Safe Learning in Robotics

2025-07-21

Contents

Abstract

This article provides a concise but holistic review of the recent advances made in using machine learning to achieve safe decision making under uncertainties, with a focus on unifying the language and frameworks used in control theory and reinforcement learning research.

1 Introduction

Robotics researchers strive to design system taht can operate autonomously in increasingly complex scenarios, often in close proximity to humans. Uncertainties arise from various sources. For example, the robot dynamics may not be perfectly modeled, sensor measurements may be noisy, and/or the operating environment may not be well-characterized or may include other agents whose dynamics and plans are not known. In recent years, the research community has multiplied its efforts to leverage data-based approaches to address this problem. (learning-based approach) A crucial, domain-specific challenge of learning for robot control is not only for the optimized policy but also during training, to avoid costly hardware failures and improve convergence. Both control theory and machine learning (reinforcement learning) have proposed approaches to tackle this problem.

- Control theory has traditionally taken a model-driven approach: it leverages a given dynamics model and provides guarantees with respect to known operating conditions.
- RL has traditionally taken a data-driven approach, which makes it highly adaptable to new contexts at the expense of providing formal guarantees.
  - RL, however, transfer to real systems remains a research area in itself.

In this article, we provide a bird's eye view of the most recent work in learning-based control and reinforcement learning that implement safety and provide safety guarantees for robot control.

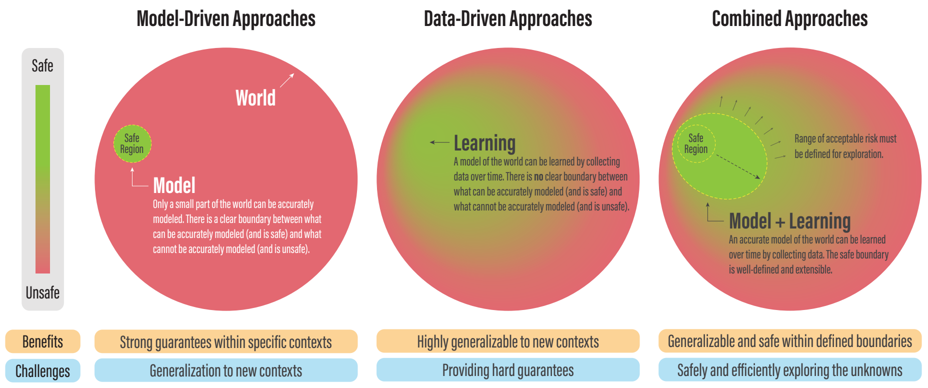
Model-Driven Approaches
Data-Driven Approaches
Combined Approaches
Safe
Unsafe
World
Safe Region
Model
Only a small part of the world can be accurately modeled. There is a clear boundary between what can be accurately modeled (and is safe) and what cannot be accurately modeled (and is unsafe).
Learning
A model of the world can be learned by collecting data over time. There is no clear boundary between what can be accurately modeled (and is safe) and what cannot be accurately modeled (and is unsafe).
Safe Region
Range of acceptable risk must be defined for exploration.
Model + Learning
An accurate model of the world can be learned over time by collecting data. The safe boundary is well-defined and extensible.
Benefits
Strong guarantees within specific contexts
Highly generalizable to new contexts
Generalizable and safe within defined boundaries
Challenges
Generalization to new contexts
Providing hard guarantees
Safely and efficiently exploring the unknowns

Figure 1: A comparison of model-driven, data-driven, and combined approaches.

2 Preliminaries

2.1 Problem Statement

We formulate the safe learning control problem as an optimization problem to capture the efforts of both the control and RL communities. The optimization problem has three main components:

- **System model** that describe the dynamic behavior of the robot.
- **Cost function** that defines the control objective or task goal.
- **Set of constraints** that specify the safety requirements.

The goal is to find a controller or policy that computes commands (also called inputs) that enable the system to fulfill the task while respecting given safety constraints.

