RadarConf'23
2023 IEEE RADAR CONFERENCE

Reinforcement Learning

For Radar Waveform Optimization

Mario Coutino, Faruk Uysal

Radar Technology Department, TNO, The Netherlands

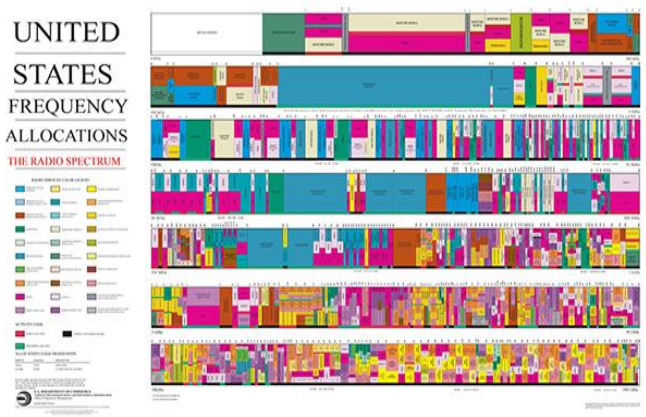
San Antonio, TX, US | May 2023

Outline

- **Radar and Radar Waveform Adaptivity**
- **Reinforcement Learning for Radar Adaptivity**
- **Waveform Design using Reinforcement Learning**
  - Setting description
  - Design Decisions
  - Illustrative Examples
- **Conclusion and Future Directions**

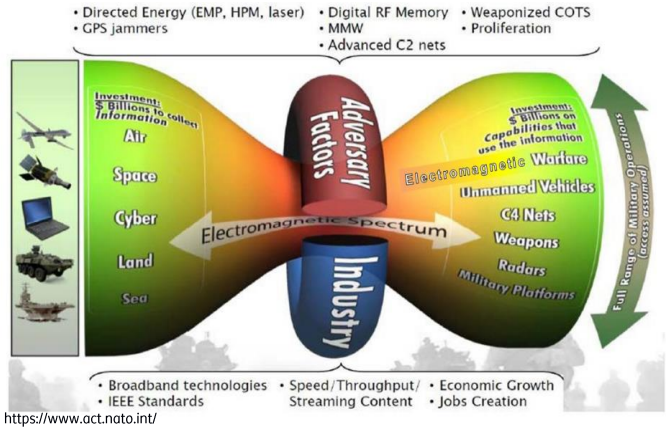
Radar Adaptivity

Modern radar systems are exposed to **increasingly challenging dynamic scenarios**



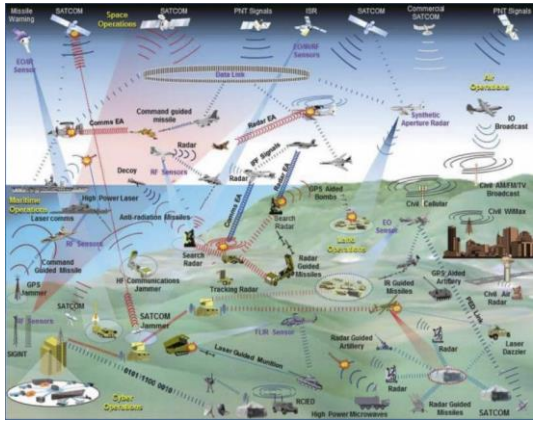
https://ntia.gov/

increasingly congested environments



https://www.act.nato.int/

contested dynamic operations



https://www.act.nato.int/

dynamic (non)coordinated deployments

Thus, **the ability to adapt**[1] to operational conditions **is critical** for modern radar systems...

Among the possible venues for adaptivity, **radar waveform optimization**[2] already **has proved itself**.

[1] Guerci, Joseph R. "Cognitive radar: A knowledge-aided fully adaptive approach." 2010 IEEE Radar Conference. IEEE, 2010.
[2] Blunt, Shannon D., and Eric L. Mokole. "Overview of radar waveform diversity." IEEE Aerospace and Electronic Systems Magazine 31.11 (2016): 2-42.

Radar Waveform Adaptivity

Radar waveforms can be optimized/adapted for

- **improve** a radar task[1], e.g., detection, tracking or classification
- **mitigate** clutter effects[2]
- **enable** coexistence of different sensing and communication platforms[3]

This is typically achieved by

- consulting a **look-up-table** :: fast but limited
  - dictionary of waveforms, e.g., responding to a mode -“tracking waveform” or a “classification waveform”
- solving an **optimization problem** :: flexible but demanding
  - given environmental inputs, a waveform is generated on-the-fly by solving an optimization procedure

Besides searching for even faster optimization algorithms,
is there other possibilities to have the best of both worlds?

