Safe Human-Interactive Control via Shielding

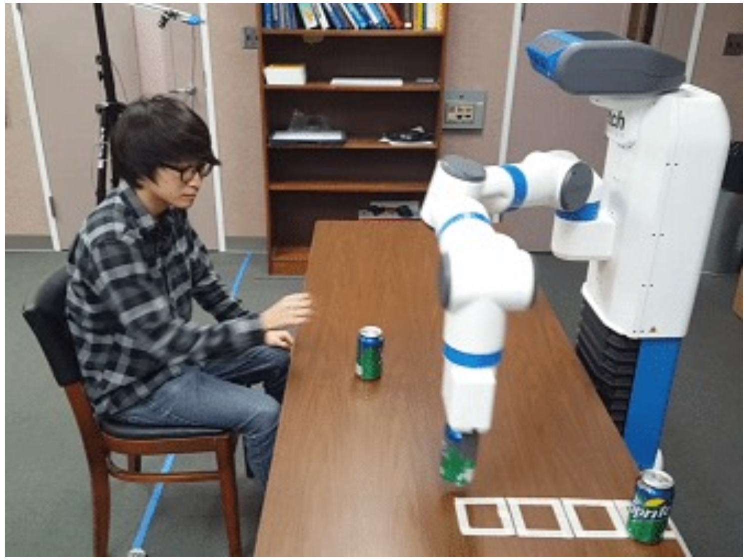
Jeevana Priya Inala[1], Yecheng Jason Ma[2],

Osbert Bastani[2], Xin Zhang[3], Armando Solar-Lezama[1]

1 MIT 2 University of Pennsylvania 3 Peking University

Human-Robot Interaction

- Robots are increasingly being deployed in settings where they must interact with humans
  - Both **cooperative** and **non-cooperative** (not necessarily zero-sum!)

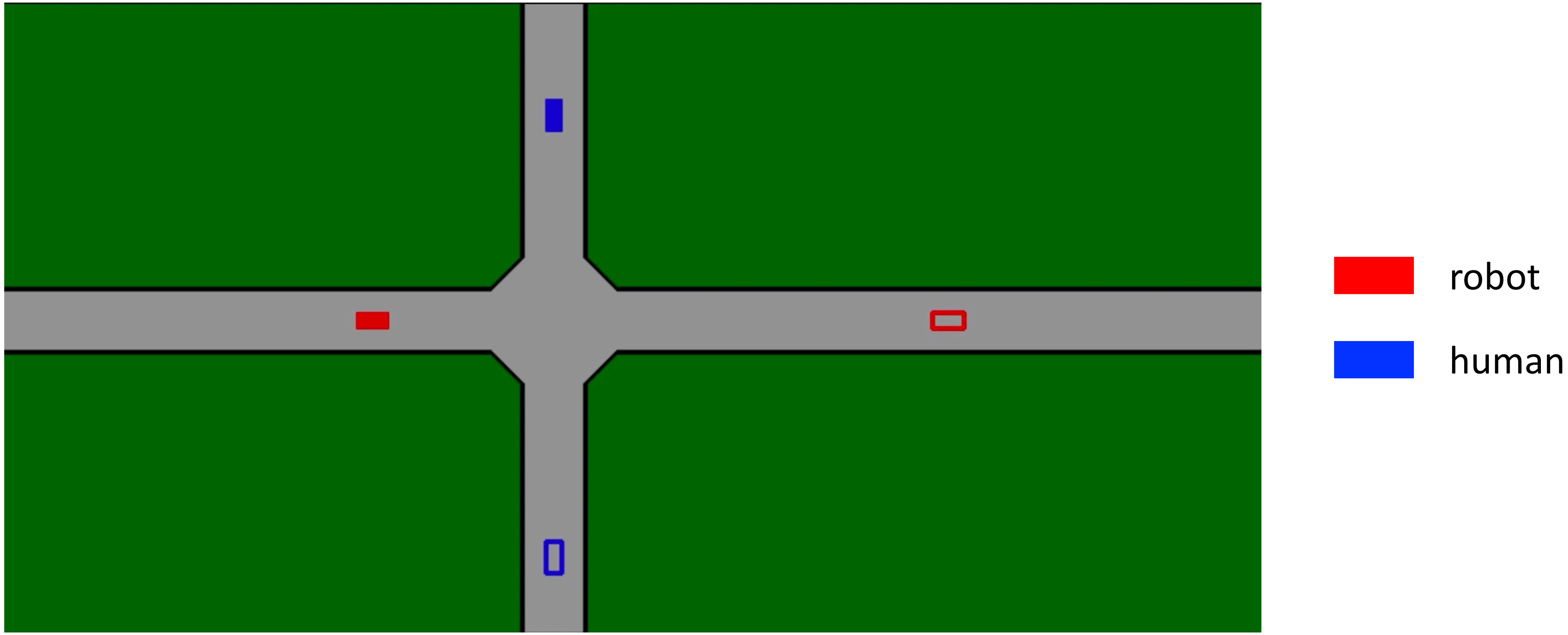


Park et al. Intention-Aware Motion Planning Using Learning Based Human Motion Prediction. RSS 2017

Human-Robot Interaction

- **Key question:** How to model the human?
- **General solution:** Two-player dynamic game

Human-Robot Interaction



(Actual) human and robot negotiate who passes first at an intersection

Roadmap

- Problem formulation
- Background on human interactive control
- Background on shielding
- Our algorithm
- Theoretical guarantees
- Experiments

Roadmap

- **Problem formulation**
- Background on human interactive control
- Background on shielding
- Our algorithm
- Theoretical guarantees
- Experiments

Game Theoretical Formulation

- **Two-player dynamic game**
  - Dynamics $x_{t+1} = f(x_t, u_{R,t}, u_{H,t})$
  - Agent $\delta \in \{R, H\}$ has utility

$$J_\delta(x; \vec{u}_R, \vec{u}_H) = \sum_{t=1}^{T} r_\delta(x_t, u_{\delta,t})$$

- **Human strategy**
  - Nash equilibrium action sequence

Control Problem

- **Robot control**
  - Construct a robot controller $u_{R,t} = \pi_R(x_t)$
- **Goal reaching**
  - Goal region $\mathcal{X}_{\text{goal}}$
  - Reach $x_t \in \mathcal{X}_{\text{goal}}$ for some t
- **Safety**
  - Safe region $\mathcal{X}_{\text{safe}}$
  - Ensure $x_t \in \mathcal{X}_{\text{safe}}$ for all $t \in \{1, \dots, T\}$

Challenges

- **Challenge 1:** Computational complexity
  - Hard to compute Nash equilibrium strategies

- **Challenge 2:** Unknown human reward function
  - Human reward function r_H is unknown

Roadmap

- Problem formulation
- **Background on human interactive control**
- Background on shielding
- Our algorithm
- Theoretical guarantees
- Experiments

Prior Work (Sadigh et al. 2016)

- **Challenge 1:** Computational complexity
  - Re-formulate as Stackelberg game
- **Challenge 2:** Unknown human reward function
  - Use inverse reinforcement learning to infer human reward function

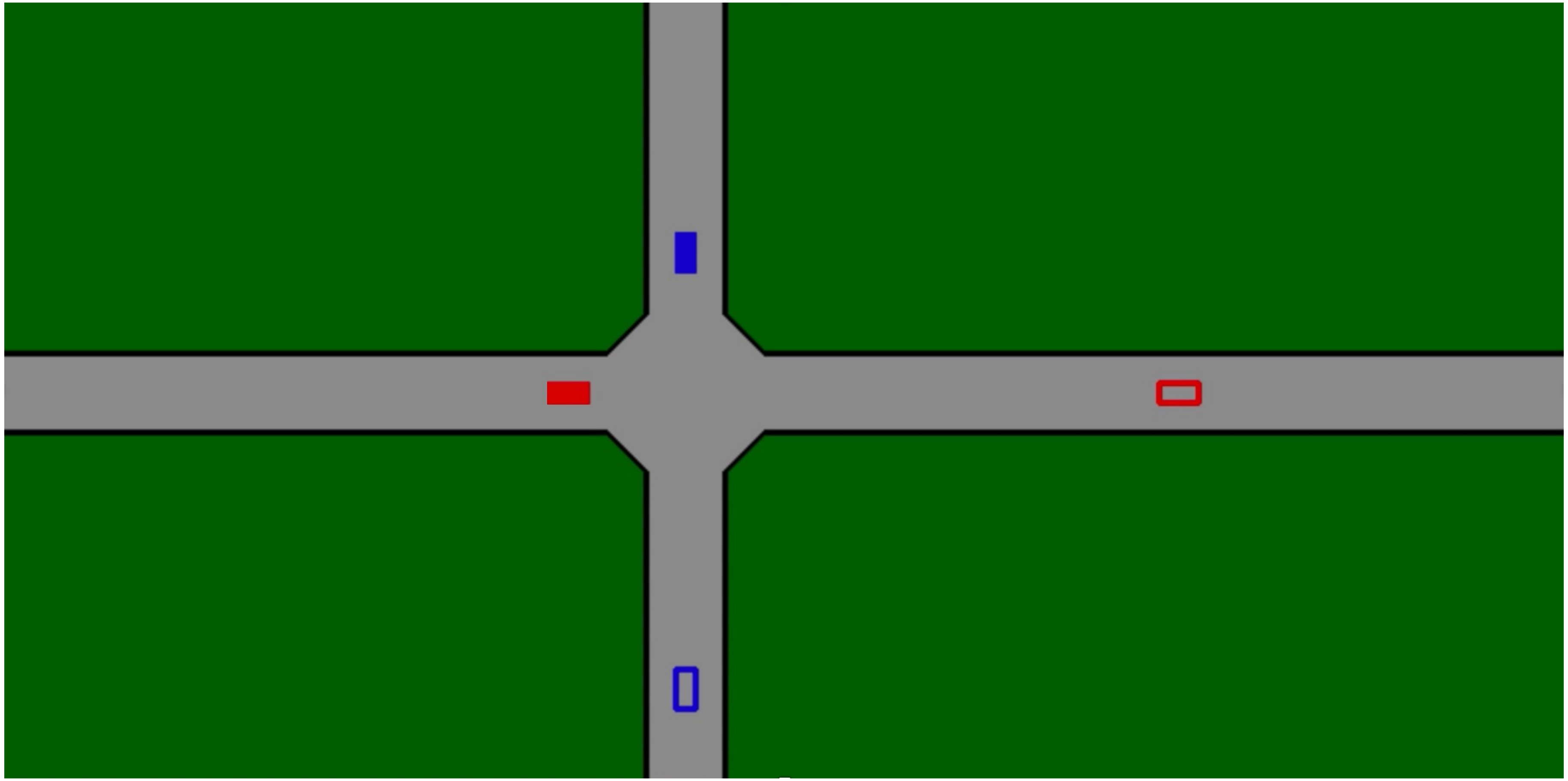
Sadigh, Sastry, Seshia, Dragan, Planning for Autonomous Cars that Leverage Effects on Human Actions. RSS, 2016.

Computational Complexity

- **Stackelberg Game**
  - Agents play sequentially (not simultaneously)
  - Dynamics $x_{t+1} = f_H(f_R(x_t, u_{R,t}), u_{H,t})$

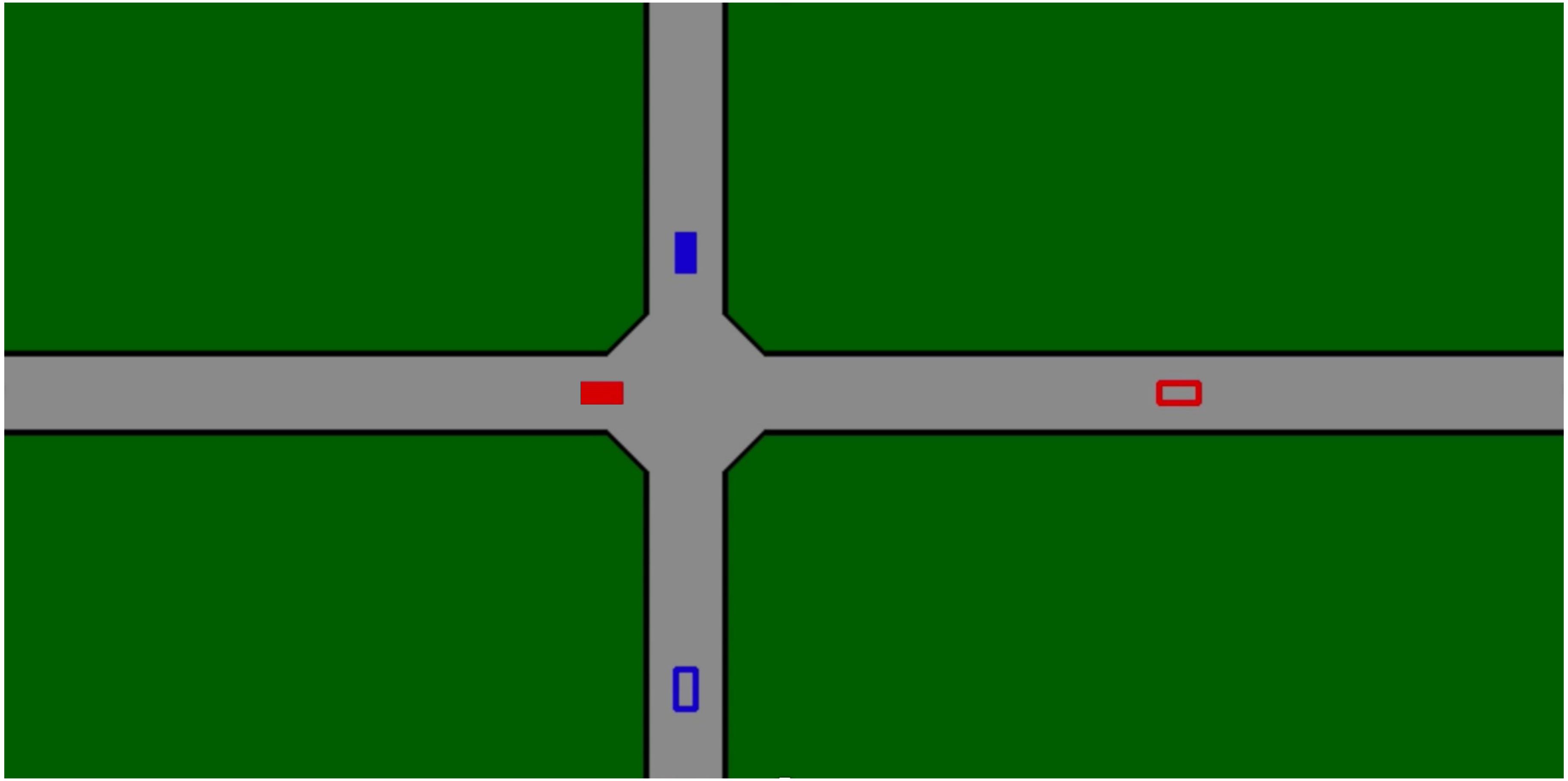
Sadigh, Sastry, Seshia, Dragan, Planning for Autonomous Cars that Leverage Effects on Human Actions. RSS, 2016.

