Using Reinforcement Learning to Create Control Barrier Functions for Explicit Risk Mitigation in Adversarial Environments

Edvards Scukins and Petter Ögren

***Abstract*— Air Combat is a high-risk activity carried out by trained professionals operating sophisticated equipment. During this activity, a number of trade-offs have to be made, such as the balance between risk and efficiency. A policy that minimizes risk could have very low efficiency, and one that maximizes efficiency may involve very high risk.**

In this study, we use Reinforcement Learning (RL) to create Control Barrier Functions (CBF) that captures the current risk, in terms of worst-case future separation between the aircraft and an enemy missile. CBFs are usually designed manually as closed-form expressions, but for a complex system such as a guided missile, this is not possible. Instead, we solve an RL problem using high fidelity simulation models to find value functions with CBF properties, that can then be used to guarantee safety in real air combat situations. We also provide a theoretical analysis of what family of RL problems result in value functions that can be used as CBFs in this way.

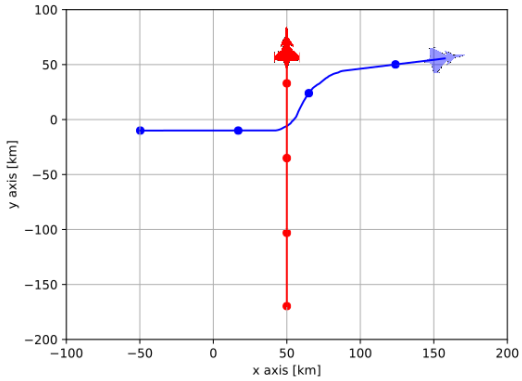
The proposed approach allows the pilot in an air combat scenario to set the exposure level deemed acceptable and continuously monitor the risk related to his/her own safety. Given input regarding acceptable risk, the system limits the choices of the pilot to those that guarantee future satisfaction of the provided bound.

***Index Terms*— Control Barrier Functions, Reinforcement Learning, Safe Exploration**

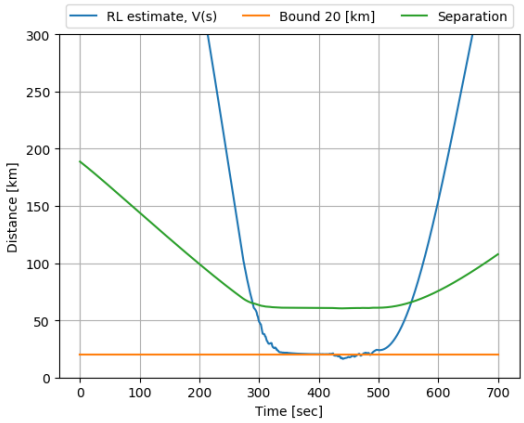
I. Introduction

In air combat, constant trade-offs have to be made between reaching the mission objectives and being exposed to risk. The mission objectives might be to gather information regarding an adversarial aircraft or to fire a weapon towards an aircraft that threatens friendly forces in the air or on the ground. In this work, we consider a scenario where an adversarial aircraft is launching a missile towards the defending aircraft where the proposed system guarantees future separation while not limiting the options of the pilot too much.

The problem above might be posed like an end-to-end reinforcement learning (RL) problem. However, we are looking for an approach that is more transparent to the human operator, the pilot, in a way that is illustrated in Figure 1. The blue aircraft of the scenario has two conflicting objectives, first, keeping the risk of the own aircraft low, which is typically achieved by a large separation of the two aircraft, and second, reaching a particular position, which cannot be achieved if that separation is too large. Thus, the pilot needs to constantly balance risk exposure with the mission progress. Furthermore, the proper choice of acceptable risk might vary between missions. Sometimes, if friendly forces are nearby, survivability is paramount, and the adversarial aircraft can be allowed to control the airspace for a short time before being pushed away by the reinforcements. In other situations, the airspace must be protected at all costs, e.g., to prevent a city from being attacked. Thus, there is no single policy solving this problem, as the acceptable risk



(a)



(b)

Fig. 1: (a) The proposed system helps the blue aircraft to make a detour that guarantees survival with a certain margin, even if a missile is fired by the red aircraft. (b) The actual separation between aircraft (green line), the RL predicted final separation between blue aircraft and a potential red aircraft's missile (blue line, $V(s)$), and the bound that can be set by the pilot on the acceptable separation between blue aircraft and a potential missile fired by the red aircraft (orange line).

