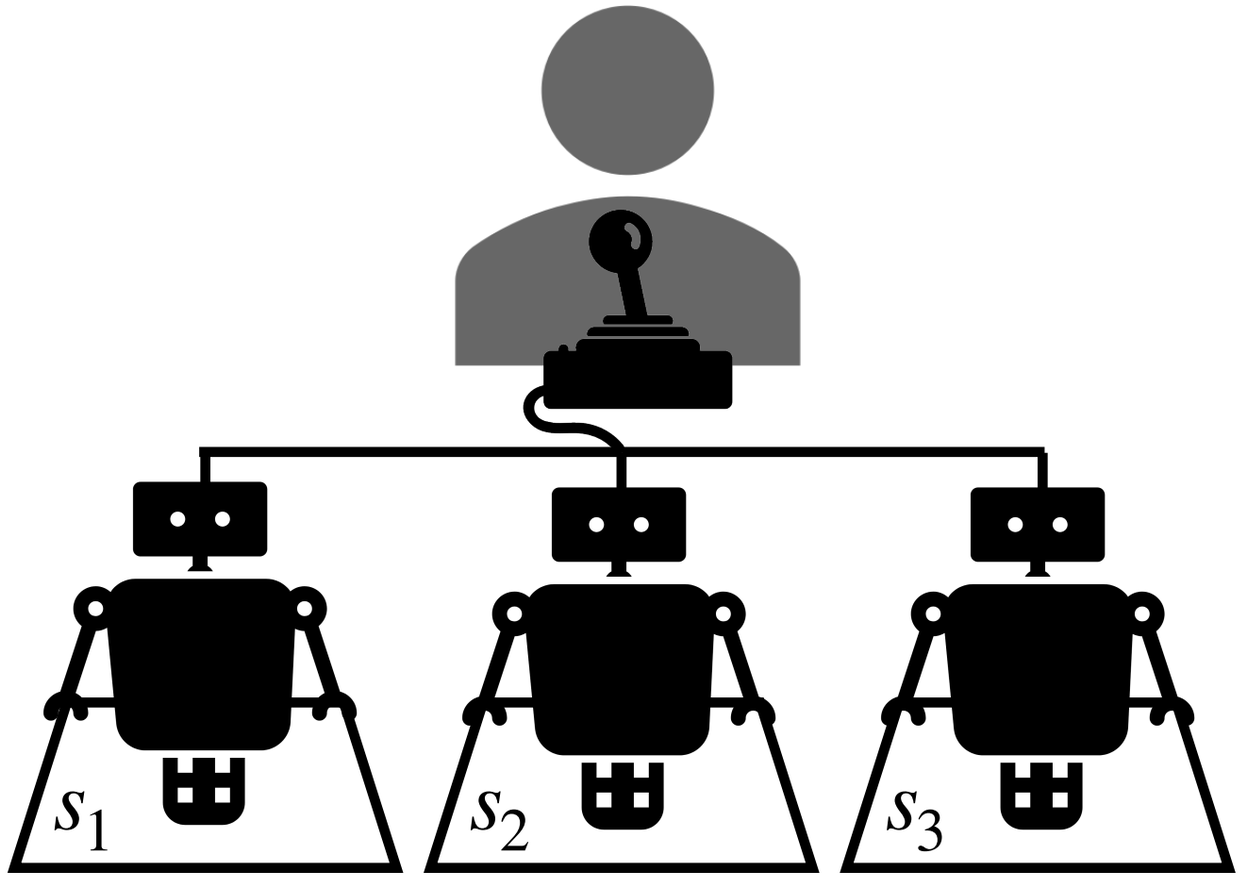
Naive Approach via Classification



Naive Approach via Classification

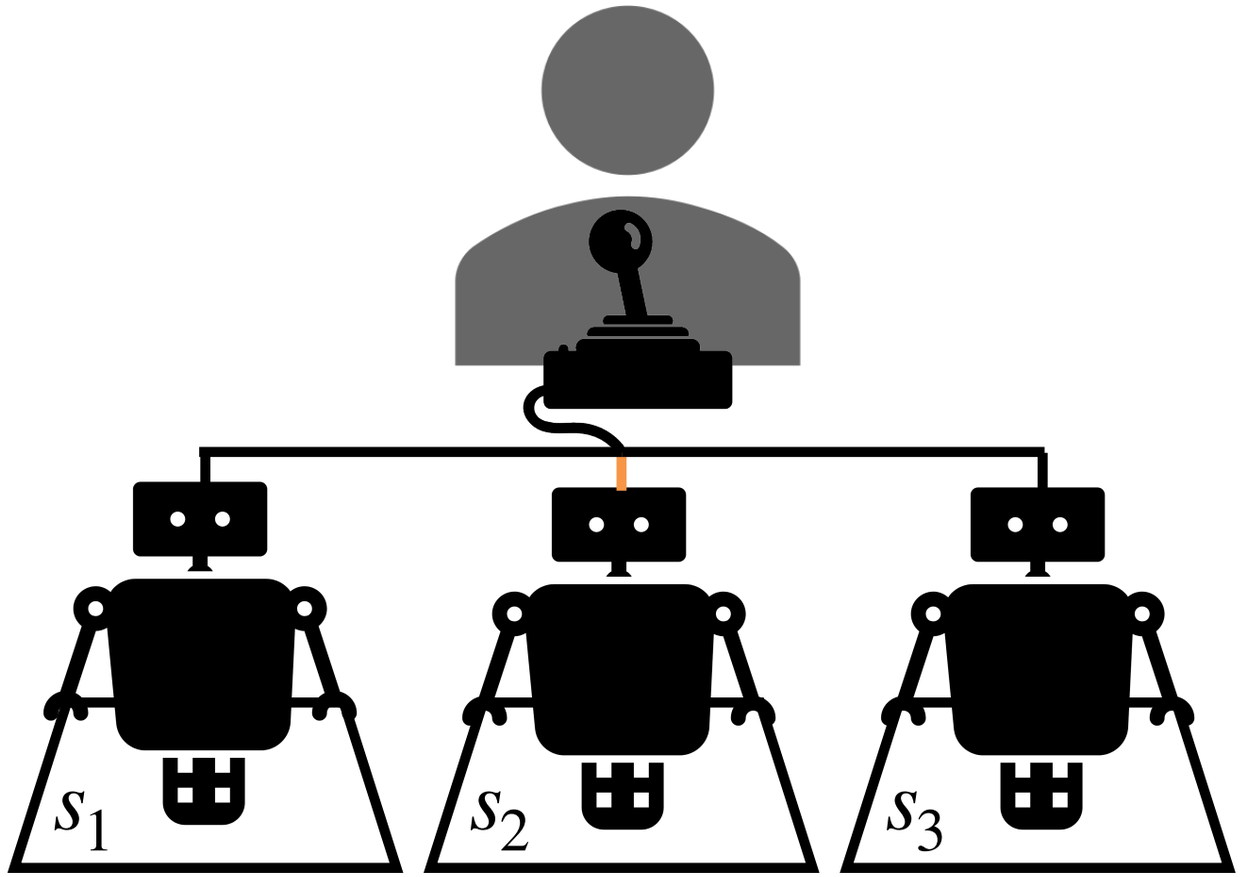


Naive Approach via Classification



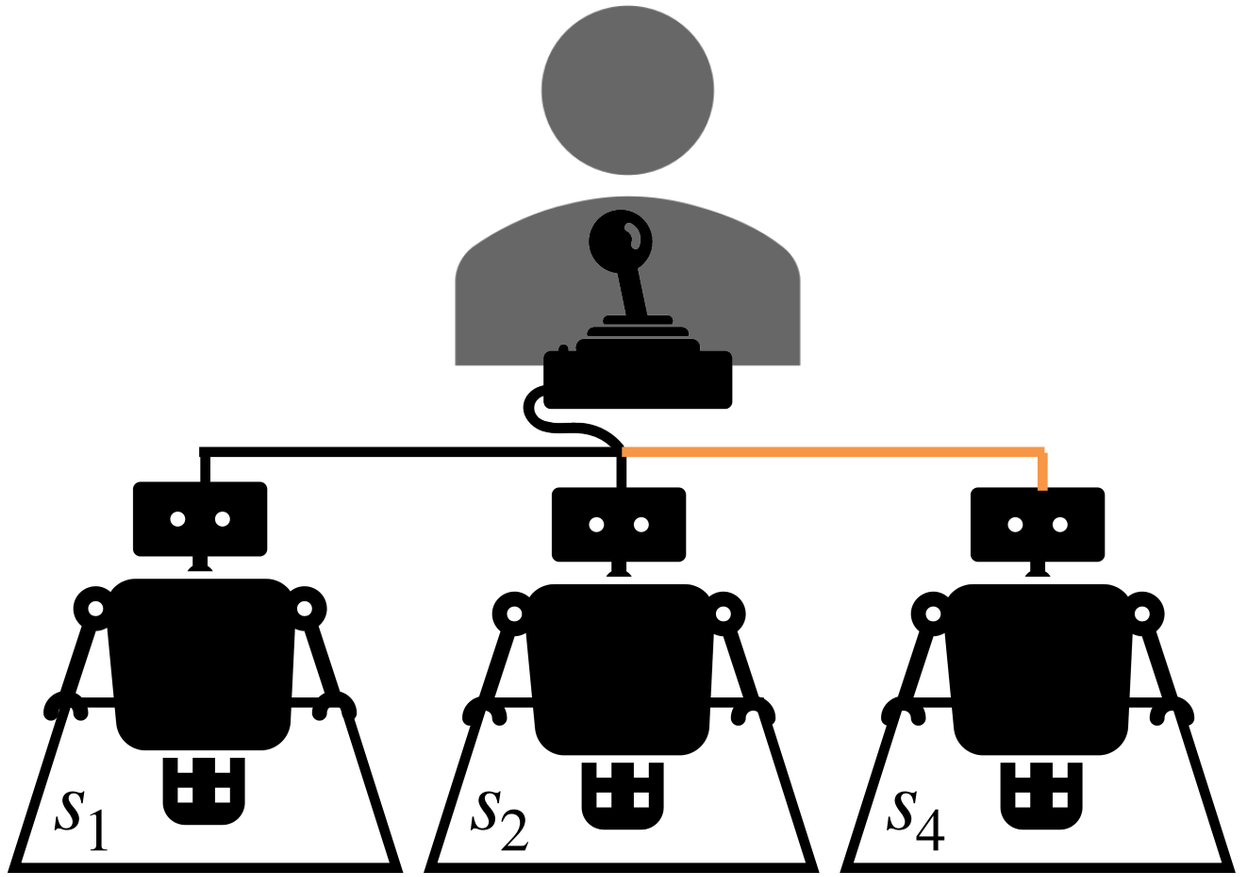
