Human-Feedback Shield Synthesis for Perceived Safety in Deep Reinforcement Learning

Daniel Marta, Christian Pek, Gaspar I. Melsión, Jana Tumova, and Iolanda Leite

***Abstract*—Despite the successes of deep reinforcement learning (RL), it is still challenging to obtain safe policies. Formal verification approaches ensure safety at all times, but usually overly restrict the agent's behaviors, since they assume adversarial behavior of the environment. Instead of assuming adversarial behavior, we suggest to focus on perceived safety instead, i.e., policies that avoid undesired behaviors while having a desired level of conservativeness. To obtain policies that are perceived as safe, we propose a shield synthesis framework with two distinct loops: (1) an inner loop that trains policies with a set of actions that is constrained by shields whose conservativeness is parameterized, and (2) an outer loop that presents example rollouts of the policy to humans and collects their feedback to update the parameters of the shields in the inner loop. We demonstrate our approach on a RL benchmark of Lunar landing and a scenario in which a mobile robot navigates around humans. For the latter, we conducted two user studies to obtain policies that were perceived as safe. Our results indicate that our framework converges to policies that are perceived as safe, is robust against noisy feedback, and can query feedback for multiple policies at the same time.**

***Index Terms*—Safety in HRI, Human Factors and Human-in-the-Loop, Reinforcement Learning.**

I. Introduction

DEEP reinforcement learning (RL) has proven to be successful in enabling autonomous robots to solve complex tasks in challenging and high-dimensional, real world environments. In RL, an agent interacts with an environment by executing (continuous) actions in it and collecting rewards at each time step. These experiences (tuples of states, actions and rewards) are used to train an optimal policy that maximizes the cumulative reward of the agent in the environment. In addition, for many RL agents, such as the one illustrated in Fig. 1, the trained policy needs to be *safe*. Broadly speaking, it needs to avoid undesired states, or undesired state sequences.

Safety in RL has been approached from various angles; one of them is to encode negative rewards that penalize undesired behaviors of the system during training, or to handcraft policies. However, reward engineering and policy tuning are tedious tasks and often not very effective to increase safety in a systematic way. Another (quite conservative) approach is to limit the set of applicable actions in each state of a Markov Decision Process when an immediate next action may lead to catastrophic failures [1]. *Shielding* does so with the use of formal methods [2], [3] and evaluates safety (i.e. satisfaction of a formal specification) of each action in the current situation. It allows the agent to only apply actions that are *provably safe* under all circumstances. In human-in-the-loop scenarios, the worst-case scenario corresponds to the overly conservative assumption that a human is an adversarial agent. The resulting policies are hence often overly conservative in real-world settings (see Fig. 1.c) if not leading to the freezing robot problem in [4].

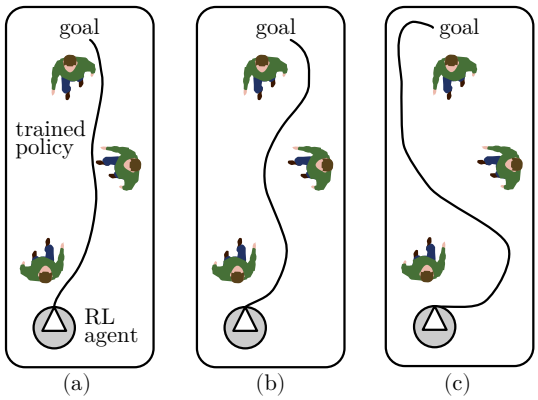


Fig. 1: In social navigation tasks, safe policies need to avoid colliding with obstacles in the environment. Yet, keeping too much safe distance to obstacles may lead in overly conservative behavior. Determining the safe distance that users perceive as safe requires human feedback when learning optimal RL policies.

In this paper, we focus on RL agents interacting with humans, and undesired behaviors defined as those that are not *perceived safe*. Instead of building shields that assume that humans are adversarial, we consider their feedback as a way to improve the trained policies while avoiding undesired behaviors (see Fig. 1.b). We propose to parameterize the conservativeness of shields and dynamically update these parameters through human feedback during learning.

A. Contributions

Our approach and contributions can be summarized as follows. We propose:

1) a parameterized shield synthesis for constraining the

Manuscript received: September, 9, 2021; Accepted October, 25, 2021.

This paper was recommended for publication by Editor Gentiane Venture upon evaluation of the Associate Editor and Reviewers' comments. This research has been carried out as part of the Vinnova Competence Center for Trustworthy Edge Computing Systems and Applications at KTH Royal Institute of Technology and partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

All of the authors are with the Division of Robotics, Perception and Learning, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, 114 28 Stockholm, Sweden. The authors are also affiliated with Digital Futures. Mail addresses: {dlmarta, pek2, gimp, tumova, iolanda}@kth.se

Digital Object Identifier (DOI): see top of this page.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2021.3128237, IEEE Robotics and Automation Letters

continuous action space of an RL agent with desired level of conservativeness,
2) an outer feedback loop in which humans evaluate perceived safety of trained policies, and
3) an update mechanism of the shield parameters to synthesize shields that reflect the humans' safety preferences even if the feedback is noisy.

We evaluate the effectiveness of our approach by parameterizing the shields in a Lunar landing benchmark scenario and demonstrate that the parameters of the shield significantly influence the perceived safety of trained policies. In a second scenario with a mobile robot that needs to reach goals while avoiding collisions with humans, we obtain trained policies that are perceived as safe by querying human preferences in two user studies in environments with different number of obstacles. We demonstrate that our framework converges to the optimal shield parameters for perceived safety even with noisy feedback of humans (inherently subjective), while accelerating learning during iterations.