Stackelberg Game



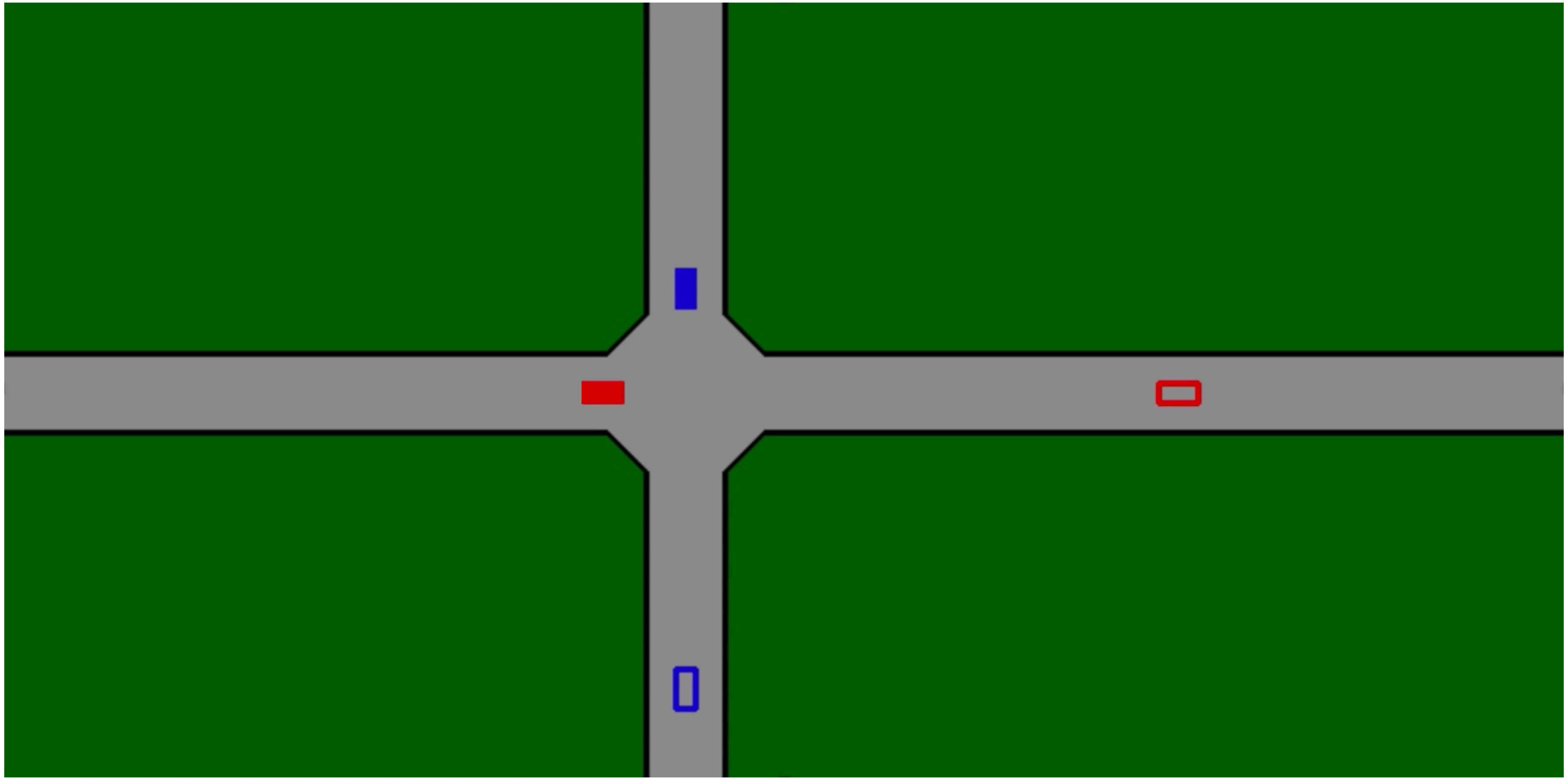
Sadigh, Sastry, Seshia, Dragan, Planning for Autonomous Cars that Leverage Effects on Human Actions. RSS, 2016.

Stackelberg Game



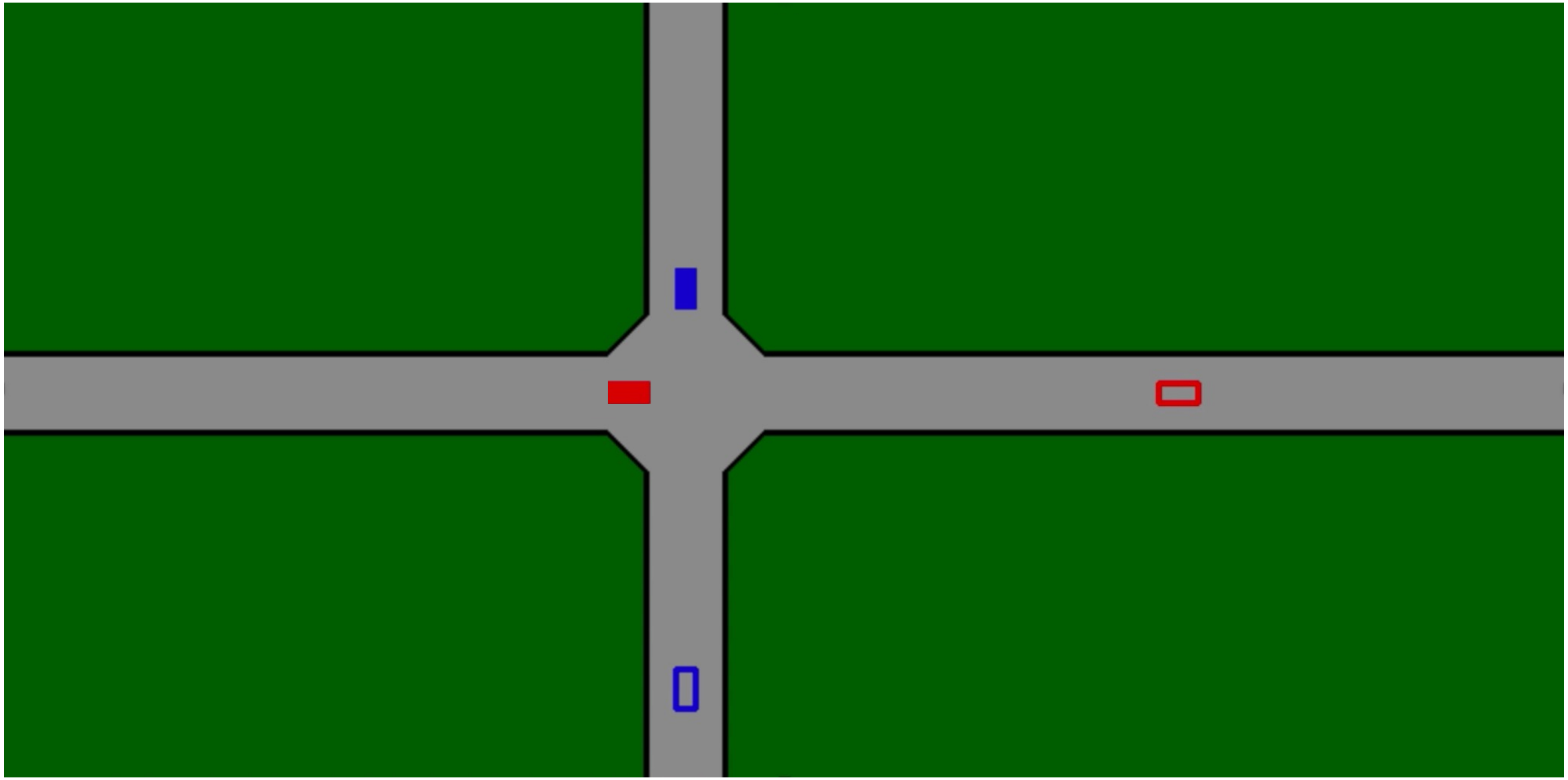
Sadigh, Sastry, Seshia, Dragan, Planning for Autonomous Cars that Leverage Effects on Human Actions. RSS, 2016.

Stackelberg Game



Sadigh, Sastry, Seshia, Dragan, Planning for Autonomous Cars that Leverage Effects on Human Actions. RSS, 2016.

Stackelberg Game



Sadigh, Sastry, Seshia, Dragan, Planning for Autonomous Cars that Leverage Effects on Human Actions. RSS, 2016.

Stackelberg Game

- **Solution strategy**
  - Finite state, finite horizon
  - Then, we can use backward induction:

$$u_{R,t}^*(x) = \arg\max_{u_R} \left\{ r_R(x) + J_{R,t+1}^*\left(f_H\left(f_R(x, u_R), u_{H,t}^*\big(f_R(x, u_R)\big) \right) \right) \right\}$$

$$J_{R,t}^*(x) = \max_{u_R} \left\{ r_R(x) + J_{R,t+1}^*\left(f_H\left(f_R(x, u_R), u_{H,t}^*\big(f_R(x, u_R)\big) \right) \right) \right\}$$

  - Here, $u_{R,t}^*(x)$ is the optimal policy and $J_{R,t}^*(x)$ is the value function
  - Computes a subgame-perfect Nash equilibrium

Sadigh, Sastry, Seshia, Dragan, Planning for Autonomous Cars that Leverage Effects on Human Actions. RSS, 2016.

Stackelberg Game

- **Solution strategy**
  - Finite state, finite horizon
  - Then, we can use backward induction:

$$u^*_{H,t}(x) = \arg\max_{u_H} \left\{ r_H(x) + J^*_{H,t+1}\left(f_R\left(f_H(x, u_H), u^*_{R,t+1}(f_H(x, u_H)) \right) \right) \right\}$$

$$J^*_{H,t}(x) = \max_{u_H} \left\{ r_H(x) + J^*_{H,t+1}\left(f_R\left(f_H(x, u_H), u^*_{R,t+1}(f_H(x, u_H)) \right) \right) \right\}$$

  - Here, $u^*_{H,t}(x)$ is the optimal policy and $J^*_{H,t}(x)$ is the value function
  - Computes a subgame-perfect Nash equilibrium

Sadigh, Sastry, Seshia, Dragan, Planning for Autonomous Cars that Leverage Effects on Human Actions. RSS, 2016.

Stackelberg Game

- **Solution strategy**
  - Continuous state, finite horizon (MPC)
  - Optimize $u_{R,t}^*$ using gradient descent
  - Compute derivative of arg-max using implicit differentiation

Sadigh, Sastry, Seshia, Dragan, Planning for Autonomous Cars that Leverage Effects on Human Actions. RSS, 2016.

Human Reward Function

- **Inverse Reinforcement Learning**
  - Given demonstrations of the human's behavior, choose the reward function that best "describes" the human behavior:

$$\hat{r}_H = \arg\max_r L(\pi(r); D)$$

  - Here, $\pi(r)$ is the optimal policy if the reward function is r, D is the observed dataset of human state-action pairs, and L is a loss function

- **Human Reward Function Inference**
  - Gather demonstrations of human behavior
  - Use off-the-shelf inverse reinforcement learning algorithms to estimate r_H

Sadigh, Sastry, Seshia, Dragan, Planning for Autonomous Cars that Leverage Effects on Human Actions. RSS, 2016.

Shortcomings

- **Challenge 1:** Computational complexity
  - Formulate as Stackelberg game
  - Susceptible to local minima, so not guaranteed to be a Nash equilibrium
- **Challenge 2:** Unknown human reward function
  - Use inverse reinforcement learning to infer human reward function
  - No guarantee that inferred reward function is correct

Sadigh, Sastry, Seshia, Dragan, Planning for Autonomous Cars that Leverage Effects on Human Actions. RSS, 2016.