The first author is with the Aeronautics division, SAAB, SE-582 54, Linköping, Sweden. The second author is with the Robotics, Perception and Learning Lab., School of Electrical Engineering and Computer Science, Royal Institute of Technology (KTH), SE-100 44 Stockholm, Sweden. e-mail: edvards.scukins@saabgroup.se, petter@kth.se

level varies significantly over time. Therefore, we need a transparent solution where the pilot can continuously monitor the risk, as well as the overall situation, and choose the acceptable level accordingly. Note, however, that human-in-the-loop experiments are beyond the scope of this paper, thus we replace the pilot with a high-level controller such as the one aiming to fly eastwards in Figure 1.

In adversarial scenarios like the one above, estimating risk is not a straightforward task. The risk of getting hit by a complex guided missile cannot be derived as a closed-form expression or hand-coded, instead it has to be learned from interaction with high fidelity simulation models. The risk measure we use is the future minimum separation between the aircraft and the missile, when the aircraft performs an optimal evasive maneuver, with large separations corresponding to lower risks. Thus we solve an RL problem to find such evasive maneuvers, using the smallest missile-aircraft distance measured over the entire missile trajectory as a reward, without any discounting. The value function of this RL problem then gives a measure of the agent's vulnerability with respect to the adversarial aircraft. Finally, we use this value function as a Control Barrier Function (CBF) [1]–[3] when pursuing other objectives.

The key idea of CBF is to reactively choose controls that satisfy a constraint on the time derivative of a particular function, the CBF, and thereby prevent the value of the CBF from entering some undesired interval. This combination of RL and CBFs allows us to do the transparent risk-efficiency trade-off described above, where the desired risk bound can be updated by the operator at any time.

The main contribution of this paper is that we show that a value function of an RL problem with reward only at the end of an episode, zero discounting, and a known system model, exhibits CBF properties. Thus we can create a CBF using RL as described above, in a way that makes the efficiency/risk trade-off transparent, in terms of allowing the operator to set and update the acceptable miss distance throughout the mission. Note that using RL to create the CBF in this way differs significantly from earlier combinations of RL and CBF, where a hand-coded CBF is used to guarantee safety while running the RL algorithm.

The outline of the work is as follows: In Section II we review the related work and background on CBFs and RL is provided in Section III. Then, Section IV describes the problem and the proposed solution is presented in Section V. Finally, simulation results are shown in Section VI, and conclusions are drawn in Section VII.

II. Related Work

Deep reinforcement learning has shown remarkable success in recent years [4] and it is a powerful method, which can be applied to a large variety of control tasks, utilizing deep neural networks for action-value function approximation [5], [6]. As the use of AI frameworks will increase in the near future, there is a need for users to be able to understand and influence the reasoning process of the system, a need for system transparency [7]. One can increase transparency by using Hierarchical Reinforcement Learning (HRL), an approach where actions are structured in multiple levels of abstraction. In such an approach, a low-level task can be learned by one type of network, that is afterward used to provide information to another higher level network [8], [9], [10]. In addition to transparency, safety is an important aspect of RL models that are to be deployed alongside humans. There has been a substantial amount of work done to address this problem. Authors of [11] used a RL approach to learn optimal policies that include constrained risk. Other proposed methods include first generating safe strategies, and then continue exploring the main tasks, while utilizing the safety constraints generated beforehand [12]. Similarly, a model-based method for learning obstacle avoidance can be used to estimate uncertainty for generating safe exploration strategies [13]. Another safety-related problem is associated with air traffic control and avoiding aircraft collisions. In [14], the authors apply RL for conflict resolution of two aircraft in the presence of uncertainties. Taking into account aircraft dynamics and surroundings, the problem formulation has a very large observation space, as mentioned in [15]. Thus it makes sense to break down the main problem into smaller sub-problems. Authors of [16] have utilized a hierarchical deep agent algorithm to help aircraft to (i) maintain safe separation, (ii) resolve conflicts, and (iii) arrive at their desired position.

In this article, we have taken inspiration from research done in [1], [17], where the authors present the CBF framework, for controlling safety-critical systems. But instead of handcrafting a model-based CBF, we use RL to create CBF in the sense of finding a value function that can be used as a CBF and we also show for what family of RL problems this can be done. Authors in [18] start with a CBF and use it to guarantee safety when applying RL. We use RL and a high fidelity simulation model to create the CBF, which is then used to guarantee safety when controlling the real system.