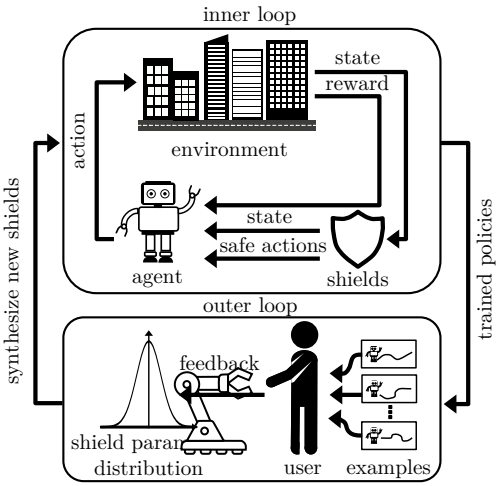


Fig. 2: Proposed shield synthesis framework for perceived safety. The inner loop trains the agent based on a provided set of shields. The outer loop generates examples of the trained policies and queries users for their feedback. Based on the feedback, the outer loop updates shield parameter distributions and synthesizes a new set of shields for the inner loop.

B. Related Work

Perceived safety is a key aspect to obtain comfortable and socially accepted Human-Robot Interaction (HRI) [5], where robot features (e.g, embodiment, gaze, speech) and abilities adhere to social norms. Human-aware motion planners [6]–[8] have been studied and demonstrated to make users feel more safe around robots [9], [10]. Larger and faster robots have been shown to lower comfort levels in humans [11]. Others have studied how a robot's proxemic behavior can influence the user experience in HRI settings [12]. Recent work has tried to incorporate human feedback within a reward shaping mechanism with HRI metrics such as frustration [13]. Unlike inverse reinforcement learning and other reward function synthesis approaches encoding human preferences [14]–[16], in this paper we maintain a simple and global objective, while preventing undesired actions.

Several frameworks address (perceived) safety in RL by deploying human intervention (human-in-the-loop) as a central theme. However, they differ in how extensive and qualitative this intervention should be. In [17], the authors propose human experts to concretely intervene on different stages of the training loop, thus avoiding catastrophic actions. Some works adapt actor-critic and policy gradient methods to incorporate human feedback, since the quality of trajectories can be evaluated through the advantage function [18]. These models drastically reduce the quantity of required human inputs. In addition, other frameworks are able to keep the human at the center of the agent's design [19], by encoding tasks in a number of subtasks and using behavioral primitives through shorter demonstrations. However, the human input is portrayed as concretely selecting sets of demonstrations as primitives, as opposed to our work which considers feedback on a small subset of rollouts of the trained policies.

Despite these works successfully including human feedback as a core theme of their framework, they usually query humans extensively throughout the learning, which can be a limitation in real world scenarios, e.g., due to the availability of humans or delays in their feedback. We aim to query humans in an online fashion as few as possible.

The concept of shielding has been introduced by [2], where high-level restrictions are modeled through linear temporal logic and converted to sets of safe actions. However, previous approaches considered static shielding strategies, either modeled *à priori* or devoid of any iterative human-feedback. To our knowledge, this is the first time an interactive high-level human feedback loop is included in shielding.

II. Shield Synthesis through User-feedback

Shielding in RL allows designers to restrict the feasible actions of the agent in order to achieve safe behaviors according to a given safety specification, e.g., the agent needs to keep a certain safe distance to obstacles. These shields generate a set of safe actions that restricts the policy to follow the safety specification. Yet, these shields often overly restrict the action space, leading to conservative behaviors. We suggest to 1) parameterize the conservativeness of shields, e.g., the minimum required distance to obstacles; and 2) to determine the optimal parameters to achieve a desired level of conservativeness for a given task or application by incorporating human feedback.

As a result, we combine the benefits of shielding the RL agent and having a human in the learning loop to determine optimal parameters of shields to obtain policies that are perceived as safe. To apply our framework to existing RL setups, we build a feedback loop around a given RL setup that uses shielding. Fig. 2 illustrates our proposed shield synthesis framework for perceived safety that is divided into an inner and an outer loop.

The inner loop of the framework contains the given RL setup and considers a set of designer-defined shields. The initial shield parameters may be chosen to overly restrict the agent's set of actions to achieve safety. The agent chooses one of the safe actions and applies it in the environment. The policy is trained for a pre-defined number of episodes with the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2021.3128237, IEEE Robotics and Automation Letters

current set of shields. The trained policy is then forwarded to our proposed outer loop.

In our added outer loop, we generate example rollouts of the trained policies. These examples are shown to users, e.g., through crowdsourcing, to get feedback on the policies with respect to their perceived safety given the current set of shield parameters. We use this human feedback to update internal shield parameter distributions, which model the guess of the optimal shield parameters. With the updated distributions in mind, we synthesize a new set of shields with updated shield parameters and provide it to the inner loop. Both loops alternate until the change of shield parameter distributions is below a pre-defined threshold. As a result, we obtain the optimal shield parameters and policy that users perceive as safe, since human feedback converged.

A. Parameterized shields

We consider continuous deep reinforcement learning setups with state spaces $\mathcal{S} \subseteq \mathbb{R}^n, n \in \mathbb{N}_+$, and action spaces $\mathcal{A} \subset \mathbb{R}^m, m \in \mathbb{N}_+$. The overall goal of the RL agent is to determine a policy $\pi : \mathcal{S} \to \mathcal{A}$ that maps a given state s to an action a. The optimal policy $\pi^\star$ maximizes the expected cumulative reward of the agent.

We model shields as functions $\Psi : \mathcal{S} \times \mathbb{R} \to \mathrm{pow}(\mathcal{A})$ (where pow denotes the power set) that map the agent's state and a shield parameter θ to a set of safe actions. The set of safe actions is obtained as $\mathcal{A}_{\mathrm{safe}}(s, \theta) := \Psi(s, \theta) \subseteq \mathcal{A}$. For instance, θ might encode the minimum required safe distance to obstacles in meters and the set of safe actions $\mathcal{A}_{\mathrm{safe}}$ ensures that the agent adheres to the required safe distance. Our framework can consider a set of $N_\Psi \in \mathbb{N}_+$ shields $\Psi_i, i \in \{1, ..., N_\Psi\}$, at the same time. In this case, we obtain the safe actions as

$$\mathcal{A}_{\mathrm{safe}}(s, \theta) := \bigcap_{i=1}^{N_\Psi} \Psi_i(s, \theta_i), \tag{1}$$

where $\theta = (\theta_1, \ldots, \theta_2)$.