2.2 System Model

We consider a robot whose dynamics can be represented by the following discrete-time model:

$$\mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k), \tag{1}$$

where $\mathbf{k} \in \mathbb{Z}_{\geq 0}$ is the discrete-time index, $\mathbf{x}_k \in \mathbb{X}$ is the state with $\mathbb{X}$ denoting the state space. $\mathbf{u}_k \in \mathbb{U}$ is the input with $\mathbb{U}$ denoting the input space, $\mathbf{f}_k$ is the dyanmics model of the robot, $\mathbf{w}_k \in \mathbb{W}$ is the process noise distributed according to a distribution W. Throughout this review, we assume direct access to (possibly noisy) measurements of the robot state $\mathbf{x}_k$ and neglect the problem of state estimation. The dynamics model in Eq. (1) can be equivalently as a state transition probability function $T_k(\mathbf{x}_{k+1}|\mathbf{x}_k, \mathbf{u}_k)$, commonly seen in RL approaches.

2.3 Cost Function

The robot's task is defined by a cost function. We consider a finite-horizon optimal control problem with time horizon N. Given an initial state $\mathbf{x}_0$, the cost function is computed based on the sequence of states $\mathbf{x}_{0:N} = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_N\}$ and the sequence of inputs $\mathbf{u}_{0:N-1} = \{\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N-1}\}$:

$$J(\mathbf{x}_{0:N}, \mathbf{u}_{0:N-1}) = l_N(\mathbf{x}_N) + \sum_{k=0}^{N-1} l_k(\mathbf{x}_k, \mathbf{u}_k), \tag{2}$$

where $l_k : \mathbb{X} \times \mathbb{U} \mapsto \mathbb{R}$ is the stage cost incurred at each time step k, and $l_N : \mathbb{X} \mapsto \mathbb{R}$ is the terminal cost incurred at the end of N-step horizon. The stage and terminal cost functions map the state and input sequences, which may be random variables, to a real number. We formulate a cost minimization problem for safe learning control, while RL typically solves a reward maximization problem.

2.4 Safety Constraints

Safety constraints ensure, or encourage, the safe operation of the robot and include:

- state constraints $\mathbb{X}_c \subseteq \mathbb{X}$, which defines the set of safe operating states
- input constraints $\mathbb{U}_c \subseteq \mathbb{U}$
- stability guarantees
  - To encode the safety constraints, we define n_c constraint functions: $\mathbf{c}_k(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) \in \mathbb{R}^{n_c}$ with each constraint c_k^j being a real-valued, time-varying function.
  - We introduce three levels of safety: hard, probabilistic(chance), and soft.

2.4.1 Safety level 3: Constraint Satisfaction Guaranteed

The system satisfies hard constraints:

$$c_k^j(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) \leq 0 \tag{3}$$

for all times $k \in \{0, \ldots, N\}$ and constraint indexes $j \in \{1, \ldots, n_c\}$.

2.4.2 Safety level 2: Constraint Satisfaction with Probability p

The system satisfies probabilistic constraints:

$$\Pr\left(c_k^j(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) \leq 0\right) \geq p^j \tag{4}$$

where $\Pr(\cdot)$ denotes the probability and $p^j \in (0, 1)$ defines the likelihood of the j-th constraint being satisfied, with $j \in \{1, \ldots, n_c\}$ and for all times $k \in \{0, \ldots, N\}$. The chance constraint in Eq. (4) is identical to the hard constraint in Eq. (3) for $p^j = 1$.

2.4.3 Safety Level 1: Constraint Satisfaction Encouraged

The system encourages constraint satisfaction. This can be achieved in different ways. One way is to add a penalty term to the objective function that discourages the violation of constraints with a high cost. A non-negative ϵ_j is added to the right-hand side of the inequality in Eq. (3), for all times $k \in \{0, \ldots, N\}$ and constraint indexes $j \in \{1, \ldots, n_c\}$:

$$c_k^j(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) \leq \epsilon_j \tag{5}$$

and an appropriate penalty term $l_\epsilon(\epsilon) \geq 0$ with $l_\epsilon(\epsilon) = 0 \iff \epsilon = 0$ is added to the objective function. (It means there is no penalty when the constraint is

satisfied.) The vector ϵ includes all elements ϵ_j and is an additional variable of the optimization problem.
Alternatively, although $c^j_k(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k)$ is a step-wise quantity, some approaches only aim to provide guarantees on its expected value $\mathbb{E}[\cdot]$ on a trajectory level:

$$J_{c^j} = \mathbb{E}\left[\sum_{k=0}^{N-1} c^j_k(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k)\right] \leq d_j, \tag{6}$$

where J_{c^j} represents the expected total constraint cost, and d_j defines the corresponding constraint threshold. The constraint function can optionally be discounted as $\gamma^k c^j_k(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k)$ with the discount factor $\gamma \in [0, 1]$.