III. Background

This paper draws on results from both control theory and RL, but to keep the notation clear we use the RL notation of s,a to denote the state and action, instead of x,u which are more common in the control literature. Below we describe the CBF and RL results we will make use of in this paper.

A. Control Barrier Functions

The key idea behind Control Barrier Functions [1]–[3] is to specify a barrier function $h(s)$ such that the so-called *safe set* $\mathscr{C}$ is characterized by:

$$\mathscr{C} = \{s : h(s) \geq 0\},$$

Given the continuous system dynamics $\dot{s} = f_C(s,a)$, if we choose controls a that satisfy

$$\frac{dh}{dx} f_C(s,a) \geq -\alpha(h(s)), \tag{1}$$

where $\alpha \in \mathscr{K}$ the class K functions, we are guaranteed to stay in the safe set $s \in \mathscr{C}$.

For the discrete time dynamics case

$$s_{t+1} = f_D(s_t) \tag{2}$$

we get a similar result described by the following definition and lemma from [17].

Definition 1: The continuous function $h : \mathbb{R}^n \to \mathbb{R}$ is a discrete-time barrier function DTBF for the set $\mathscr{D} \subset \mathbb{R}^n$, $r \in \mathbb{R}_+$ if there exists $\alpha \in \mathscr{K}, \alpha(r) < r$ for all $r > 0$ such that

$$h(s_{t+1}) - h(s_t) \geq -\alpha(h(s_t)), \forall s \in \mathscr{D} \tag{3}$$

Lemma 1: Given a discrete time system (2), and a set $\mathscr{C} = \{x : h(x) \geq 0\} \subset \mathscr{D}$. Then the set is invariant if and only if there exists a DTBF.

The extension to discrete time control systems

$$s_{t+1} = f(s_t, a_t) \tag{4}$$

through a policy $a_t = a_t(s_t)$ giving $s_{t+1} = f(s_t, a_t(s_t)) = f_D(s_t)$ is straightforward.

B. Reinforcement Learning

In this work, we consider a standard reinforcement learning problem, where the agent needs to interact with an environment to learn something about it. The goal of reinforcement learning is to maximize the expected sum of future rewards. At each time step, the agent is in state $s \in S$, has the ability to apply action $a \in A$, and receives a reward $r_t = r(s_t, a_t)$. A transition to the next state is based on the dynamics $f : S \times A \to S$. For a given policy $\pi : S \to A$ we can now define the value function $v_\pi(s)$ as the expected total reward that can be obtained from that state

$$v_\pi(s) = \mathbb{E}\,[\sum_{i=0}^{\infty} r_i\, \gamma^i \,|S_t = s],$$

where the discount factor $\gamma \in [0, 1]$ and the corresponding optimal value function is

$$v^*(s) = \max_\pi v_\pi(s).$$

For the agent training, we make use of the Proximal Policy Optimization (PPO) algorithm [19], with Clipped surrogate objective function:

$$L_t(\theta) = \min(\, \mathrm{r_t}(\theta)\hat{A}_t,\, clip(r_t(\theta)),\, 1-\varepsilon,\, 1+\varepsilon)\hat{A}_t,$$

where $\mathrm{r_t}(\theta)$ is the ratio between the new policy (π_θ) and the old policy $(\pi_{\theta old})$ defined as

$$\mathrm{r_t} = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta old}(a_t|s_t)}.$$

In this approach ε is used to limit the size of update and the $\hat{A}_t$ represents an estimator of the advantage function at time step t.

IV. Problem Formulation

To enable the pilot to do a transparent tradeoff between mission objective and safety we formalize the problem as follows:

The state is composed of distance and orientation to the adversarial aircraft and missile, $s = (\rho, \mu, \nu, \delta) \subset S = \mathbb{R}^4$, see Figure 2.

ρ - Distance to adversarial aircraft
μ - Angle to adversarial aircraft
ν - Distance to incoming missile
δ - Angle to incoming missile

The actions available to the blue aircraft are 11 different turn rates equally spaced as follows $a_t \in A = \{-a_{max}, \ldots, 0, \ldots, a_{max}\}$, assuming the aircraft flies at a constant speed.