[1] Cui, Guolong, et al., eds. *Radar waveform design based on optimization theory*. SciTech Publishing, 2020.
[2] Hughes, Evan J., and Clive M. Alabaster. "Medium PRF radar PRF optimisation using evolutionary algorithms." *Proceedings of the 2003 IEEE Radar Conference (Cat. No. 03CH37474)*. IEEE, 2003.
[3] Aubry, Augusto, et al. "A new radar waveform design algorithm with improved feasibility for spectral coexistence." *IEEE Transactions on Aerospace and Electronic Systems* 51.2 (2015): 1029-1038.

Alternative:

instead of per-case optimization

better to design a policy to

act given the environment's state.

Enabling Radar Waveform Adaptivity

A **policy**

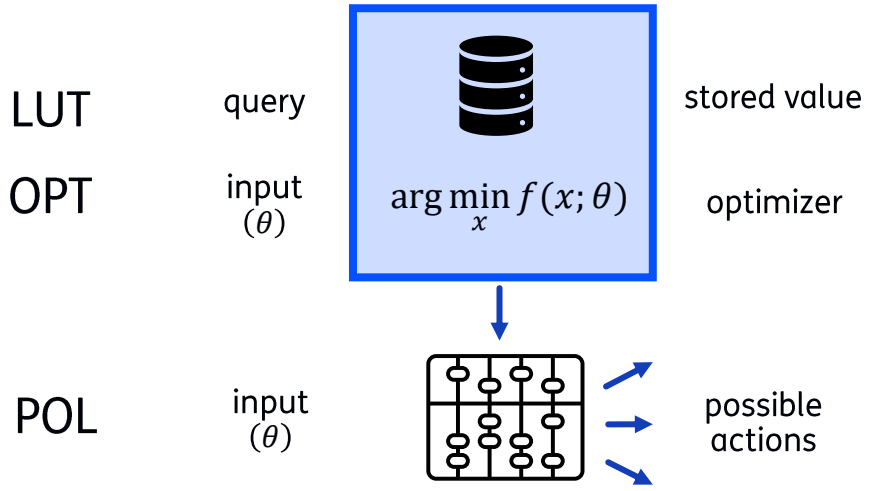
can be interpreted as:

as **mapping** representing a **compressed look-up table** capable of **solving** a family of **optimization problems.**

Thus, it can have

- predictable and low time requirements
- affordable memory footprint
- input-dependent behavior
- multiple responses to the same query

How can we learn a policy for radar waveform design?

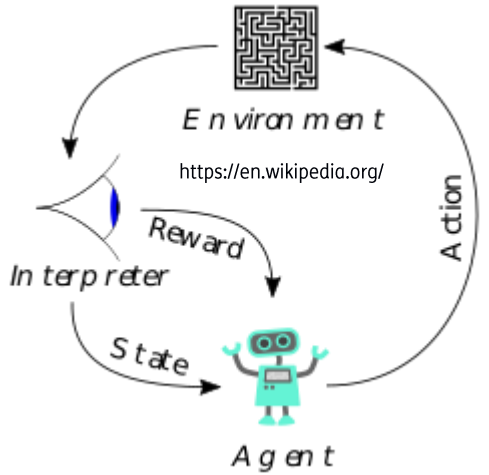


Reinforcement Learning

A way to **learn policies** for particular problems is by **reinforcement learning** (RL) algorithms.

Reinforcement learning[1]

- is **a feedback-based** technique aiming to **teach agents** to **maximize rewards** as it **interacts with an environment**.
- it has been **applied to several domains**, including radar, e.g.,
  - strategy games[2]
  - cognitive beamforming[3]
  - notched waveforms[4]
  - radar resource allocation[5]
  - etc..



RL Agent's learning cycle

In this work, we study its application to burst design.

[1] Sutton, Richard S., and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
[2] https://www.deepmind.com/research/highlighted-research/alphago
[3] Ahmed, Aya Mostafa, et al. "A reinforcement learning based approach for multitarget detection in massive MIMO radar." *IEEE Transactions on Aerospace and Electronic Systems* 57.5 (2021): 2622-2636.
[4] Durst, Sebastian, and Stefan Brüggenwirth. "Quality of service based radar resource management using deep reinforcement learning." *2021 IEEE Radar Conference (RadarConf21)*. IEEE, 2021.
[5] Smith, Graeme E., and Taylor J. Reininger. "Reinforcement learning for waveform design." *2021 IEEE Radar Conference (RadarConf21)*. IEEE, 2021.