Our Goal

- **Design a robot controller that**
  - Guarantees safety
  - Best attempt to reach goal (but no guarantees)

- **Assumptions**
  - Necessary to make some assumptions about human reward function
  - Goal is to minimize the required assumptions

Roadmap

- Problem formulation
- Background on human interactive control
- **Background on shielding**
- Our algorithm
- Theoretical guarantees
- Experiments

Shielding

- **Untrustworthy Policy $\hat{\boldsymbol{\pi}}$**
  - Achieves good performance
  - May be unsafe
- **Backup Policy $\boldsymbol{\pi}_{\mathbf{backup}}$**
  - May perform poorly
  - Can safely bring the system to a stop from $x \in \mathcal{X}_{\text{rec}} \subseteq \mathcal{X}_{\text{safe}}$
  - Say x is **recoverable**
- **Strategy**
  - **Safety:** Override $\hat{\pi}$ using π_{backup} to guarantee safety
  - **Goal-reaching:** Minimally override $\hat{\pi}$ to ensure performance (no guarantees)

Recoverability

We have $x \in \mathcal{X}_{\text{rec}}$ if π_{backup} safely brings the robot to a stop ($v = 0$)

Shielding Algorithm

- **Algorithm**
  - Use $\hat{\pi}$ if $\hat{x}_{t+1} = f\left(x_t, \hat{\pi}(x_t)\right) \in \mathcal{X}_{\text{rec}}$
  - Use π_{backup} otherwise
- **Theorem:** This algorithm maintains the invariant

$$x_t \in \mathcal{X}_{\text{rec}} \Rightarrow x_{t+1} \in \mathcal{X}_{\text{rec}}$$

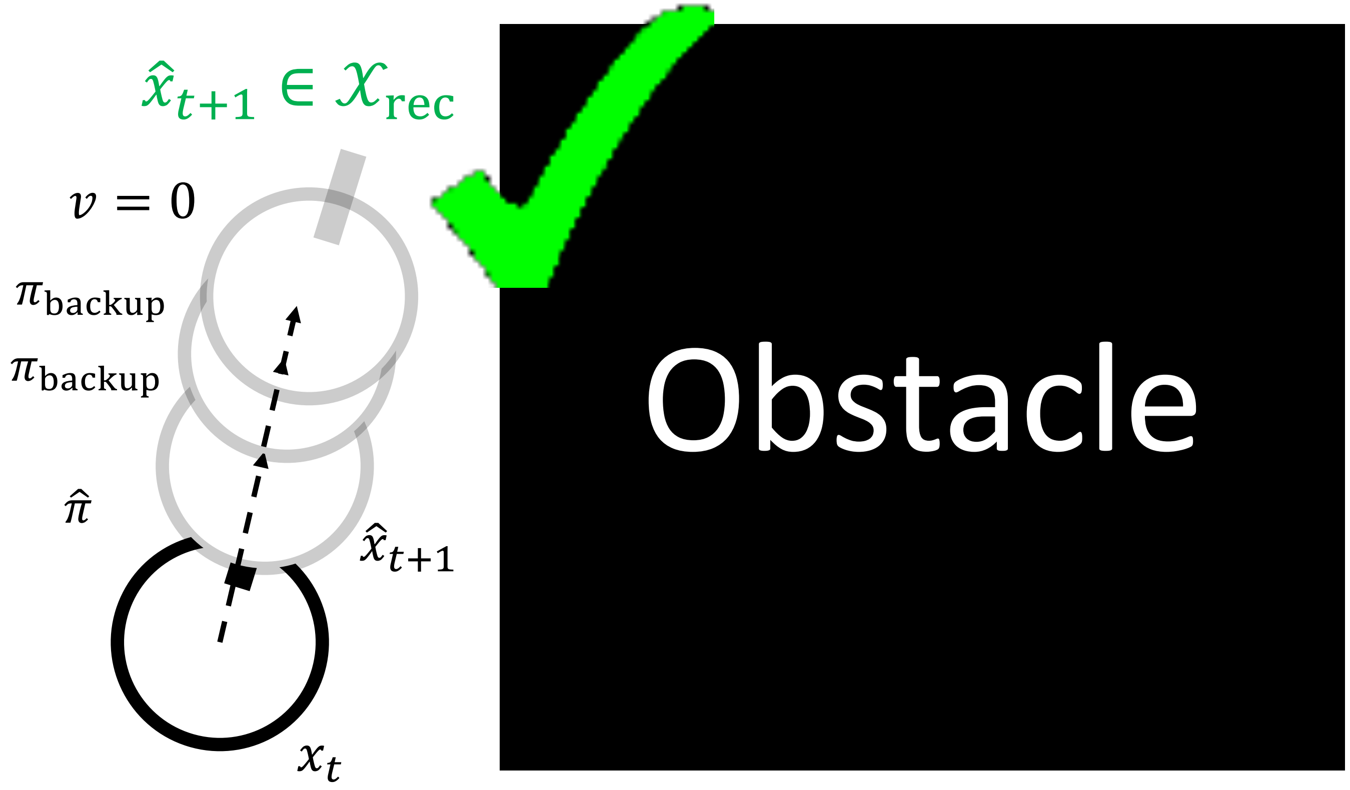
- **Proof**
  - **Case 1:** It uses $\hat{\pi}$; then, the result follows by the condition
  - **Case 2:** It uses π_{backup}; then, it follows since $x_t \in \mathcal{X}_{\text{rec}}$ implies that using π_{backup} is safe, so $x_{t+1} = f\left(x_t, \pi_{\text{backup}}(x_t)\right)$ must also be recoverable

Model Predictive Shielding

- **Challenge:** $\mathcal{X}_{\mathrm{rec}}$ is often hard to compute in closed form
- **Key idea:** We can check $x \in \mathcal{X}_{\mathrm{rec}}$ using model-based simulation

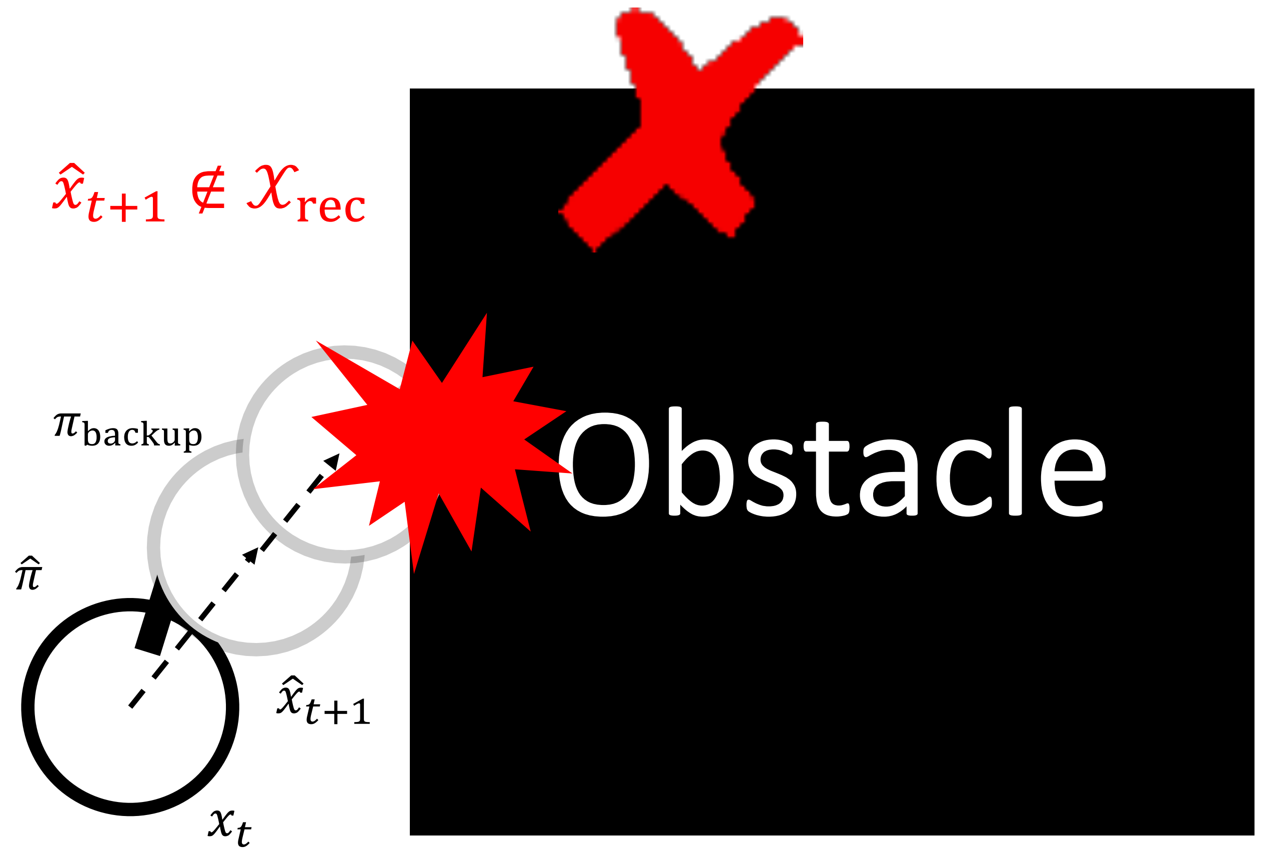
Li & Bastani 2020, Bastani 2021

Checking Recoverability



Simulation to see if $\hat{x}_{t+1} \in \mathcal{X}_{\text{rec}}$

Checking Recoverability



Simulation to see if $\hat{x}_{t+1} \in \mathcal{X}_{\text{rec}}$

Model Predictive Shielding

if $\hat{x}_{t+1} \in \mathcal{X}_{\mathrm{rec}}$ ✓ **then use** $\hat{\pi}$

if $\hat{x}_{t+1} \notin \mathcal{X}_{\mathrm{rec}}$ ✗ **then use** π_{backup}

Roadmap

- Problem formulation
- Background on human interactive control
- Background on shielding
- **Our algorithm**
- Theoretical guarantees
- Experiments

Our Approach

- **High-level strategy**
  - Do **not** try to compute a Nash Equilibrium solution
  - Instead, act conservatively with respect to rational human

- **Key idea**
  - Assume human prioritizes safety
  - We only need to prove that the human **can** maintain safety
  - Then, rationality implies that the human **does** maintain safety

Simplified Human Model

- **Simplified Stackelberg Game**
  - Assume that human plays conservatively with respect to possible future robot actions $\vec{u}_R \subseteq \hat{\mathcal{U}}_R$:

$$\vec{u}_R^* = \arg\max_{\vec{u}_R} J_R\left(x; \vec{u}_R, \vec{u}_H^*(\vec{u}_{R,1})\right)$$

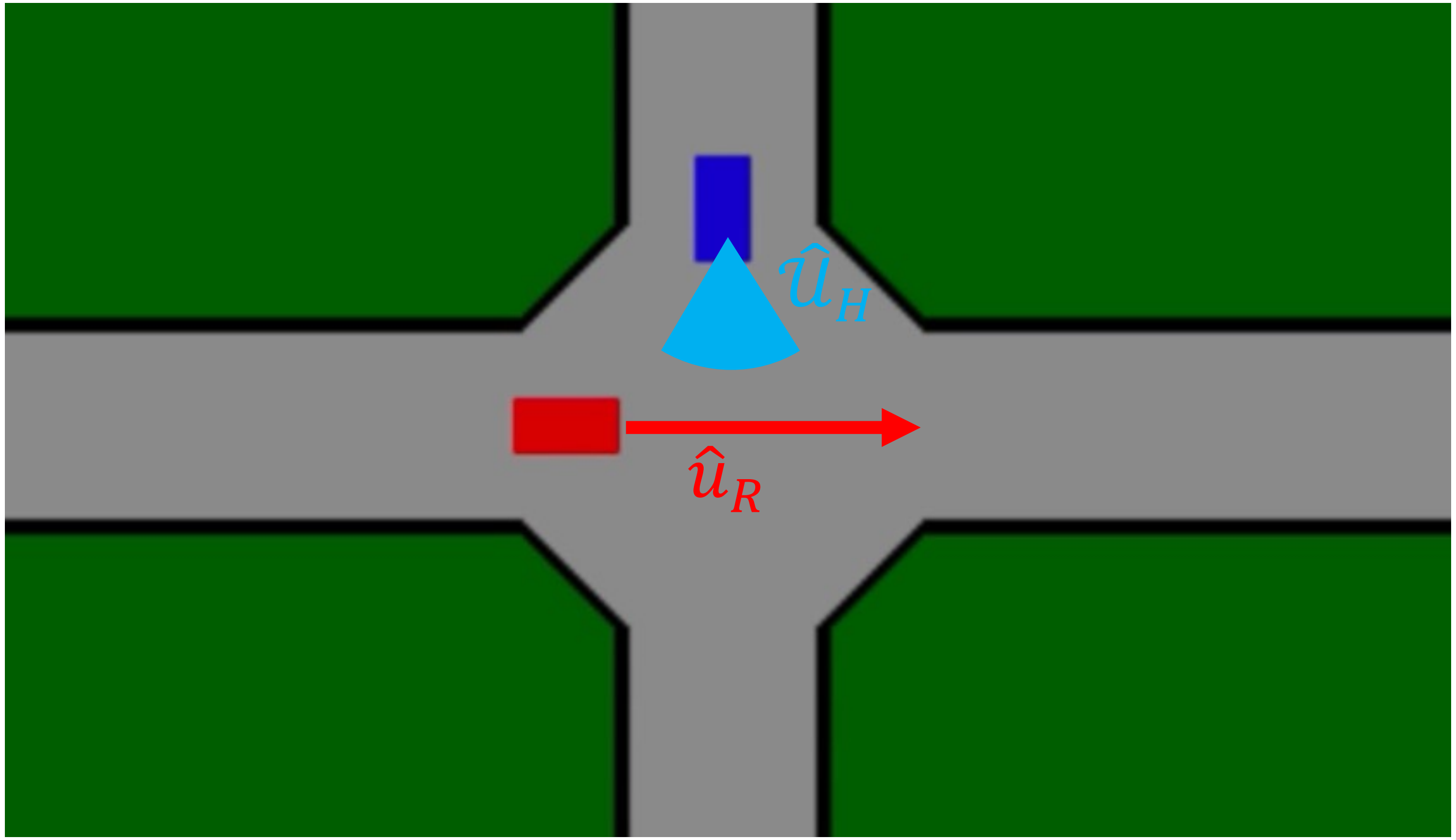
$$\vec{u}_H^*(u_{R,1}) = \arg\max_{\vec{u}_H} \min_{\vec{u}_{R,2:T}} J_H(x; u_{R,1} \circ \vec{u}_{R,2:T}, \vec{u}_H)$$

