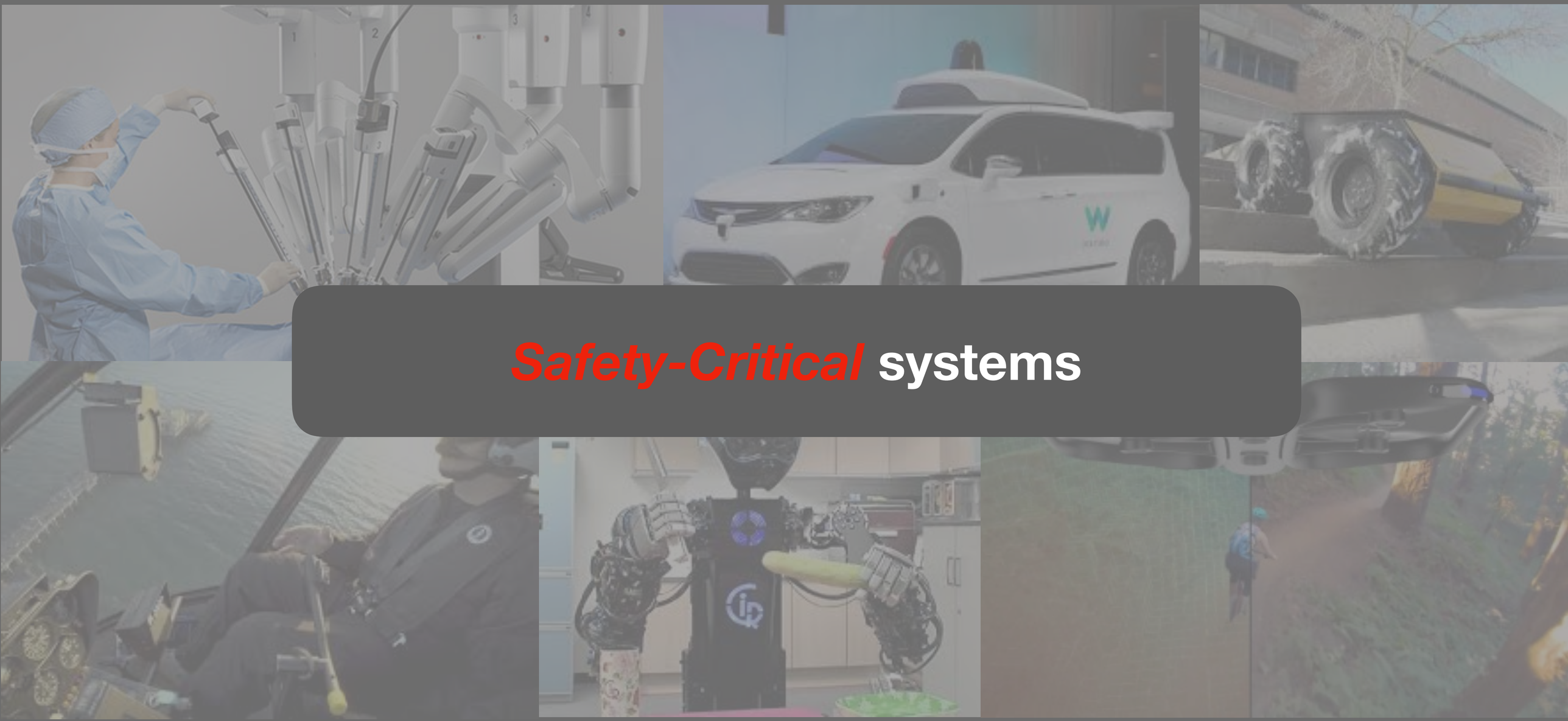
Scaling Hamilton-Jacobi Reachability Analysis for Robotics

Somil Bansal
University of California, Berkeley

Hybrid Systems Laboratory

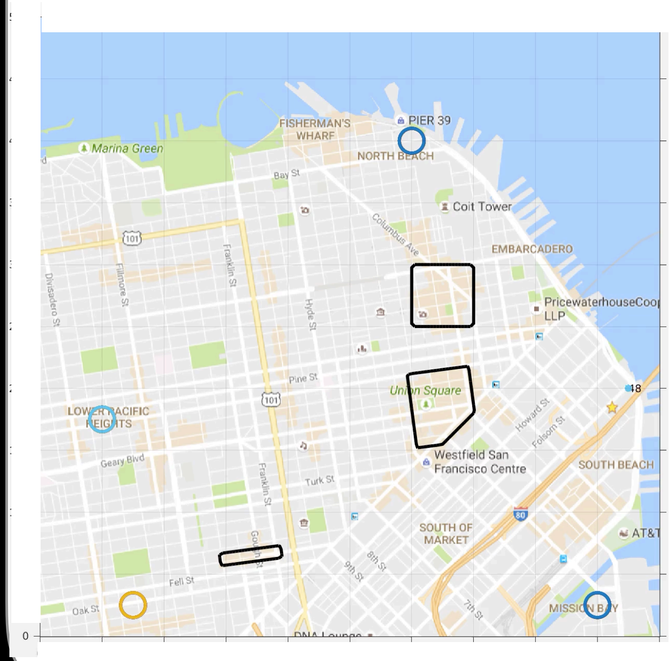
BAIR
BERKELEY ARTIFICIAL INTELLIGENCE RESEARCH



Safety-Critical systems

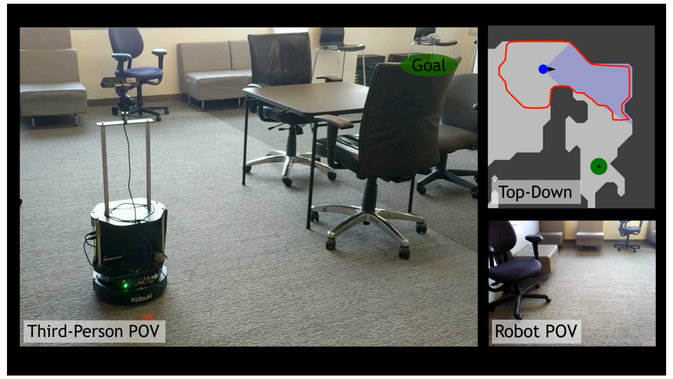
Hamilton-Jacobi Reachability for Robotics

Safe Multi-Vehicle Trajectory Planning



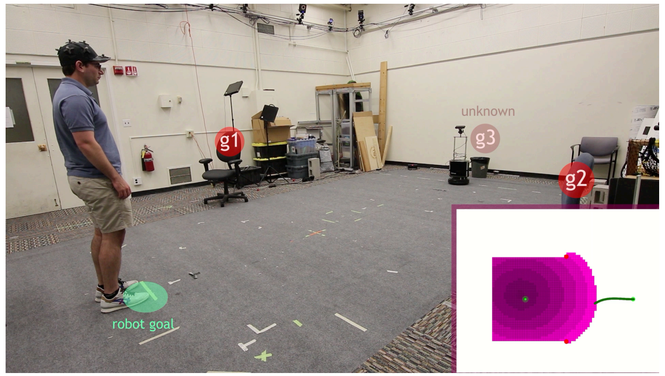
[ACC'17, TCST'17, TCST'19]

Safe Monitoring of Learning-Enabled Systems



[CDC'19]

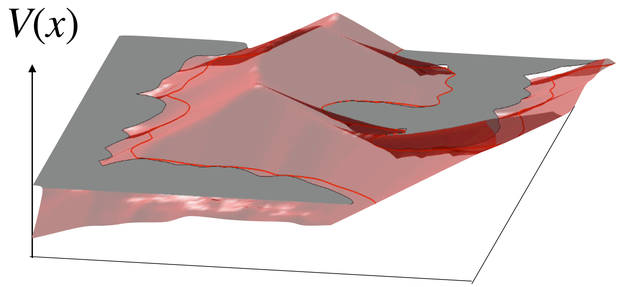
Safe Operation Around Humans



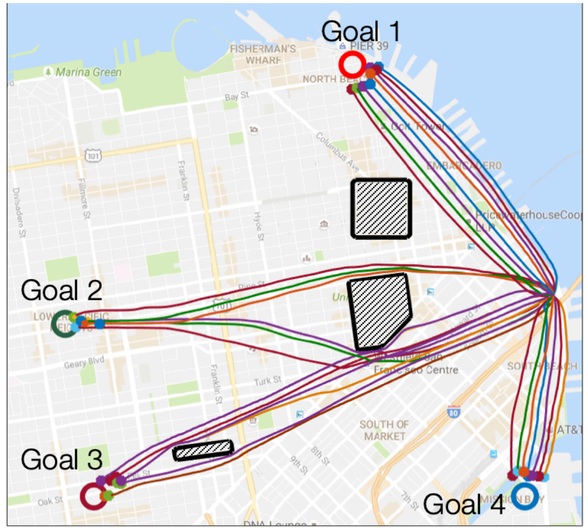
[ICRA'20]

Outline

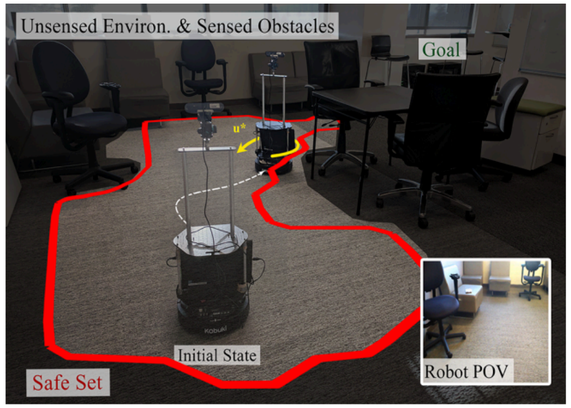
Hamilton-Jacobi Reachability Analysis

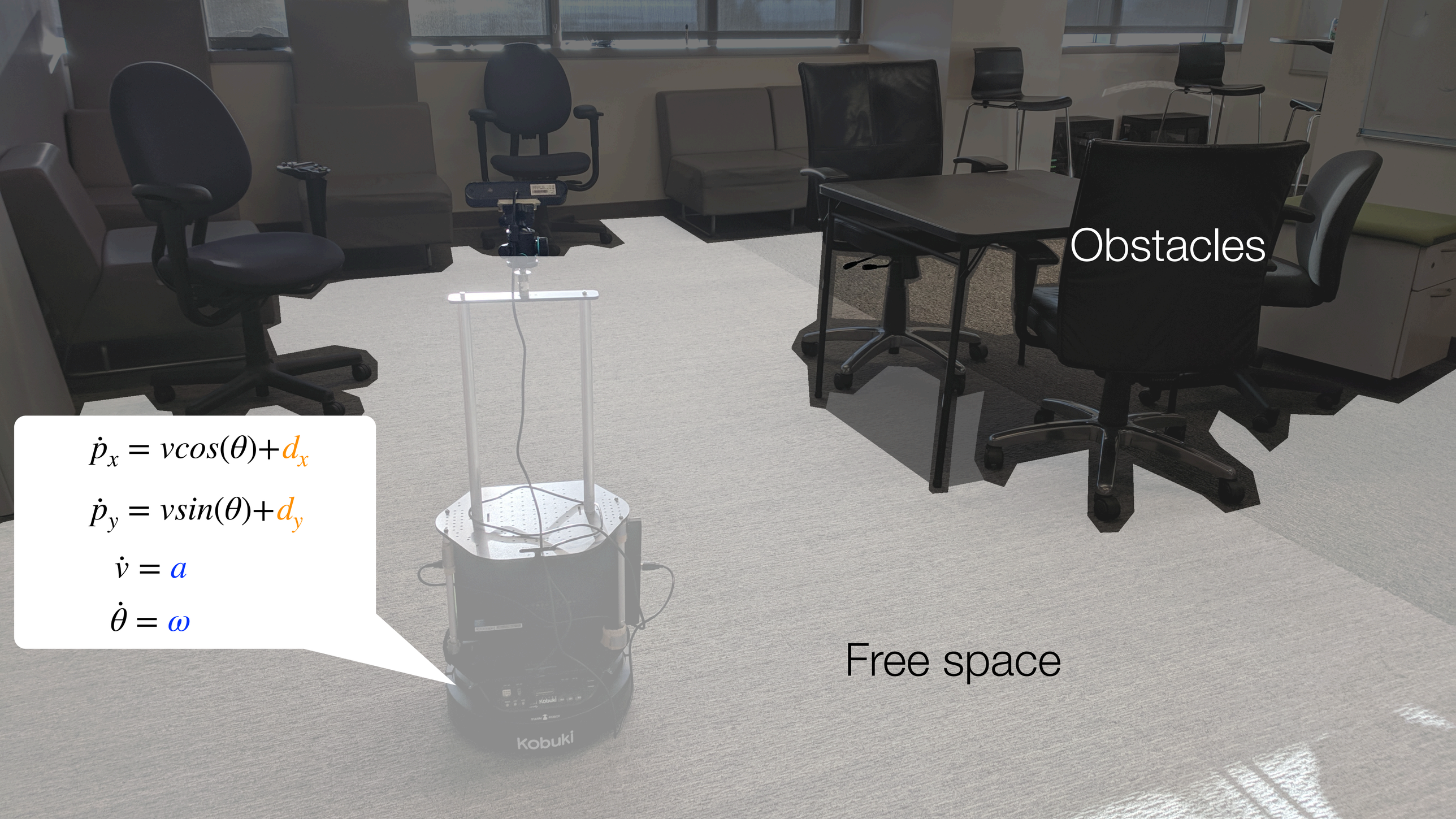


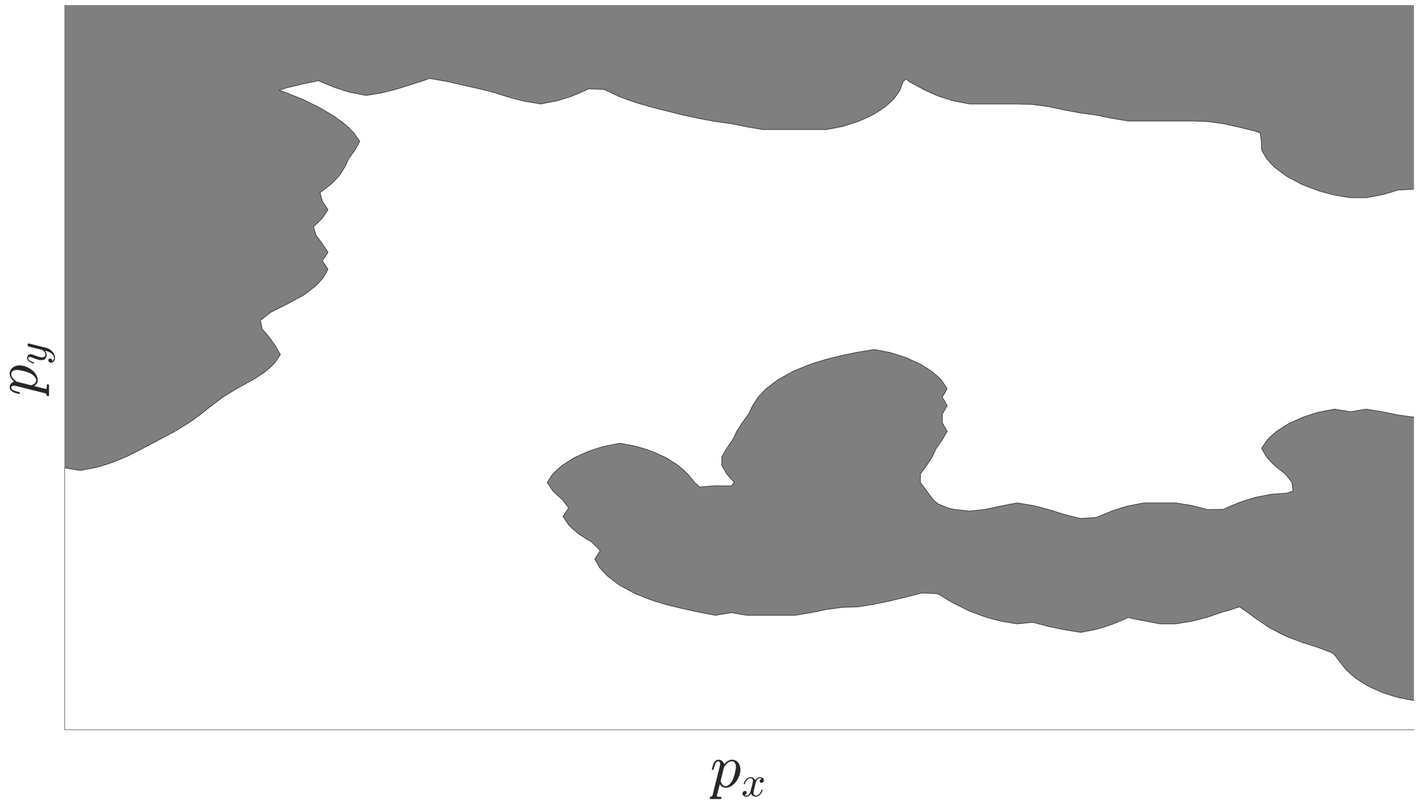
Safe Multi-Vehicle Trajectory Planning

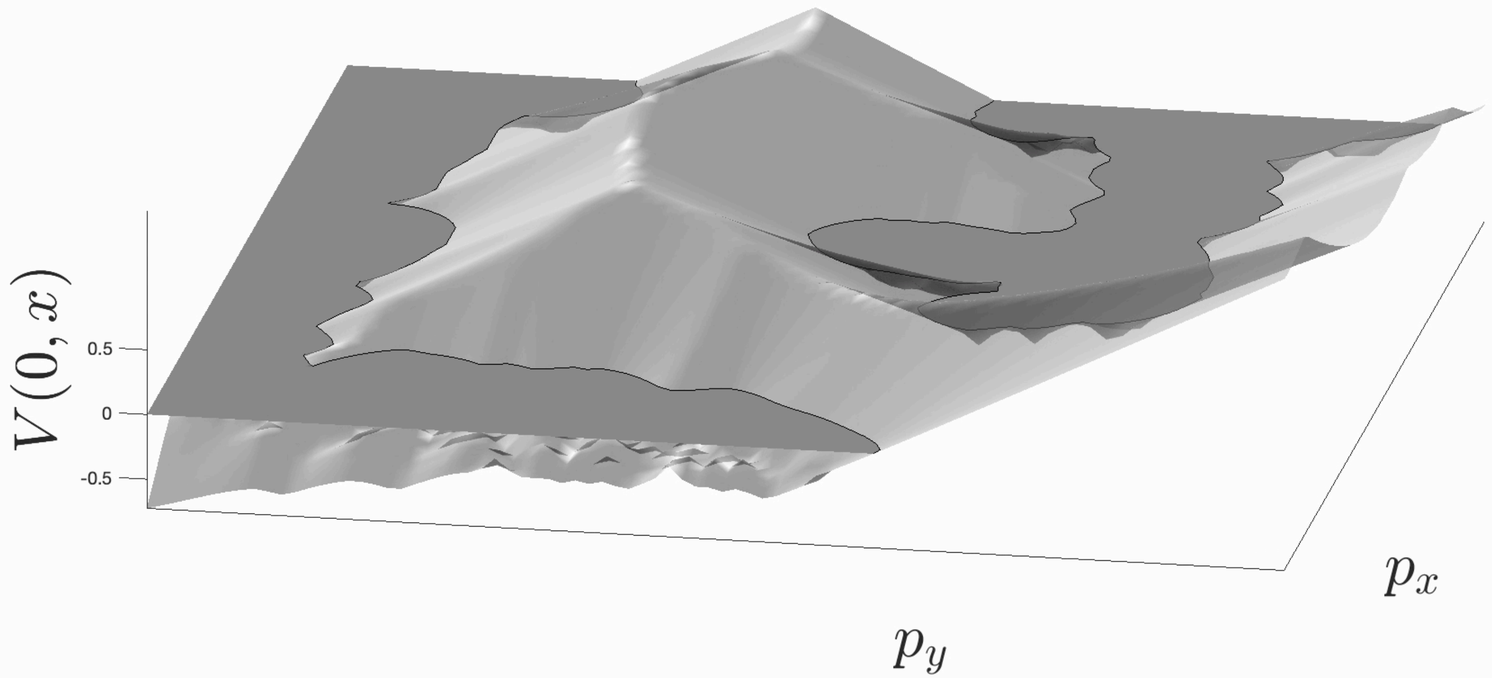


Safe Learning-based Perception Systems




Obstacles
Free space
$\dot{p}_x = v\cos(\theta) + d_x$
$\dot{p}_y = v\sin(\theta) + d_y$
$\dot{v} = a$
$\dot{\theta} = \omega$
Kobuki