Reinforcement Learning

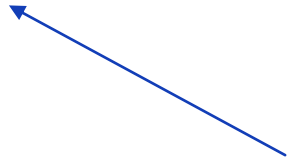
Most of RL starts by modelling a problem by a Markov decision process $\mathcal{M}(\mathcal{S}, \mathcal{A}, T, R)$ (or any of its variants)

- $\mathcal{S}$ set of permissible environment states
- $\mathcal{A}$ set of available actions
- $T: \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}_0^+$ probability of reaching an state $s' \in \mathcal{S}$ after performing action $a \in \mathcal{A}$ at state $s \in \mathcal{S}$
- $R: \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ *immediate* reward after an action $a \in \mathcal{A}$ is taken at state $s \in \mathcal{S}$

The **goal of RL is to learn a policy** $\pi: \mathcal{A} \times \mathcal{S} \to [0,1]$ which **maximizes** the expected **cumulative reward.**

To make use of the RL machinery for waveform design

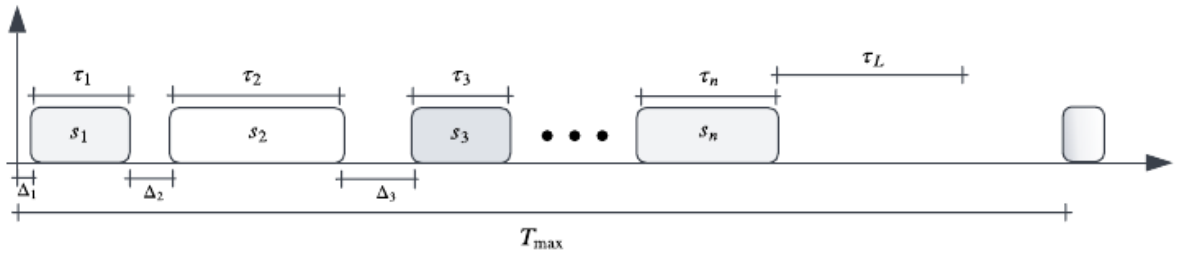
- the above components needs to be defined for the problem and
- an appropriate RL algorithm needs to be chosen



This by interacting with the environment

Reinforcement Learning For Waveform Optimization

Let us consider the following setting for **burst-level waveform optimization**



- irregular but bounded **transmit intervals** possible :: $\Delta_{\min} \leq \Delta_n \leq \Delta_{\max}$
- upper bound on **number of pulses** in burst :: $n \leq N$
- irregular but bounded **pulse durations** :: $\tau_{\min} \leq \tau_n \leq \tau_{\max}$
- **pulses modulations** selected from a dictionary :: $\{s_1(t), \dots s_K(t)\}$
- **total duration**, including listening time τ_L, is upper bounded :: $\sum_n(\tau_n + \Delta_n) + \tau_L \leq T_{\max}$

Constraints of the design "game"

Game over constraints
Actionable constraints

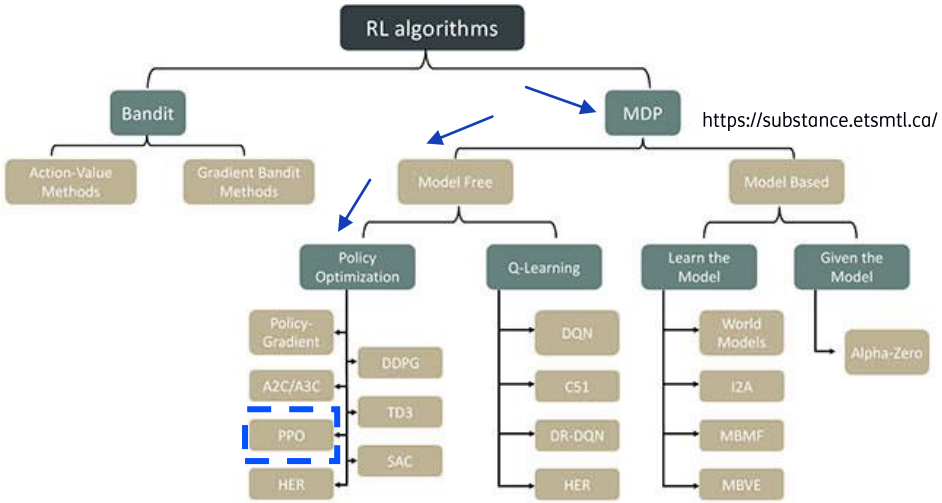
At every "time instant", the agent needs to select (take) a pulse configuration (action):

- **duration of pulse** :: $\tau_n \in \mathcal{T} \coloneqq [\tau_{\min}, \tau_{\max}]$
- **separation with respect previous pulse** :: $\Delta_n \in \mathcal{D} \coloneqq [\Delta_{\min}, \Delta_{\max}]$
- **modulation type** :: $k \in \mathcal{K} \coloneqq \{1 \dots, K\}$

Action Space

Reinforcement Learning For Waveform Optimization

Given a base scenario(as before), **algorithm selection** comes next.