- **Intuition:** Reduces problem to 1-step Stackelberg game
- **Justification:** Human cannot anticipate exactly what the robot is going to do, so they must act conservatively

Assumptions on Human Objective

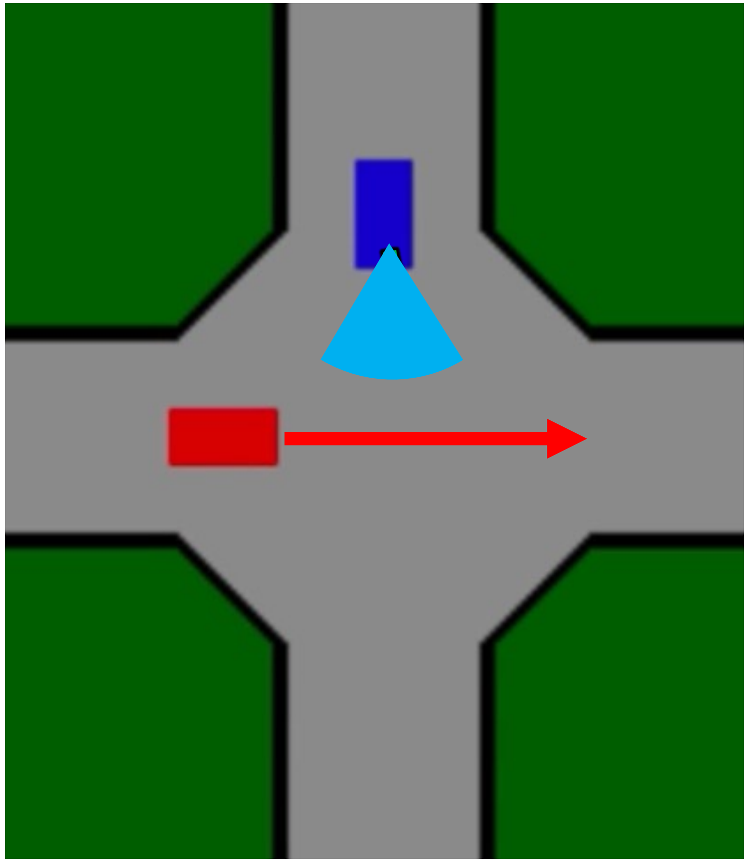
- **Human policy:** $\vec{u}_H^*(u_{R,1}) = \arg\max_{\vec{u}_H} \min_{\vec{u}_{R,2:T}} J_H(x; u_{R,1} \circ \vec{u}_{R,2:T}, \vec{u}_H)$
- **Assumption 1:** Human rewards for unsafety
  - We have $r_H(x_t, u_{H,t}) = -\infty$ if $x_t \notin \mathcal{X}_{\text{safe}}$
- **Assumption 2:** Human predicted robot backup action
  - We are given a **robot backup action** $\hat{u}_R$ that the robot can use to come to a stop
  - The human acts conservatively with respect to this action
- **Assumption 3:** Human backup actions
  - We are given **human backup actions** $\hat{\mathcal{U}}_H$ that the human can use to come to a stop
  - If the human's objective value is $-\infty$, then they use some action $u_H \in \hat{\mathcal{U}}_H$

Assumptions on Human Objective



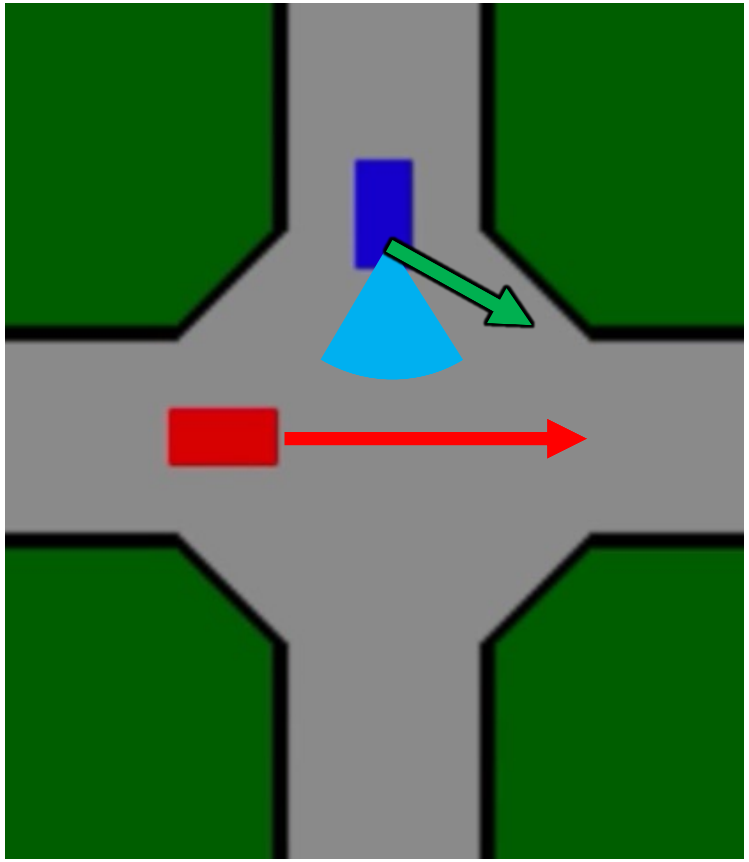
Algorithm Overview

- **Human model**
  - $\vec{u}_H^* = \arg\max_{\vec{u}_H} \min_{\vec{u}_{R,2:T}} J_H(x; u_{R,1} \circ \vec{u}_{R,2:T}, \vec{u}_H)$
- **Recoverability**
  - Say x is **recoverable** if using $\hat{u}_R$ from x safely brings the system to a stop
  - Depends on the **unknown human policy**
- **Algorithm**
  - Say x is **recoverable** if the system safely comes to a stop for $\vec{u}_{R,2:T} = \vec{\hat{u}}_R$ and for all $\vec{u}_H \subseteq \hat{U}_H$



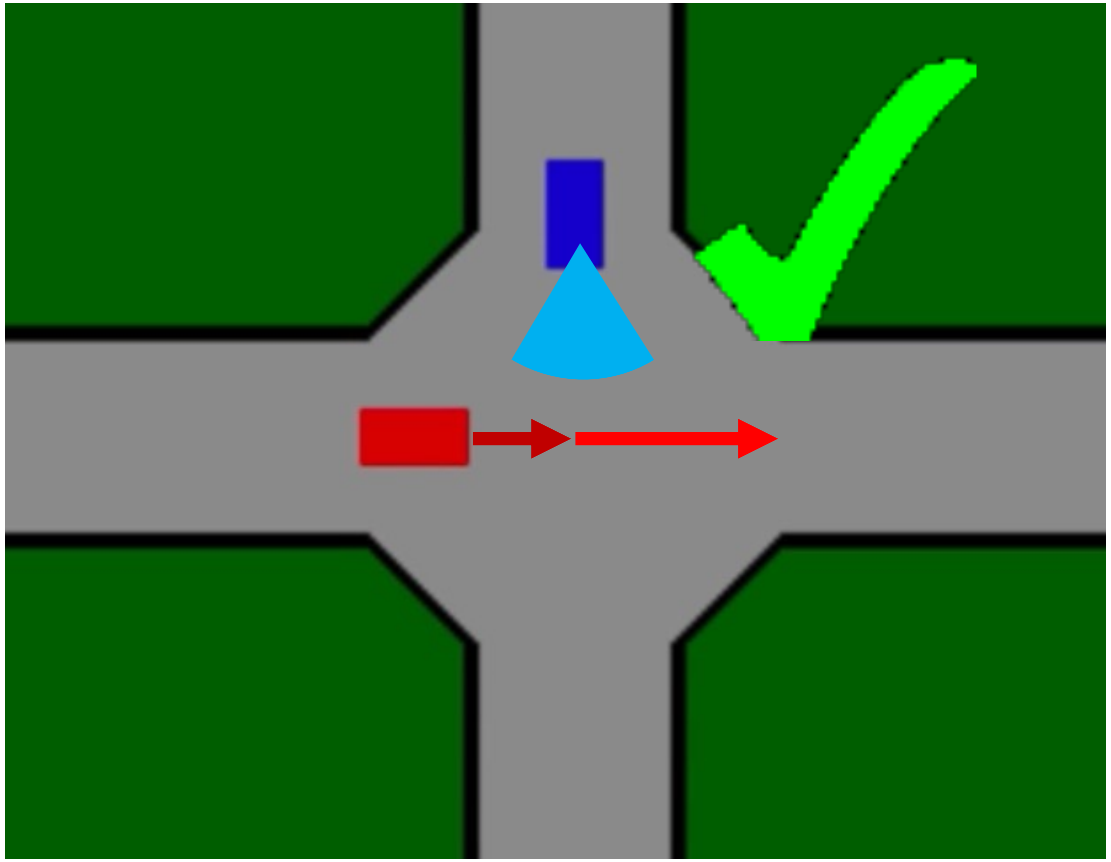
Algorithm Overview

- **Human model**
  - $\vec{u}_H^* = \arg\max_{\vec{u}_H} \min_{\vec{u}_{R,2:T}} J_H(x; u_{R,1} \circ \vec{u}_{R,2:T}, \vec{u}_H)$
- **Recoverability**
  - Say x is **recoverable** if using $\hat{u}_R$ from x safely brings the system to a stop
  - Depends on the **unknown human policy**
- **Algorithm**
  - Say x is **recoverable** if the system safely comes to a stop for $\vec{u}_{R,2:T} = \vec{\hat{u}}_R$ and for all $\vec{u}_H \subseteq \hat{U}_H$
  - The human may not take such a $\vec{u}_H$, but then they take an action $\vec{u}_H'$ that is **better** than $\vec{u}_H$



Algorithm

- **Step 1:** Compute the **human backup region** that the human can reach using $u_H \in \hat{\mathcal{U}}_H$
  - It is **not a reachable set**; the human may drive outside of it
- **Step 2:** Compute the **robot backup trajectory** that the robot takes using $\hat{\pi}$ for one step followed by $\hat{u}_R$
- **Step 3:** Use $\hat{\pi}$ if these do not overlap, and $\hat{u}_R$ otherwise

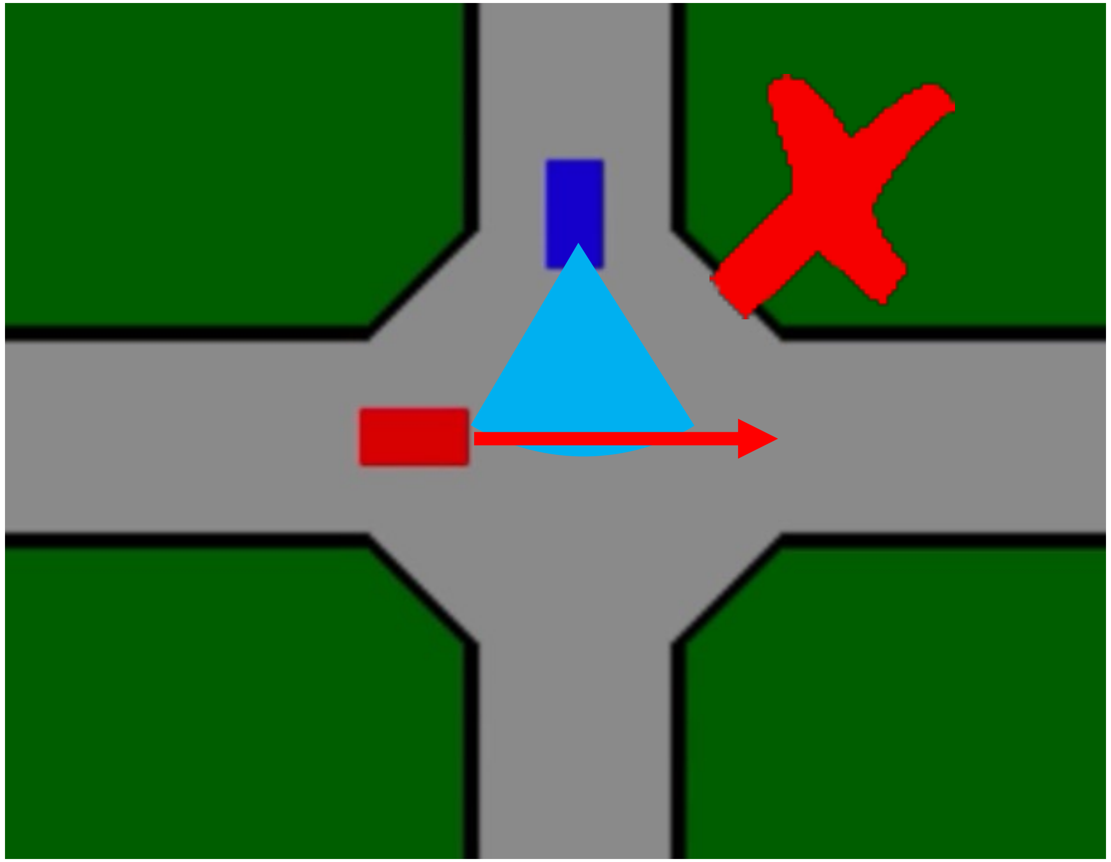


Algorithm

- **Step 1:** Compute the **human backup region** that the human can reach using $u_H \in \hat{\mathcal{U}}_H$
  - It is **not a reachable set**; the human may drive outside of it