p_y
p_x

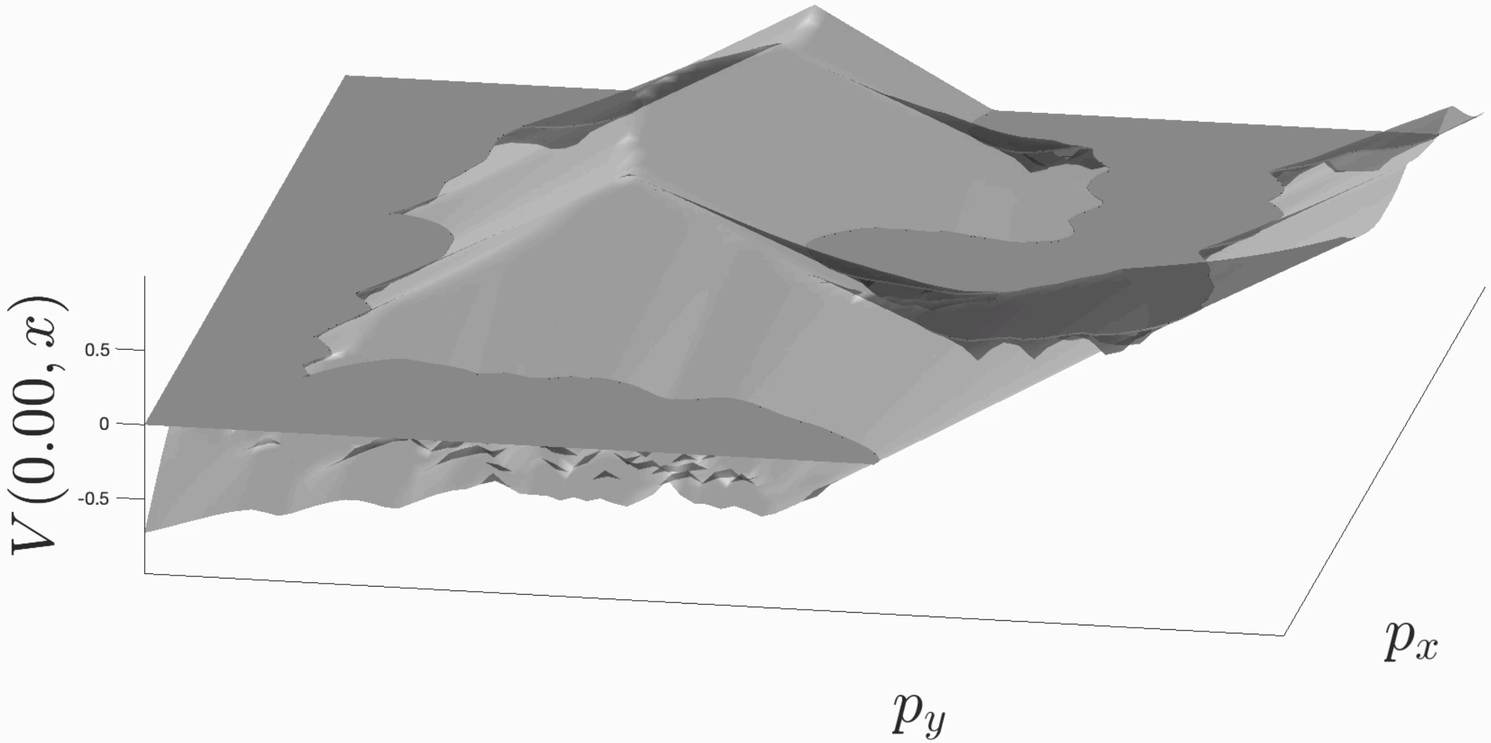


Unsafe States

$$\mathscr{L} = \{x : l(x) \leq 0\}$$

Value Function

$$V(T,x) = \min_{\mathbf{d}[\cdot]} \max_{\mathbf{u}(\cdot)} \min_{t \in [0,T]} l(\zeta(t; x, \mathbf{u}, \mathbf{d}[\mathbf{u}]))$$



Value Function

$$V(T,x) = \min_{\mathbf{d}[\cdot]} \max_{\mathbf{u}(\cdot)} \min_{t\in[0,T]} l(\zeta(t;x,\mathbf{u},\mathbf{d}[\mathbf{u}]))$$

Dynamic Programming

$$min\left\{\frac{\partial V}{\partial t} + H(x,\nabla V), \quad l(x) - V(t,x)\right\} = 0$$

$$V(0,\ x) = l(x)$$

Backward Reachable Set

$$\mathscr{V} = \{x : V(T,x) \leq 0\}$$

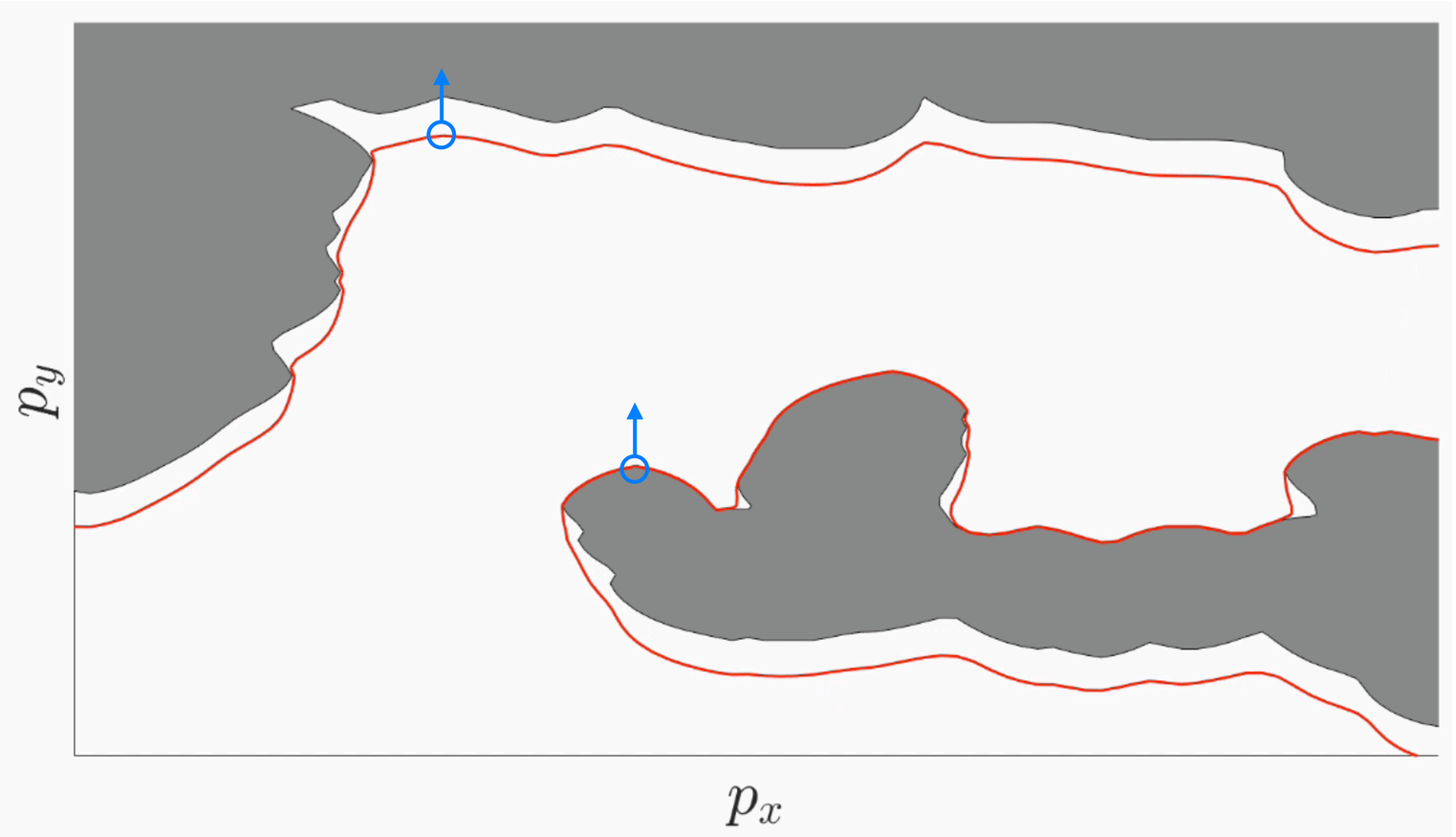
Optimal Control

$$\mathbf{u}^* = \arg\max_{\mathbf{u}} \min_{\mathbf{d}} \nabla V(t,x) \cdot f(x,\mathbf{u},\mathbf{d})$$

Hamiltonian

$$H(x,\nabla V) = \max_{\mathbf{u}} \min_{\mathbf{d}} \nabla V(t,x) \cdot f(x,\mathbf{u},\mathbf{d})$$

Related work: [Mitchell 2005, Lygeros 2005, Kress-Gazit 2009, Oishi 2010, Vasudevan 2011, Ames, Grizzle and Tabuada 2013, Egerstedt 2015…]



Dynamics

$$\dot{p}_x = v\cos(\theta) + d_x$$

$$\dot{p}_y = v\sin(\theta) + d_y$$

$$\dot{v} = a$$

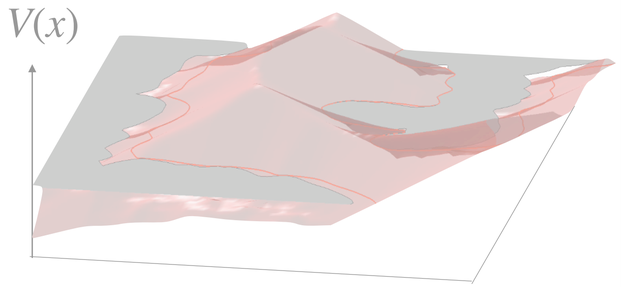
$$\dot{\theta} = \omega$$

Slice : $\theta = \frac{\pi}{2}, \quad v > 0$

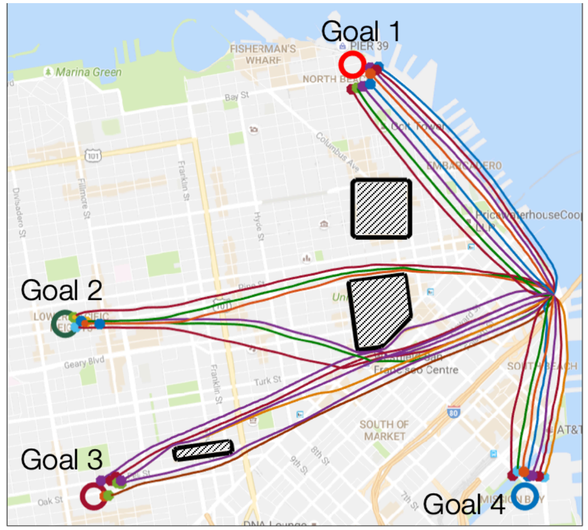
Related work: [Mitchell 2005, Lygeros 2005, Kress-Gazit 2009, Oishi 2010, Vasudevan 2011, Ames, Grizzle and Tabuada 2013, Egerstedt 2015…]

Outline

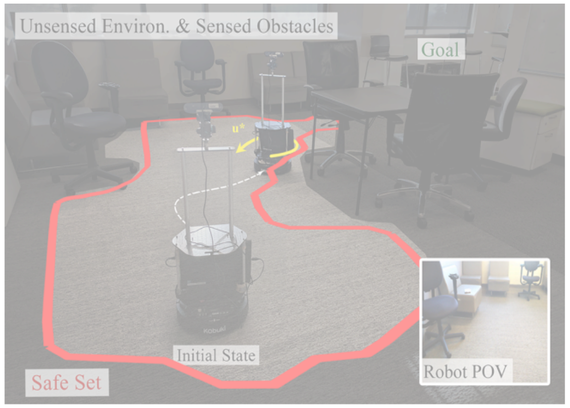
Hamilton-Jacobi Reachability Analysis



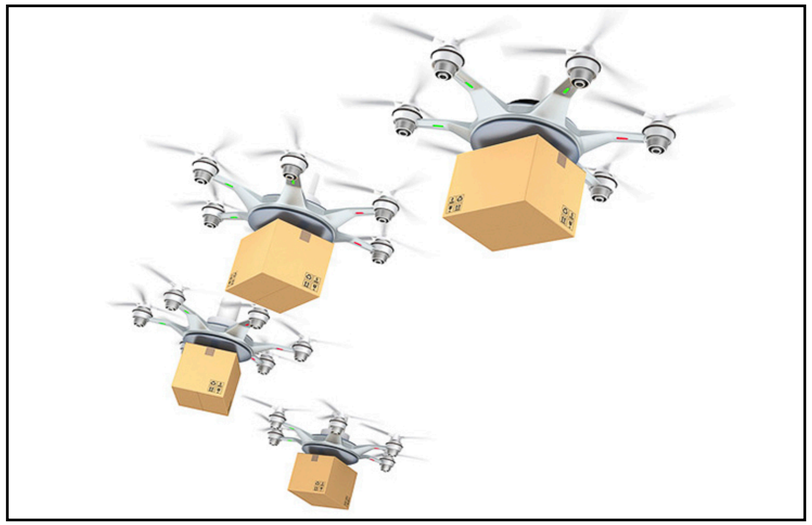
Safe Multi-Vehicle Trajectory Planning



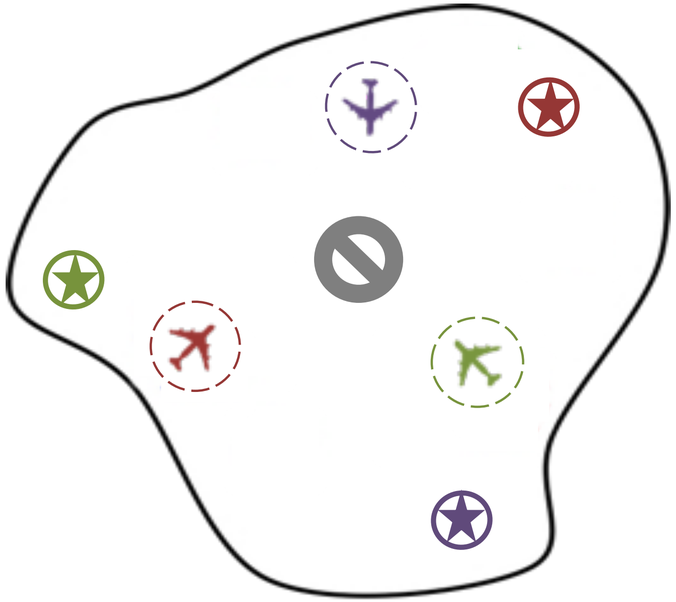
Safe Learning-based Perception Systems

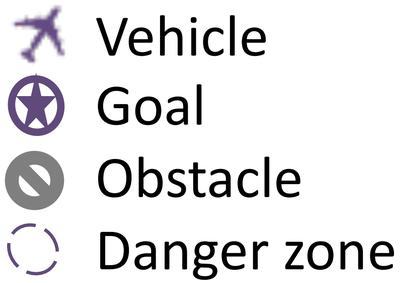


Safe Multi-Vehicle Trajectory Planning

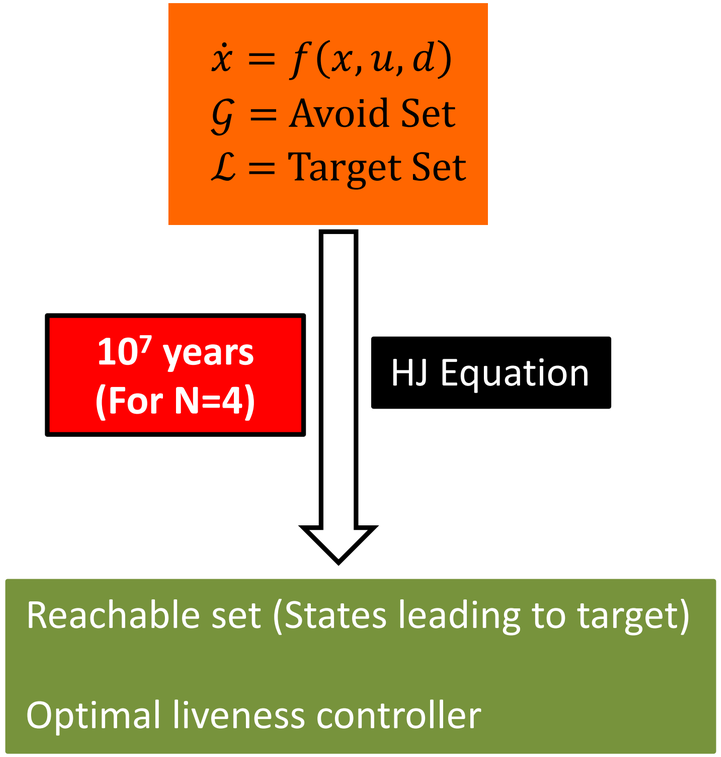


How a group of robots can reach their ***goals,*** ***without colliding*** with obstacles or other vehicles despite the presence of ***external disturbances***?