Naive Approach via Classification

$t = 1$

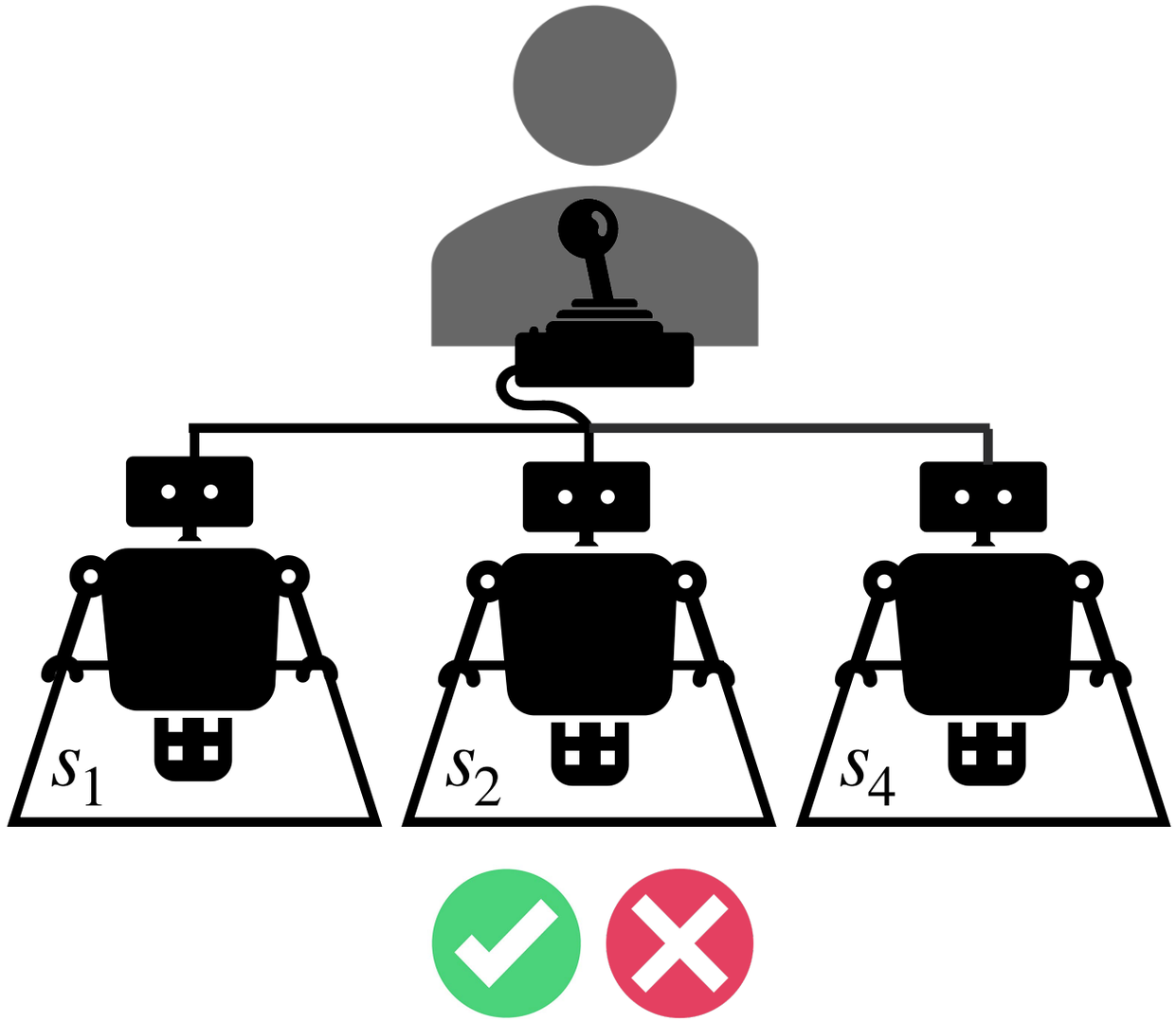


Naive Approach via Classification

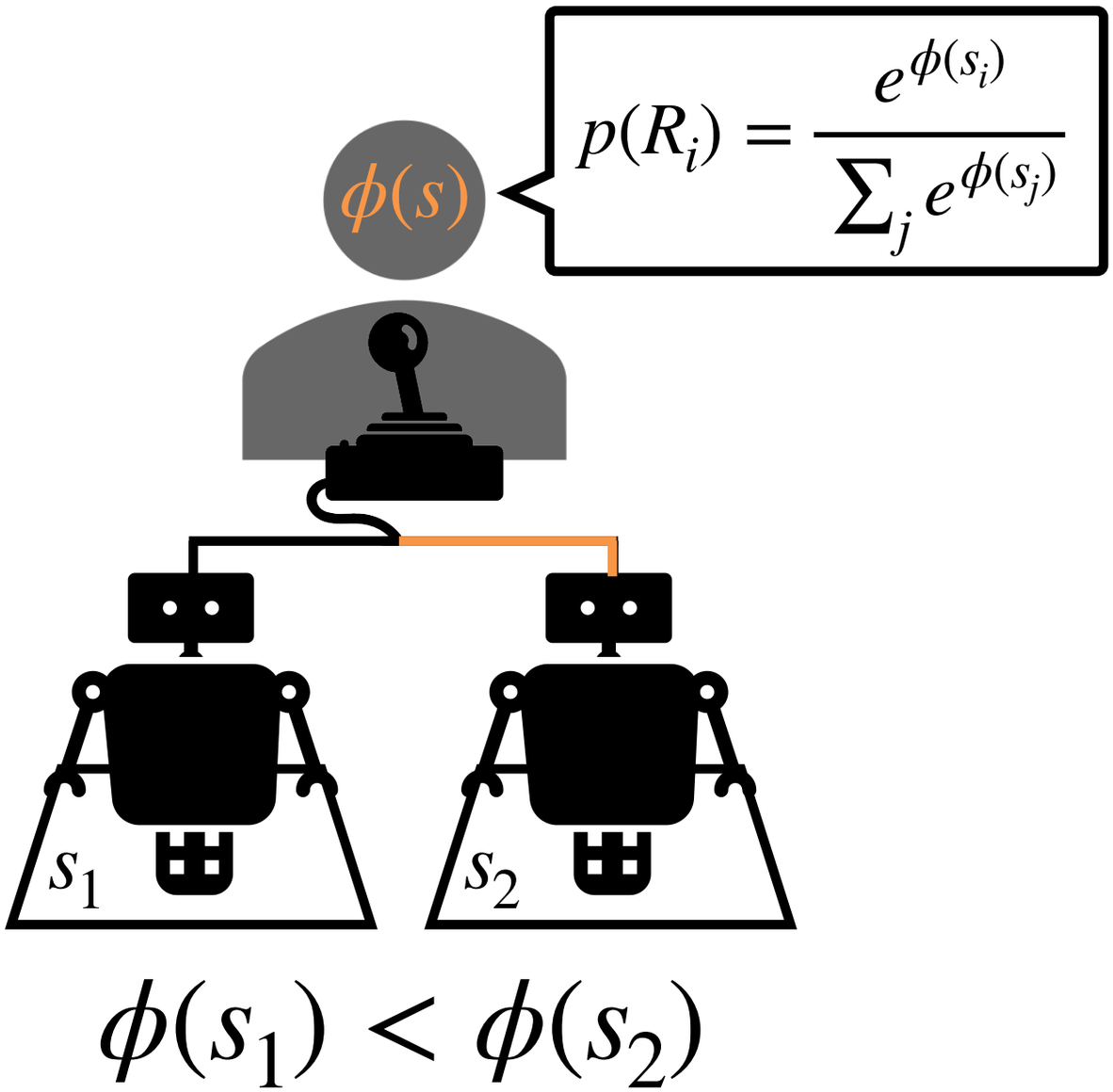
$t = 2$



Naive Approach via Classification