B. Shield and feedback distributions

We use distinct probability distributions to model the shield parameters and the human feedback. This choice is motivated by the fact that users might provide uncertain feedback, since their ratings are subjective. With separate distributions, we are able to provide estimates independently of the uncertainty in the other distribution. Moreover, large shifts in the human distribution will not immediately cause large shifts in the shield distribution, resulting in smoother parameter changes when restricting the number of samples taken from the human distribution. The inner loop of our framework considers the shield parameter distribution $\theta_i \sim f_{\theta_i}$, where f_{θ_i} is a probability distribution that models shield parameter θ_i. The shield parameter distribution is used to directly synthesize new shields $\Psi_i, i \in \{1, ..., N_\Psi\}$ by sampling candidate parameters θ_i from f_{θ_i}.

The outer loop translates the user feedback for parameter i into parameter updates using the distribution f_{h_i}. To simplify the notation, we consider the parameter update for a single parameter, allowing us to omit the index i. All of the following steps are similarly applied to all parameters θ_i.

The distributions, f_θ and f_h, are differently updated throughout the learning phase. While f_h is inferred from the empirical data of the users' feedback using estimators for the mean and variance, f_θ is sequentially updated using Bayesian inference from samples generated by f_h. Both f_θ and f_h can be represented by any mixture model, but for simplicity and to take advantage of closed form solutions (conjugate priors to infer the posterior of f_θ), we consider them to be Normal distributions, i.e., $f_\theta = \mathcal{N}(\mu_\theta, \sigma_\theta^2)$ and $f_h = \mathcal{N}(\mu_h, \sigma_h^2)$ with means μ_θ, μ_h and variances $\sigma_\theta^2, \sigma_h^2$.

C. Mapping human feedback

In the beginning of our framework, we initialize μ_θ and μ_h with initial guesses or given initial values. By showing users example rollouts of the current trained policies, we acquire feedback for the parameter θ from humans in numerical or symbolic form. For instance, one might use the following symbols $g \in \mathcal{H} = \{\text{very unsafe}, \ldots, \text{fine}, \ldots, \text{very safe}\}$, where unsafe translates into the wish of decreasing (or increasing) the parameter, fine to not update the parameter, and very unsafe the wish of increasing (or decreasing) the parameter. Nevertheless, developers may also design other symbols and mappings in our approach. To integrate different symbolic representations in our approach, we introduce a mapping function that maps the user feedback into numerical values for updating the feedback distribution:

$$\mathrm{map}(g_j) = \begin{cases} \mu_h - \frac{|\mathcal{H}|\sigma}{2} & \text{if } g_j = \text{very unsafe}, \\ \vdots & \vdots \\ \mu_h & \text{if } g_j = \text{fine}, \\ \vdots & \vdots \\ \mu_h + \frac{|\mathcal{H}|\sigma}{2} & \text{if } g_j = \text{very safe}. \end{cases} \tag{2}$$

Note that in between those symbols, one can use an arbitrary fine resolution depending on the application. Moreover, one can also incorporate other mapping functions in our approach.

To update the feedback distribution, we first collect a dataset of user feedback samples $\mathcal{D} = \{g_1, ..., g_j\}, g_j \in \mathcal{H}, j \in \{1, \ldots, N_{\mathrm{user}}\}$, where $N_{\mathrm{user}} \in \mathbb{N}_+$ is the number of collected samples. Afterwards, we update the human parameter distribution f_h by computing the new mean ${}^u\mu_\theta$ and variance ${}^u\sigma_h^2$ as estimators from $\mathcal{D}$ with:

$${}^u\mu_h = \frac{1}{N_{\mathrm{user}}} \sum_{j=1}^{N_{\mathrm{user}}} \mathrm{map}(g_j), \tag{3}$$

$${}^u\sigma_h^2 = \max\left(\frac{1}{N_{\mathrm{user}}} \sum_{j=1}^{N_{\mathrm{user}}} \big(\mathrm{map}(g_j) - ({}^u\mu_h^2) \big), \sigma_{\min}^2 \right), \tag{4}$$

where $\sigma_{\min}^2$ is the minimum considered human uncertainty, provided to our framework by users as a design parameter.

D. Update of the shield distribution

After acquiring the human feedback, we use the updated human feedback distribution f_h as a generative model to

improve our inner loop shield distribution f_θ that is used to synthesize new shield parameters for the next run of the inner loop. This improvement is done by generating a pre-defined amount of samples from f_h. Since we are exemplifying with Normal distributions for both, we can use Bayesian updates.

Let us consider a dataset of generated samples $\mathcal{G}$ from the updated human distribution f_h. We are interested in estimating the true mean μ (which is equal to the mean of f_θ and f_h after convergence to the human feedback) and assume σ to be known and small, e.g., provided by designers. The likelihood of the true parameter distribution is given by:

$$p(\mathcal{G}|\mu,\sigma^2) = \prod_{j=1}^{|\mathcal{G}|} p(x_j|\mu,\sigma^2), x_j \in \mathcal{G} \tag{5}$$

and the Bayesian update of f_θ is given by the posterior

$$\hat{p}_\theta(\hat{\mu}_\theta|\mathcal{G},\hat{\sigma}_\theta^2) \propto p(\mathcal{G}|\mu,\sigma^2)p_\theta(\mu|\mu_\theta,\sigma_\theta^2), \tag{6}$$

where μ_θ and σ_θ^2 are the prior and $\hat{\mu}_\theta$ and $\hat{\sigma}_\theta^2$ the posterior means and variances. Since we are using Gaussian distributions, we can compute $\hat{\mu}_\theta$ and $\hat{\sigma}_\theta^2$ in closed-form [20]:

$$\hat{\sigma}_\theta^2 = \frac{1}{\frac{n}{\sigma} + \frac{1}{\sigma_\theta^2}}, \tag{7}$$

$$\hat{\mu}_\theta = \hat{\sigma}_\theta^2\left(\frac{\mu_\theta}{\sigma_\theta^2} + \frac{n\bar{x}}{\sigma^2}\right), \tag{8}$$

where $n = |\mathcal{G}|$ and $\bar{x}$ is the mean of the samples in $\mathcal{G}$.