Problem 4.1: Given an action space A and a state space S and a bound $b \geq 0$ on acceptable missile distances, find a set of safe actions $A_s(s) \subset A$, such that the bound b is guaranteed for any policy $\pi_s : S \to A$ that choose actions from the safe set, that is $\pi_s(s) \in A_s(s)$ for all s.

As noted above, a user study is beyond the scope of this paper. Thus, we will not make experiments with humans in the loop, but instead replace the pilot with a high-level controller striving to either (i) fly safely towards a given goal, see Figure 1, or (ii) use RL to safely learn a new skill. Note that in the latter case, we first use RL to create a CBF using a simulation model, and then use RL again, with actions limited by the CBF, to learn an additional skill, see Figure 9.

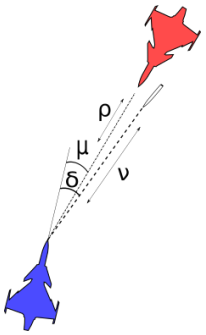


Fig. 2: Illustration of the state.

V. Proposed Solution

In this section, we describe the proposed CBF-RL approach and specify the family of RL problems that can be used to create CBFs, thus solving Problem 4.1. First, we define the set of RL problems with CBF characteristics.

Definition 2 (CBF-RL problem): Let a CBF-RL problem be an RL problem such that the system dynamics is given by a known function $s_{t+1} = f(s_t, a_t)$ and the reward $r_t = r(s_t, a_t)$ is such that $r_t = 0$ for all t except the final time $t = T$. Finally, the discount factor is $\gamma = 1$.

Now we state the following lemma, that provides a solution to Problem 4.1.

Lemma 2 (CBF-RL solution): Given a CBF-RL problem with a corresponding optimal value function $v^*(s)$, and a

desired lower bound on the total reward b, such that $v^*(s_0) > b$, then we can use $h(s) = v^*(s) - b$ as a DTBF to guarantee $v^*(s) \geq b$ for the execution, by always choosing an action $a_t \in A_s(s) \neq \emptyset$, where

$$A_s(s) = \{a \in A : v^*(f(s_t, a_t)) - v^*(s_t) \geq -k(v^*(s_t) - b)\}, \quad (5)$$

and $k \in (0,1)$.

Proof: The proof builds upon Lemma 1, and we want to show that the set where $v^*(s_t) > b$ is invariant. Let $\alpha(r) = kr < r$, since $k < 1$. First we note that $A_s(s) \neq \emptyset$ in (5). By Bellmans equation

$$v^*(s_t) = \max_a \Sigma_{s_{t+1}, r} p(s', r|s, a)[r + \gamma v^*(s_{t+1})] \quad (6)$$

$$= \max_a [r + \gamma v^*(s_{t+1})] \quad (7)$$

$$= \max_a v^*(s_{t+1}) \quad (8)$$

$$= \max_a v^*(f(s_t, a)) \quad (9)$$

thus there is an a such that $v^*(s_t) = v^*(s_{t+1})$. This $a \in A_s(s)$ since $v^*(s_t) > b$ gives $-k(v^*(s_t) - b) < 0 = v^*(s_{t+1}) - v^*(s_t)$. Now with the inequality in (5) satisfied we substitute $v^*(s) = h(s) + b$ and get $h(s_{t+1}) - h(s_t) \geq -k(h(s_t)) = -\alpha(h(s_t))$ which is (3). Thus, there is a policy choosing $a \in A_s(s)$, and any such policy also satisfies (3), making $v^*(s_t) > b$ invariant. ■

Thus, by solving a CBF-RL problem, and applying a policy choosing actions from $A_s(s)$ described in the lemma above, we have a solution to Problem 4.1 above. Now we will solve a few such problems to illustrate the approach.

VI. Numerical Results

In this section, we first solve the CBF-RL problem for our adversarial scenario. Then we apply the results to a simple problem and finally use them to address a more complex problem, applying RL once again to learn a new policy, while safety is guaranteed by the CBF what was created using CBF-RL. All numerical results were created using algorithms from the PyTorch and SciPy packages [20], [21].

A. The CBF-RL problem for avoiding threats

We formulate an RL problem, where the main objective of the agent is to explore the threat posed by an adversarial aircraft and establish a policy for evading threatening situations. In the scenario, the adversarial aircraft (red) starts at $(100, 0)$ and fires its onboard missile as soon as the agent is within firing range, see Figure 3. The initial location of the agent (blue) is randomized within an area of a semi-annulus with an inner and outer radius of 40 km and 75 km, respectively.