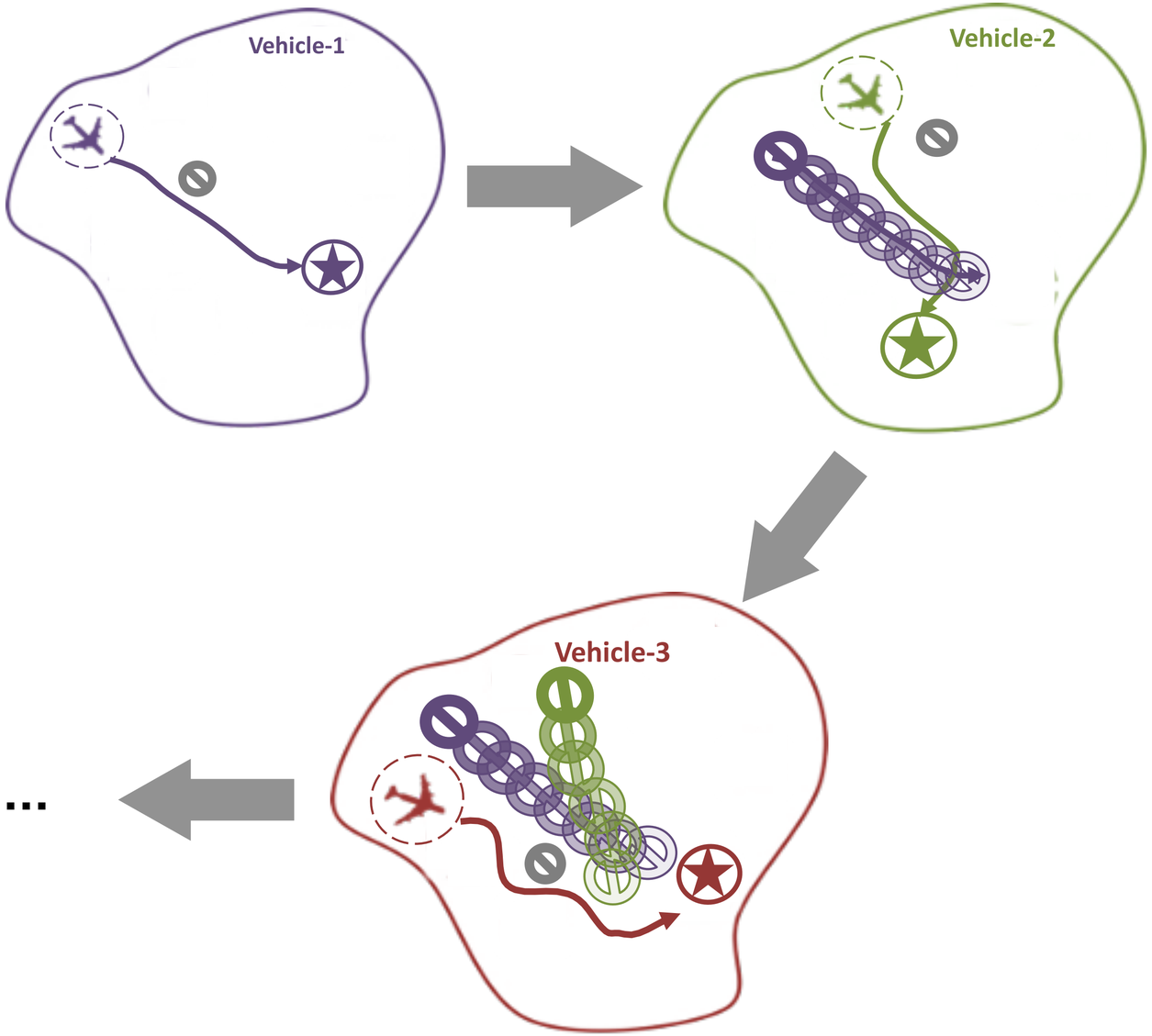


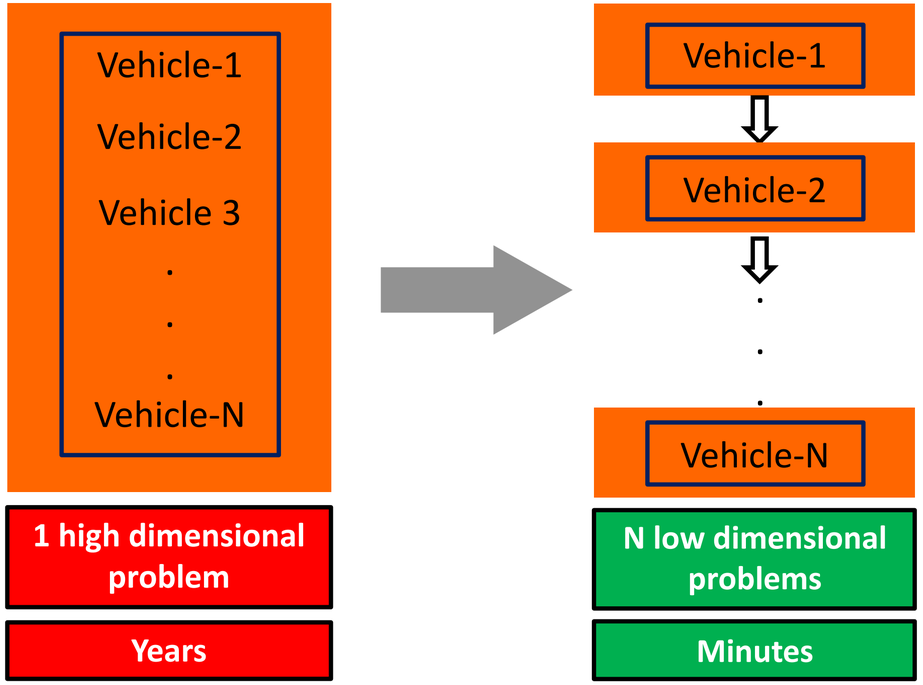


But….Curse of Dimensionality!



Sequential Trajectory Planning





Large Scale Multi-Vehicle Trajectory Planning

Simulation Setup

50 UAVs originating from the Blue star

Going to one of the 4 destinations (circles)

UAV Model

$$\dot{s} = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{matrix} v\cos(\theta) + d_x \\ v\sin(\theta) + d_y \end{matrix}$$

$$0 \le v \le 25\ m/s$$
$$|\omega| \le 2\ rad/s$$

Algorithmic developments alone are not enough, computational advancements are also required!

200 seconds
[Level Set Toolbox]

~3 hours
[for 50 vehicles]

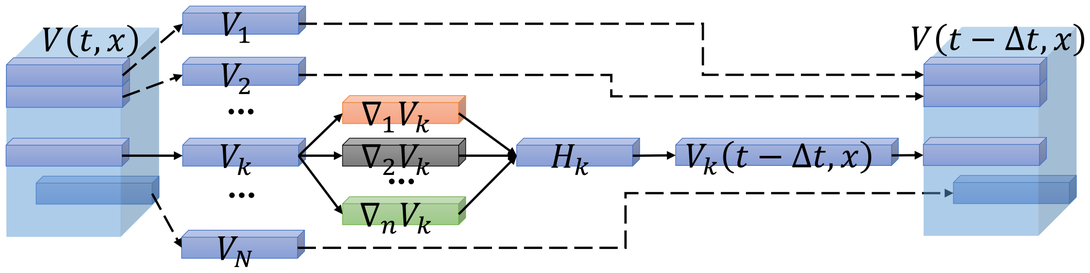
~half a day!
[for 200 vehicles]

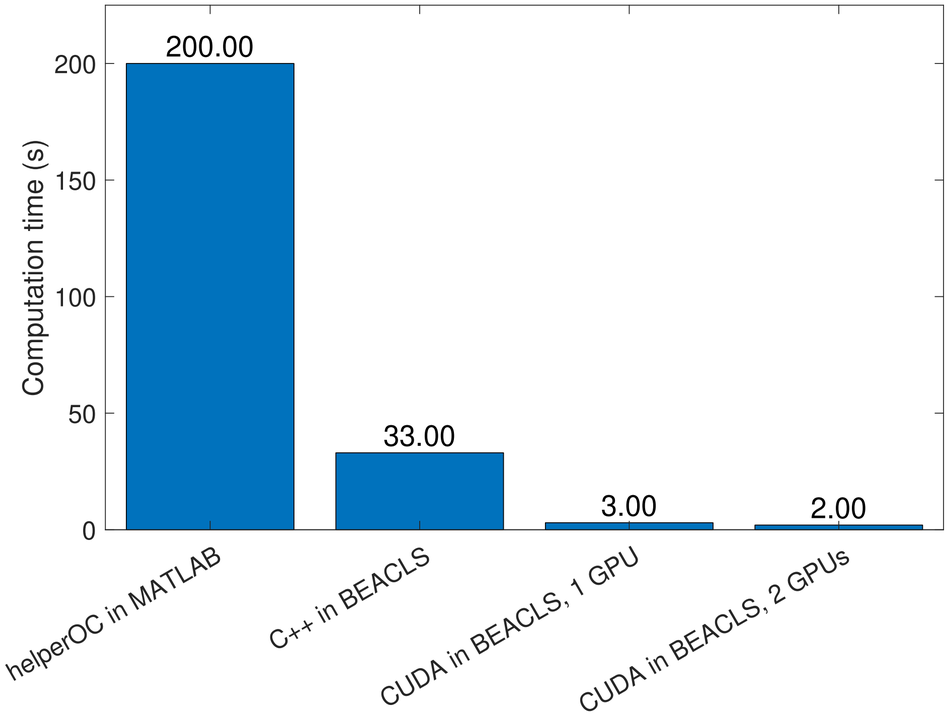
BEACLS: Berkeley Efficient API in C++ for Level Set Methods

One-one correspondance

MATLAB Implementation ⟷ **BEACLS**

Parallel Computation of Value Function





~3 hours
[for 50 vehicles]

~2 minutes
[for 50 vehicles]

~half a day!
[for 200 vehicles]

~8 minutes
[for 200 vehicles]

BEACLS Website: *https://github.com/HJReachability/beacls*

Wind Speed

High

Low

High

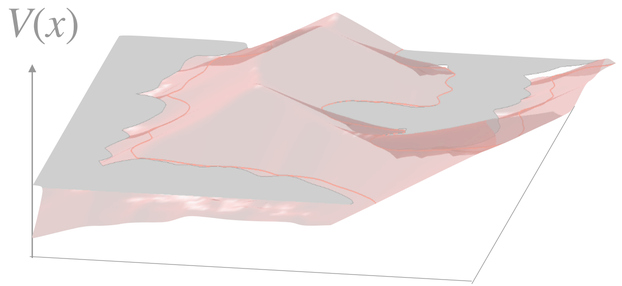
Medium

Low

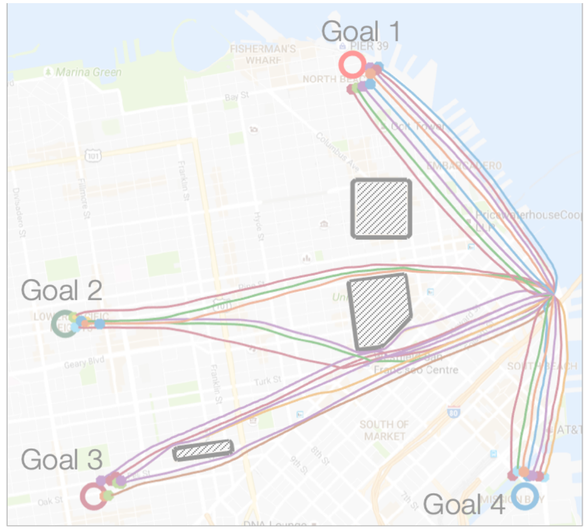
Vehicle Density

Outline

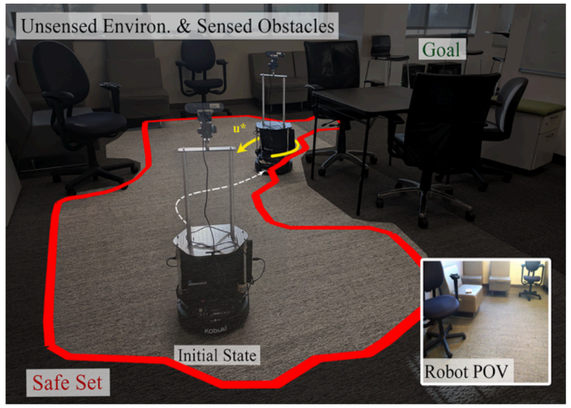
Hamilton-Jacobi Reachability Analysis



Safe Multi-Vehicle Trajectory Planning

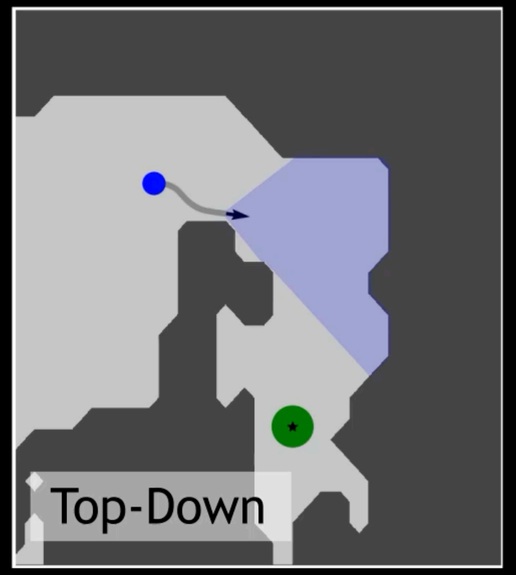


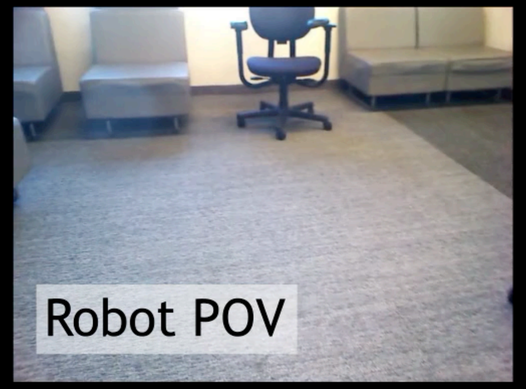
Safe Learning-based Perception Systems



Safety Critical **systems need to update safety guarantees in real-time.**

Learning-based modules can make errors
that lead to safety violations

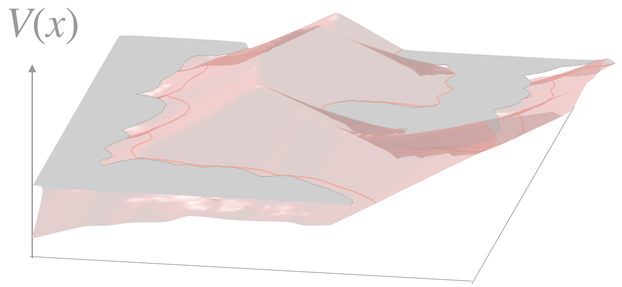




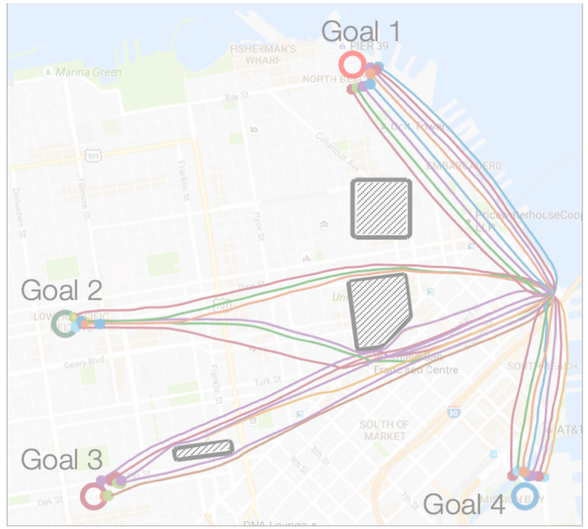
Learning-based modules can make errors
that lead to safety violations