Since querying humans for feedback might be costly (e.g., time-wise), our approach also allows for querying feedback on multiple policies at the same time. These multiple policies can be trained with different shield parameters sampled from f_{θ_i}. Given independent feedback sample datasets $D_1, ..., D_j, j \in \{1, \ldots, N_{\text{policies}}\}$, obtained through different policies with shield values sampled from f_{θ_i}, we may jointly assess them as $\mathcal{D} = D_1 \cup D_2 \cup ... \cup D_j, j \in \{1, \ldots, N_{\text{policies}}\}$. Thus evaluating $p(\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_{N_{\text{policies}}}|\mu,\sigma^2)$ is equivalent to $p(\mathcal{G}|\mu,\sigma^2)$. In this way, designers can accelerate the convergence of our algorithm, since we can evaluate N_{policies} policies in each outer loop.

E. Algorithm and convergence criterion

Algorithm 1 summarizes the steps of our proposed human-feedback shield synthesis for perceived safety, that is comprised by two distinct loops as illustrated in Fig 2. The outer loop, which requests feedback from humans to update shield distribution f_h, and an inner loop where policy π interacts with the environment, restricted by a shield $\Psi_i(s,\theta_i)$ with shield parameters sampled from f_θ. We repeat both loops, gathering feedback (outer loop) from policies generated in the inner loop, and training with synthesised shields until convergence (inner loop), generating new policies which require more feedback until convergence.

To allow designers more control on the convergence of our algorithm and the frequency of querying humans, we further introduce three hyper-parameters $\beta \in \mathbb{R}_{\geq 0}$, $\omega \in \mathbb{N}$ and $\zeta \in \mathbb{N}$ to denote convergence thresholds for the human feedback, frequency of training epochs before showing a new policy to the user, and global training steps, respectively. To assess whether the human and shield feedback converged, we use the KL-divergence between the human feedback and shield distributions as a stopping criteria (see line 24 of Alg. 1) The threshold β describes the minimum similarity in the KL-divergence between human feedback and shield distributions as a stopping criteria. Note that setting a lower boundary for the KL-divergence sets a trade-off between allowing for high-level human feedback to be contradictory and avoiding infinite learning loops when consensus of feedback is reached. The parameter ζ (see line 12 of Alg. 12), functions as maximum number of learning steps/episodes/epochs to train our policy π before showing its rollouts as examples to humans. Finally, $\omega \in \mathbb{R}$ (see line 26 of Alg. 1) represents the number of learning epochs of the agent's policy π^*, commonly used in a DRL loop. Whenever there is a change of shield parameters, the state-action space distribution changes, possibly resulting in a completely new policy.

Algorithm 1: Shield synthesis for perceived safety through human feedback

1 **Input**:mapping function map, divergence threshold β, learning steps ζ, number of learning epochs ω;
2 **Output**: Optimal policy $\pi^\star$ that is perceived as safe;
3 $t \leftarrow 0$;
4 $f_\theta, f_h \leftarrow$ initializeShieldDistributions();
5 env, $s_t \leftarrow$ initializeEnvironment();
6 $\Pi \leftarrow$ initializePolicies();
7 converged $\leftarrow False$;
// Outer loop
8 **do**
// Inner loop
9 **for** π_i ***in*** Π **do**
10 $\theta_i \leftarrow \text{sample}(f_\theta)$;
11 ${}^i\mathcal{A}_{\text{safe}} \leftarrow \Psi(s_t, \theta_i)$;
12 **for** 1 **to** ζ **do**
13 $a_t \leftarrow \text{getSafeAction}(\pi_i, {}^i\mathcal{A}_{\text{safe}})$;
14 $s_{t+1}, r_t \leftarrow \text{env.step}(a_t)$;
15 $\pi_i \leftarrow \text{train}(\pi_\text{i}, \text{s}_\text{t}, \text{a}_\text{t}, \text{s}_{\text{t+1}}, \text{r}_\text{t})$;
16 $s_t \leftarrow s_{t+1}$;
17 $t \leftarrow t + 1$;
18 **if** not converged **then**
19 $\mathcal{D} \leftarrow \text{collectFeedback}(\pi_\text{i})$;
20 $\mathcal{G} \leftarrow \text{map}(\mathcal{D})$;
21 $f_h \leftarrow \text{fitModel}(f_h, \mathcal{G})$;
22 $\mathcal{Z} \leftarrow \text{generateSamples}({}^u f_h)$;
23 $f_\theta \leftarrow \text{computePosterior}(f_\theta, \mathcal{Z})$;
24 converged $\leftarrow \text{KL}(f_h, f_\theta) \leq \beta$;
25 $\Psi(:, \theta) \leftarrow \text{generateShield}(f_\theta)$;
26 **while** $t < \omega$;
27 **return** π^*

III. Experimental Results

We demonstrate the effectiveness of our human-feedback shield synthesis for perceived safety in different deep RL

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2021.3128237, IEEE Robotics and Automation Letters

applications. Each of our experiments focuses on different aspects of our algorithm:

1) in Sec. III-B, we focus on analyzing the continuous generation of shields and the convergence of our algorithm;
2) in Sec. III-C, we investigate the effect of contradictory human feedback on the convergence of our algorithm;
3) in Sec. III-E, we demonstrate how to query human feedback for multiple policies at the same time; and
4) in Sec. III-F, we highlight the update of the shield distribution with human feedback and how to sample new shield parameters for the next loop of our algorithm.

For experiments 1 and 2, we use the synthetic benchmark of safe Lunar landing, since we can control various parameters. For the last two experiments 3 and 4, we use a social navigation environment (similar to Fig. 1) and query real humans in an online user study. Videos of our experiments can be found in the media attachment of this paper.