In this particular set-up, the missile's operational range is roughly 75 kilometers, which implies that the missile will catch up with the agent unless the agent executes some sort of evasive maneuver.

The problem is set up as follows:

- The reward is 0 for all times except the final time.
- The final reward is equal to the minimal separation throughout the scenario, thus if the aircraft was hit the final reward is 0, otherwise, it is larger than 0.

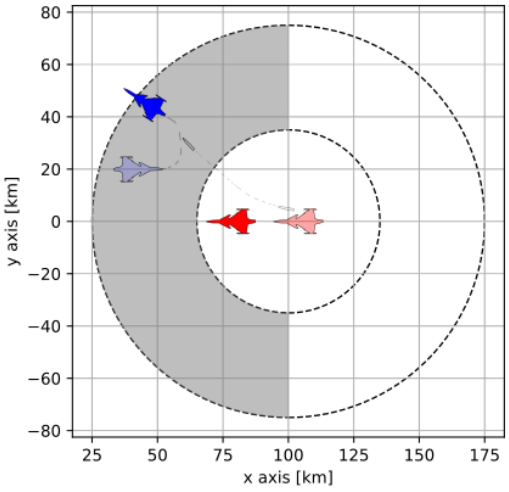


Fig. 3: The initial position of the agent (blue aircraft) is randomized within an area of a semi-annulus with an inner and outer radius of 40 km and 75 km, respectively. The adversarial aircraft (red) is placed at $(100, 0)$. The lighter color shapes indicate the initial positions of the aircraft and the darker colored ones illustrate possible final positions.

As can be seen, this RL problem satisfies Definition 2. The optimal value function for this problem is denoted $v_{safe}(s)$ and the results are illustrated in Figure 4 and Figure 5.

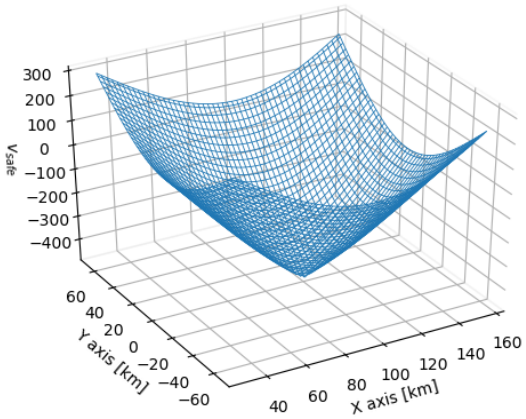


Fig. 4: $v_{safe}(s)$ as a function of the position of the own aircraft, with the enemy aircraft at (100,0).

As can be seen in Figure 4, the expected reward, which is an approximation of the final separation between the agent and the final position of the missile, is the smallest close to the adversarial aircraft and increases when the agent is positioned further away from the opponent.

A key property of the CBF-RL is that it satisfies the Bellman equation, leading to $v^*(s_t) = \max_a v^*(f(s_t, a))$, see (9). However, we can only expect the RL algorithm to provide an estimation of the true $v^*(s_t)$. To investigate how large the estimation errors are, we run the simulation in Figure 5 with actions that maximize $v_{safe}(f(s_t, a_t))$. In theory

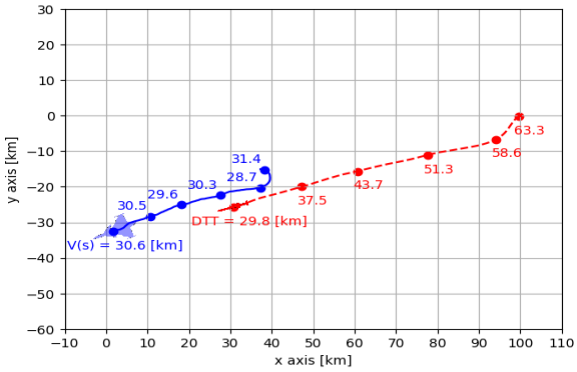


Fig. 5: The opponent fires a missile from (100,0) towards the blue aircraft, starting at (38,-10). Maximizing $v_{safe}(f(s_t,a_t))$ gives a trajectory (solid blue) that turns away from the incoming missile (dashed red). A set of values that represent the estimate of the final separation between the missile and the aircraft, $v_{safe}(s)$, are plotted along the blue trajectory. The distance to the target (DTT) is shown along the red (dashed) trajectory. Note that DTT decrease with time as the missile gets closer to the target while $v_{safe}(s)$ stays constant with some oscillation due to estimation errors.