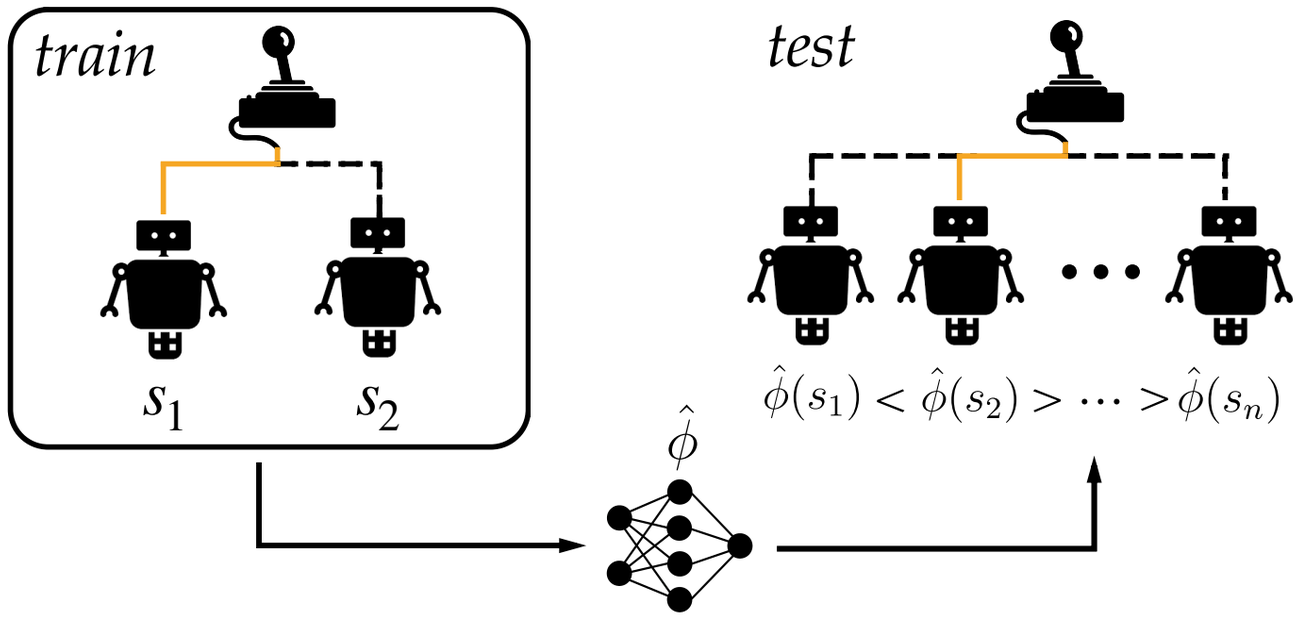
Scaled Autonomy



Luce et al.

Scaled Autonomy

***Step 1**: Let user freely choose which of a few robots to teleoperate.*



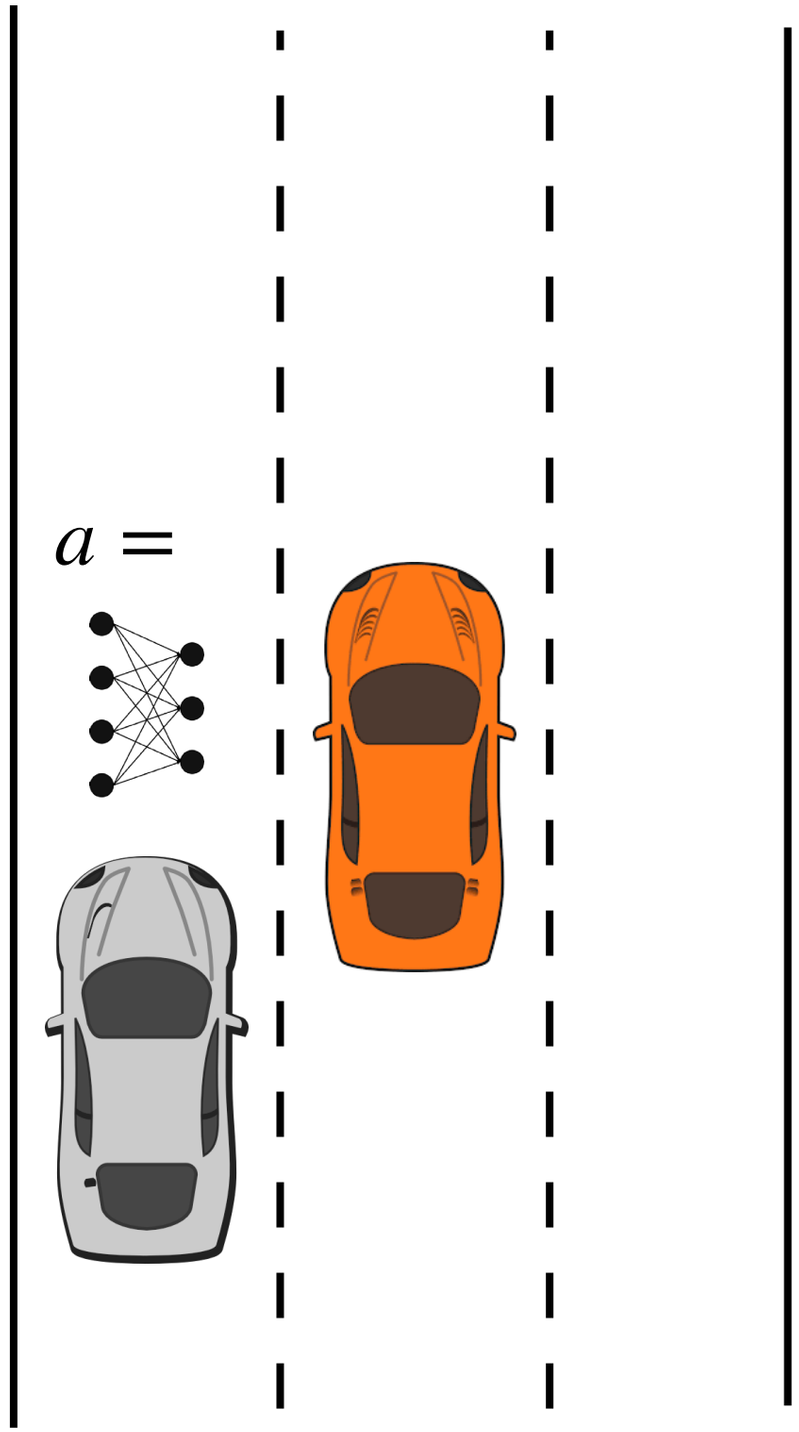
***Step 3**: Take the argmax over the learned function to automatically choose a robot for the user.*

***Step 2**: Train a network to mimic user choices by maximizing the likelihood of the demonstrated*

Learning to Model Other Drivers via "Imitation"

Behavior Prediction

- Naive Approach
  - Fit network to map from state of all cars to action of a particular car
- **Q**: Why might this not work well?
- **A**: Your actions influence the actions of other cars