Based on what we want

- **learning a policy**

and what we have

- **mix of discrete and continuous** variables
- (so-far) **no model** of the environment

we can make a selection from the RL algorithm zoo[1].

Here, we advocate* the use of the **Proximal Policy Optimization** (PPO) algorithm.

- designed to achieve **stable learning** :: optimizes performance
- **sample-efficient** compared to other on-policy methods* :: proximal optimization*
- it has a simpler implementation and has **“few moving” parts** :: easier to tune
- policies tend to be **fast to evaluate** :: neural networks with few parameters

Quality of action

Parametrized policy

$$L(\theta) = \hat{\mathbb{E}}_t\left[\pi_\theta(a_t|s_t)\hat{A}_t\right]$$

Expected quality of policy

[1] Part 2: Kinds of RL Algorithms — Spinning Up documentation (openai.com)

* Details in publication.

With this setup

different games can be played...

it is not yet defined what the agent sees
(state description)

and how their actions are scored.
(reward)

Optimization Example

Optimization of Maximum Sidelobe Level (MSL) under Duty Cycle (DC) Constraints

Example consists on designed a pulsed waveform with the objective of

- achieving a *particular duty* cycle DC_0, while
- *maintaining the MSL* of a region of interest $\mathcal{R}$ below a desired level MSL_0

Reward function:

$$R(n) = \begin{cases} 0 & \forall\, n : n < N \text{ and } T_{\text{design}} < T_{\max} \\ -\text{FOM} & \text{otherwise} \end{cases}$$

$$\text{FOM} = \frac{|\text{DC} - \text{DC}_0|}{\text{DC}_0} + \max\left\{ \frac{\text{MSL} - \text{MSL}_0}{|\text{MSL}_0|}, 0 \right\}$$

Aims to match the DC goal

It does not promote waveforms with lower MSL than target.

Collection all previous actions

State Space: $s_t = [s_{t-1}, a_{t-1}]$

Naïve but straightforward description of design

Model: Neural Network [32,16] w/ ReLus

Frameworks: pytorch[1] ray[2]