- **Step 2:** Compute the **robot backup trajectory** that the robot takes using $\hat{\pi}$ for one step followed by $\hat{u}_R$

- **Step 3:** Use $\hat{\pi}$ if these do not overlap, and $\hat{u}_R$ otherwise



Step 1 & 2: Robot/Human Backup Regions

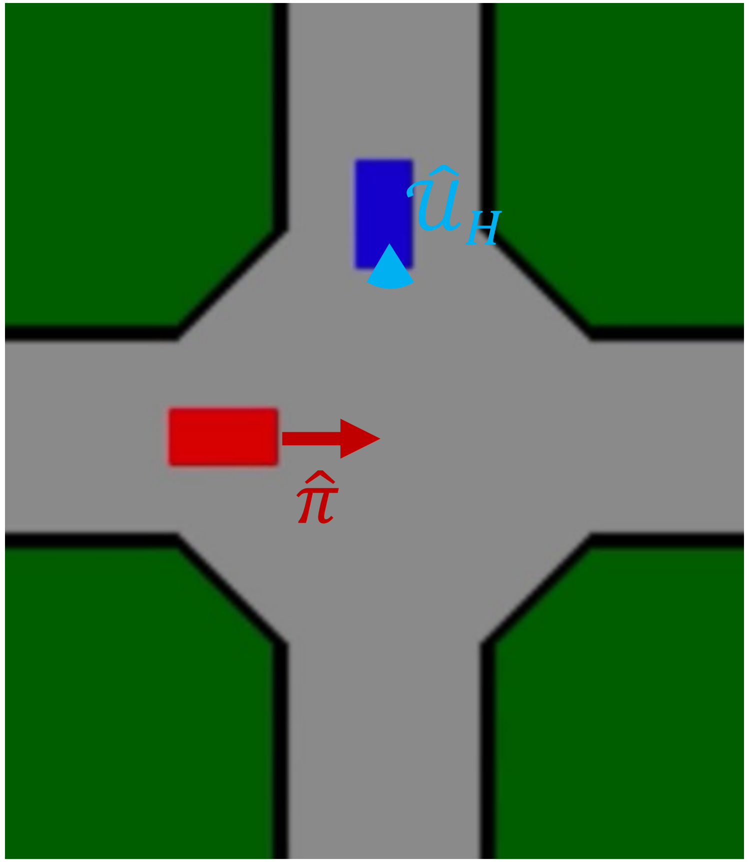
- **Goal**
  - Compute reachable set under robot backup action $\hat{u}_R$ and human backup actions $\hat{\mathcal{U}}_H$

- **Algorithm**
  - Given $F_\delta: 2^{\mathcal{X}} \times 2^{\mathcal{U}_\delta} \to 2^{\mathcal{X}}$ for $\delta \in \{R, H\}$ s.t.

$$\{ f_\delta(x, u) \mid x \in X, u \in U \} \subseteq F_\delta(X, U)$$

  - Compute

$$X_2 = F_H\big(F_R(\{x\}, \{\hat{\pi}(x)\}), \hat{\mathcal{U}}_H\big)$$
$$X_{t+1} = F_H\big(F_R(X_t, \{\hat{u}_R\}), \hat{\mathcal{U}}_H\big)$$



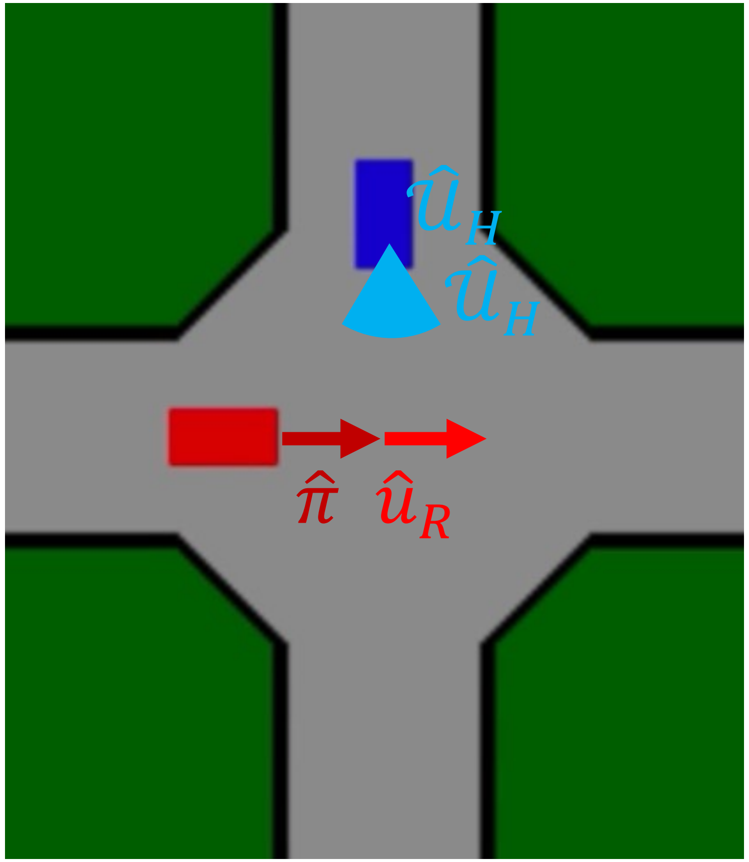
Step 1 & 2: Robot/Human Backup Regions

- **Goal**
  - Compute reachable set under robot backup action $\hat{u}_R$ and human backup actions $\hat{\mathcal{U}}_H$
- **Algorithm**
  - Given $F_\delta: 2^{\mathcal{X}} \times 2^{\mathcal{U}_\delta} \to 2^{\mathcal{X}}$ for $\delta \in \{R, H\}$ s.t.

    $$\{\, f_\delta(x, u) \mid x \in X, u \in U \,\} \subseteq F_\delta(X, U)$$

  - Compute

    $$X_2 = F_H\left(F_R(\{x\}, \{\hat{\pi}(x)\}), \hat{\mathcal{U}}_H\right)$$
    $$X_{t+1} = F_H\left(F_R(X_t, \{\hat{u}_R\}), \hat{\mathcal{U}}_H\right)$$



Step 1 & 2: Robot/Human Backup Regions

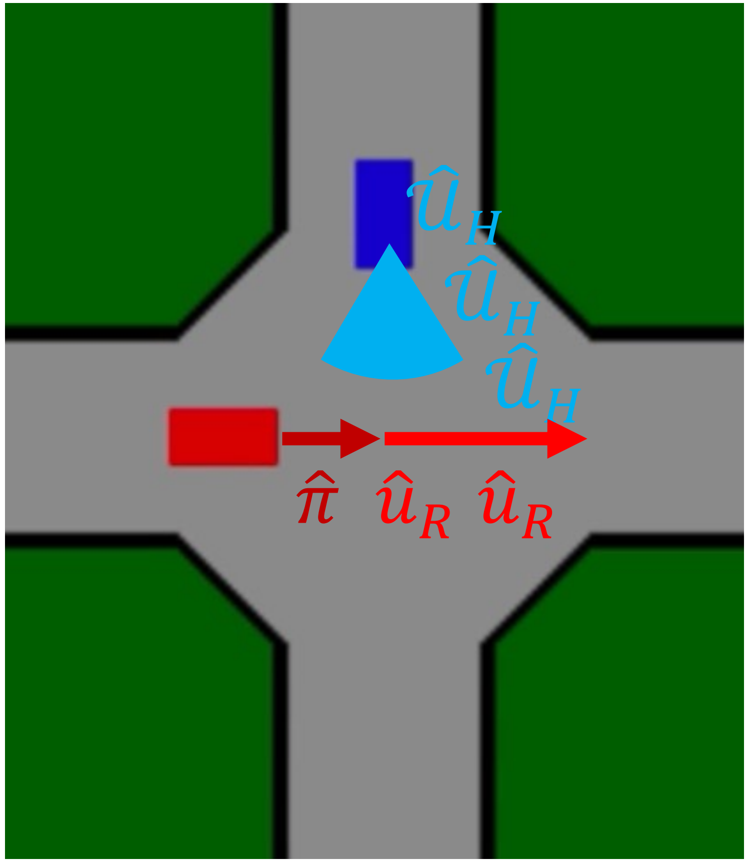
- **Goal**
  - Compute reachable set under robot backup action $\hat{u}_R$ and human backup actions $\hat{\mathcal{U}}_H$

- **Algorithm**
  - Given $F_\delta: 2^{\mathcal{X}} \times 2^{\mathcal{U}_\delta} \to 2^{\mathcal{X}}$ for $\delta \in \{R, H\}$ s.t.

    $$\{\, f_\delta(x, u) \mid x \in X, u \in U \,\} \subseteq F_\delta(X, U)$$

  - Compute

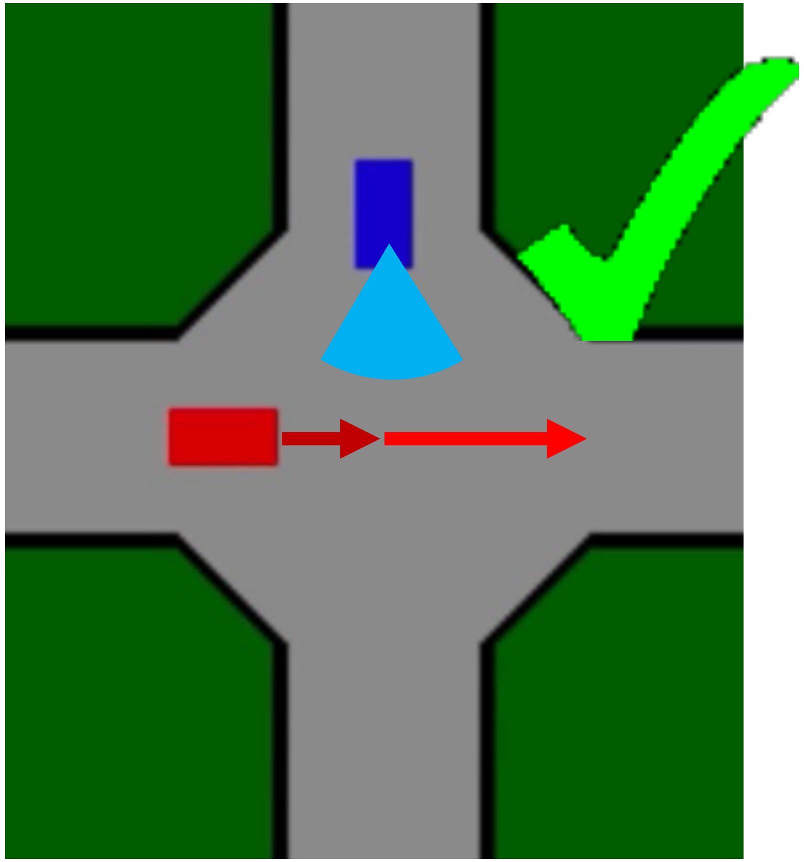
    $$X_2 = F_H\big(F_R(\{x\}, \{\hat{\pi}(x)\}), \hat{\mathcal{U}}_H\big)$$
    $$X_{t+1} = F_H\big(F_R(X_t, \{\hat{u}_R\}), \hat{\mathcal{U}}_H\big)$$



Step 3: Robot/Human Backup Regions

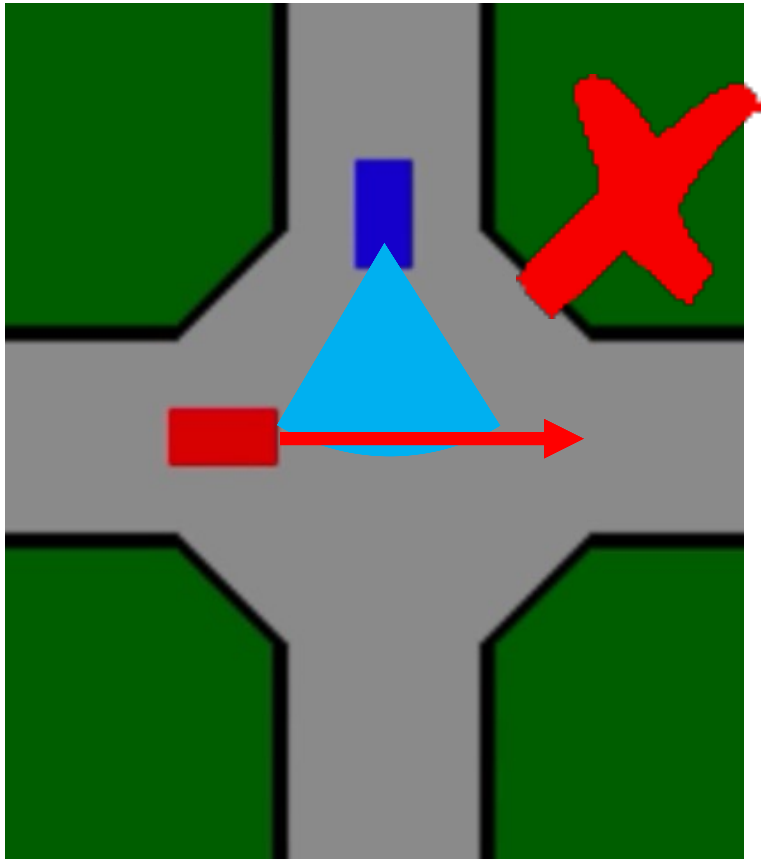
- **Goal**
  - Check safety
  - Check if the system comes to a stop