Outline

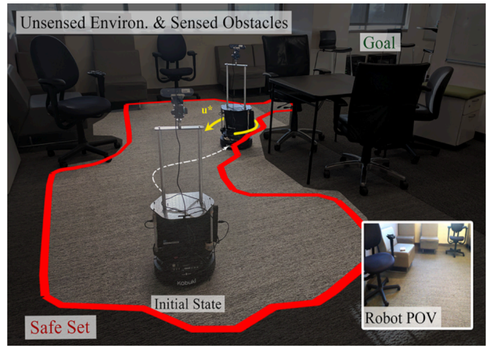
Hamilton-Jacobi Reachability Analysis



Safe Multi-Vehicle Trajectory Planning



Safe Learning-based Perception Systems



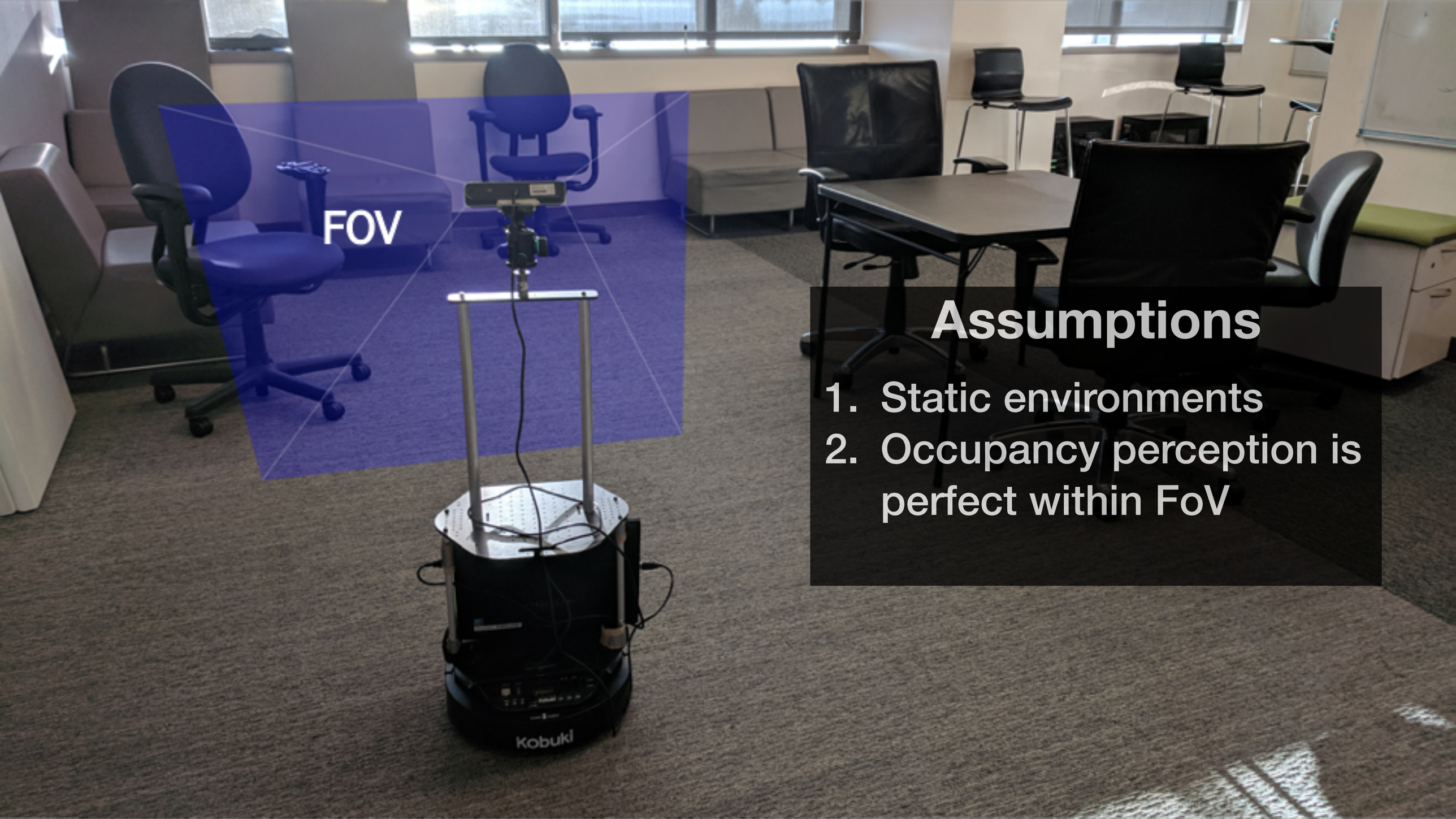
Approach

Online Safety Updates

Hardware experiments

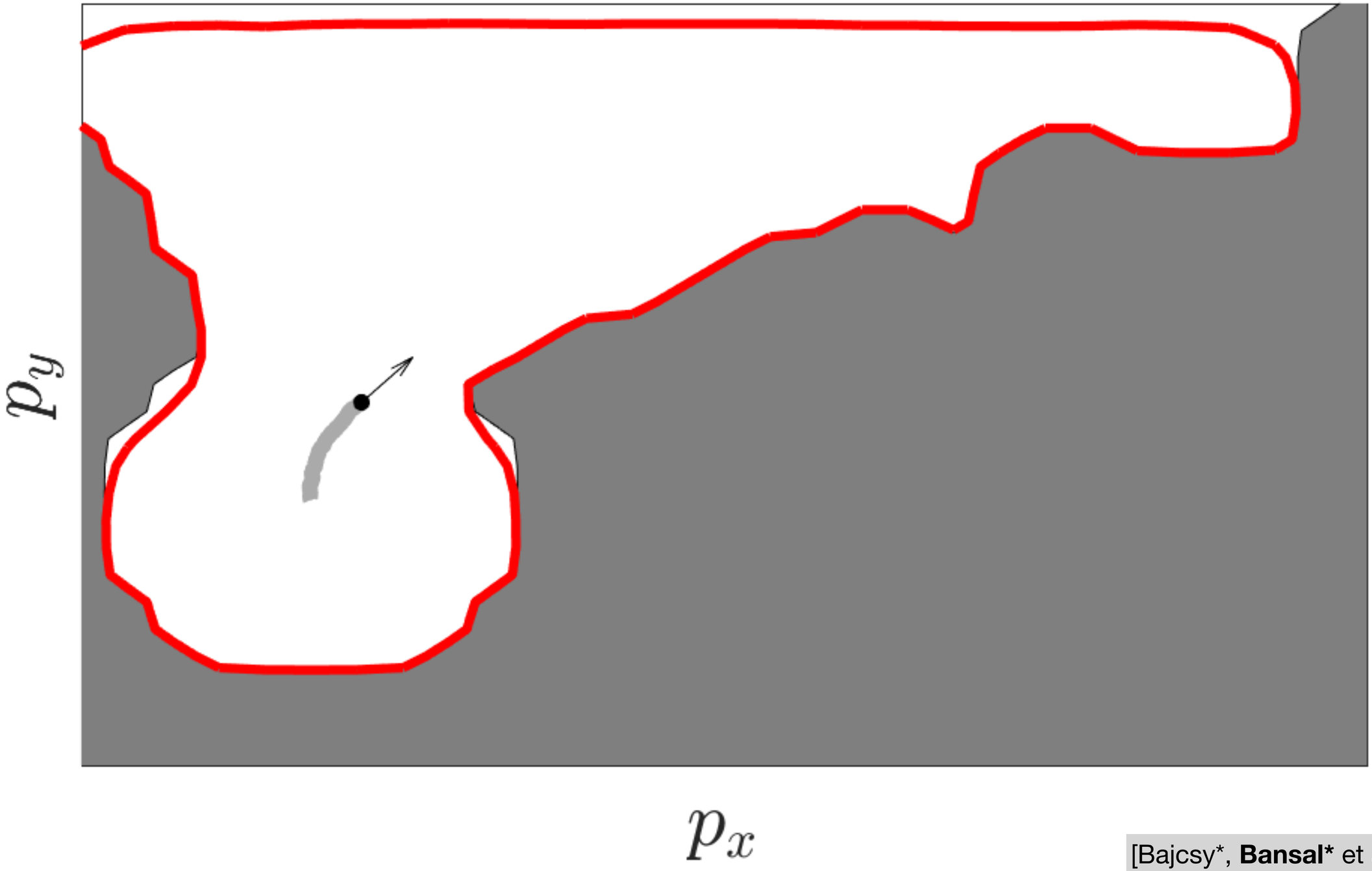
Key Idea:

Monitor the output of learning module and provide a *corrective safe* action whenever necessary.

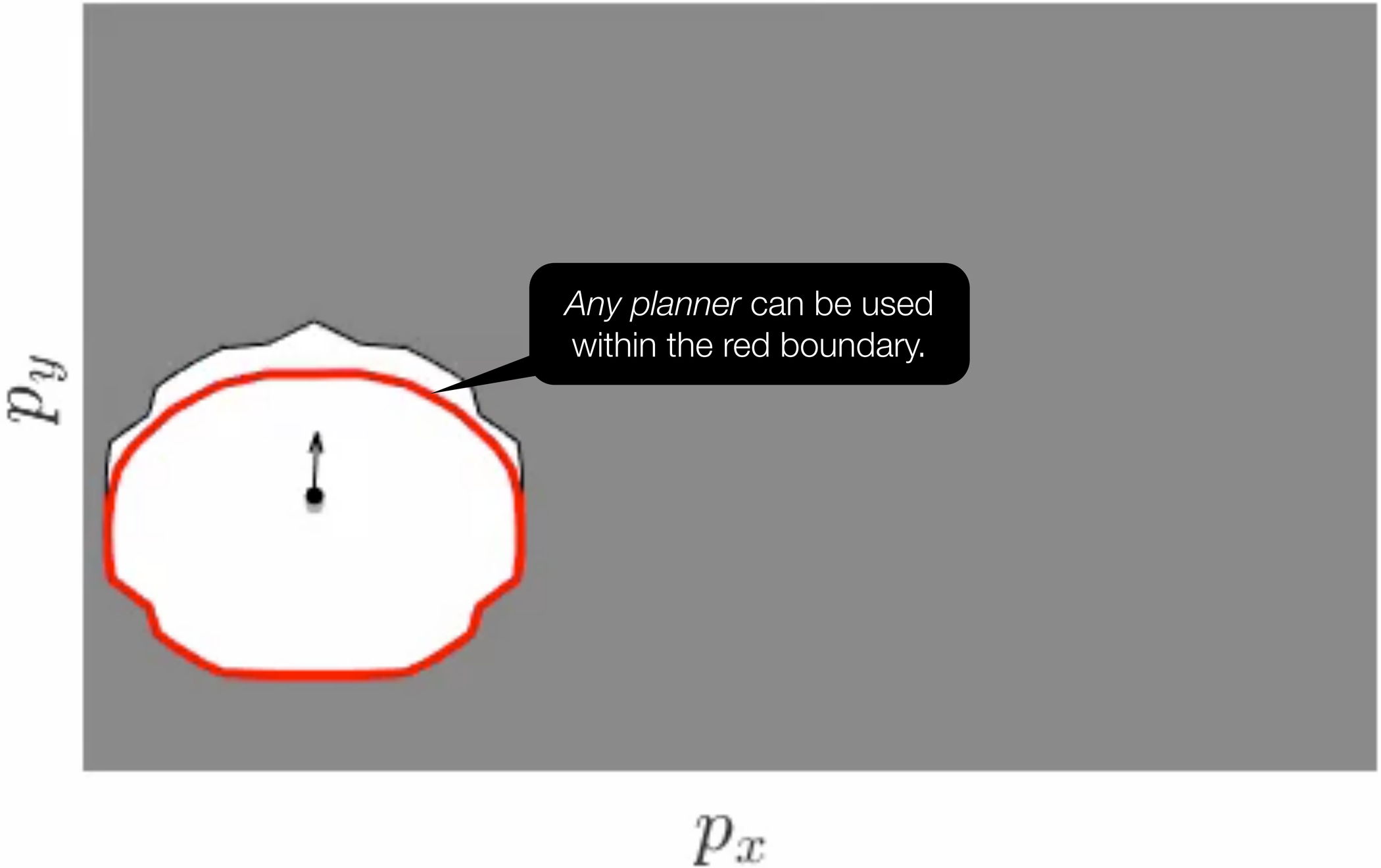


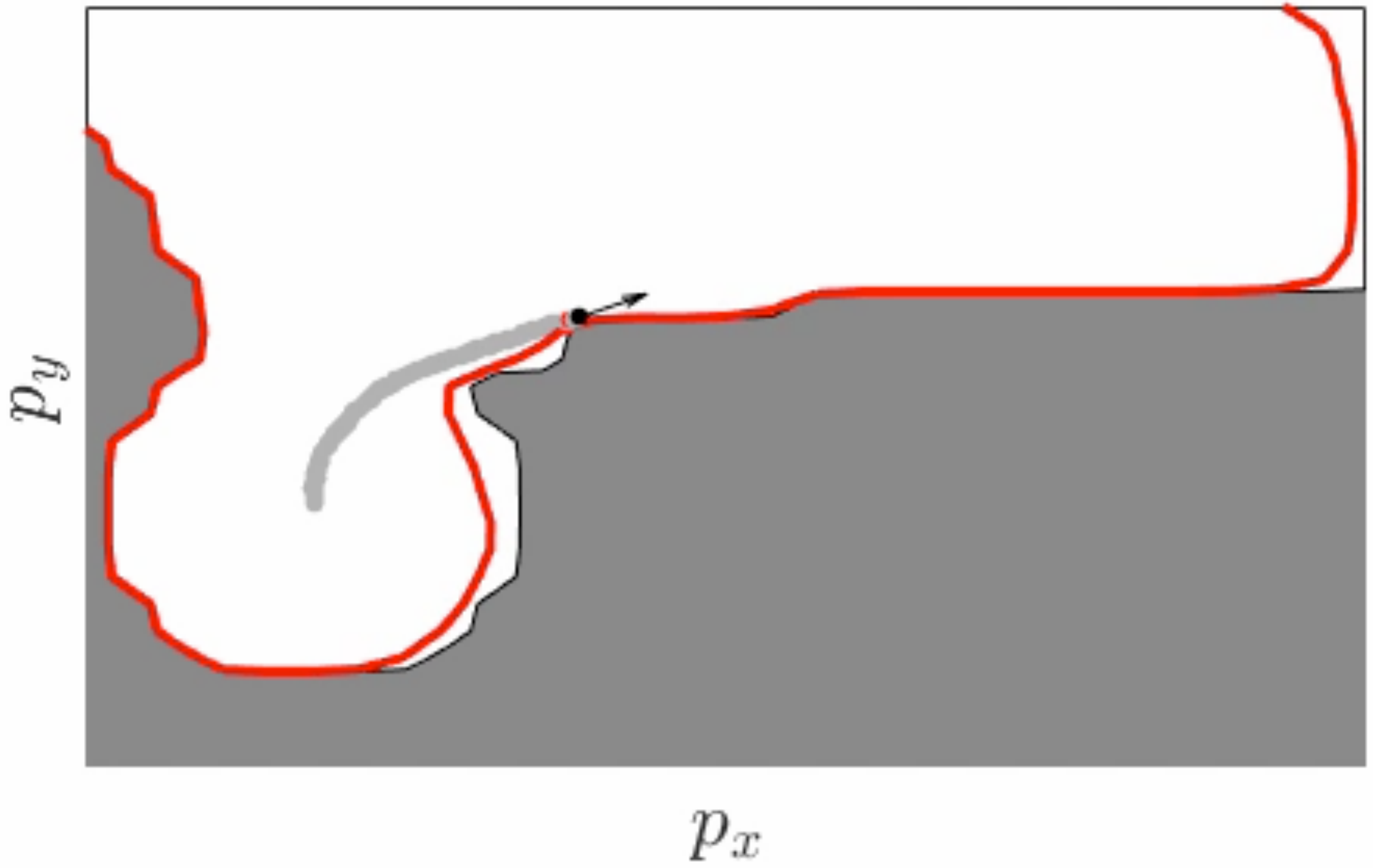
Assumptions

1. Static environments
2. Occupancy perception is perfect within FoV



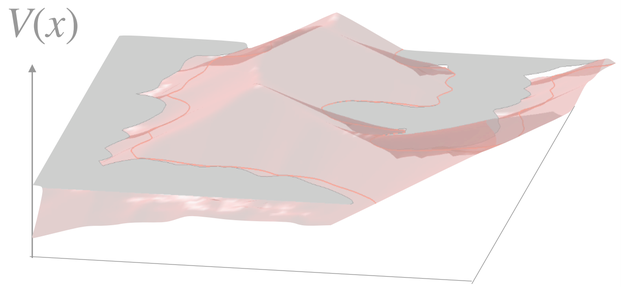
[Bajcsy*, **Bansal*** et al, CDC 2019]


Any planner can be used within the red boundary.
p_y
p_x

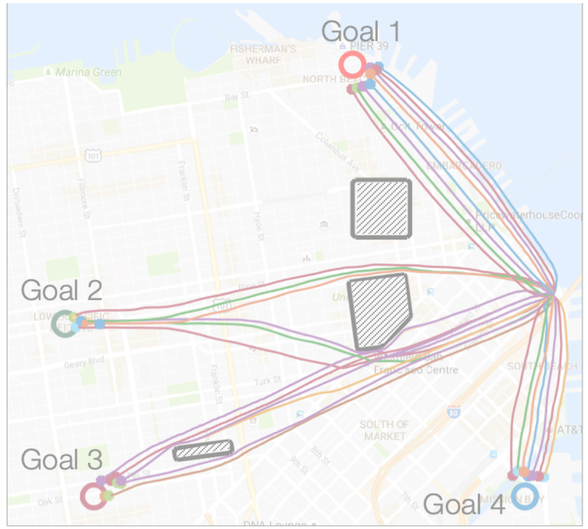

p_y
p_x

Outline

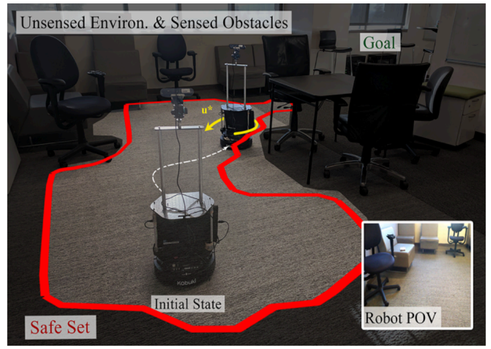
Hamilton-Jacobi Reachability Analysis



Safe Multi-Vehicle Trajectory Planning



Safe Learning-based Perception Systems



Approach

Online Safety Updates

Hardware experiments

Key Idea:

Only ***locally*** update the safe set based on the new information about the environment.