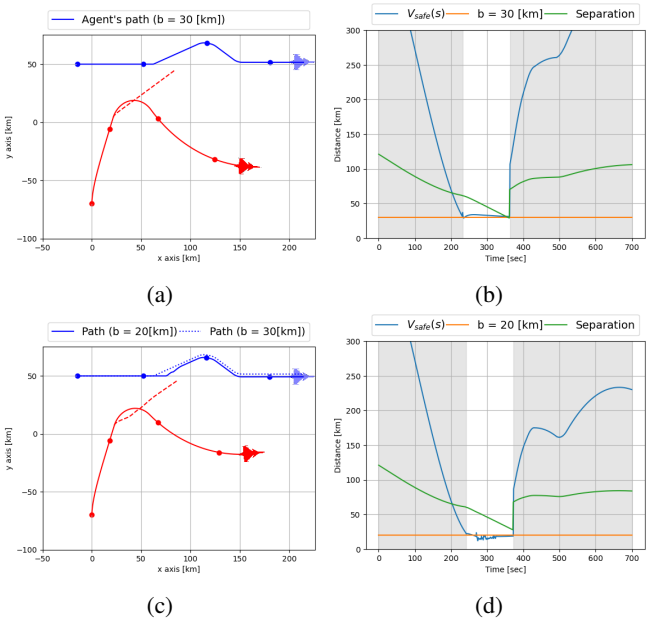


Fig. 6: The blue agent's objective v_{mo} is to fly eastward, with safety bounds of $b = 30km$(top) and $b = 20km$(bottom) respectively. The red agent approaches and fires a missile (dashed line) towards the blue. In (c), the blue trajectory from (a) is dotted, and we see that the larger safety margin in (a) results in a larger detour. In (b), (d), v_{safe} is blue, the actual separation is green and the bound is orange. The time the missile is active is indicated with a white background. Note how in both (b),(d), the predicted separation v_{safe} approaches the bound and stays close to the bound (sometimes being below due to estimation errors), but still accurately predicting the future minimal separation as the missile (green) keeps coming closer until it runs out of fuel very close to the bound.

this should lead to $v^*(s_t) = v^*(s_{t+1})$ but in practice we get values in the range $(29, 32)$. We will see below how these errors sometimes makes the agent brake the safety bound, but only with a small margin.

B. Applying the approach to low complexity mission objectives

In this section, we will apply the proposed approach to low complexity mission objectives, such as flying east while limiting risk exposure. To keep the setup general, we assume that the low complexity mission objectives are captured in a value function $v_{mo}(s)$. Thus, a natural approach for maximizing $v_{mo}(s)$ while respecting the bound $v_{safe}(s) \geq b$ is the following.

$$a = \text{argmax}_a v_{mo}(f(s_t,a)) \quad \text{s.t. } a \in A_s(s) \tag{10}$$

Examples of the resulting behavior with $v_{mo}(s)$ striving to fly eastwards can be seen in Figures 1 and 6. Two scenarios are visualized in Figure 6a with a safety threshold of $b = 30km$ and $b = 20km$, i.e., the final separation between the agent and the missile should not be lower than 30 and 20 km respectively.

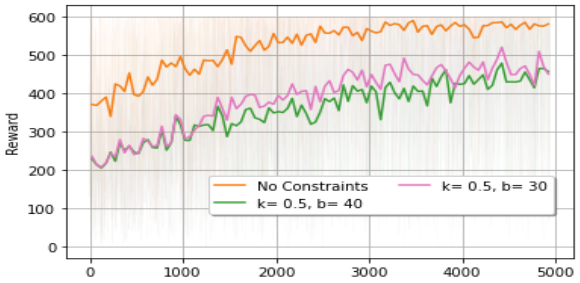
The advantage of using CBF-RL is the ability to predict expected reward at a given state. In other words, we can predict what the final separation between the agent and the hostile object will be. This can be seen in Figure 6b where the actual distance is illustrated by a green line and the estimated final separation by the blue line, $v(s)$. As the missile approaches the agent, the estimate of future separation $v(s)$ is roughly constant, very close to the chosen bound, while the actual distance between the agent and the incoming missile decreases until the missile runs out of fuel very close to the predicted value. Note that while the red aircraft has not fired its missile, the missile is at the same location as the adversarial aircraft. The same applies when the missile has run out of fuel. We then assume that the red aircraft has more onboard missiles.