- **Algorithm**
  - Check $X_t \subseteq \mathcal{X}_{\text{safe}}$ (for all t)
  - Check $X_T \subseteq \mathcal{X}_{\text{eq}}$



Step 3: Robot/Human Backup Regions

- **Goal**
  - Check safety
  - Check if the system comes to a stop
- **Algorithm**
  - Check $X_t \subseteq \mathcal{X}_{\text{safe}}$ (for all t)
  - Check $X_T \subseteq \mathcal{X}_{\text{eq}}$



Roadmap

- Problem formulation
- Background on human interactive control
- Background on shielding
- Our algorithm
- **Theoretical guarantees**
- Experiments

Theoretical Guarantee

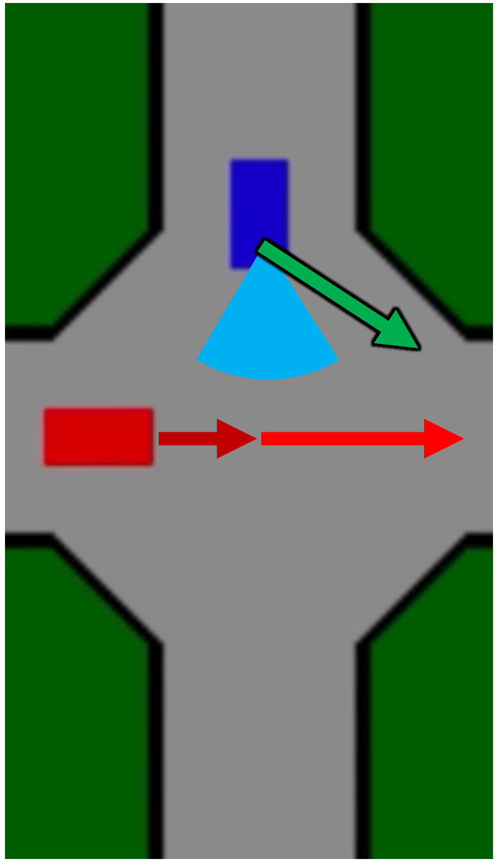
- **Assumptions**
  - Our model of human behavior holds
  - Assumptions 1, 2, & 3
  - The human and robot are at rest at x_1 (i.e., $x_1 \in \mathcal{X}_{\text{eq}}$)

- **Theorem**
  - Our algorithm ensures $x_t \in \mathcal{X}_{\text{safe}}$ for all t

- **Proof**
  - Prove by induction that $x_t \in \mathcal{X}_{\text{rec}} \subseteq \mathcal{X}_{\text{safe}}$
  - **Case 1:** Robot uses $\hat{\pi}$
  - **Case 2:** Robot uses $\pi_{\text{backup}}(x) = \hat{u}_R$

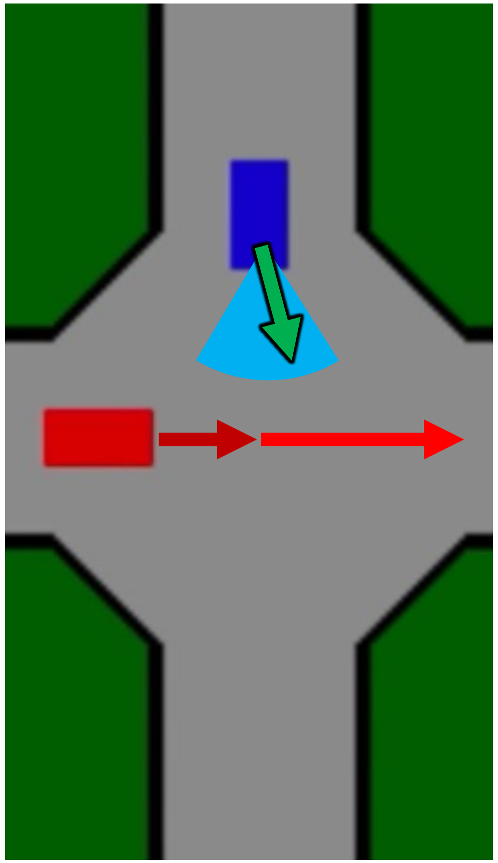
Case 1: Robot uses $\hat{\pi}$

- **Human model**
  - $\vec{u}_H^*(\vec{u}_{R,2:T}) = \arg\max_{\vec{u}_H} \tilde{J}_H(x; u_{R,1}, \vec{u}_H)$
  - $\tilde{J}_H(x; u_{R,1}, \vec{u}_H) = \min_{\vec{u}_{R,2:T}} J_H(x; u_{R,1} \circ \vec{u}_{R,2:T}, \vec{u}_H)$
- **Proof that $x_t \in \mathcal{X}_{\text{rec}} \Rightarrow x_{t+1} \in \mathcal{X}_{\text{rec}}$**
  - Since the robot uses $\hat{\pi}$, the **human backup region** and the **robot backup trajectory** do not overlap
  - If $\tilde{J}_H(x; u_{R,1}, \vec{u}_H) > -\infty$, then the human action is safe for $\vec{u}_{R,2:T}$ (Assumption 1), which includes $\vec{\hat{u}}_R$ (Assumption 2)



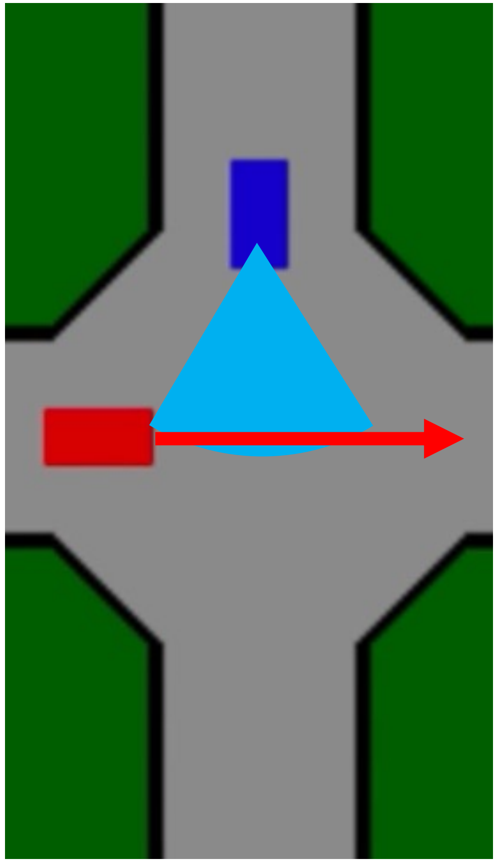
Case 1: Robot uses $\hat{\pi}$

- **Human model**
  - $\vec{u}_H^*(\vec{u}_{R,2:T}) = \arg\max_{\vec{u}_H} \tilde{J}_H(x; u_{R,1}, \vec{u}_H)$
  - $\tilde{J}_H(x; u_{R,1}, \vec{u}_H) = \min_{\vec{u}_{R,2:T}} J_H(x; u_{R,1} \circ \vec{u}_{R,2:T}, \vec{u}_H)$
- **Proof that $x_t \in \mathcal{X}_{\text{rec}} \Rightarrow x_{t+1} \in \mathcal{X}_{\text{rec}}$**
  - Since the robot uses $\hat{\pi}$, the **human backup region** and the **robot backup trajectory** do not overlap
  - If $\tilde{J}_H(x; u_{R,1}, \vec{u}_H) > -\infty$, then the human action is safe for $\vec{u}_{R,2:T}$ (Assumption 1), which includes $\vec{\hat{u}}_R$ (Assumption 2)
  - If $\tilde{J}_H(x; u_{R,1}, \vec{u}_H) = -\infty$, then the human takes an action $u_{H,1} \in \hat{\mathcal{U}}_H$ (Assumption 3), which is safe



Case 2: Robot uses π_{backup}

- **Proof that** $x_t \in \mathcal{X}_{\text{rec}} \Rightarrow x_{t+1} \in \mathcal{X}_{\text{rec}}$
  - The **human backup region** and the **robot backup trajectory** may overlap
  - However, by definition of $x_t \in \mathcal{X}_{\text{rec}}$, using π_{backup} from x_t safely brings the system to a stop
  - Since we used π_{backup}, the same must be true of x_{t+1}



Roadmap

- Problem formulation
- Background on human interactive control
- Background on shielding
- Our algorithm
- Theoretical guarantees
- **Experiments**

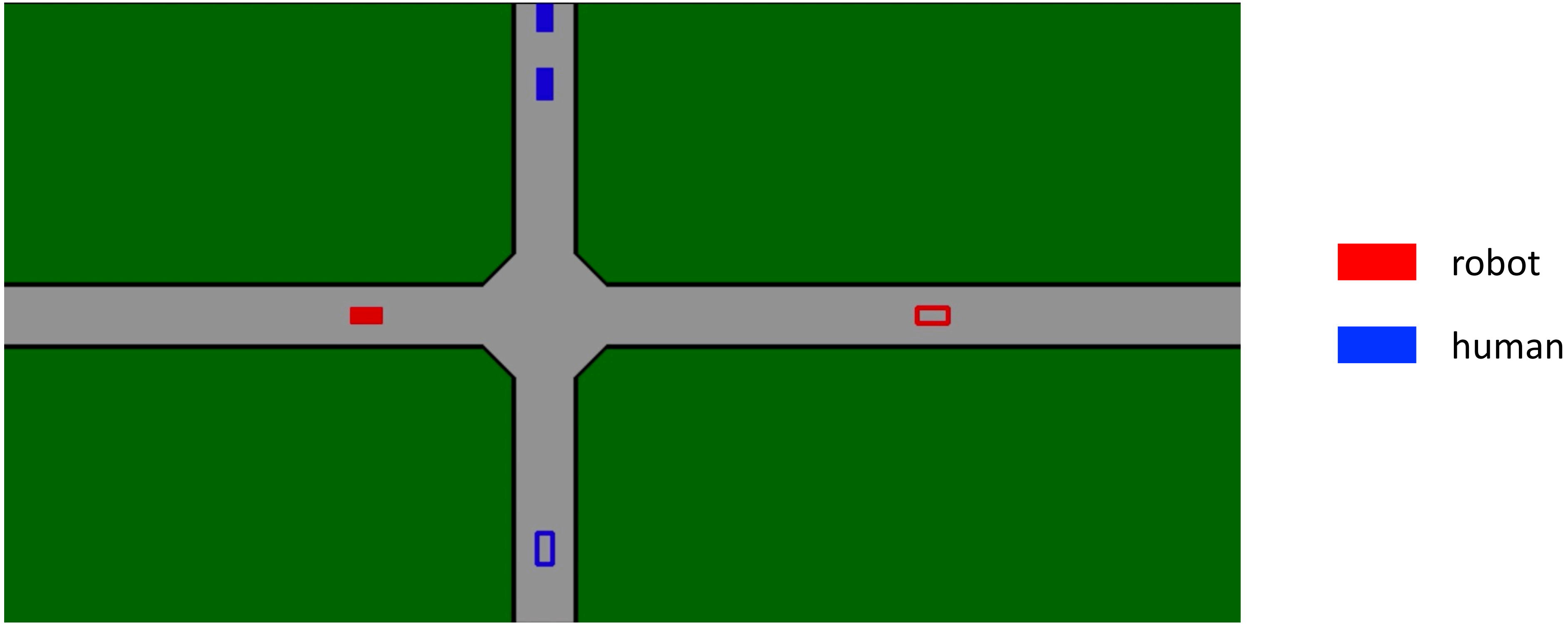
Experimental Setup

- **Environment**
  - Cars with bicycle dynamics
  - Control input is acceleration and steering angle
  - Several different driving tasks
- **Robot**
  - Our approach (MPS) + Aggressive controller that drives straight to goal
  - Model predictive control (MPC) baseline
- **Humans**
  - Simulated humans (social forces model)
  - Real humans interacting with the simulation via keyboard