Local Update of the BRS

$V_{old} \leftarrow V_t(0, Q)$

$Q \leftarrow new\ free\ states\ and\ neighbors$

while Q *is not empty* do:

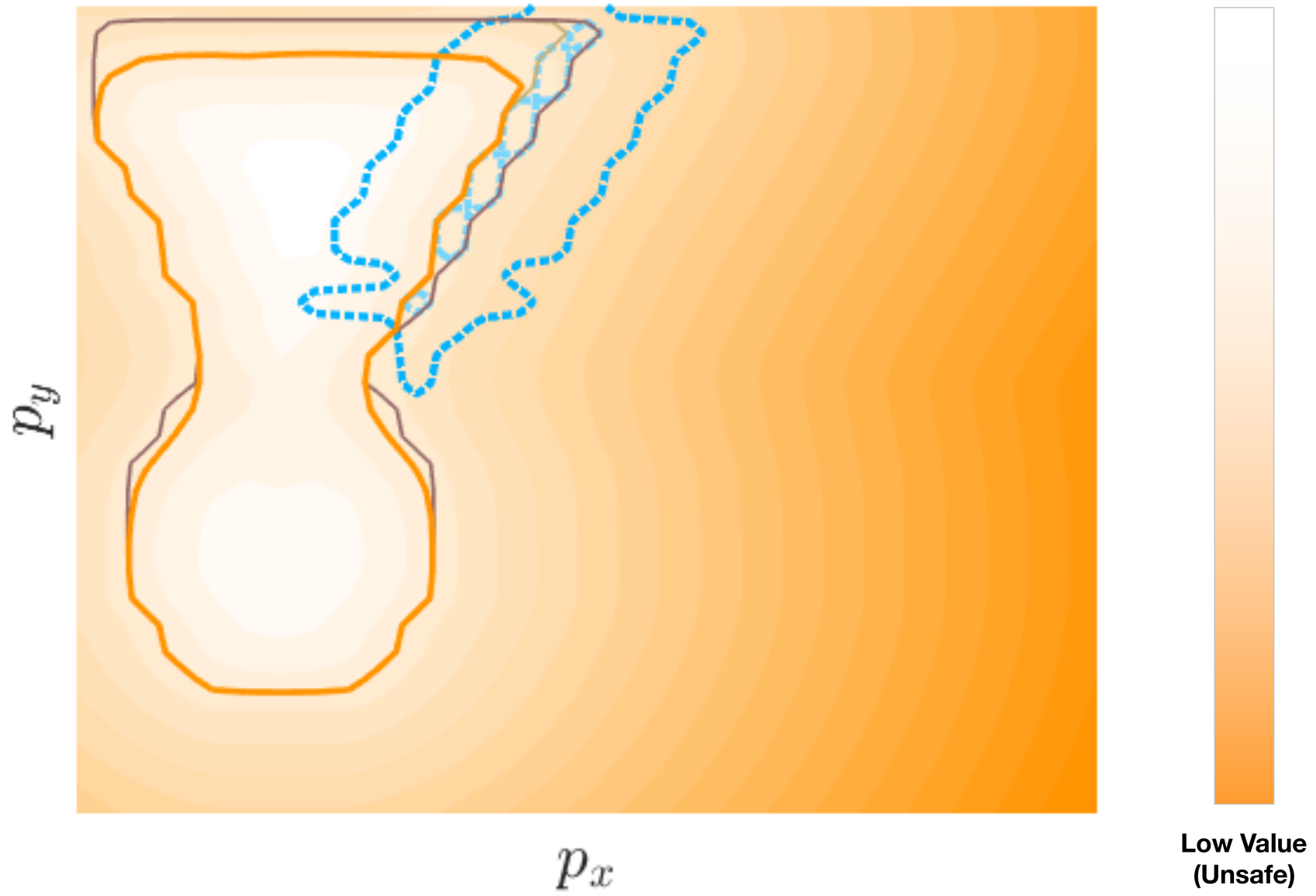
$V_{update} \leftarrow update\ V_{old}\ for\ \Delta T$

$\Delta V = ||V_{update} - V_{old}||$

$Q \leftarrow remove\ states\ with\ \Delta V = 0$

$Q \leftarrow add\ neighbors$

$V_{old} \leftarrow V_{update}$

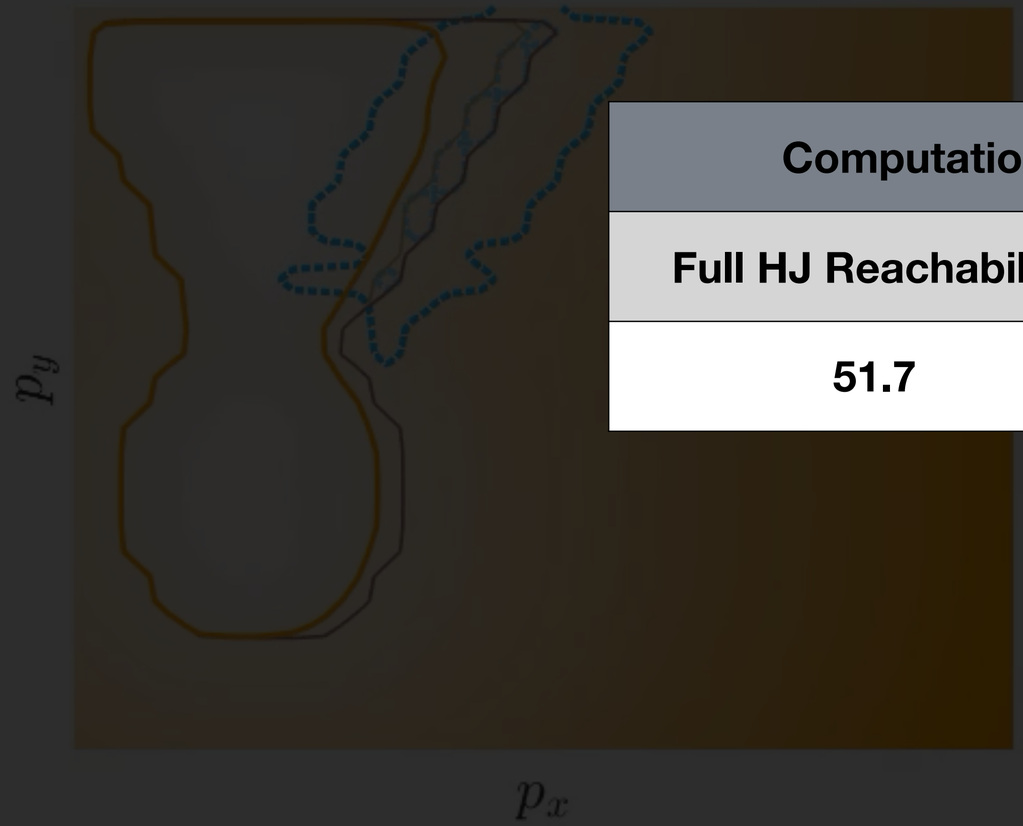


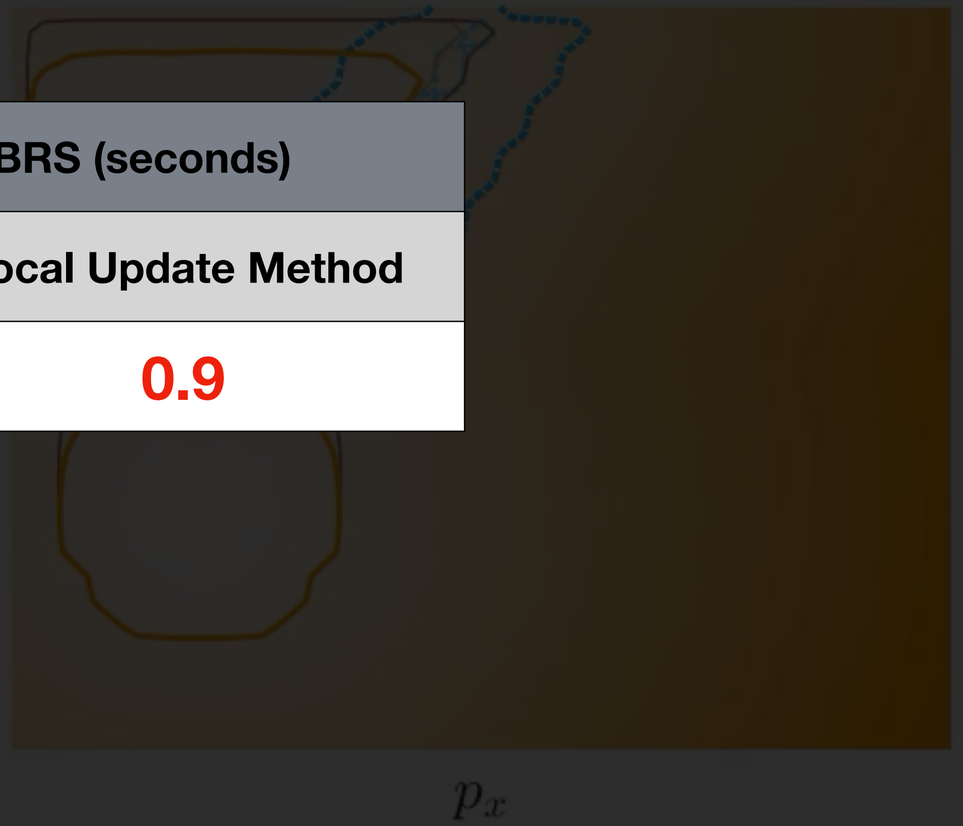
Local Update of the BRS

$Slice\ \theta = 0$

$Slice\ \theta = \frac{\pi}{2}$

Guaranteed under-approximation!



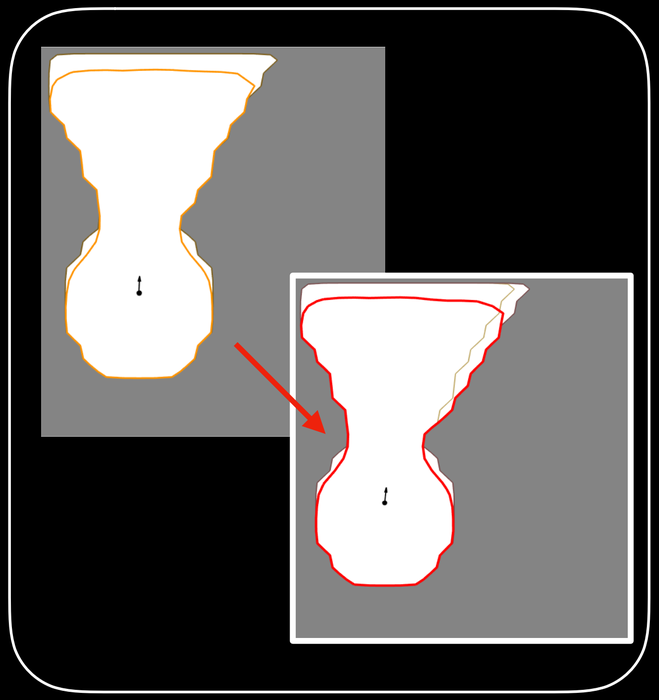


Computation time for BRS (seconds)	
Full HJ Reachability	**Local Update Method**
51.7	**0.9**

[Bajcsy*, **Bansal*** et al, CDC 2019; Herbert, **Bansal** et al, CDC 2019]

Local Update Beyond Obstacles…

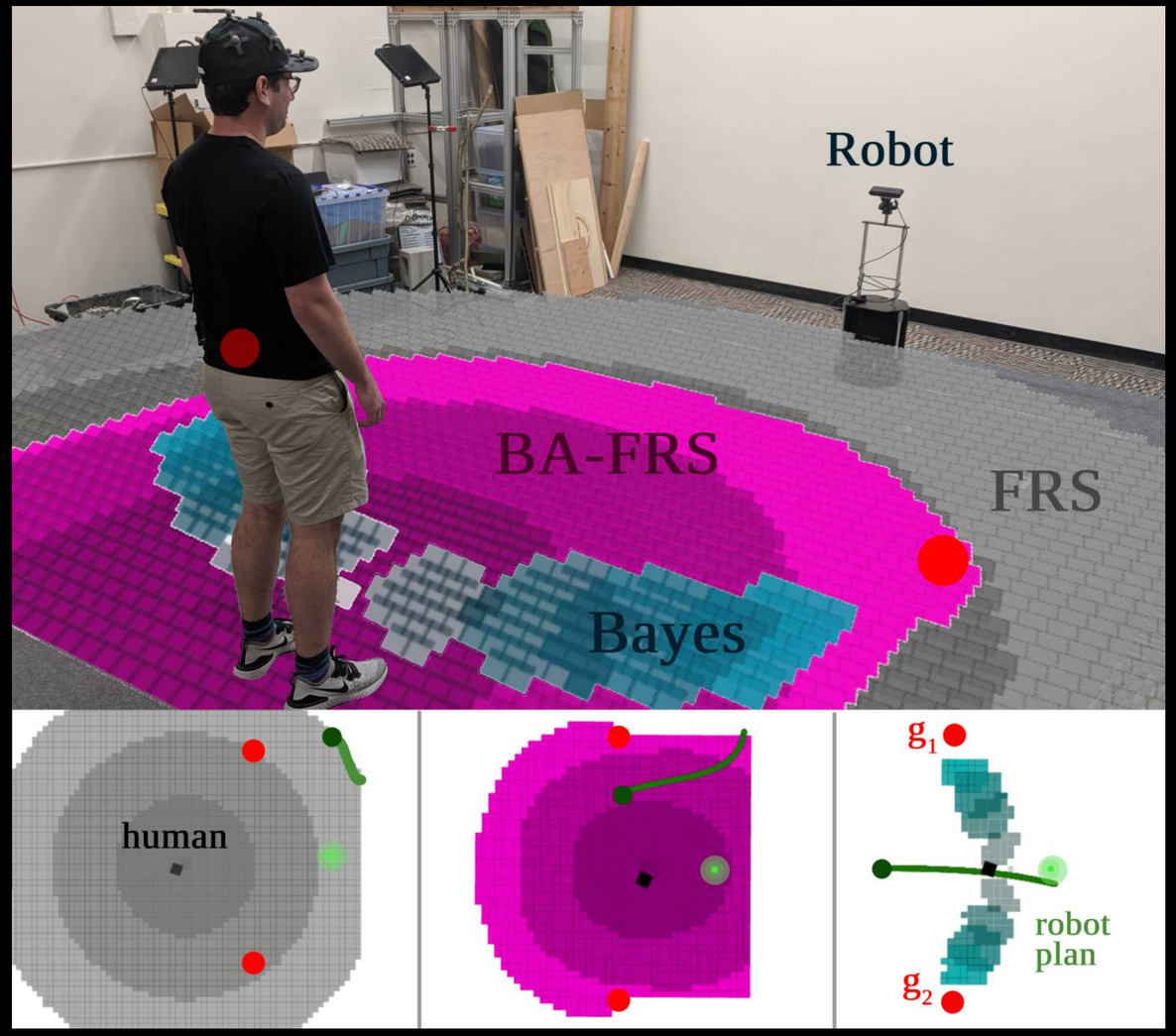
Unknown obstacles



Unknown environment conditions

Unknown behavior of other agents

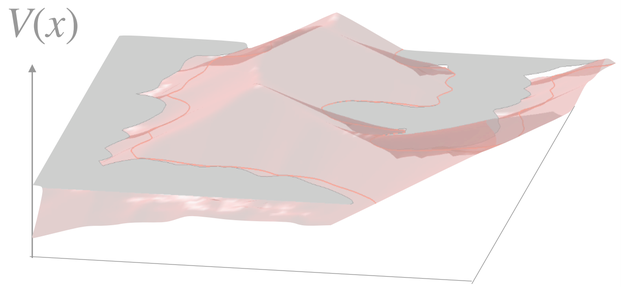
Local Update For Safe Human Motion Prediction



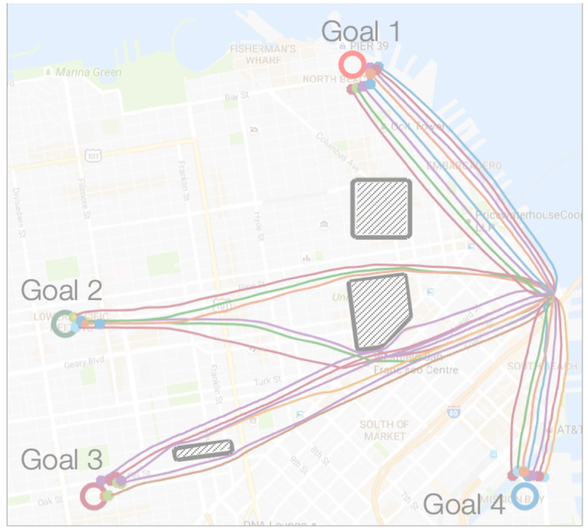
[**Bansal** et al, ICRA 2020]