As mentioned above, during air combat it is desirable to have the ability to adjust risk exposure depending on the circumstances. Therefore, in the same scenario, we decrease the acceptable distance, Figure 6c, and compare it to Figure 6a. In the results, the agent keeps a smaller distance to the adversarial aircraft, thus illustrating the ability to change the acceptable risk setting depending on the situation.

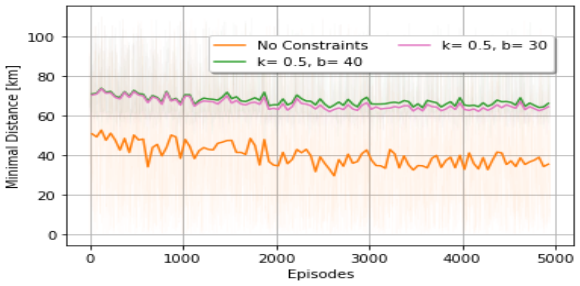
C. Learning high complexity mission objectives while staying safe

In Section VI-B, we described an approach of using CBF-RL to constrain a predefined objective function $v_{mo}(s)$. In this section, we further expand the capabilities of CBF-RL in a scenario where an additional high-level policy needs to be learned through RL while staying safe.

We consider a scenario where the agent needs to keep



(a)



(b)

Fig. 7: (a) illustrates the obtained reward during the training. (b) illustrates the average distance that is kept between the aircraft. The policy trained without activating the CBF-RL is denoted as "No Constraints", while the other policy was trained while being constrained by CBF-RL. Thus, by varying the parameter b, governing the extent of the safety constraints, we are able to adjust the trade-off between effectiveness and safety.

track of the adversarial aircraft's position. This is done by keeping the aircraft within the field of view and range of the onboard radar unit, see Figure 2.

To address this problem, we apply the approach suggested in [22] and illustrated in Figure 8, using our CBF-RL to find the safe action set. Rewards for the RL problem are provided

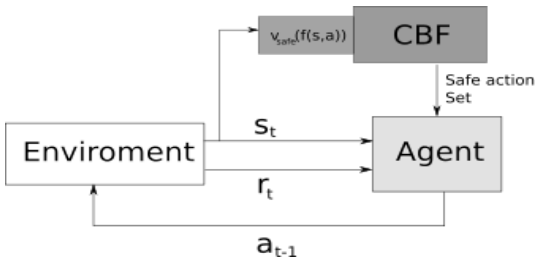
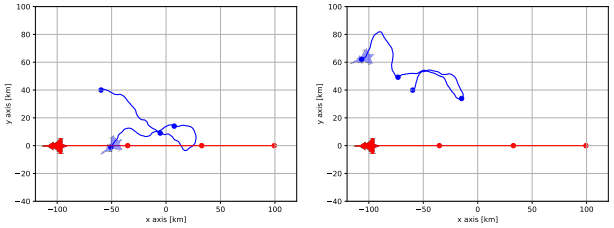


Fig. 8: Observation of the state s is sent to the RL agent and to the safety network $v_{safe}(f(s,a))$. The RL agent is then only allowed to chose from actions satisfying (5).

by the following rules:

- if the agent has the adversarial aircraft within its own radar range and field of view, a reward of +1 is received during the current time-step
- else a reward of 0 is received

Thus, to maximize accumulated reward, the agent needs to keep observation of the adversarial aircraft as long as



(a) (b)

Fig. 9: In (a) the blue agent has learned to keep the red agent within sensor coverage. In (b) the blue agent has the same objective, but with the requirement of staying safe.

possible. The learning history of the non-constrained learning is illustrated in Figure 7a and is denoted as "No Constraints". We see that during the learning phase, the agent is able to reach near maximum accumulated reward already after 3000 episodes, where one episode consists of 600 time steps, giving a maximal reward of 600. In Figure 7b, we can see that the agent's relative distance to the adversarial aircraft averages at about 35 *km*, and as the agent increases accumulated reward, the distance between aircraft's tends to decrease.