MPS Parameters

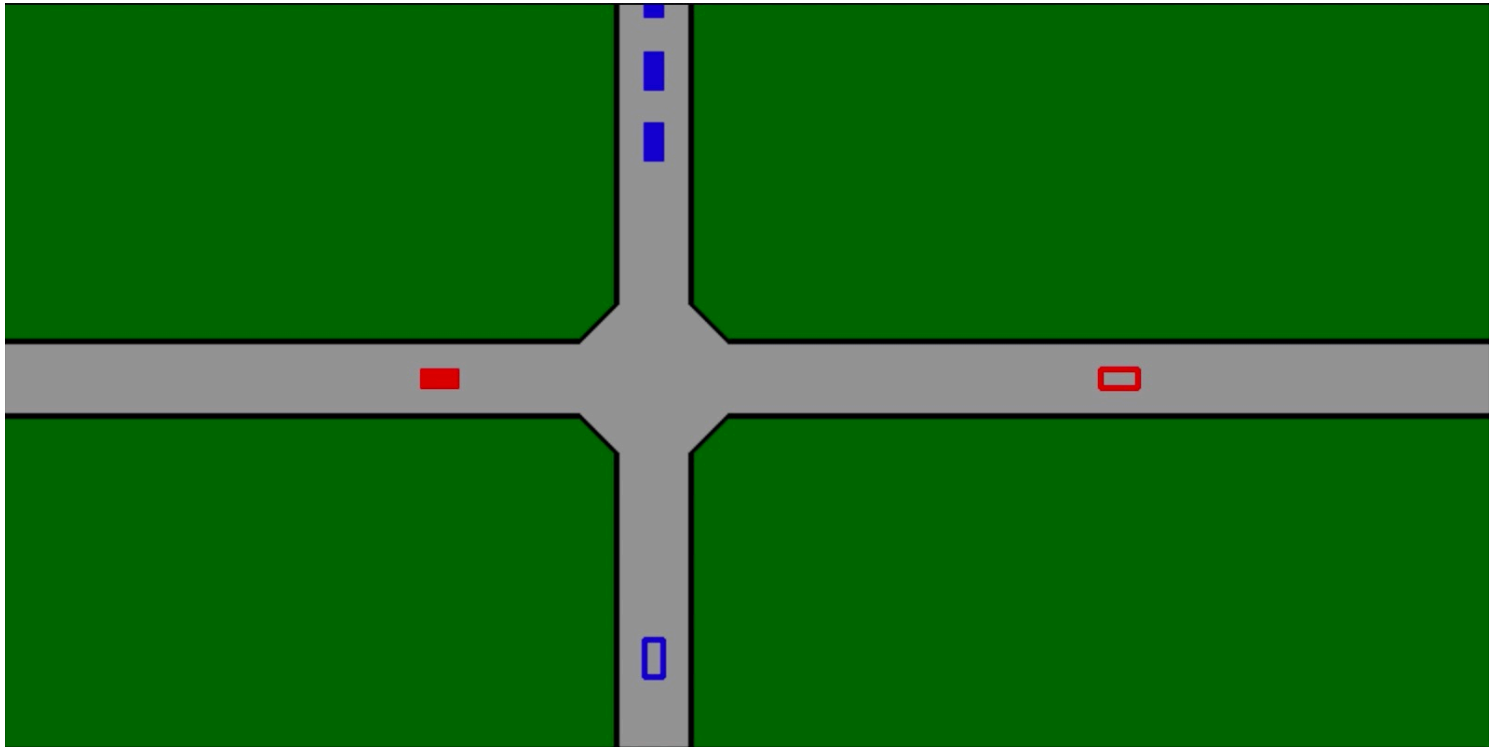
- **Robot backup action $\hat{\boldsymbol{u}}_{\boldsymbol{R}}$**
  - Brake at a given deceleration $a = -a_R$
  - Steering angle $\phi = 0$
- **Human backup actions $\widehat{\mathcal{U}}_{\boldsymbol{H}}$**
  - Brake at a deceleration $a \in [-a_H, -a_H']$
  - Steering angle $\phi \in [-\phi_H, \phi_H]$

Our Approach + Simulated Humans



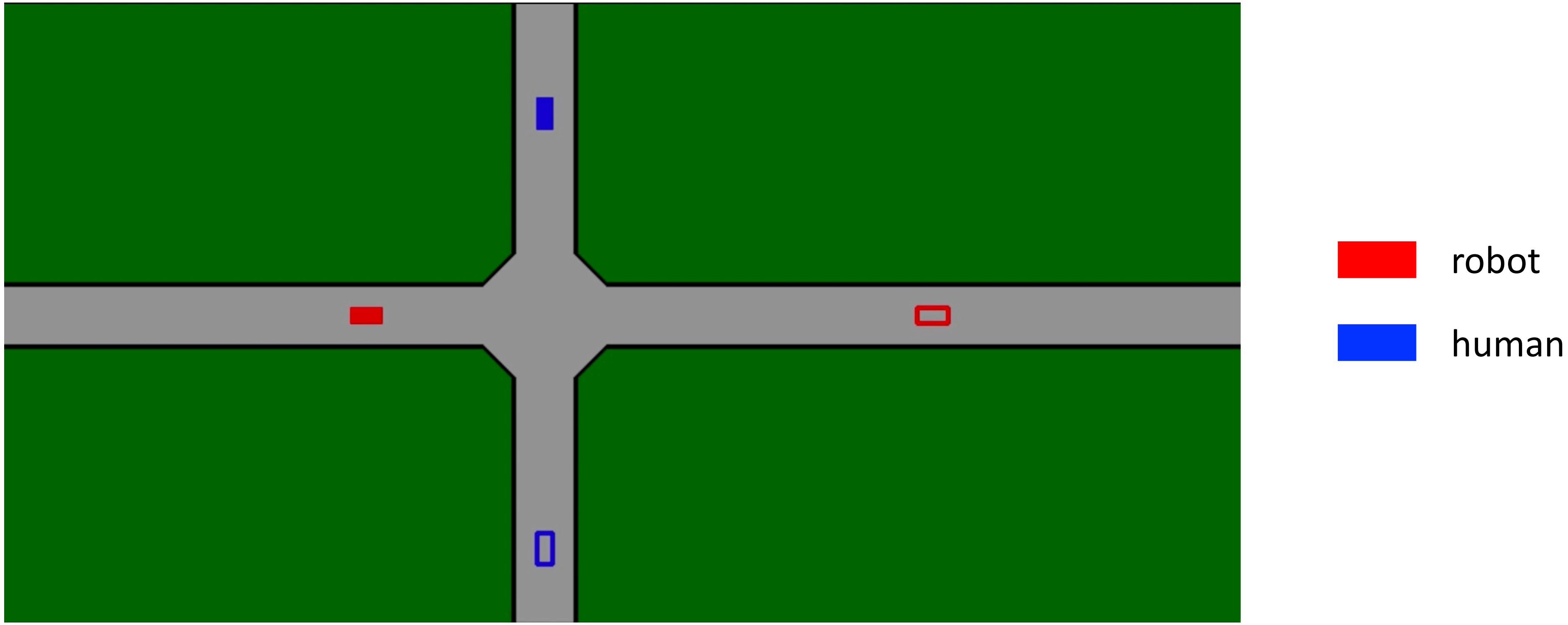
Shielded aggressive controller cuts in front of the human leveraging the fact that a responsible human driver will slightly brake

Our Approach + Simulated Humans



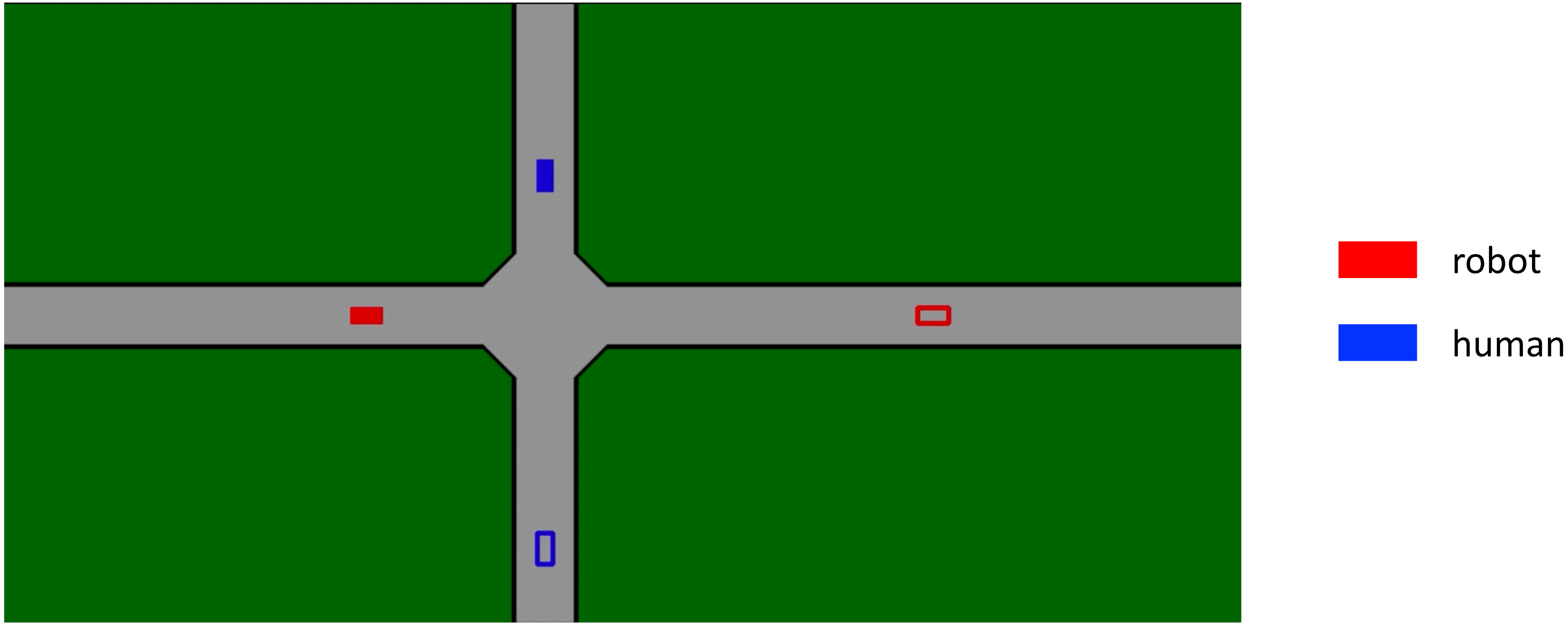
The robot triggers the shield to brake and allow the human to pass safely

Our Approach + Real Humans



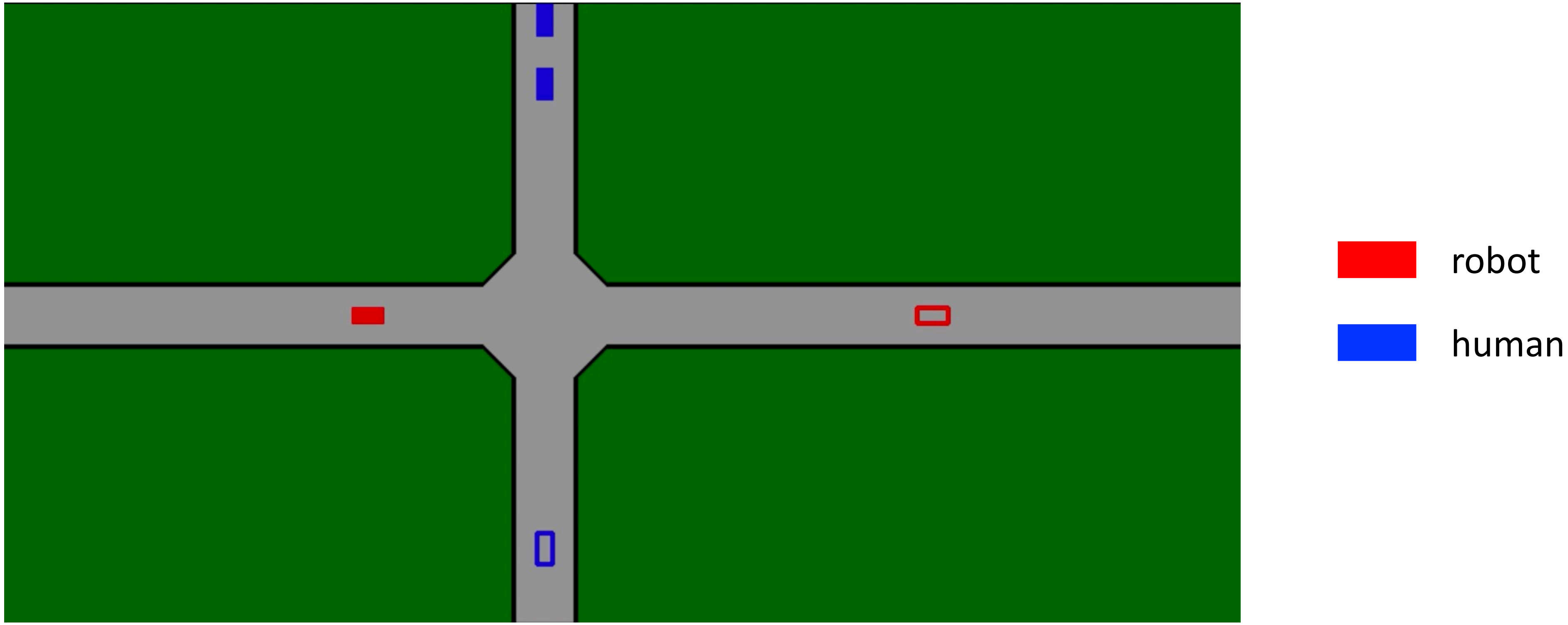
Shielded aggressive controller cuts in front of the human leveraging the fact that a responsible human driver will slightly brake