Outline

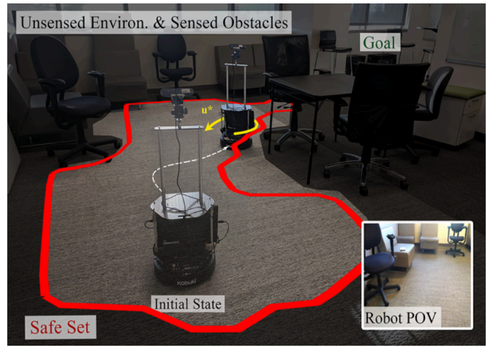
Hamilton-Jacobi Reachability Analysis



Safe Multi-Vehicle Trajectory Planning



Safe Learning-based Perception Systems



Approach

Online Safety Updates

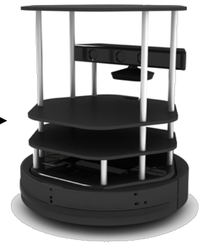
Hardware experiments

Mapping

Perception

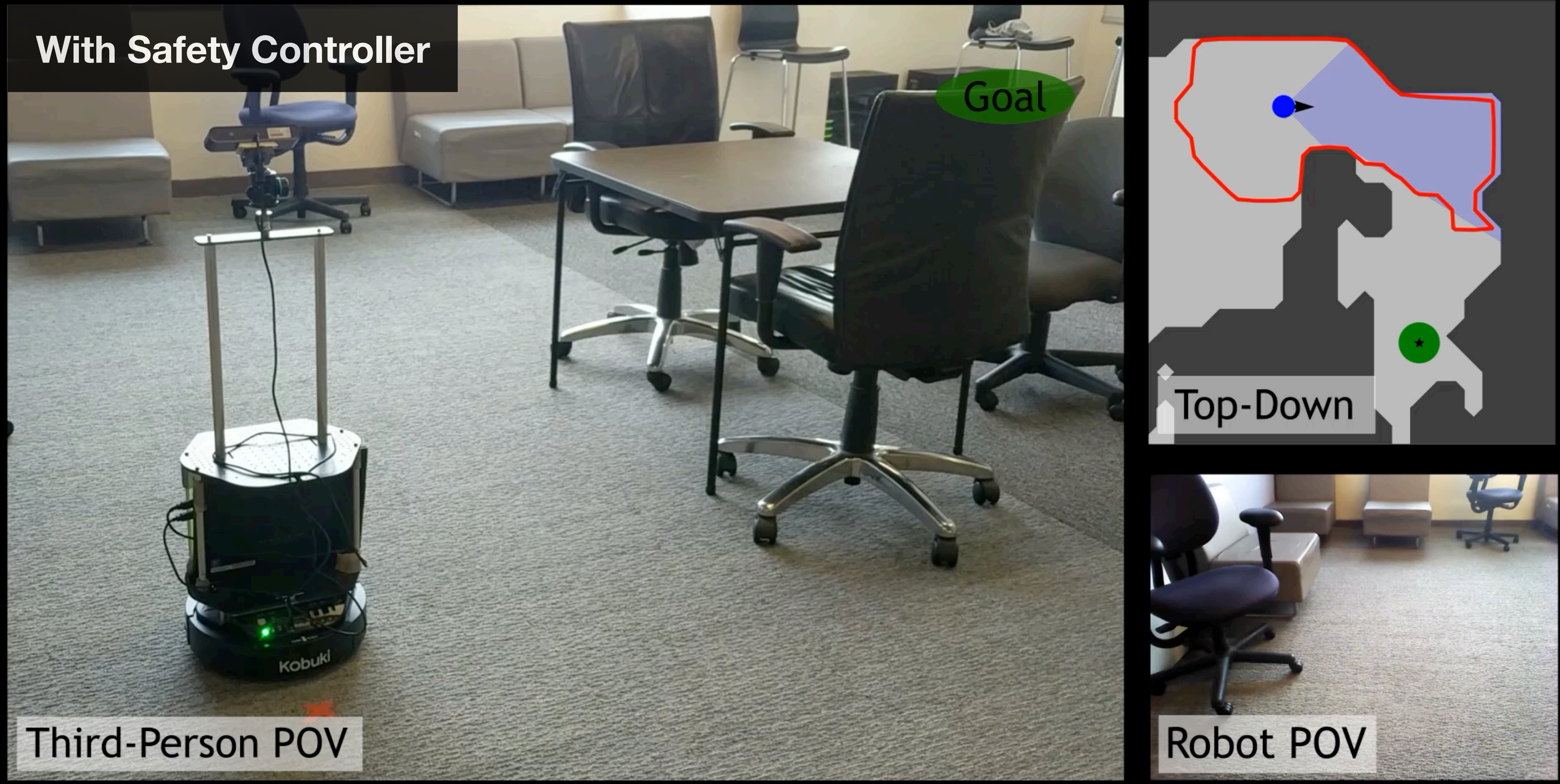
Planning

Safety Verifier



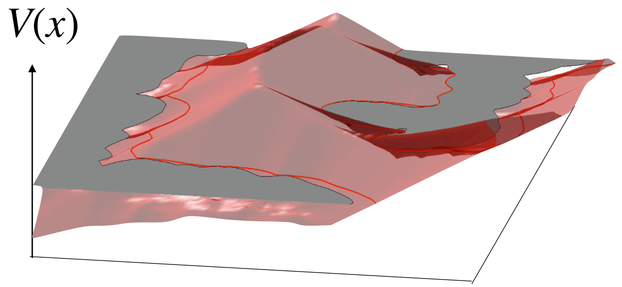
Learning Enabled Planner

Safe Autonomy Stack

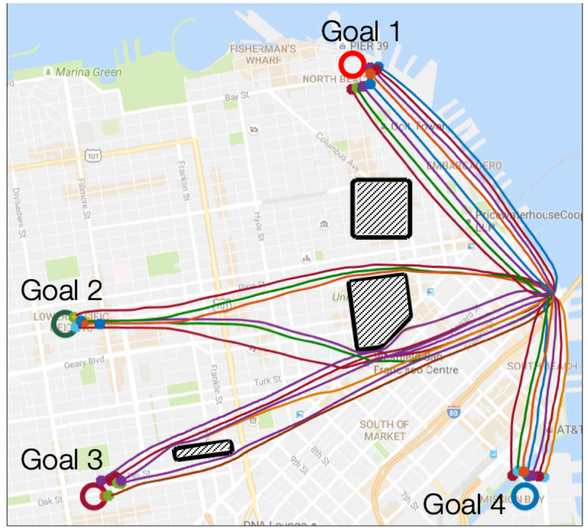

With Safety Controller
Goal
Kobuki
Third-Person POV
Top-Down
Robot POV

Summary

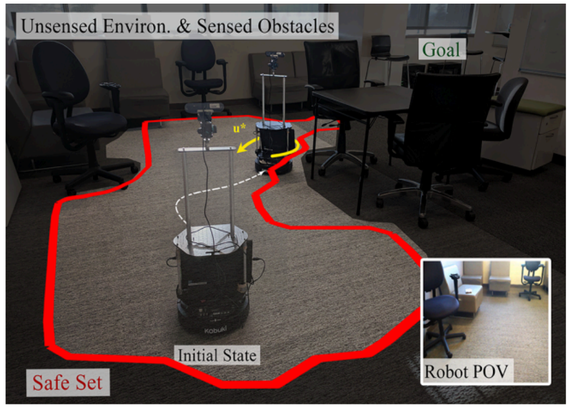
Hamilton-Jacobi Reachability Analysis

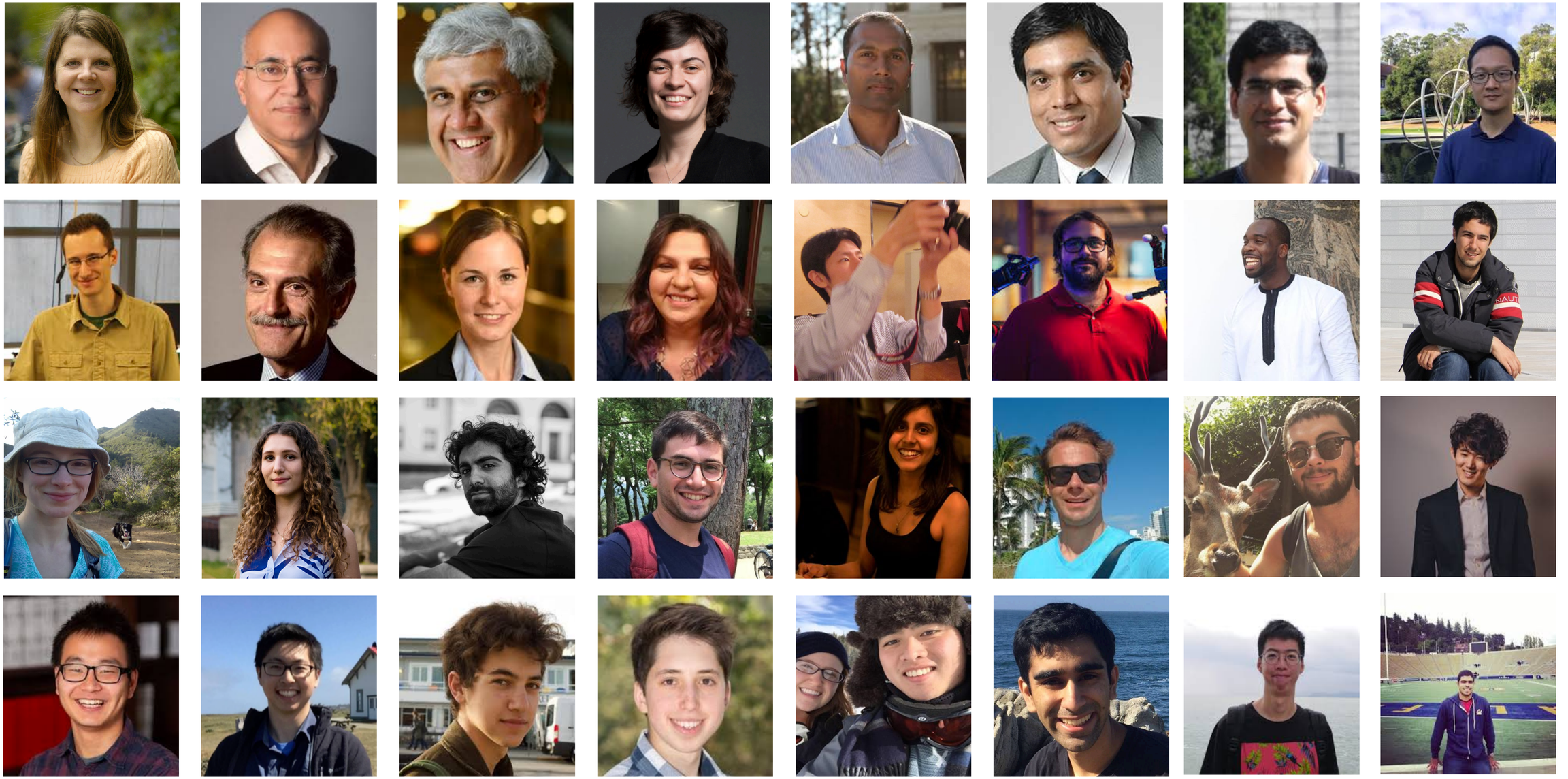


Safe Multi-Vehicle Trajectory Planning



Safe Learning-based Perception Systems





Scaling Hamilton-Jacobi Reachability Analysis for Robotics

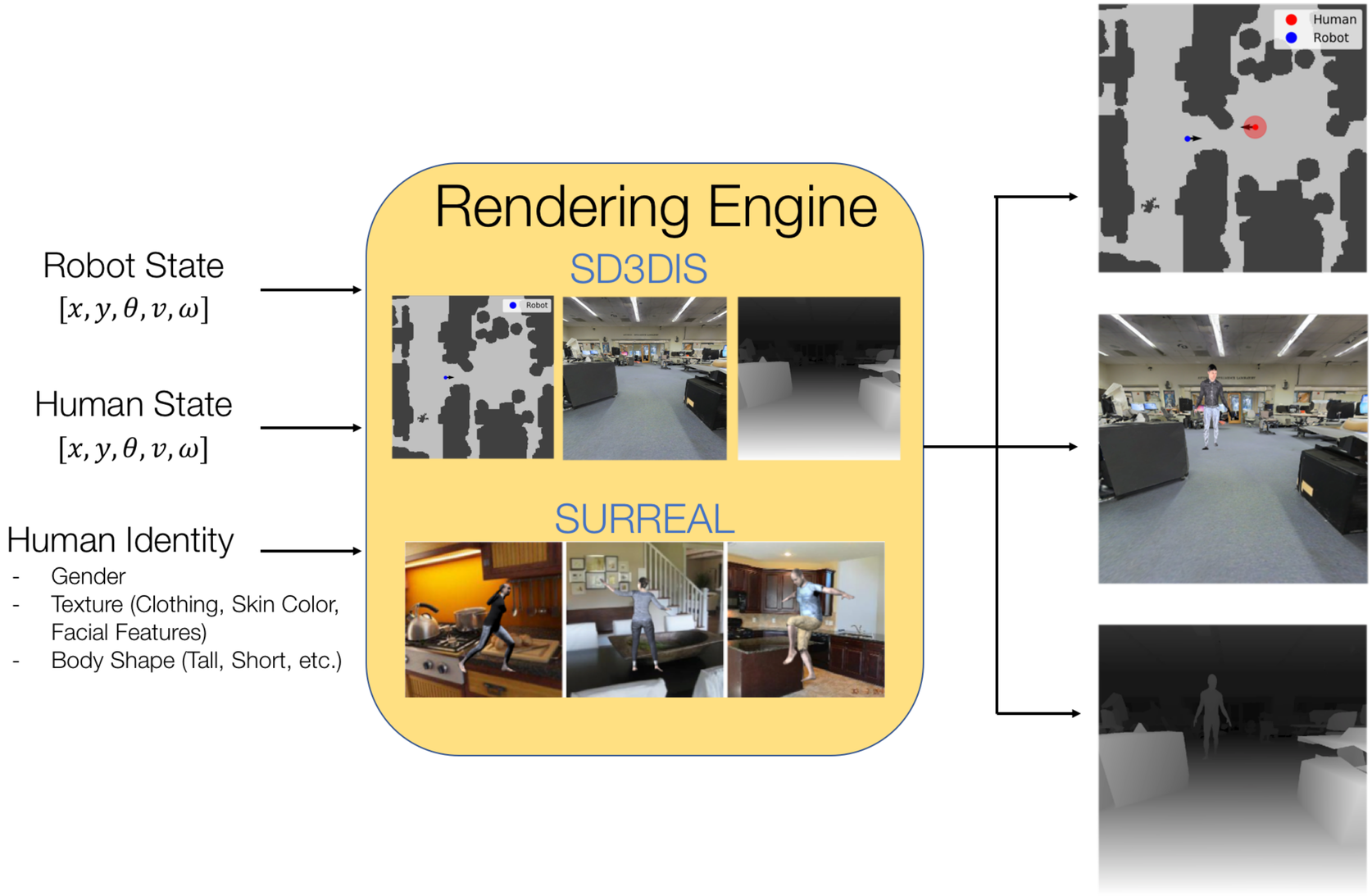
Somil Bansal
University of California, Berkeley

Thank You!

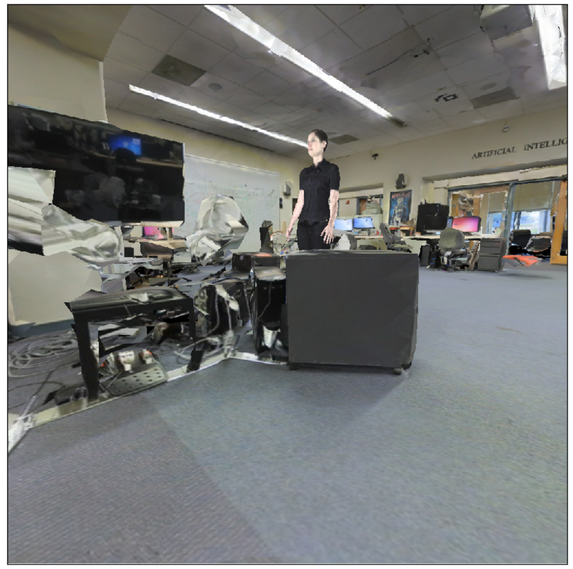
Contact: *somil@berkeley.edu*

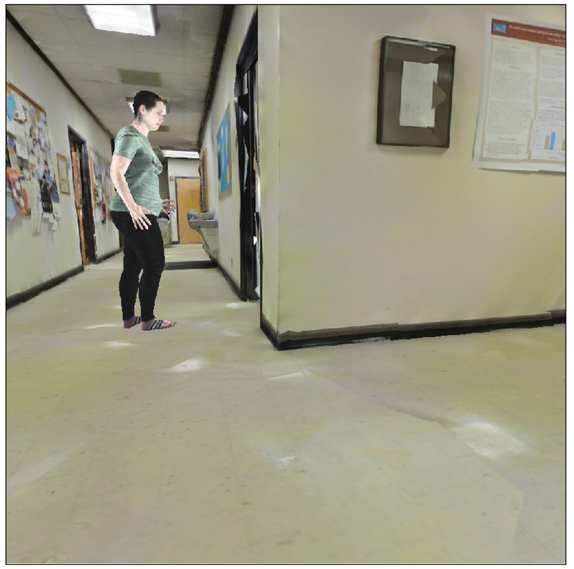
Appendix

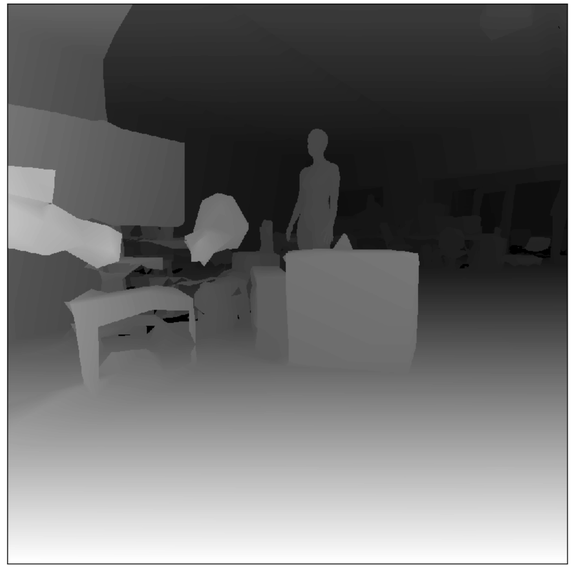
Navigation in Human-Centric Environments

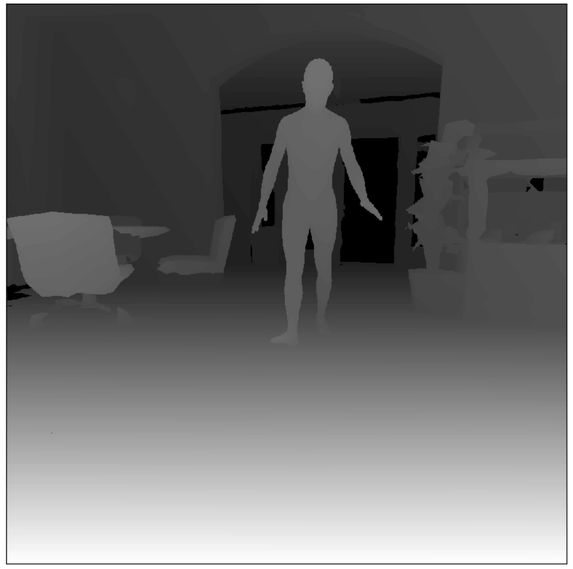


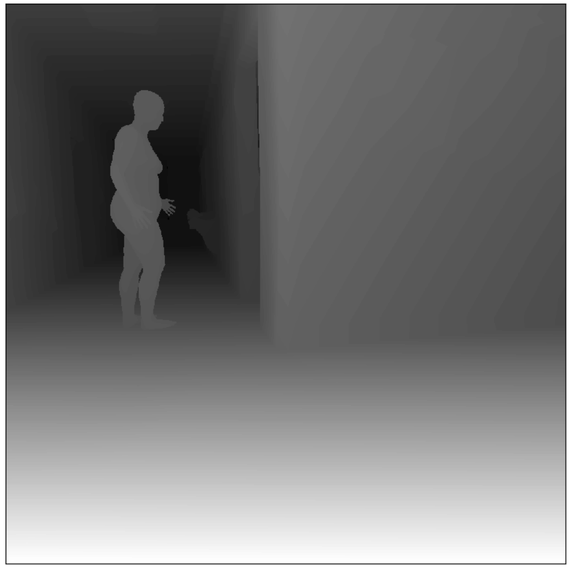
An active dataset for navigation in human-centric environments.