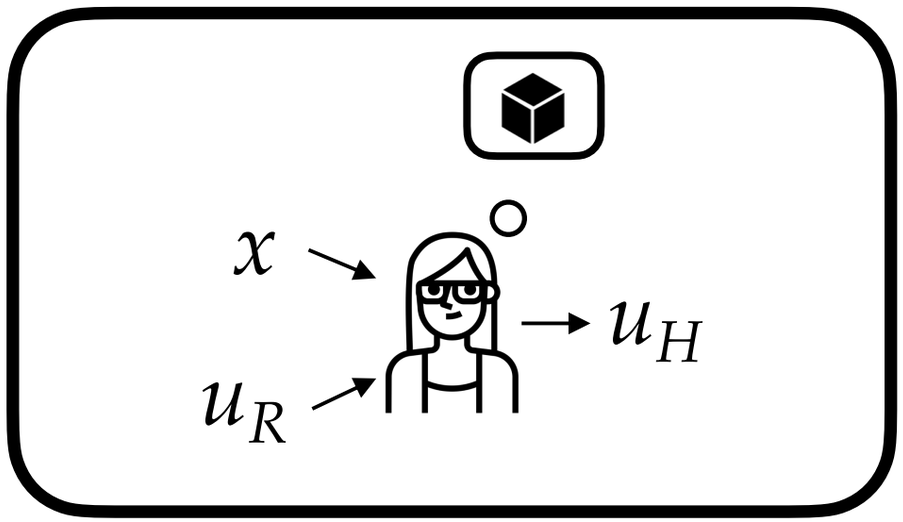


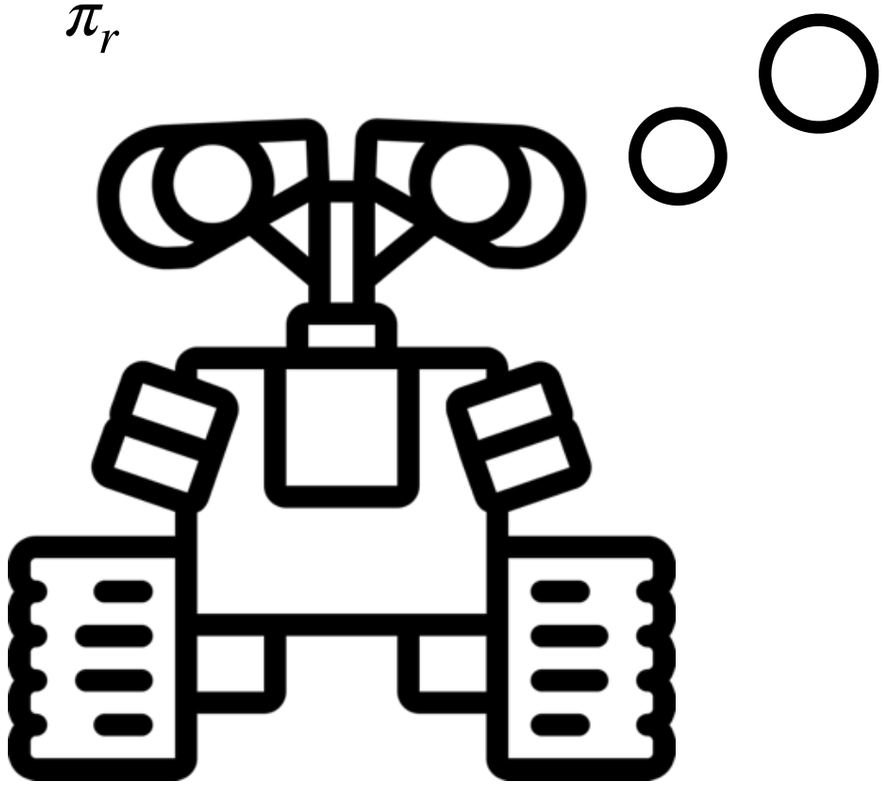
Behavior Prediction

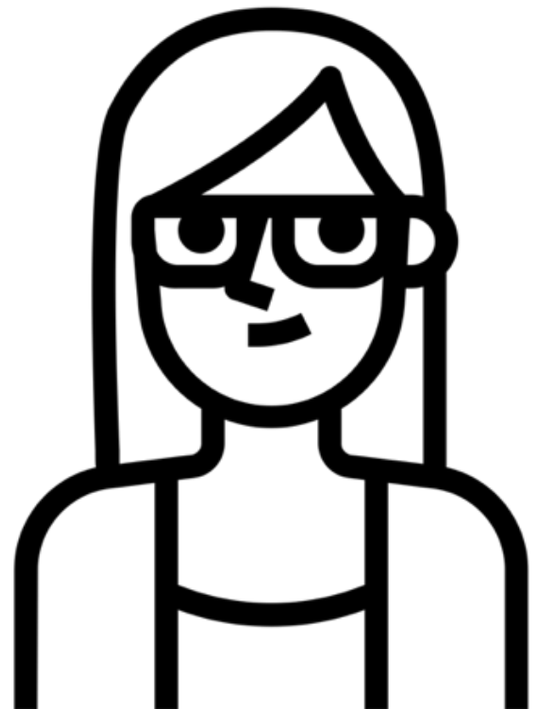
- **Problem**: other drivers ***react*** to the positions and actions of your car
  - Predictions → Your Actions → Other's Actions → Pred.
    - So if you update your policy, you change input distribution to behavior network
- **Solution**: learn a function that models the person's response to a robot's action
  - Question: what kind of function to learn?