[1] https://pytorch.org/
[2] https://www.anyscale.com/ray-open-source

Optimization Example

Optimization of Maximum Sidelobe Level (MSL) under Duty Cycle (DC) Constraints

Waveforms Generated with Optimized Policy

$$\mathcal{T} \coloneqq \left[\frac{1}{B}, \frac{T_{\max}}{15}\right]$$

$$\mathcal{D} \coloneqq \left[\frac{10}{B}, \frac{T_{\max}}{15}\right]$$

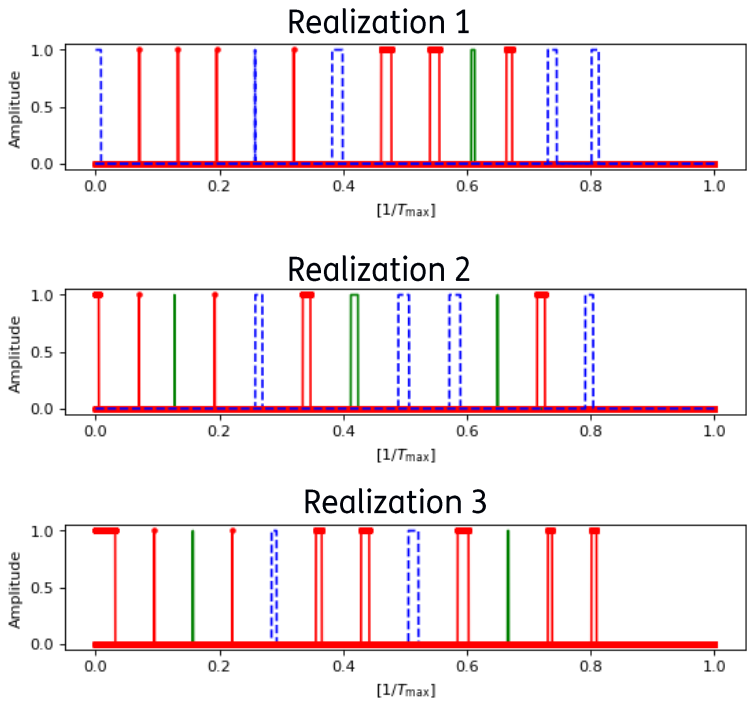
$$N = 32$$

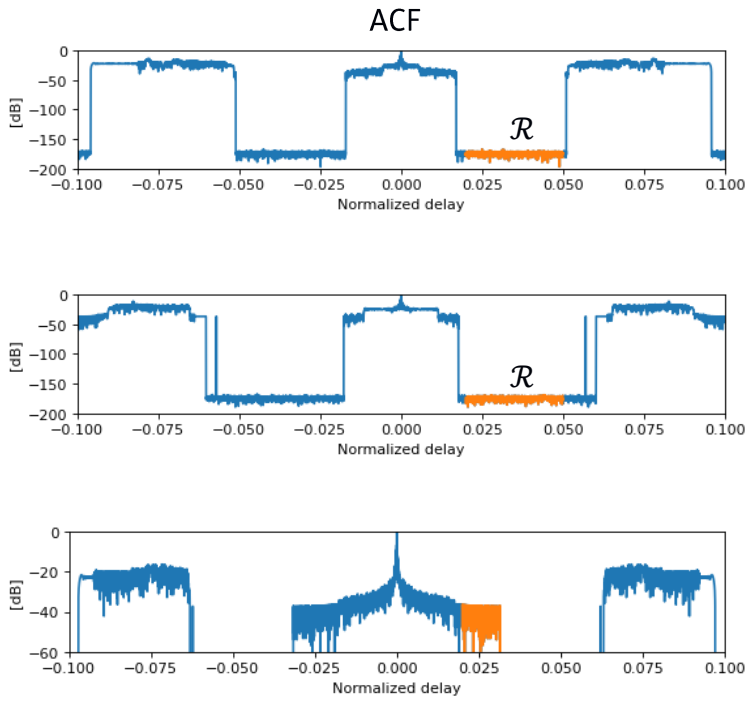
$$\mathrm{DC}_0 = 10\%$$

$$\mathrm{MSL}_0 = -40\mathrm{dB}$$

$$\mathcal{R} \coloneqq [0.020, 0.05]$$

Strong dependency on time distribution
(temporally signals looks similar)





Modulations type: (red) down- (blue) up- (green) up-down chirp

Small, but some diversity on the waveforms when the policy is sampled.

Designing with respect to a

single configuration is not the killer application of RL*.

Generation of waveforms for arbitrary inputs is.

* Elaborated in publication.

Design Example

Automatic Design with Control of Maximum Sidelobe Level under Duty Cycle Constraints

Instead of only optimizing for a single set of target parameters

- the goals, MSL and DC, can be included in the state

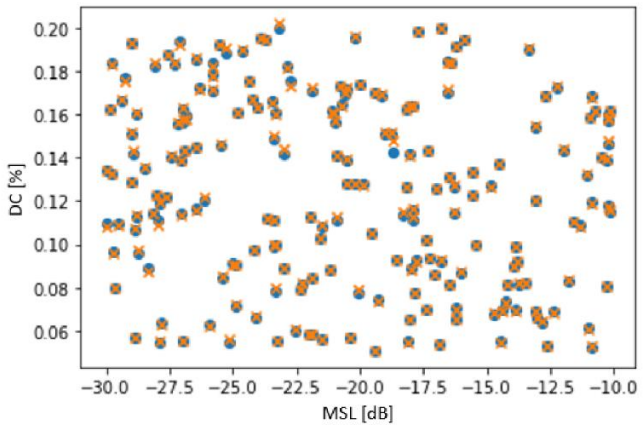
$$\textbf{State Space: } \boldsymbol{s_t} = [s_{t-1}, a_{t-1}]; s_0 \coloneqq [\mathrm{DC}_0, \mathrm{MSL}_0]$$

Initial conditions

This will allow to obtain different waveforms by *querying* the agent's policy with *different initial conditions*.

without making any other change in the environment setting

- Target requirements (blue circles)
- Achieved parameters (orange crosses)