For all experiments we use similar architectures and hyperparameters. Both environments are tested with PPO [21] [22]. In the *Safe Lunar Landing* experiment, we use three densely-connected neural-network layers, with $\{512, 256, 64\}$ neurons, while in the *Social Navigation* experiment we use two layers with $\{512, 512\}$ neurons. To adapt to a continuous action space, both actors have a final layer at the output corresponding to one *tanh* per action.

A. Safe Lunar landing - Problem setup

The safe Lunar landing environment (see Fig. 3) is a slightly modified version of the continuous Lunar landing problem [23] and serves as a benchmark for our framework to demonstrate the update of shield parameters during training. In this setup, we do not query real humans but created an artificial population of oracles which evaluate the perceived safety of the agent. The population contains oracles, which always give perfect feedback in terms of perceived safety of the spaceship until the shield distribution converges, and imperfect oracles which give random feedback. The continuous state space is $\mathcal{S}_{\text{lunar}} = \mathbb{R}^8 \times [-1, 1]$ with horizontal and vertical coordinates, speed, angular speed, and leg contacts. We consider the safety as not over-using the main engine of the Lunar, i.e., usage greater than 85% (which corresponds to a maximum input of 0.7) power. Therefore, we added a ninth state variable as a warning state that indicates over-use of the main engine and is used by our oracle to evaluate safety. The continuous action is $a = (a_1, a_2)^T \in \mathcal{A}_{\text{lunar}} = [-1, 1]^2$, where a_1 fires the main engine, and a_2 the lateral engines. A negative reward of -0.3 is given when the agent uses the engine. We add an additional rule to make the problem slightly harder and showcase potential shortcomings of training without safety constraints. A linear penalty of between $[-2, 0]$ is given when using the main engine action, i.e., if $|a_1| > 0.7$, then an extra negative reward $r_e = -2\left(\frac{|a_1|-0.7}{0.3}\right)$ is provided. We trained the agent through Open AI gym's API [23] with parallel environments.

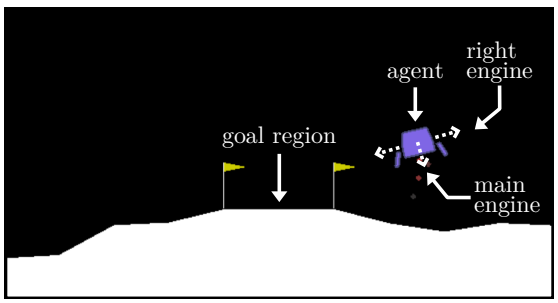


Fig. 3: In the lunar landing example, the agent needs to land a space ship with three engines (main, right and left) in the designated goal region without colliding and with correct orientation of the ships' feet.

B. Safe Lunar landing - Convergence of shield updates

To demonstrate the parameter update of our framework, we use a simple shield in the safe lunar landing environment:

$$\Psi_{\text{lunar}}(s, \theta) = [-\theta, \theta] \times [0, 1], \tag{9}$$

with the parameter $\theta \in [0, 1]$. This shield constrains the use of the main engine. The oracle receives rollouts of the policy and evaluates the warning state. It perceives the policy as safe if the agent does not over-use the main engine. When the agent over-uses the main engine, it provides the feedback to reduce engine power, otherwise to keep the policy. The oracle is queried each training iteration, i.e., $\zeta = 1$.

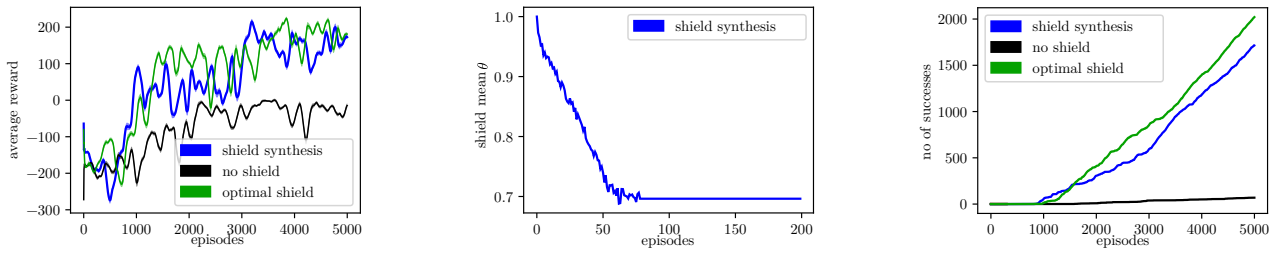
Figure 4a shows the learning curves when using no shields, continuously updating the shield, and learning with the optimal shield parameter from the beginning. We validated our results by retraining with different random seeds. Our results show that using a simple shield already significantly accelerates learning of the agent and respects the preferences of the artificial population. The continuously updated and optimal shield policies converge to the same average reward.

The shield parameter update is shown in Fig. 4b. Through the continuous feedback of the oracles, θ is lowered over time until converging to the optimal parameter at $\theta \approx 0.7$ after around 300 episodes. Although the shield is constantly updated, the agent is able to quickly learn the task. The baseline (no shield) policy of the agent is heavily over-using the main engine of the lunar (see Fig. 4a). In contrast, the policy that is perceived as safe by our oracles never over-uses the engine while successfully completing the task (landing in the goal region) more often (see Fig. 4c).

C. Safe Lunar landing - Contradictory human feedback

Human feedback may be noisy or even contradictory. To estimate how such feedback impacts the convergence of our algorithm, we varied the percentage of random oracles within our artificial population. Each oracle within the population provides feedback regarding the engine of the spaceship from three options, e.g, $\mathcal{H} = \{\text{underused}, \text{correctly used}, \text{overused}\}$. True oracles will always correctly identify when the engine is overused, while random oracles will offer random feedback. To assess the convergence of shield distributions, we created four populations: (1) 10 true oracles and zero random oracles; (2) 9 true oracles and 1 random oracle (10% randomness); (3) 7 true oracles and 3 random oracles (30% randomness); and (4) 5 true oracles and 5 random oracles (50% randomness).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2021.3128237, IEEE Robotics and Automation Letters



(a) (b) (c)

Fig. 4: Results of the synthetic benchmark. (a) Average collected reward of the agent over 5000 episodes without shielding (black), with the proposed shield synthesis (blue), and with the optimal shield from the start (green). The standard deviation is shown as a shaded region for each graph. (b) Update of the shield parameter distribution's mean over episodes. (c) Cumulative number of successful landings without and with the proposed shield synthesis.