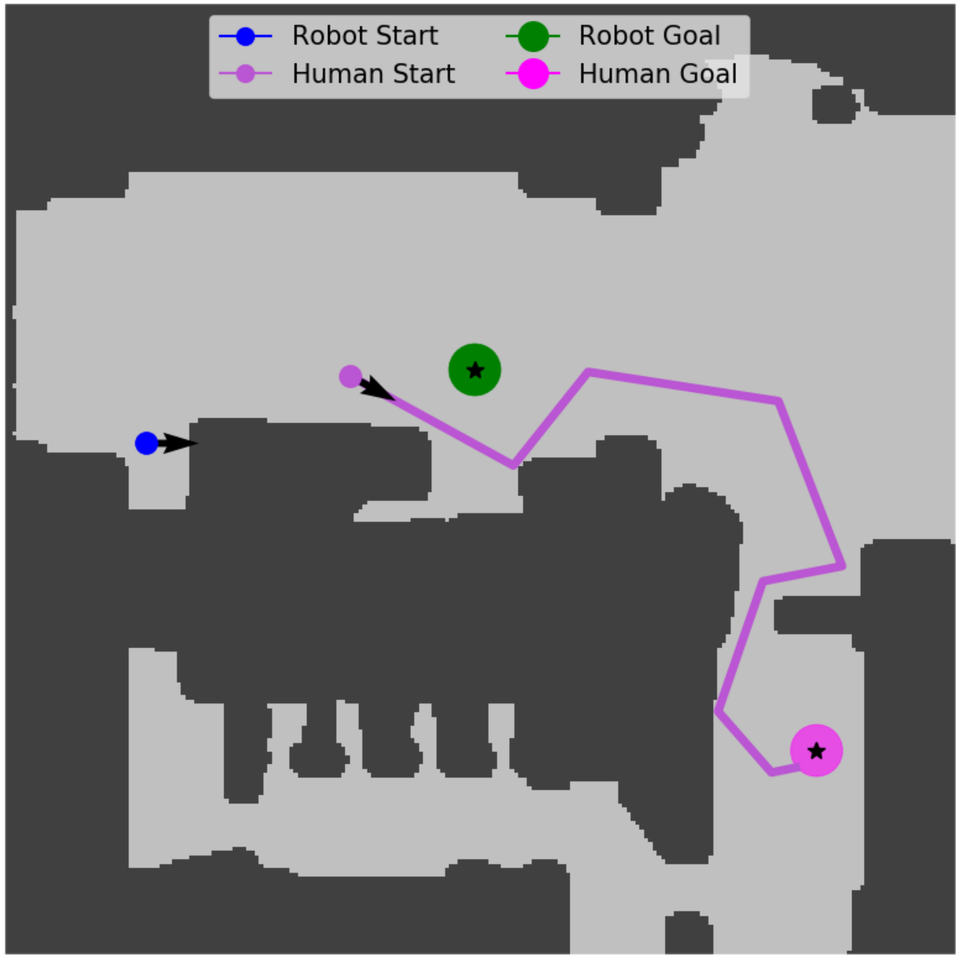








Training Data



$$w_t^* = argmin_w \sum_{i=t}^{t+H_P} J_i^V(z_i^V, u_i^V)$$

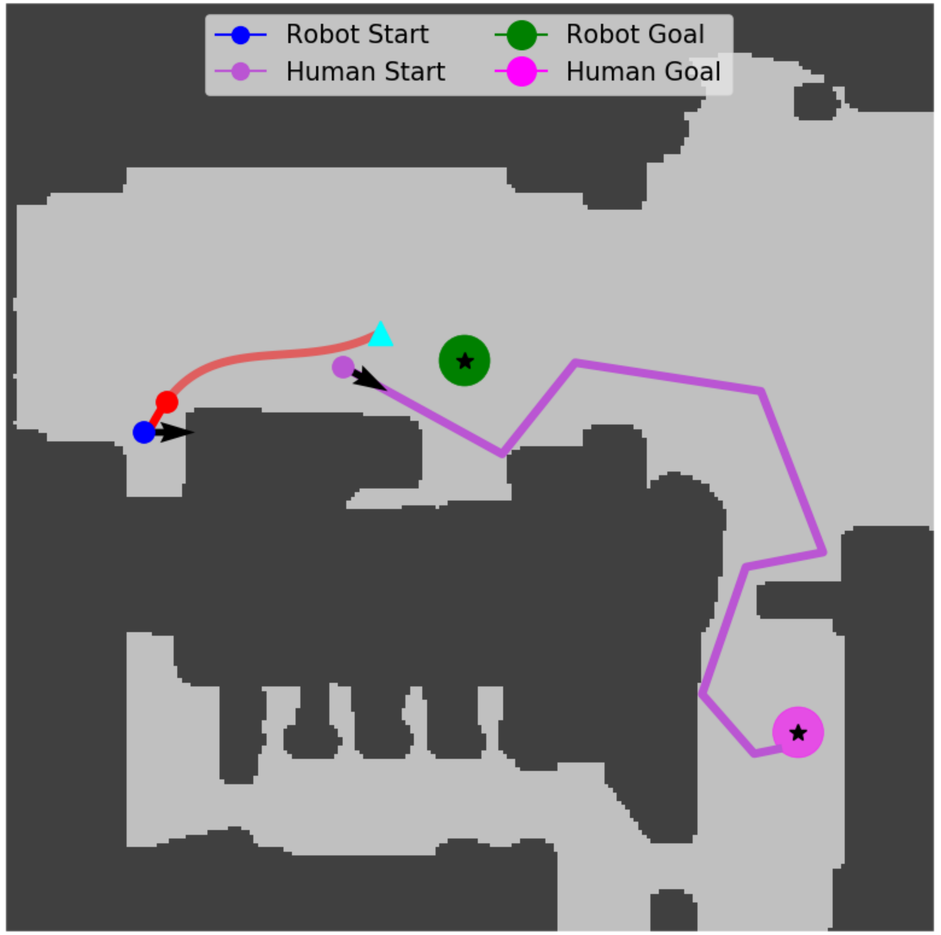
$$subject\ to: \quad \left[(z_t^V, u_t^V), \ldots, (z_{t+H_p}^V, u_{t+H_p}^V)\right] = \tau(z_t^V, u_t^V, w)$$

$$J_i^V(z_i^V, u_i^V) = \lambda_1 d_V^{goal}(z_i^V)^2 +$$

$$\lambda_2(\max\{0, d_{cutoff}^{obs} - d^{obs}(z_i^V)\})^3 +$$

$$\lambda_3(\max\{0, d_{cutoff}^{human} - d_i^{human}(z_i^V)\})^3$$

Training Data



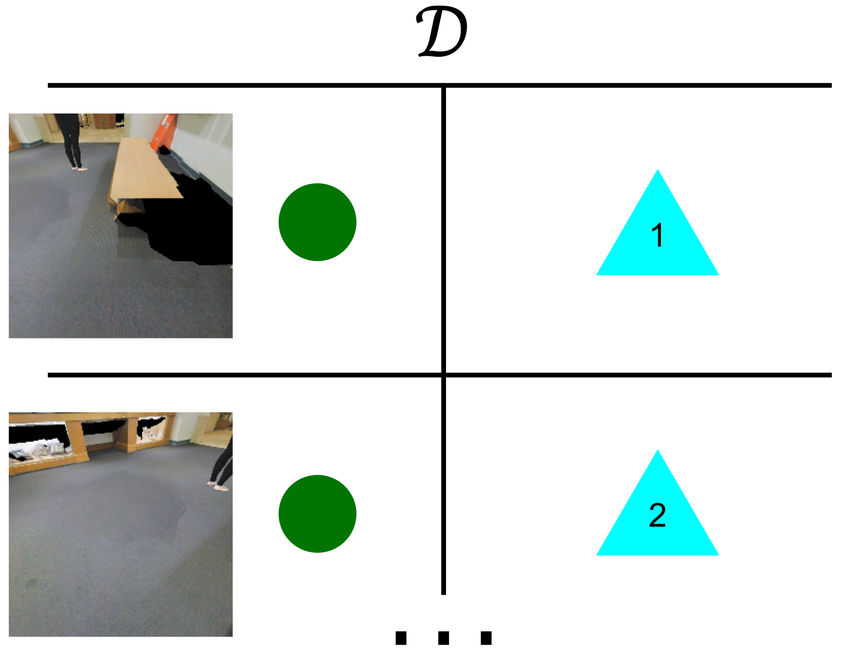
$$w_t^* = argmin_w \sum_{i=t}^{t+H_P} J_i^V(z_i^V, u_i^V)$$

$$subject\ to: \quad \left[(z_t^V, u_t^V), \ldots, (z_{t+H_p}^V, u_{t+H_p}^V)\right] = \tau(z_t^V, u_t^V, w)$$

$$J_i^V(z_i^V, u_i^V) = \lambda_1 d_V^{goal}(z_i^V)^2 +$$

$$\lambda_2(\max\{0, d_{cutoff}^{obs} - d^{obs}(z_i^V)\})^3 +$$

$$\lambda_3(\max\{0, d_{cutoff}^{human} - d_i^{human}(z_i^V)\})^3$$



Robust Interfaces Between Perception and Control

Mapping

Perception

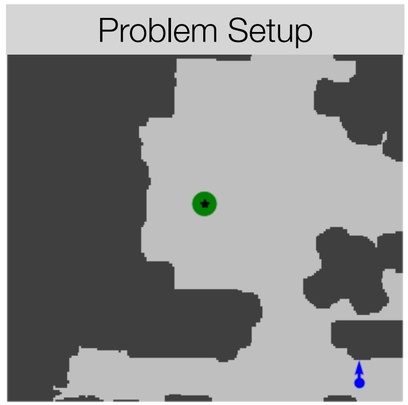
Planning

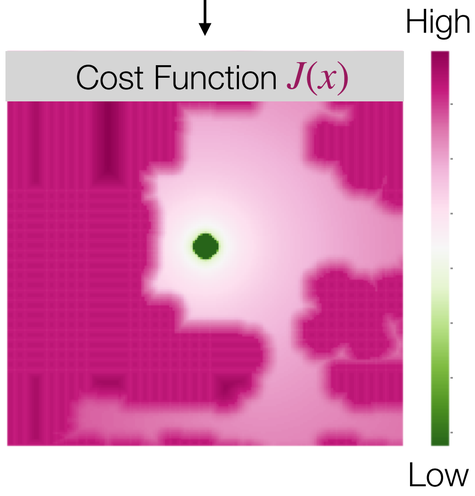
Robust Control

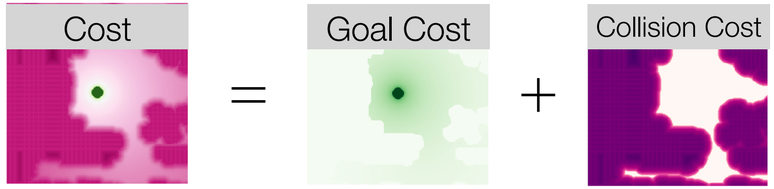
Learning Enabled Planner

Prediction Error
Δ

Robust Data Generation Using Optimal Control







$$\min_{\mathbf{x},\hat{w}} \sum_{i=0}^{T} \max J\left(x_i\right)$$

Cost of the trajectory

subject to $\mathbf{x} = FitSpline(x_0, \hat{w})$ ***Vehicle dynamics***

$x_T = \hat{w}$ ***Waypoint***

$x_0 - Given$ ***Current state***

[Li, **Bansal** et al, arXiv 2020]

2x

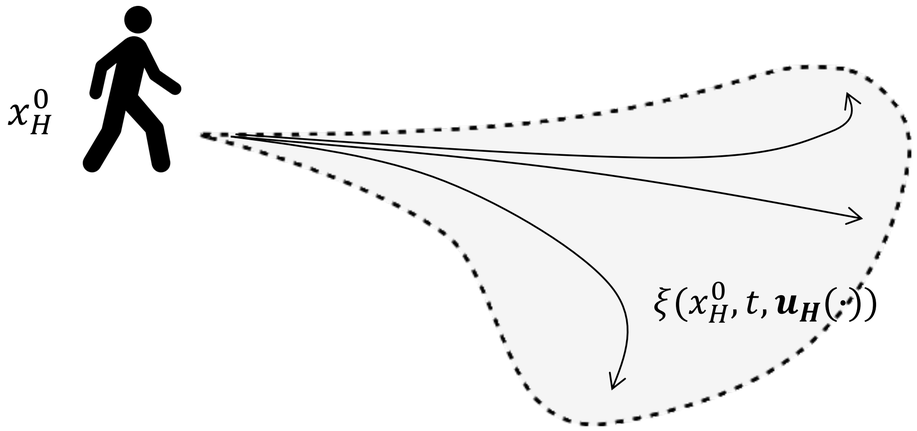
Joint Dynamics

$$\dot{z}^t = f(z, u_H) = \begin{bmatrix} \dot{h}_x \\ \dot{h}_y \\ \dot{P}^t(\boldsymbol{\theta} = \underline{\theta}) \\ \dots \\ \dot{P}^t(\boldsymbol{\theta} = \overline{\theta}) \end{bmatrix} = \begin{bmatrix} v_H \cos(u_H^t) \\ v_H \sin(u_H^t) \\ \underline{g}\,(P^t(\theta), u_H^t, x_H^t) \\ \dots \\ \bar{g}(P^t(\theta), u_H^t, x_H^t) \end{bmatrix}$$

The dynamics are the Bayesian update and the 'input' is the human's action observation

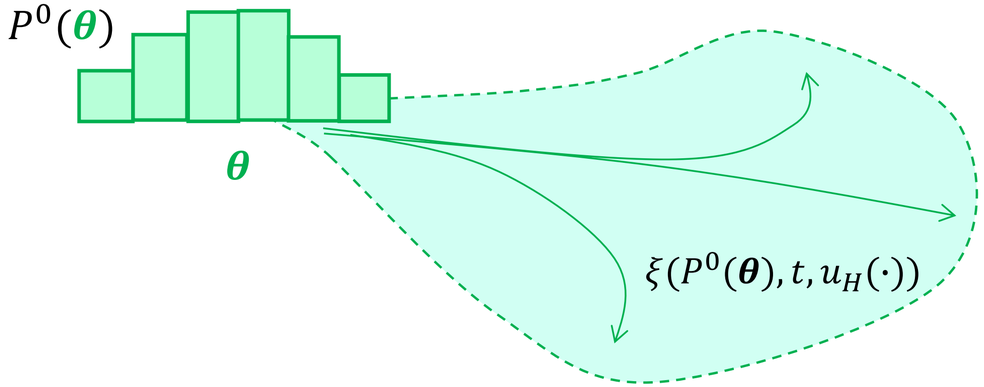
$\dot{x}_H = f_H(x_H, u_H)$

$u_H \in U$



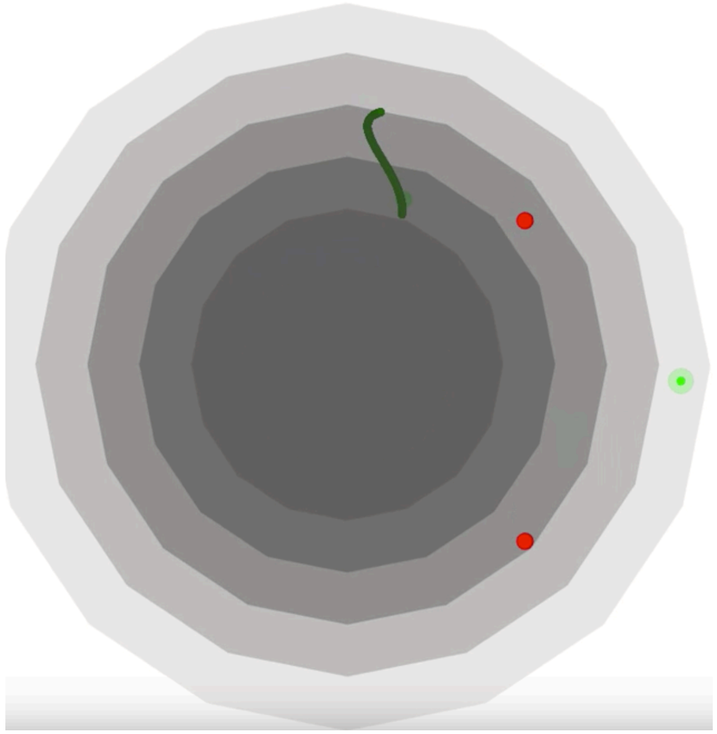
$\dot{P}^t\,(\boldsymbol{\theta}) = g(P^t(\boldsymbol{\theta}), u_H, x_H)$