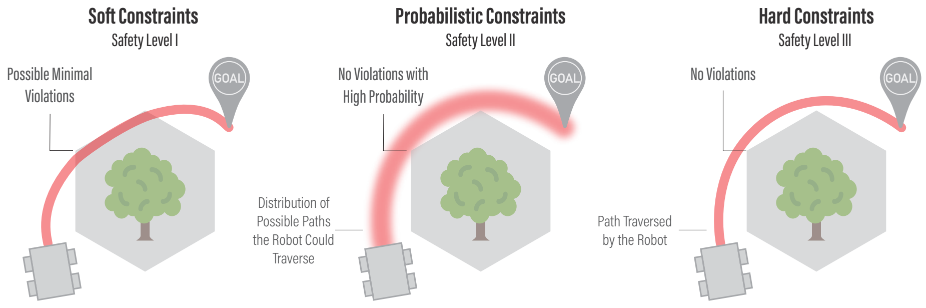
Soft Constraints
Safety Level I
Possible Minimal Violations
GOAL
Probabilistic Constraints
Safety Level II
No Violations with High Probability
Distribution of Possible Paths the Robot Could Traverse
GOAL
Hard Constraints
Safety Level III
No Violations
Path Traversed by the Robot
GOAL

Figure 2: Illustation of the different safety levels.

2.5 Formulation of the Safe Learning Control Problem

The function introduced above, i.e., the system model $\mathbf{f}$, the constraints $\mathbf{c}$, and the cost function J, represent the true functions of the robot control problem. In practice, $\mathbf{f}, \mathbf{c}$, and J may be unknown or partially known. Without loss of generality, we assume that each of the true functions $\mathbf{f}, \mathbf{c}$, and J can be decomposed into a nominal component, $(\bar{\cdot})$, reflecting our prior knowledge, and an unknown component $(\hat{\cdot})$, to be learned from data. For instance, the dynamics model f can be decomposed as:

$$\mathbf{f}_k(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) = \bar{\mathbf{f}}_k(\mathbf{x}_k, \mathbf{u}_k) + \hat{\mathbf{f}}_k(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k), \tag{7}$$

where $\bar{\mathbf{f}}$ is the prior dynamics mode and $\hat{\mathbf{f}}$ are the uncertain dynamics. Safe learning control (SLC) leverages our prior knowledge $\mathcal{P} = \{\bar{\mathbf{f}}, \bar{\mathbf{c}}, \bar{J}\}$ and the data collected from the system $\mathcal{D} = \{\mathbf{x}^{(i)}, \mathbf{u}^{(i)}, \mathbf{c}^{(i)}, l^{(i)}\}_{i=0}^{i=\mathcal{D}}$ to find a policy (or controller) $\pi_k(\mathbf{x}_k)$ that achieves the given task while respecting all safety constraints. $(\cdot)^{(i)}$ denotes a sample of a quantity $(\cdot)_k$ and D is the data set size. More specifically, we aim to find a policy π_k that best approximates the true optimal policy π_k^*, which is the solution to the following optimization problem:

$$\begin{aligned} J^{\pi^*}(\bar{\mathbf{x}}_0) = \min_{\pi_{0:N-1},\epsilon} \; & J(\mathbf{x}_{0:N}, \mathbf{u}_{0:N-1}) + l_\epsilon(\epsilon) \\ \text{s.t} \quad & \mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k), \quad \mathbf{w}_k \sim \mathcal{W}, \forall k \in \{0, \dots, N-1\}, \\ & \mathbf{u}_k = \pi_k(\mathbf{x}_k), \\ & \mathbf{x}_0 = \bar{\mathbf{x}}_0, \\ & \text{Safety constraints according to either} \\ & \text{Eq. (3), Eq. (4), Eq. (5) or Eq. (6), and } \epsilon \geq 0, \end{aligned} \tag{8}$$

where $\bar{\mathbf{x}}_0 \sim \mathcal{X}_0$ is the initial state with $\mathcal{X}_0$ being the initial state distribution, and ϵ and l_ϵ are introduced account for the soft safety constraint case (Eq. (5)) and are set to zero, for example, if only hard and probabilistic safety constraints are considered. (Eq. (3), Eq. (4))

2.6 A Control Theory Perspective

Typical assumptions are that a model of the system is available and it is either parameterized by an unknown parameter or it has bounded unknown dynamics and noise.