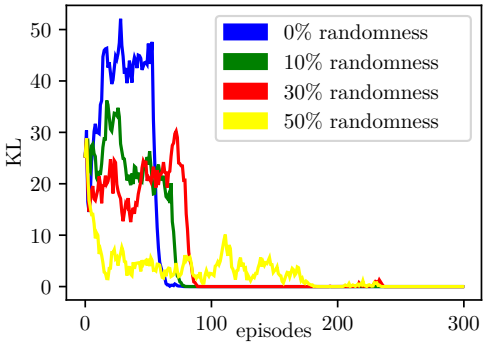


Fig. 5: KL-divergence between f_h and f_θ used as a convergence criteria to stop asking humans for feedback

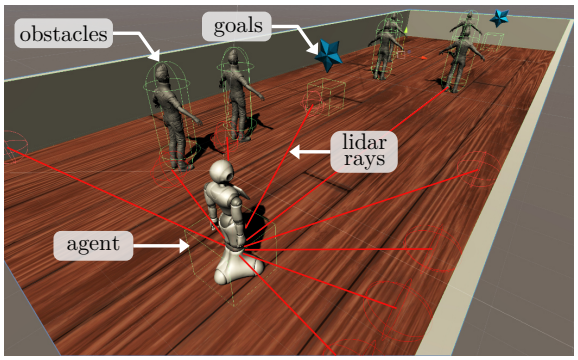


Fig. 6: Environment of the social navigation task. The agent needs to reach two randomly placed goals while avoiding moving humans. The agent can detect both the obstacles and goals using nine lidar rays.

Figure 5 illustrates how the KL-divergence evolves between f_h and f_θ when setting a lower boundary of $\beta = 0.05$. The more random feedback a population provides, the more episodes are needed for convergence. The higher margins at the beginning for population (2) are due to a larger shift of f_h, since more oracles are agreeing to decrease the shield value. Even when considering the unrealistic contradictory feedback of population (4), where 50% of the feedback is random, our algorithm still converges and only requires three times more episodes than the perfect population (1).

D. Social navigation - Problem setup

We designed a social navigation environment and conducted two user studies. In the environment, the agent (a mobile robot) needs to reach two random goals while avoiding humans (see Fig. 6). To generate safe policies, we consider the Social Force Model [24] [25] as a starting point. In this model, obstacles are represented as repelling directed force fields that influence the actions of the agent. For instance, if the agent is approaching the boundary of the force field, it will be pushed away at the same time. Interested readers are referred to [24] for more details.

To evaluate perceived safety, we want to estimate how strong the force field should be, i.e., the shield parameter encodes how much the shield considers the full interaction force of the Social Force Model (SFM). The state space is comprised by the agent's position and velocity, the velocity of other obstacles in the environment, and nine rays of a lidar-like sensor, commonly used in navigation robots. The agent's actions are composed of the accelerations in x- and y-directions, representing the driving force and a third action proportional to the interaction force of the SFM. For each ray, the agent detects either a goal, one of the humans or walls. The rays are one-hot encoded in addition to the distance between the robot and a specific element. In total, there are 9 rays opening in a field of view (FOV) of 200^o, see Fig. 6.

The agent receives a positive reward of 1 and 1.5 when reaching the middle and final goals (generated randomly in both halves of the room), respectively, while a collision with an obstacle or a wall yields a negative reward of -1. Additionally, a small negative reward of $\frac{-1}{5000}$ is given per step. The social navigation experiment was designed from scratch and implemented in Unity. To train our agents, we took full advantage of the communication pipelines available in the Unity ML-Agents API [26].

E. Social Navigation - Human feedback on multiple policies

In this experiment, we look at obtaining real human feedback instead of having an artificial population. In the social navigation task, we consider shields that remove actions that will result in a conflict with the force field of the SFM. The shield depends on the current state of the agent and constraints the action to not conflict with a given force field intensity interval. The shield parameter $\theta \in [0, 1]$ denotes the center of this interval, where the values 0 and 1 correspond to no or full force field, respectively. The design parameter $\chi \in [0, \min(|0 - \theta|, |1 - \theta|)]$ is the width of the interval. For

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2021.3128237, IEEE Robotics and Automation Letters

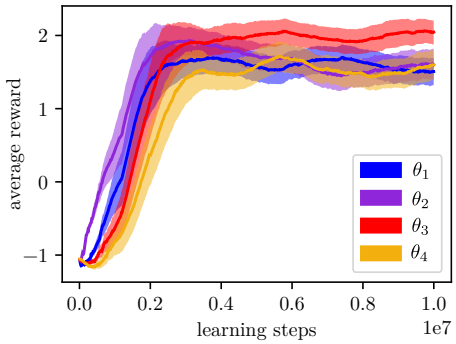


Fig. 7: Average reward of the agents with different shields over the learning steps. The standard deviation is shown as a shaded region for each graph.

instance, for $\theta = 0.25$ and $\chi = 0.25$, the agents needs to account for $[\theta - \frac{\chi}{2}, \theta + \frac{\chi}{2}]$ of the force field at any time.

Since our approach allows designers to query feedback for multiple policies at the same time, we sampled four initial and distinct shield values (more details in Sec. III-F) which are used to train four policies in parallel. We trained the four different policies with $\chi = 0.25$ and different shield parameters: $\theta_1 = 0.125, \theta_2 = 0.375, \theta_3 = 0.625$, and $\theta_4 = 0.875$ to cover the whole parameter space. Fig. 7 shows the learning curves of the four policies.