Black-Box Models

$$\max_{\pi_r} E[U_r(\pi_r, \pi_h)]$$

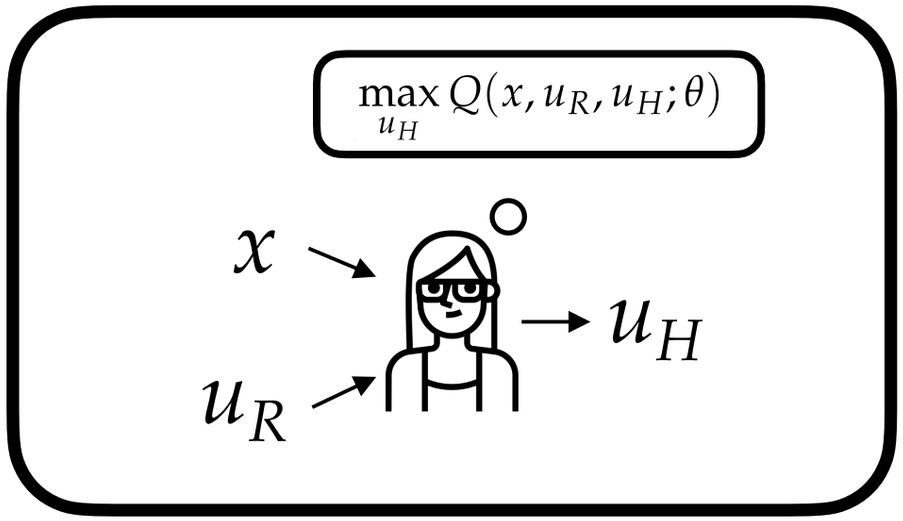


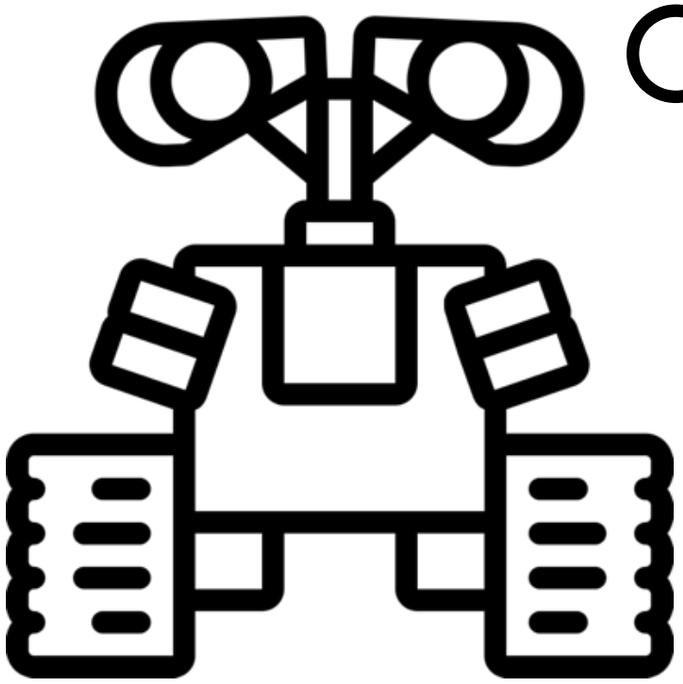




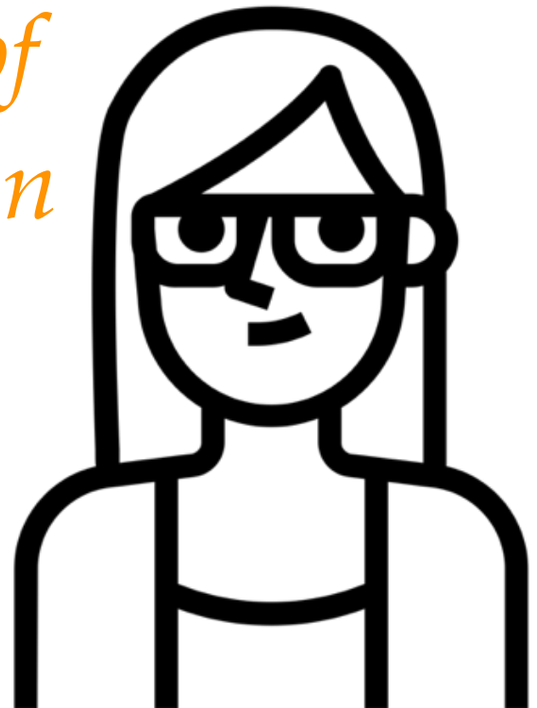
Theory of Mind

$$\max_{\pi_r} E[U_r(\pi_r, \pi_h)]$$





Takes advantage of structure in human decisions!