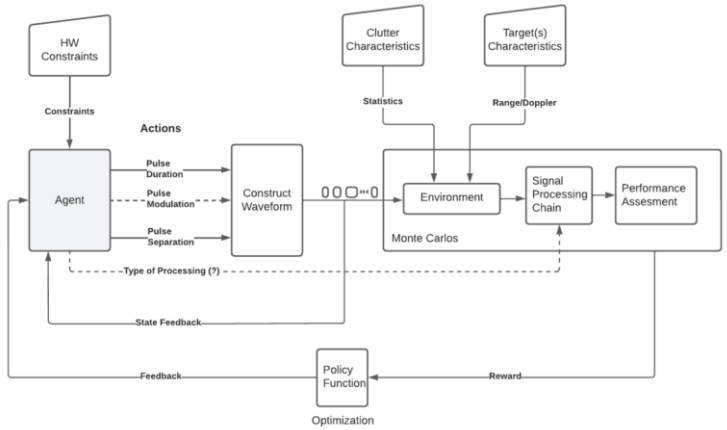


Policy evaluation tend to be fast
(small models)

Random waveforms can be designed for new configurations once the policy has been learnt.

Conclusions and Future Directions

- **Policy** learning using RL is a promising enable technology for **real-time radar adaptivity**
  - policies allow to have the **speed of LUTs** while being as **responsive as optimization problems**,
  - **structure knowledge** to gain experience –after optimization all that has been evaluated is thrown away
  - and **allow diversity** when the solution to the problem accepts it.
- **Proper** definition of environment and **"game" rules** are crucial
  - under the same setting, multiple "design games" can be played –change on reward or available observations
- **Waveform design using RL** was demonstrated with MSL/DC goals
  - several parameters were optimized at once **even for varying goals** –initial conditions
- **Extensions** of "games" to **complete scenarios** of practical interest
  - holistic view of, e.g., a radar processing chain
- Application of RL to **design "testers"** of systems –finding edge/corner cases
- Consideration of **Multi-agent RL** to
  - **alleviate sample complexity** for larger problems, e.g., dwell design.



Thank you

mario.coutinominguez@tno.nl

Reinforcement Learning for Radar Waveform Optimization

Mario Coutino and Faruk Uysal
Radar Technology, *Netherlands Organisation for Applied Scientific Research (TNO)*, The Hague, The Netherlands
{mario.coutinominguez, faruk.uysal}@tno.nl

***Abstract*—Recently, it has been shown that reinforcement learning (RL) is able to solve decision-based problems through a series of action-observation-reward cycles. In this paper, we pose the problem of constrained waveform optimization as a sequential decision problem and show how it can be solved by an RL agent. The proposed RL-based method is an alternative to mix-integer optimization, evolutionary algorithms, and Bayesian optimization, which is capable of dealing directly with a variable parameter space dimension while considering designs with different processing algorithms in the (optimization) loop. To illustrate the effectiveness of the proposed method, we demonstrate the optimization of an agent's policy capable of defining the number of pulses as well as their duration and modulation parameters of radar waveform while optimizing an user-defined figure of merit.**

***Index Terms*—artificial intelligence, radar, reinforcement learning, optimization, waveform optimization**

I. Introduction

With the increased computing capabilities of modern radar systems and technological advances, full adaptivity of such systems, on both transmit and receive, is becoming a reality [1]–[3]. However, these new possibilities do not come without challenges. Among them, we find the automatic selection, design or optimization of the transmit radar waveform intended to fulfill a particular task. Due to the benefits of radar adaptivity to the environment or the task at hand, sometimes referred as *cognition* [1]), several techniques have been devised for realizing, for instance, waveform optimization, see, e.g., [4]. This adaptation and optimization could allow for better detection, tracking or classification performance.

Although several works have defined different figures of merit (FOMs) by means of application-specific cost functions, most of those works have been focused on the received filter, e.g., [5], [6]; and the modulation (or spectral content) of the transmit waveform, e.g., [4], [7]. Among those that have investigated different optimization parameters, such as pulse locations for blind zone optimization [8] or sparse sensing [9], their scope has been limited to a single parameter. Thus a framework capable of leveraging the insights, FOMs and waveform parametrizations developed in the literature is much needed towards defining an integral autonomous radar waveform designer.

This study is carried out within the framework of the Dutch Radar Centre of Expertise (D-RACE), a strategic alliance of Thales Netherlands B.V. and TNO.

In addition to the need to consider different parameters when radar waveforms are optimized, the latency requirements challenges the possibility of doing optimization, and thus adaptation, during system operation at particular time horizons. Therefore, methods capable to provide the much needed adaptivity but that can be executed in real time are of utmost importance. Efforts in this direction have been done in the literature for particular methods, see, e.g., [4], [10], though to the best of our knowledge not yet encompassing several waveform parameters simultaneously.