We rendered videos of 30 randomly selected rollouts of each policy for two different scenarios in the environment in Fig. 6: 3 and 6 moving obstacles (humans). These were divided into a series of 6 experiments containing 20 videos each (5 per parameter interval). 10 participants were recruited for every series, and they could decide on how many series they want to participate in. The participants were asked to rate the perceived safety taking into account the robot social navigation for each video on a 7-point Likert scale from *Very Unsafe* (rating 0) to *Very Safe* (rating 6). This resulted in 60 participants for each study, generating 300 data points for each shield parameter, in each of the studies.

The study was run online using Amazon Mechanical Turk (AMT). In total, there were 92 unique participants (59 males, 33 female and none of other gender identities). Their age ranged from 23 to 65 years old, with a median of 34; the majority were in or had completed college education (N=78) and came from the US (N=71). 62 participants reported to have never or only seen robots in media, 14 to have interacted with one robot before, and 2 to do it on a regular basis.

The perceived safety results from the user studies can be seen in Fig. 8 and Table I. One-way ANOVA resulted in statistically significant results in both studies (3 obstacles: $F(3, 1196) = 88.01, p < .001$, Fig. 8a; 6 obstacles: $F(3, 1196) = 28.71, p < .001$, Fig. 8b), showing that people have significant different perceptions of safety between the different policies. In the case of 3 obstacles in the environment, the trajectories from θ_4 were the ones rated as the safest ($M = 4.83, SD = 1.07$) and those from θ_2 as the least safe ($M = 2.83, SD = 1.89$). In Fig. 8a it can be observed a notable difference between the interquartile ranges (IQR) from the different shields, suggesting that participants agreed more on their ratings of θ_3 and θ_4. This is also true for the experiment with 6 obstacles, where θ_3 presents all answers except outliers within the upper range of the scale. Then, in the case of 6 obstacles, the trajectories perceived as safest by the participants were those from θ_3 ($M = 4.47, SD = 1.37$).

TABLE I: Mean and standard deviation of the perceived safety ratings.

Safety Ratings		θ_1	θ_2	θ_3	θ_4
3 obstacles	Mean	3.14	2.83	4.00	4.83
	SD	1.95	1.89	1.57	1.07
6 obstacles	Mean	3.25	3.63	4.47	4.09
	SD	1.94	1.76	1.37	1.75

Our results show that humans are able to evaluate the perceived safety of example rollouts of four policies with different shield parameters. Thus, we can use this information to update our shield distribution and to determine the next four parameter samples in the next loop of our algorithm.

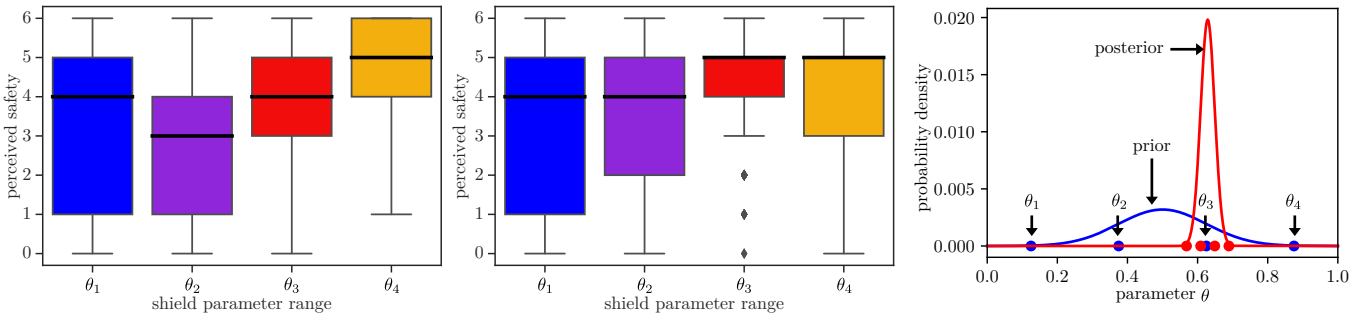
F. Social Navigation - Shield distribution update

All policies converged to similar average returns, suggesting any of them successfully accomplishes the task safely, since collisions are heavily punished. However, the user feedback showed that perceived safety per policy was clearly different, and their continuous feedback improves perceived safety in a robotic real-world social scenario. For simplicity, we considered a Normal prior $f_\theta \sim \mathcal{N}(\mu_\theta = 0.5, \sigma_\theta^2 = 0.15625)$ over shield parameters. We sampled four different shield parameters and used them to train the four different policies. To prevent duplication of values when sampling, we used a sampling mechanism based on the mean and standard deviation, using one and three standard deviations from left to right as pictured on Fig. 8c. Both datasets with 3 and 6 obstacles were mapped and used to update f_θ with posteriors $f_\theta \sim \mathcal{N}(\mu_\theta = 0.62891, \sigma_\theta^2 = 0.00041)$ and $f_\theta \sim \mathcal{N}(\mu_\theta = 0.56001, \sigma_\theta^2 = 0.00037)$, for 3 and 6 obstacles respectively. This result shows that one update step of our algorithm already significantly narrowed down the search space for the optimal parameters. In the next loop of our algorithm, we would again sample four new shield parameters amd acquire feedback until the desired KL threshold β has been reached.

IV. Conclusions

This paper presents a framework to obtain RL policies that are perceived as safe by humans, i.e., policies that avoid undesired behaviors while having a desired level of conservativeness. The inner loop of our framework trains a policy that is safeguarded by a set of shields that limit the action space of the agent. The outer loop generates example rollouts of the trained policies and collects feedback from humans. The feedback is then used to update the parameters of the inner loop shields to reflect human preferences. To this extent, we formulate two distinct shield distributions, one from the human's feedback of policy rollouts, and another which infers the true value of these parameters. Our algorithm provides users with control about the convergence of the shield parameters and the frequency of querying humans. These parameters help to overcome issues

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2021.3128237, IEEE Robotics and Automation Letters



(a) Results for 3 obstacles. **(b)** Results for 6 obstacles. **(c)** Prior and posterior distributions.