2.6.1 Adaptive Control

Adaptive control considers systems with parameteric uncertaintes and adapts the controller parameters online to optimize performance. [1] Adaptive control requires knowledge of the parameteric form of the uncertainty and typically considers a dynamics model that t hat is affine in $\mathbf{u}$ and the uncertain parameters $\theta \in \mathbf{\Theta}$:

$$\mathbf{x}_{k+1} = \bar{\mathbf{f}}_{\mathbf{x}}(\mathbf{x}_k) + \bar{\mathbf{f}}_{\mathbf{u}}(\mathbf{x}_k)\mathbf{u}_k + \bar{\mathbf{f}}_\theta(\mathbf{x}_k)\theta, \tag{9}$$

where $\bar{\mathbf{f}}_{\mathbf{x}}, \bar{\mathbf{f}}_{\mathbf{u}}$, and $\bar{\mathbf{f}}_\theta$ are known functions often derived from first principles and Θ is a possibly bounded parameter set. The control input is $\mathbf{u}_k = \pi(\mathbf{x}_k, \hat{\theta}_k)$, which is parameterized by $\hat{\theta}_k$.

- The parameter $\hat{\theta}_k$ is adapted by using either a Lyapunov function to gurantee that the closed-loop system [2] is stable.
- Model Reference Adaptive Control (MRAC) to make the system behave as a predefined stable reference model.

Adaptive control is typically limited to parameteric uncertainties and relies on a specific model structure. Moreover, adaptive control approaches tend to "overfit" to the latest observations and convergence to the true parameters is generally

[1] A parameteric model depends on a finite number of parameters that may have a physical interpretation or reflect our prior knowledge about the system structure in other ways. Parameteric uncertainty is the uncertainty in the model parameters.

[2] A general system is given by $\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$. If the input is given based on the state, i.e., $\mathbf{u}_k = \pi(\mathbf{x}_k)$, then the system is called a closed-loop system.

not guranteed. These limitations motivate the learning-based adaptive control approaches in Sec.

2.6.2 Robust Control

In contrast to adaptive control, which adapts to the parameters currently present, robust control finds a suitable controller for all possible disturbances, which can include unknown dynamics and noise, and keeps the controller unchanged after the initial design. Robust control is largely limited to linear, time-invariant (LTI) systems with linear nominal model $\bar{\mathbf{f}}(\mathbf{x}_k, \mathbf{u}_k) = \bar{\mathbf{A}}\mathbf{x}_k + \bar{\mathbf{B}}\mathbf{u}_k$ and unknown dynamics $\hat{\mathbf{f}}_k(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) = \hat{\mathbf{A}}\mathbf{x}_k + \hat{\mathbf{B}}\mathbf{u}_k + \mathbf{w}_k \in \mathbb{D}$, with $\mathbb{D}$ being known and bounded, and $\bar{\mathbf{A}}, \bar{\mathbf{B}}, \hat{\mathbf{A}}, \hat{\mathbf{B}}$ being static matrices of appropriate size. That is,

$$\mathbf{x}_{k+1} = (\bar{\mathbf{A}} + \hat{\mathbf{A}})\mathbf{x}_k + (\bar{\mathbf{B}} + \hat{\mathbf{B}})\mathbf{u}_k + \mathbf{w}_k. \tag{10}$$

Robust control design techniques yield controllers that are robustly stable for all $\hat{\mathbf{f}}_k \in \mathbb{D}$. Robust control can be extended to nonlinear system whose dynamics can be decomposed into a linear nominal model $\bar{\mathbf{f}}$ and a nonlinear function $\hat{\mathbf{f}}$ with known bound $\hat{\mathbf{f}} \in \mathbb{D}$. [3]

2.6.3 Robust Model Predictive Control (MPC)

3 Method

4 Experiment

5 Discussion

6 Conclusion

[3] Do not confuse $\mathcal{D}$, which denotes the dataset collected from the system, with $\mathbb{D}$, which represents the disturbances set.