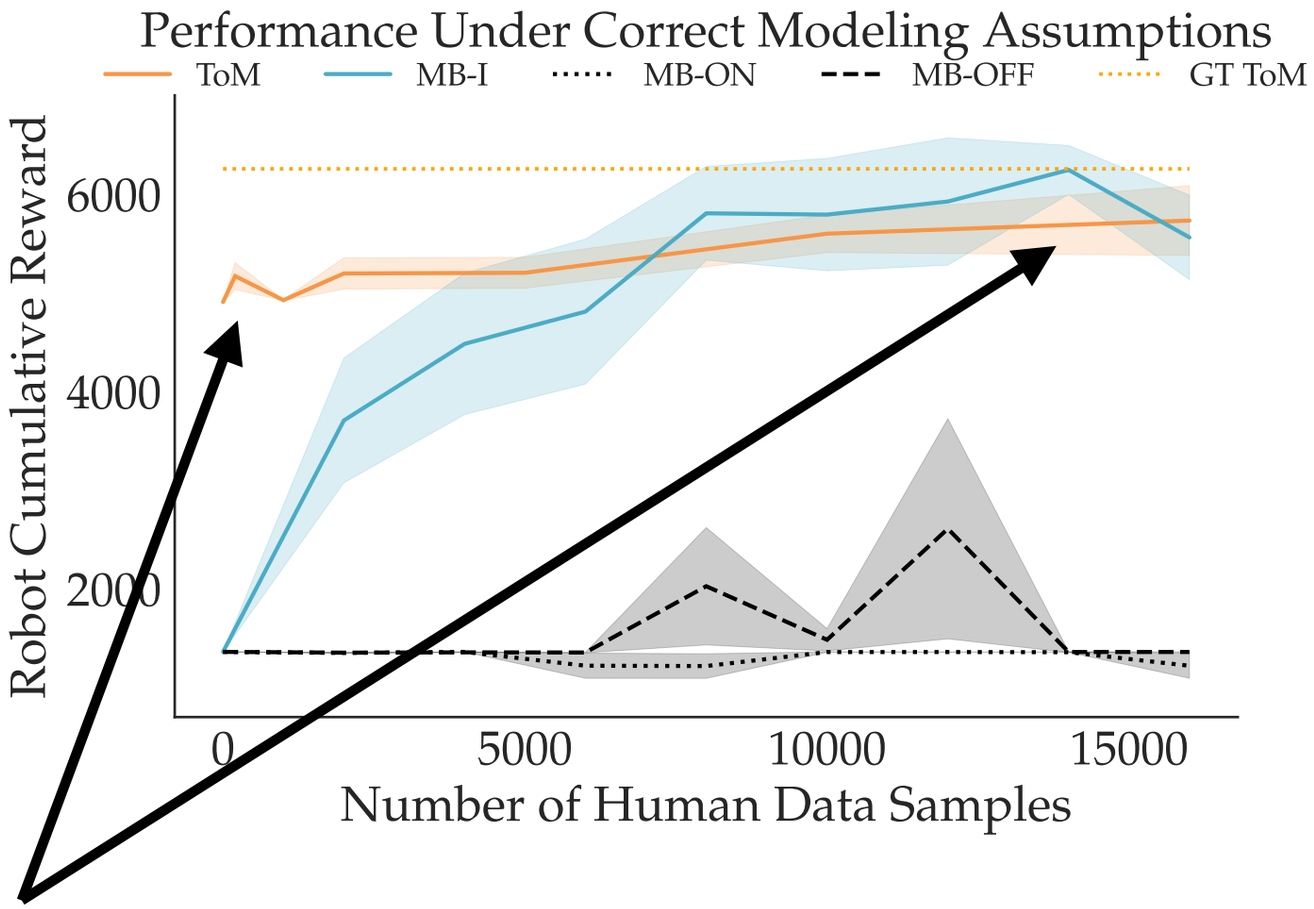
Inferring Utility Functions

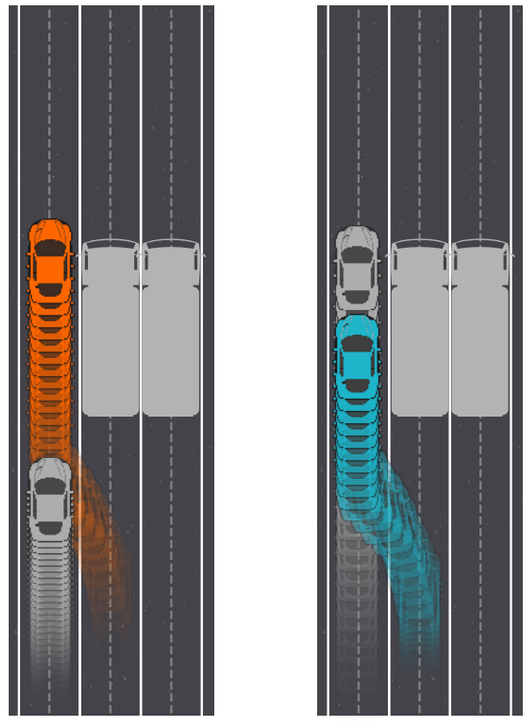
- **Approach**: Inverse Optimal Control / Inverse Reinforcement Learning

  - RL: Given reward function, find best actions
    - IRL: Given actions, find reward function that would have produced these actions

- High level blueprint: Fix a set of features a person driving could care about. Figure out what weights on these features would produce observed behavior on average.

Theory of Mind Results





Much better sample complexity

Questions?