Reinforcement learning has not only been successful in robotics and strategy games, see, e.g., [11], but also it has shown promising results in radar applications. For example, in [12], RL is used for transmit frequency selection with the goal of improving detection and tracking performance in congested spectral environments. In [13], [14], a cognitive beamforming algorithm for collocated MIMO radars based on RL is proposed. This method allows the automatic synthesis of MIMO emissions tailored to the current (and predicted) environmental situations. Finally, to mitigate runtime evaluations, in [15], an RL-enhanced quality-of-service based resource allocation model was introduced. For more instances of the usage of RL in radar, the reader is referred to [12], [16]–[18] and reference therein. In this work, we advocate for a waveform optimization framework driven by a reinforcement learning (RL)-based approach.

The paper continues as follows. In Section II the necessary background to follow this contribution is provided. Section III introduces the RL framework used for waveform optimization as well as the definitions of the RL environment. Results illustrating the proposed approach are discussed in Section IV. And finally, Section V draws conclusions from this work and discusses future research directions.

II. Preliminaries

In the following, we introduce the necessary background to discuss the contribution of this work. We first discuss the basics of reinforcement learning and then the radar waveform and system model used in this work.

A. Reinforcement Learning

The field of RL provides a general framework for defining autonomous agents, either physical or virtual, capable of making decisions (*actions*) without human intervention solely based on the *state* of the *environment*. This is done with the

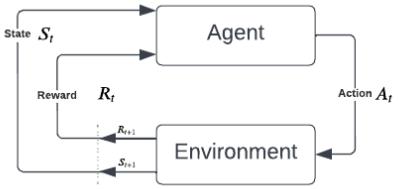


Fig. 1. Illustration of the action-observation-reward cycle [11].

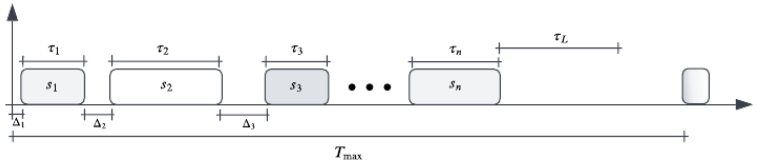


Fig. 2. Illustration of the structure of the designed pulsed waveform. The agent at each step can select the duration of the pulses (τ_i), the separation with respect to the previous pulse (Δ_i) and the modulation type (color). Here τ_L is the trailing listening time to ensure a given maximum range and $T_{\max}$ the maximum allowable waveform duration.

objective of performing a task, for example, the design of a radar waveform to meet certain operational requirements. An illustration of this procedure is seen in Fig. 1.

In RL, the tasks to be solved are modelled through a *Markov Decision Process* (MDP) or their variants, e.g., partially observable MDP (POMDP). These processes are employed to represent and tackle sequential decision-making problems under uncertainty [11]. Typically, an MDP is defined by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, T, R)$ where $\mathcal{S}$ defines the set of permissible *states*, $\mathcal{A}$ the set of available *actions*, $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}_0^+$ specifies the probability of reaching an state $s' \in \mathcal{S}$ after performing an action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$ and $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ provides the immediate reward after an action $a \in \mathcal{A}$ is taken in a state $s \in \mathcal{S}$.

As in this setting, the immediate reward provides an indication of how good or bad an action is, solving the planning problem related to the MDP amounts to find a *policy* $\pi(\cdot)$ (set of actions) that provides the highest possible reward. As a result, the goal of RL is to learn an optimal policy $\pi^*(\cdot)$ capable of selecting a set of actions that leads to the highest (accumulate/discounted) reward [11]. RL achieves this by accumulating empirical knowledge by "playing" a set of *episodes* (games), i.e., sequence of states, actions and rewards ending in a terminal state.

In this work, we thus pose the problem of radar waveform as an MDP, making the appropriate definitions for the available actions, possible states and reward definition and then make use of a proximal policy optimization (PPO) [19], a widely-used reinforcement learning algorithm, to find an optimal policy for performing automatic waveform design.

In the following, we introduce the model that is used to define the elements in $\mathcal{M}$ for this problem and then, in the next section, we model the radar waveform problem using the introduced RL terminology to then specialize it to discuss a particular use case of interest.

B. Radar Waveform Model

First, we introduce the radar system model that will be used to define the environment with which the agent will interact. We consider that the radar system can transmit radar pulses in irregular pulse reputation intervals (PRI). The transmit intervals Δ_n are assumed to be bounded as $\Delta_{\min} \leqslant \Delta_n \leqslant \Delta_{\max}$. The maximum number of pulses to be transmitted is set to N. However, the agent is allowed to use less number of pulses ($n \leqslant N$). Each transmit pulse $s_n(t)$ can be selected from a dictionary, i.e., a finite set of pulses with different modulations or a finite set of (optimization) algorithms to define modulations. The size of the dictionary is K. In addition, each of the transmit pulses can be different. The pulses' duration is bounded as $\tau_{\min} \leqslant \tau_n \leqslant \tau_{\max}$.