$u_H \in U_{obs}$

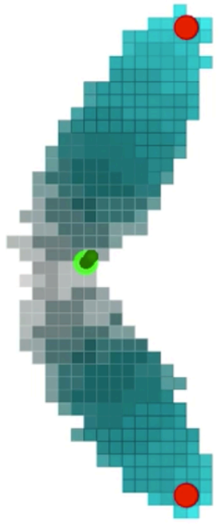


[Vinod, Oishi, 2018]
[Abate, Amin, Prandini, Lygeros, Sastry, 2007]
[Kaelbling, Littman, Cassandra, 1998]

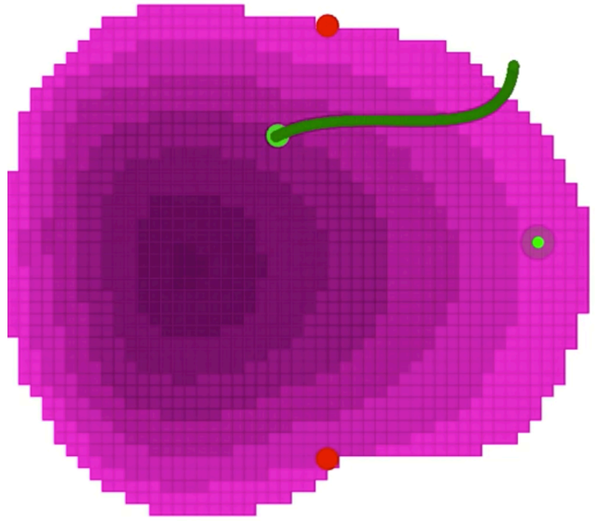
$U_{obs} = U$

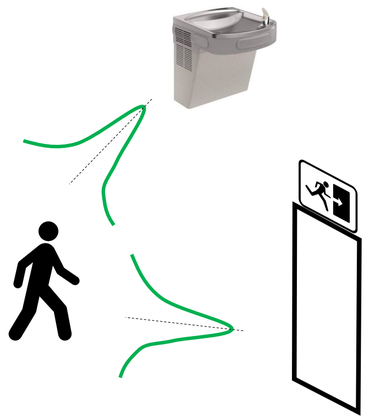


$u_{obs}^t \sim P(u_H^t \mid x_H^t; \theta^t)$

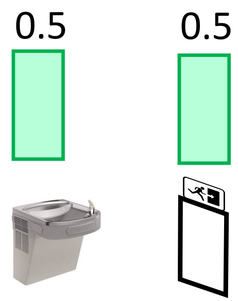


$U_{obs} = \{u_H^t : P(u_H^t \mid z^t) \geq \delta\}$





$P^0(\boldsymbol{\theta})$



Joint Dynamics

$$\dot{z}^t = f(z, u_H) = \begin{bmatrix} \dot{h}_x \\ \dot{h}_y \\ \dot{P}^t(\boldsymbol{\theta} = \underline{\boldsymbol{\theta}}) \\ \dots \\ \dot{P}^t(\boldsymbol{\theta} = \overline{\boldsymbol{\theta}}) \end{bmatrix} = \begin{bmatrix} v_H \cos(u_H^t) \\ v_H \sin(u_H^t) \\ \underline{g}\,(P^t(\theta), u_H^t) \\ \dots \\ \bar{g}(P^t(\theta), u_H^t) \end{bmatrix}$$

The dynamics are the Bayesian update and the 'input' is the human's action observation

$$\dot{x}_H = f_H(x_H, u_H)$$
$$u_H \in U$$

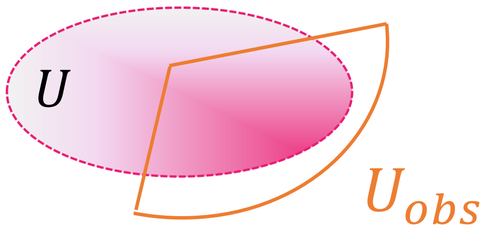
x_H^0 

$$u_H^t \sim P(u_H^t \mid x_H^t; \theta^t) \quad \approx \quad u_H^t \in U_{obs} \subseteq U$$

Stochastic reachability → highly intractable!

$$U_{obs} = \{u_H^t : P(u_H^t \mid z^t) \geq \delta\}$$

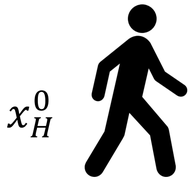
Deterministic reachability

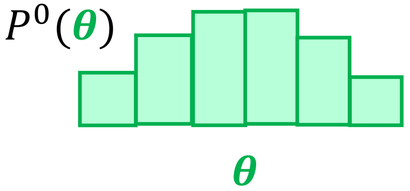


Balanced approximation between stochastic reachability and full forward reachable set

Deterministic Reachability Problem

$$FRS(z,t) \coloneqq \{z' \colon \exists u_H(\cdot), z' = \xi(z,t,u_H(\cdot))\}$$





$$\dot{z}^t = f(z^t, u_H^t)$$

$$u_H \in U_{obs}$$

$$U_{obs} = \{u_H^t : P(u_H^t \mid z^t) \geq \delta\}$$

Hamilton-Jacobi Reachability Analysis

Hamilton-Jacobi Reachability: A Brief Overview and Recent Advances

Somil Bansal*, Mo Chen*, Sylvia Herbert* and Claire J. Tomlin

***Abstract*— Hamilton-Jacobi (HJ) reachability analysis is an important formal verification method for guaranteeing performance and safety properties of dynamical systems; it has been applied to many small-scale systems in the past decade. Its advantages include compatibility with general nonlinear system dynamics, formal treatment of bounded disturbances, and the availability of well-developed numerical tools. The main challenge is addressing its exponential computational complexity with respect to the number of state variables. In this tutorial, we present an overview of basic HJ reachability theory and provide instructions for using the most recent numerical tools, including an efficient GPU-parallelized implementation of a Level Set Toolbox for computing reachable sets. In addition, we review some of the current work in high-dimensional HJ reachability to show how the dimensionality challenge can be alleviated via various general theoretical and application-specific insights.**

I. Introduction

As the systems we design grow more complex, determining whether they work according to specification becomes more difficult. Consequently, verification and validation have received major attention in many fields of engineering. However, verification of systems is challenging for many reasons. First, all possible system behaviors must be accounted for. This makes most simulation-based approaches insufficient, and thus formal verification methods are needed. Second, many practical systems are affected by disturbances in the environment, which can be unpredictable, and may even contain adversarial agents. In addition, these systems often have high dimensional state spaces and evolve in continuous time with complex, nonlinear dynamics.

Hamilton-Jacobi (HJ) reachability analysis is a verification method for guaranteeing performance and safety properties of systems, overcoming some of the above challenges. In reachability analysis, one computes the reach-avoid set, defined as the set of states from which the system can be driven to a target set while satisfying time-varying state constraints at all times. A major practical appeal of this approach stems from the availability of modern numerical tools, which can compute various definitions of reachable sets [1]–[4]. For example, these numerical tools have been successfully used to solve a variety of differential games, path planning problems, and optimal control problems. Concrete practical applications include aircraft auto-landing [5], automated aerial refueling [6], model predictive control (MPC) of quadrotors [7], [8], multiplayer reach-avoid games [9], large-scale multiple-vehicle path planning [10], [11], and real-time safe motion planning [12]. However, HJ reachability becomes computationally intractable as the state space dimension increases. Traditionally, reachable set computations involve solving an HJ partial differential equation (PDE) on a grid representing a discretization of the state space, resulting in an *exponential* scaling of computational complexity with respect to system dimensionality; this is often referred to as the "curse of dimensionality." However, recent work has made a significant leap in overcoming these challenges by exploiting system structures to decompose the computation of reachable set into several small dimensional computations [13], [14]. In addition, convex optimization applied to the Hopf-Lax formula allows real-time computation of the HJ PDE solution at any desired state and time instant when the system dynamics are linear [15], [16].

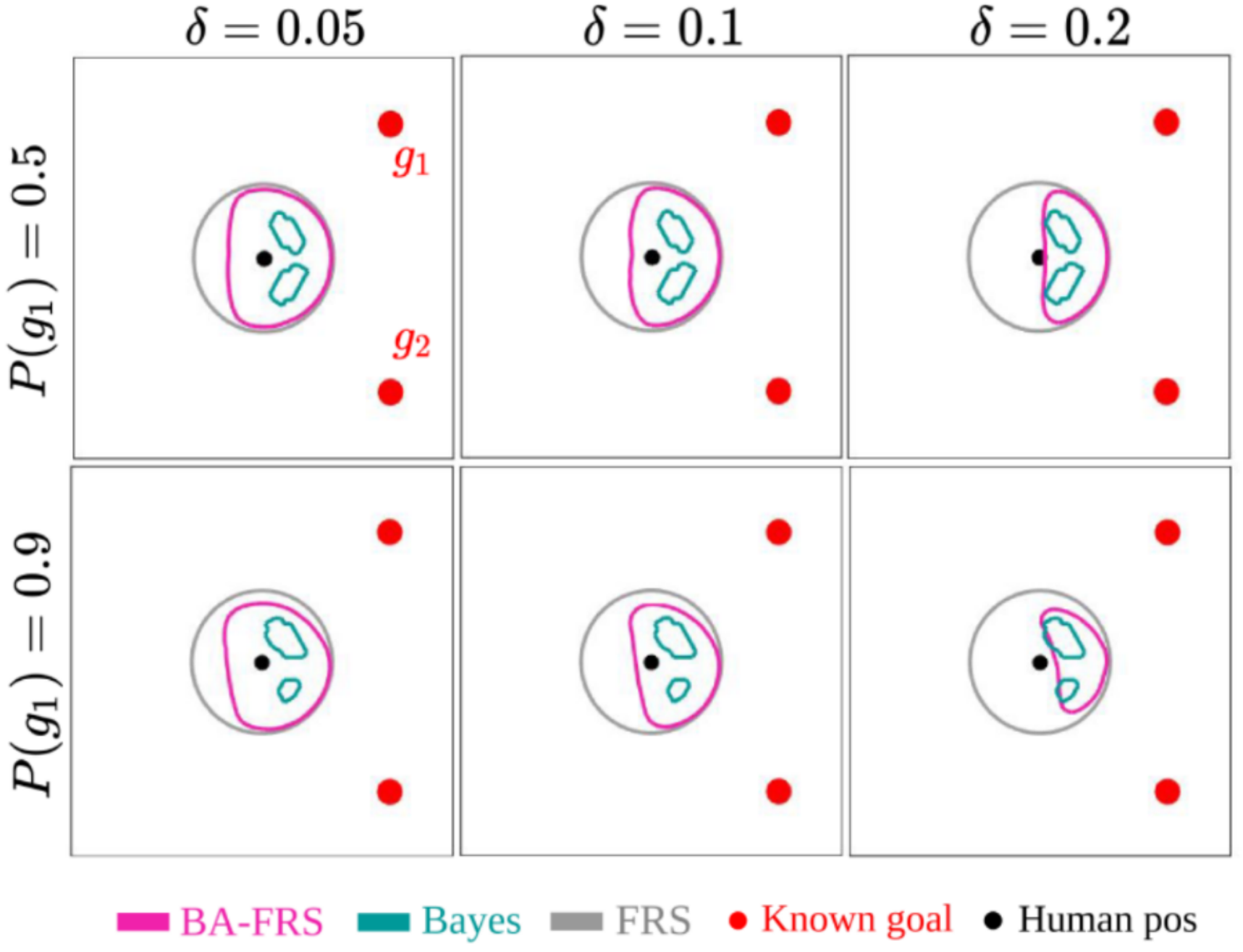
Besides HJ reachability, alternative approaches to verification exist. In particular, satisfaction of properties such as safety, liveness, and fairness in computer software and in discrete-time dynamical systems can be verified by checking whether runs of a transition system, or words of a finite automaton satisfy certain desired properties [17], [18]. These properties may be specified by a variety of logical formalisms such as linear temporal logic. For specifications of properties of interest in autonomous robots, richer formalisms have been proposed. For example, propositional temporal logic over the reals [19], [20] allows specification of properties such as time in terms of real numbers, and chance-constrained temporal logic [21] allows specification of requirements in the presence of uncertainty. Besides autonomous cars and robots, verification approaches based on discrete models have also been successfully used in the context of intelligent transportation systems [22] and human-automation interaction [23].

For continuous and hybrid systems, safety properties can be verified by checking whether the forward reachable set or an over-approximation of it intersects with a set of undesirable states, akin to checking runs of transition systems. Numerous tools such as SpaceEx [24], Flow* [25], CORA [26], C2E2 [27], [28], and dReach [29] have been developed for this purpose; the authors in [30] present a tutorial on combining different tools for hybrid systems verification. In addition, methods that utilize semidefinite programming to search for Lyapunov functions can be used to verify safety [31], [32]. This is done, for example, by constructing

* All authors contributed equally to this article. Authors' names are written in the alphabetical order. All authors are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley. {somil, mochen72, sylvia.herbert, tomlin}@eecs.berkeley.edu

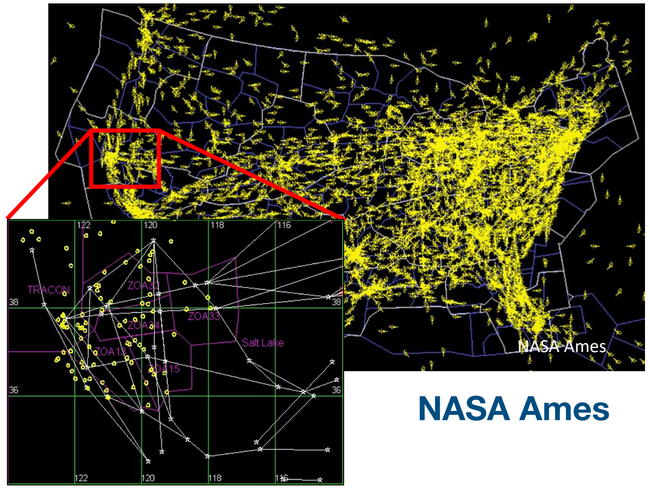
This tutorial is supported by NSF under the CPS Frontiers VehiCal project (1545126) and CPS:ActionWebs (CNS-931843), by the UC-Philippine-California Advanced Research Institute under project IID-2016-005, by the ONR MURI Embedded Humans (N00014-16-1-2206), and by NASA under grants NNX12AR18A and UCSCMCA-14-022 (UARC).

https://www.youtube.com/watch?v=iWsfc107nRc


$\delta = 0.05$
$\delta = 0.1$
$\delta = 0.2$
$P(g_1) = 0.5$
$P(g_1) = 0.9$
g_1
g_2
BA-FRS
Bayes
FRS
Known goal
Human pos

Bayesian Predictor

Air Traffic Control

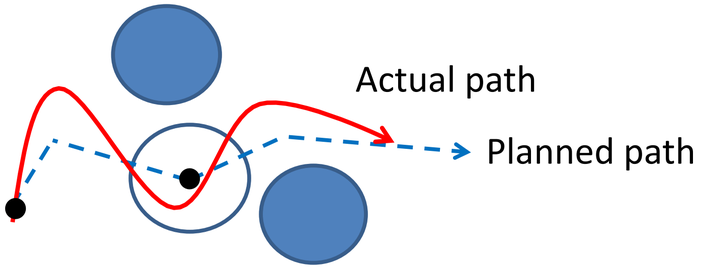


NASA Ames

UAV Applications

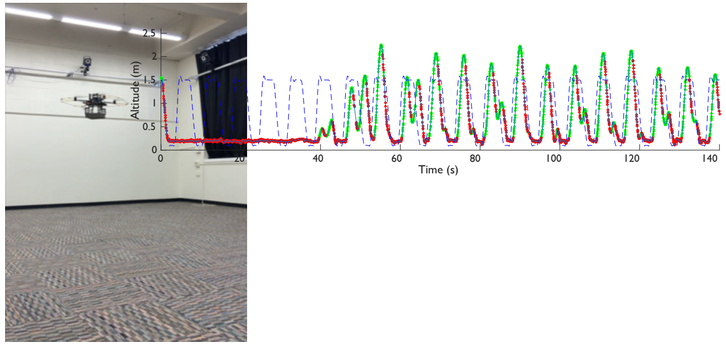
Bansal et al

Robust Planning



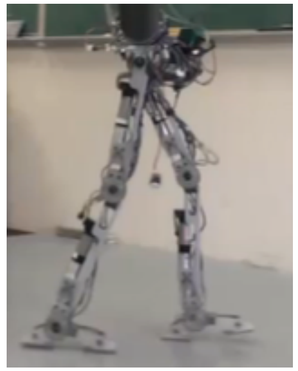
Herbert et al

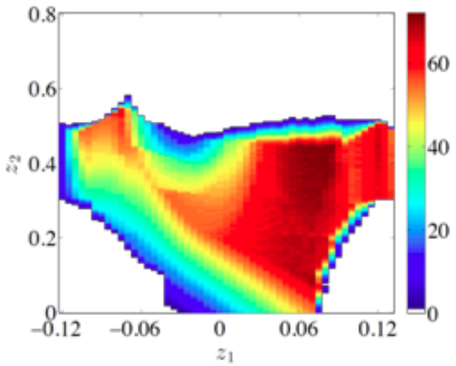
Safe Learning



Fisac et al

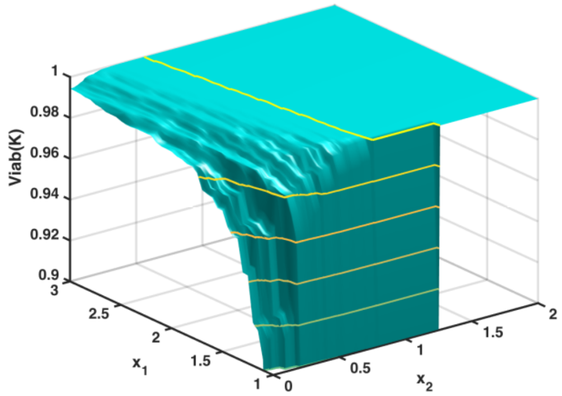
Safe Bipedal Walking





Ames et al
Vasudevan et al

Safe Anesthesia Delivery



Kaynama et al