Fig. 8: (a) and (b) Perceived safety responses from social navigation experiments. We use a 7-point Likert scale, where 0 is 'Very Unsafe' and 6 'Very Safe'. (c) Prior and posterior distributions of f_θ after one loop.

such as noisy feedback and querying for feedback when the state-action space distributions change.

In various experiments with synthetic and real human feedback, we demonstrated that our proposed algorithm effectively synthesizes shields to obtain policies that are perceived as safe. We showed that the algorithm converges even if the feedback is contradictory by adding random oracles in our synthetic benchmark. To reduce the amount of feedback queries, we allow designers to query feedback for multiple policies at the same time. We demonstrated this property in a user study of our social navigation task to obtain feedback from real humans. Our user study shows that humans are able to evaluate the perceived safety from visual inspection of example rollouts and that the feedback can be used to update the shield distribution. Our algorithm can incorporate any sampling strategy to determine the next set of parameters for the training of multiple policies.

For future work, we want to test our framework on additional social environments, different shield distributions or models, and consider other desired properties of policies, such as social acceptance or comfort. Additionally, we would like to explore the issue of scalability of human feedback when capturing preferences while using our approach.

REFERENCES

[1] T. M. Moldovan and P. Abbeel, "Safe exploration in markov decision processes," in *ICML*, 2012.

[2] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe reinforcement learning via shielding," in *Proc. of the AAAI Conf. on Artificial Intelligence*, vol. 32, no. 1, 2018.

[3] B. Könighofer, F. Lorber, N. Jansen, and R. Bloem, "Shield synthesis for reinforcement learning," in *Leveraging Applications of Formal Methods, Verification and Validation: Verification Principles*, T. Margaria and B. Steffen, Eds., 2020, pp. 290–306.

[4] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2010, pp. 797–803.

[5] P. A. Lasota, T. Fong, and J. A. Shah, "A survey of methods for safe human-robot interaction," *Foundations and Trends in Robotics*, vol. 5, no. 4, p. 261–349, 2017.

[6] E. A. Sisbot, L. F. Marin-Urias, X. Broquère, D. Sidobre, and R. Alami, "Synthesizing robot motions adapted to human presence," *Int. Journal of Social Robotics*, vol. 2, no. 3, p. 329–343, 2010.

[7] S. Bansal, A. Bajcsy, E. Ratner, A. D. Dragan, and C. J. Tomlin, "A hamilton-jacobi reachability-based framework for predicting and analyzing human motion for safe planning," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 7149–7155.

[8] A. Bajcsy, S. Bansal, E. Ratner, C. J. Tomlin, and A. D. Dragan, "A robust control framework for human motion prediction," *IEEE Robotics and Automation Letters*, vol. 6, no. 1, pp. 24–31, 2020.

[9] P. A. Lasota and J. A. Shah, "Analyzing the effects of human-aware motion planning on close-proximity human–robot collaboration," *Human Factors*, vol. 57, no. 1, p. 21–33, 2015.

[10] D. Kulic and E. Croft, "Anxiety detection during human-robot interaction," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2005, p. 616–621.

[11] J. T. Butler and A. Agah, "Psychological effects of behavior patterns of a mobile personal robot," *Autonomous Robots*, vol. 10, no. 2, p. 185–202, 2001.

[12] Y. Kim and B. Mutlu, "How social distance shapes human–robot interaction," *Int. Journal of Human-Computer Studies*, vol. 72, no. 12, p. 783–795, 2014.

[13] S. Krening, "Newtonian action advice: Integrating human verbal instruction with reinforcement learning," *arXiv preprint arXiv:1804.05821*, 2018.

[14] D. Sadigh, A. D. Dragan, S. Sastry, and S. A. Seshia, "Active preference-based learning of reward functions." in *Robotics: Science and Systems*, 2017.

[15] S. Reddy, A. Dragan, S. Levine, S. Legg, and J. Leike, "Learning human objectives by evaluating hypothetical behavior," in *Int. Conf. on Machine Learning*, 2020, pp. 8020–8029.

[16] E. Ratner, D. Hadfield-Menell, and A. Dragan, "Simplifying reward design through divide-and-conquer," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[17] W. Saunders, G. Sastry, A. Stuhlmüller, and O. Evans, "Trial without error: Towards safe reinforcement learning via human intervention," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, 2018, pp. 2067–2069.

[18] J. MacGlashan, M. K. Ho, R. Loftin, B. Peng, G. Wang, D. L. Roberts, M. E. Taylor, and M. L. Littman, "Interactive learning from policy-dependent human feedback," in *Int. Conf. on Machine Learning*. PMLR, 2017, pp. 2285–2294.

[19] M. Rahtz, J. Fang, A. D. Dragan, and D. Hadfield-Menell, "An extensible interactive interface for agent design," *arXiv preprint arXiv:1906.02641*, 2019.

[20] K. P. Murphy, "Conjugate bayesian analysis of the gaussian distribution," *def*, vol. 1, no. $2\sigma 2$, p. 16, 2007.

[21] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[22] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Int. Conf. on machine learning*. PMLR, 2015, pp. 1889–1897.

[23] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[24] M. Moussaïd, D. Helbing, S. Garnier, A. Johansson, M. Combe, and G. Theraulaz, "Experimental study of the behavioural mechanisms underlying self-organization in human crowds," *Proc. of the Royal Society B: Biological Sciences*, vol. 276, no. 1668, p. 2755–2762, 2009.

[25] D. Helbing and P. Molnár, "Social force model for pedestrian dynamics," *Physical Review E*, vol. 51, no. 5, p. 4282–4286, 1995.

[26] A. Juliani, V.-P. Berges, E. Teng, A. Cohen, J. Harper, C. Elion, C. Goy, Y. Gao, H. Henry, M. Mattar, and D. Lange, "Unity: A general platform for intelligent agents," 2020.