Our Approach + Real Humans



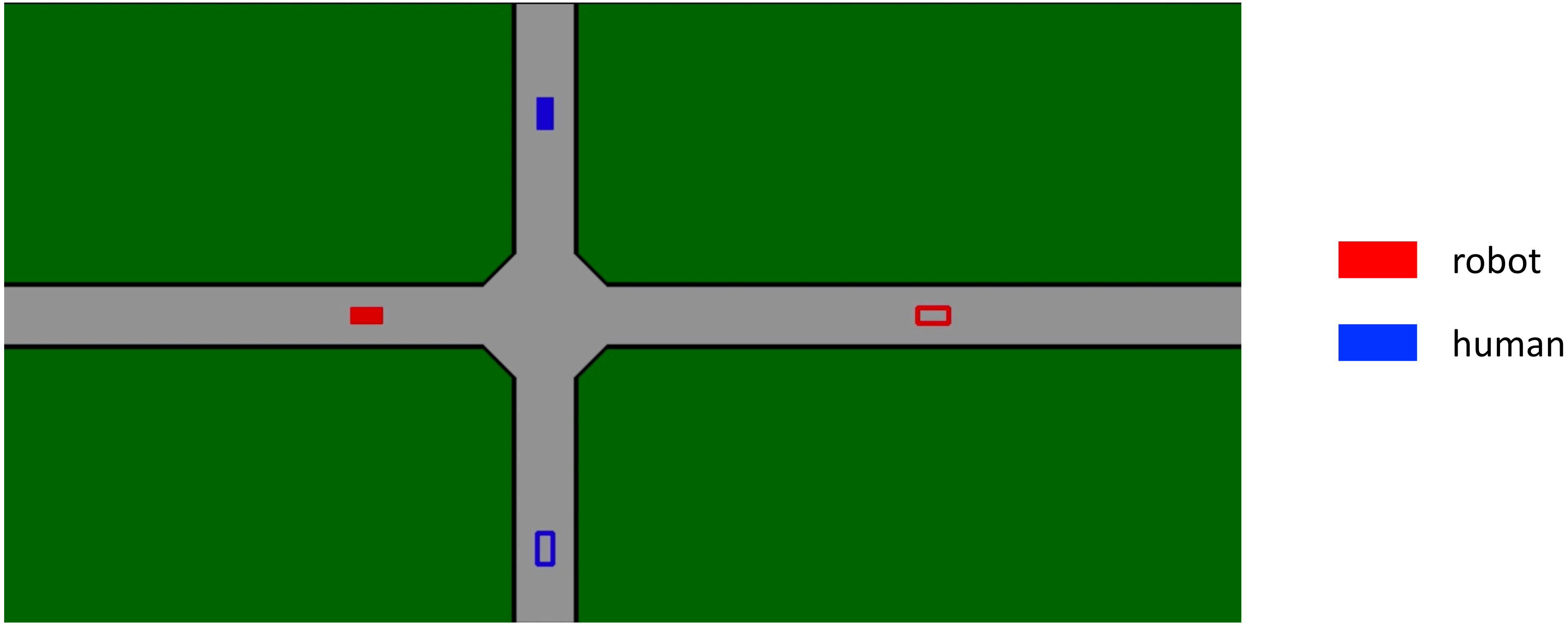
The robot triggers the shield to brake and allow the human to pass safely

MPC + Simulated Humans



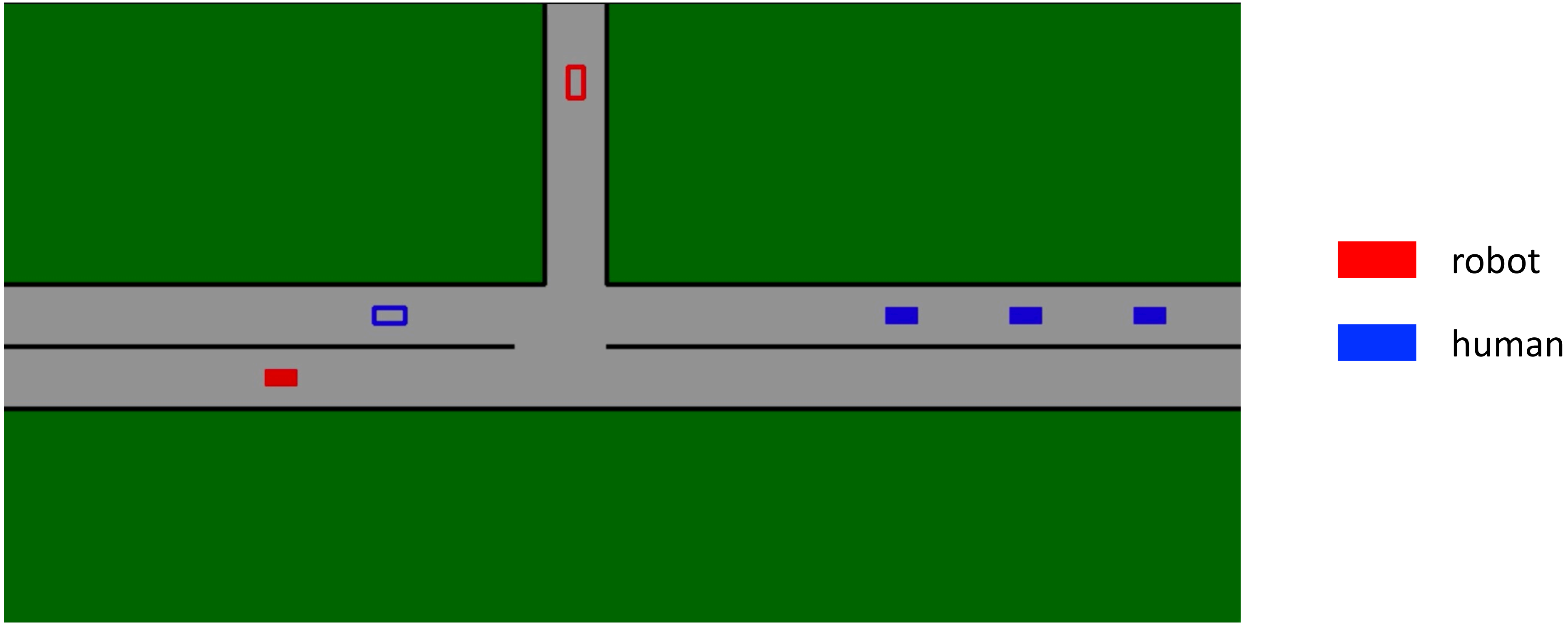
MPC control takes **longer** than the shielded aggressive control

Our Approach + Real Humans with an Accident



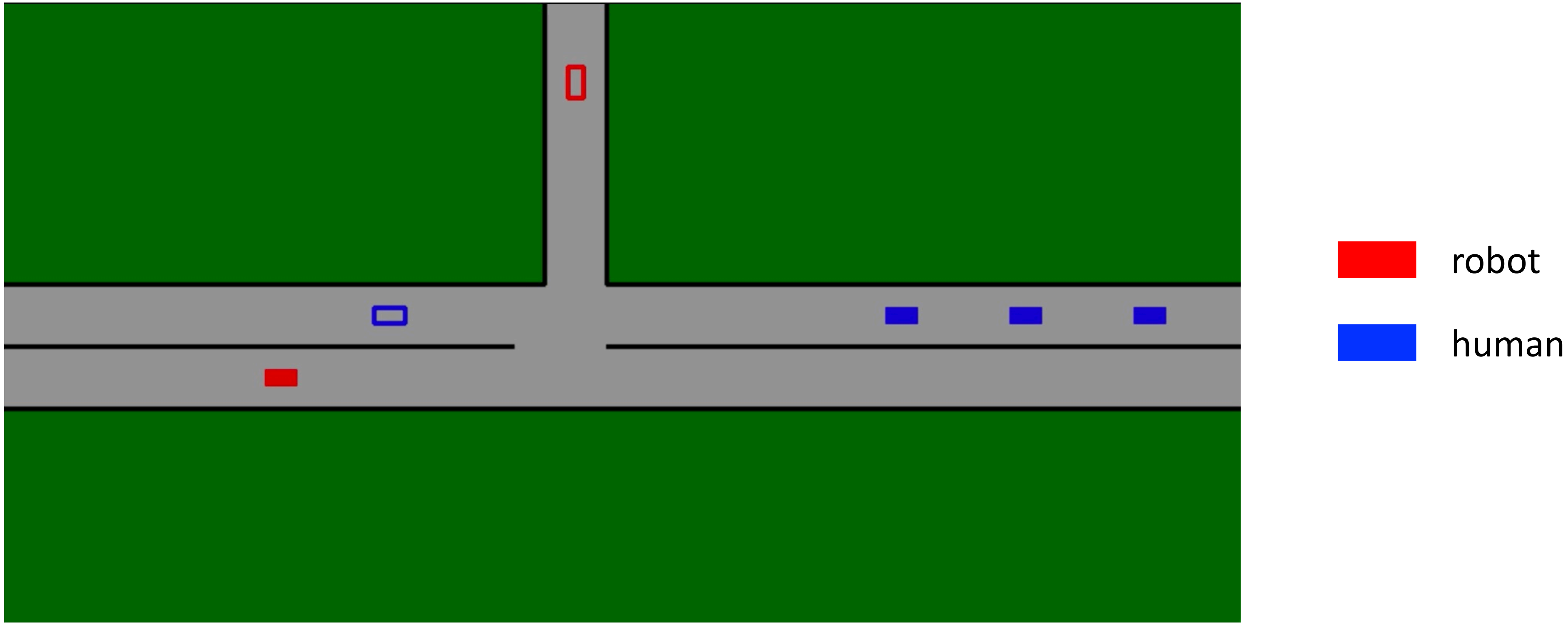
The human acted aggressively and collided with the stationary robot

No Stopping in Intersection



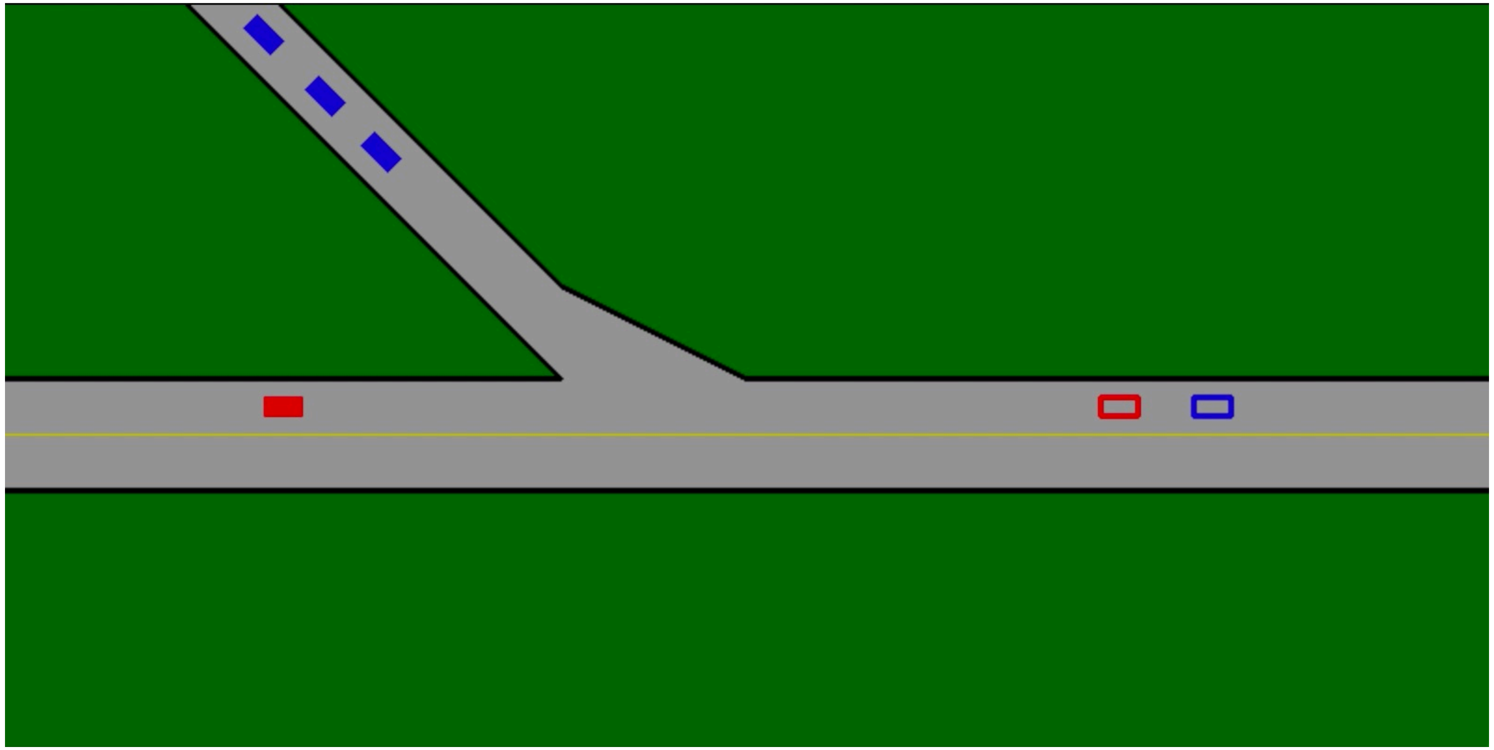
Old shielded controller **without** the no-stop-at-intersection constraint stops at the intersection, which leads to congestion

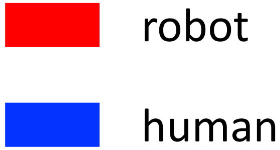
No Stopping in Intersection



New shielded controller **with** the no-stop-at-intersection constraint stops **before** the intersection

Pull-Over Backup Action





Instead of stopping in the middle of the highway, the robot pulls over to the next lane as a backup policy

Conclusion

- **Safe human-interactive control**
  - Game theoretic model of human behavior
  - Model predictive shielding + abstract interpretation to ensure safety