By activating CBF-RL as illustrated in Figure 8, we notice a trade-off between accumulated reward and the average distance. In this situation, some of the actions are no longer available in a given situation, thus the agent is not able to maintain tracking of the adversarial agent and has to use safer alternatives. Thus, by adding the safety requirement, the accumulated rewards decrease in the given environment. Two example executions can be seen in Figure 9. Note that the unconstrained version in 9(a) moves much closer than the constrained one in 9(b).

VII. Conclusions

In this paper, we provide theoretical results on using RL to create CBFs, and illustrate these with a set of adversarial aircraft scenarios. The theoretical results show what some RL value functions can be used as CBFs, and we apply the Bellman equation to show that the resulting safe sets are indeed invariant when using the designated action set. Finally, we also show that these guarantees lead to good, but not perfect, performance in the presence of estimation errors in the value function.

On the application side, we have shown that combining CBFs with RL allows the operator to set the risk level deemed to be acceptable while acting in adversarial environments. Such a feature is important, since objectives are often conflicting, and a certain amount of risk is needed to complete the mission. In such cases, we want to endow the pilot with a tool that enables him to constantly balance risk exposure with other mission objectives in a transparent way.

VIII. Acknowledgment

The authors gratefully acknowledge funding from Vinnova, NFFP7, dnr 2017-04875.

REFERENCES

[1] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2016.

[2] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," *arXiv preprint arXiv:1903.11199*, 2019.

[3] P. Ogren, A. Backlund, T. Harryson, L. Kristensson, and P. Stensson, "Autonomous ucav strike missions using behavior control lyapunov functions," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2006, p. 6197.

[4] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[5] C. Liu, X. Xu, and D. Hu, "Multiobjective reinforcement learning: A comprehensive overview," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 3, pp. 385–398, 2014.

[6] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[7] R. Iyer, Y. Li, H. Li, M. Lewis, R. Sundar, and K. Sycara, "Transparency and explanation in deep reinforcement learning neural networks," in *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, 2018, pp. 144–150.

[8] N. Heess, G. Wayne, Y. Tassa, T. Lillicrap, M. Riedmiller, and D. Silver, "Learning and transfer of modulated locomotor controllers," *arXiv preprint arXiv:1610.05182*, 2016.

[9] A. S. Vezhnevets, S. Osindero, T. Schaul, N. Heess, M. Jaderberg, D. Silver, and K. Kavukcuoglu, "Feudal networks for hierarchical reinforcement learning," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3540–3549.

[10] C. Tessler, S. Givony, T. Zahavy, D. J. Mankowitz, and S. Mannor, "A deep hierarchical approach to lifelong learning in minecraft," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[11] P. Geibel and F. Wysotzki, "Risk-sensitive reinforcement learning applied to control under constraints," *Journal of Artificial Intelligence Research*, vol. 24, pp. 81–108, 2005.

[12] S. Junges, N. Jansen, C. Dehnert, U. Topcu, and J.-P. Katoen, "Safety-constrained reinforcement learning for mdps," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2016, pp. 130–146.

[13] G. Kahn, A. Villaflor, V. Pong, P. Abbeel, and S. Levine, "Uncertainty-aware reinforcement learning for collision avoidance," *arXiv preprint arXiv:1702.01182*, 2017.

[14] D.-T. Pham, N. P. Trant, S. K. Goh, S. Alam, and V. Duong, "Reinforcement learning for two-aircraft conflict resolution in the presence of uncertainty," in *2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF)*. IEEE, 2019, pp. 1–6.

[15] E. F. Morales and C. Sammut, "Learning to fly by combining reinforcement learning with behavioural cloning," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 76.

[16] M. Brittain and P. Wei, "Autonomous aircraft sequencing and separation with hierarchical deep reinforcement learning," in *Proceedings of the International Conference for Research in Air Transportation*, 2018.

[17] M. Ahmadi, A. Singletary, J. W. Burdick, and A. D. Ames, "Safe policy synthesis in multi-agent pomdps via discrete-time barrier functions," in *IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 4797–4803.

[18] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick, "End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 3387–3395.

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

[20] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[21] E. Jones, T. Oliphant, P. Peterson, *et al.*, "Scipy: Open source scientific tools for python," 2001.

[22] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe reinforcement learning via shielding," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.