In this setting, we consider that after the last pulse is being sent the radar remains receiving until the last range of interest arrives. That is, there is a trailing region in the waveform with duration $\tau_L = (2R_{max}/c) + \tau_{max}$. As a result, for a maximum allowable time $T_{\max}$, the following holds

$$T_{\text{design}} := \sum (t_n + \Delta_n) + \tau_L \leqslant T_{\max}. \tag{1}$$

An illustration of such a waveform is shown in Fig. 2.

III. Radar Waveform Design: A Reinforcement Learning Formulation

In this section, we discuss how to pose the task of waveform optimization (WO) as a decision-based problem. To do so, we first define the learning *environment* where the agent will make decisions and then we introduce the RL method on which our work is based.

A. Environment Definition

To define a decision-based problem for WO, we first define our *action space* $\mathcal{A}$. Here, we consider an agent capable of selecting at the nth decision step the

- duration of the pulse $\tau_n \in \mathcal{T} := [\tau_{\min}, \tau_{\max}]$
- separation between pulses $\Delta_n \in \mathcal{D} := [\Delta_{\min}, \Delta_{\max}]$
- modulation scheme $k \in \mathcal{K} := \{1, \ldots, K_{\max}\}$

Thus $\mathcal{A}$ is defined by the Cartesian product of the sets, i.e., $\mathcal{A} = \mathcal{T} \times \mathcal{D} \times \mathcal{K}$, defining the constraints on the parameters. Note that while k is discrete, τ_n and Δ_n are continuous. Also, note that with this action set the number of pulses within a waveform are defined by the number of actions that the agent takes (length of episode) and not the maximum allowable.

Similarly, the *state (or observation) space* $\mathcal{S}$ is defined as a set containing all possible chains of actions together with any extra information provided by the user. Specifically, the state of the environment at the nth decision is defined as the variable-length vector

$$s_n = (a_1, \ldots, a_n, e_1, \ldots, e_n) \tag{2}$$

where $e_n \in \mathcal{E} \; \forall \; n$ is the extra information provided by the user. Example of this information could be the duty cycle (DC), the integrated sidelobe levels (ISL) or the maximum sidelobe level (MSL) at particular regions, the duration of the waveform, etc.

In our WO scenario, and in the absence of user-defined extra information [cf. $\mathcal{E}$] the *transition function* T is straightforward

as the relation between actions and states is clear from (2). Although when user-defined information is present, T can be extremely difficult to define, for the selected RL approach, only the ability of (stepping) forwarding the environment, i.e., perform the transition, is needed.

The *reward function* R on the hand is goal dependent and it is directly linked with the FOM under which the waveform will be evaluated. For example, the reward could be related to the autocorrelation and/or ambiguity function of the generated waveform, or to its time/power budget. In addition, in instances when an algorithm-in-the-loop design is considered, the reward function can be directly linked to the performance of such algorithm, e.g., accuracy of estimates or probability of detection/false alarms. In this section, we do not discuss further this aspect as it is deferred to the Section IV where a particular use case of this framework is discussed.

Constraints or Penalties?: As in the case of classical optimization, the parametrization of the design variables and the definition of their constraints are critical for the success of RL. The proper parametrization of the design variables reduces the need to add (penalization) terms to the FOM (reward function) to promote constraints easing its design. Further, it helps guaranteeing that the design meets directly the desired constraints. In the case of RL, most of the constraints that are related to the actions, e.g., non-overlapping pulses [cf. 2] or setting a particular separation (Δ_n) to a given value, are straightforward to consider as they relate to a proper parametrizations of the variables or to the definition of the T function. An example of the former is our choice for the separation between pulses instead of positions of the pulses. An example of the latter is, for instance, our implicit condition that $\Delta_1 = 0$ as this delay is irrelevant.

Other types of constraints, for example those related to objectives resulting from set of actions, need to be enforced either through a *game-stopping criterion* or by considering them in the reward function. For instance, the requirement that the number of pulses or the total waveform time do not exceed N and $T_{\max}$, respectively, are typically implemented as stopping criteria, i.e., if the agent reaches the respective limits, the episode (current agent's try) stops and a new one is started. However, constraints such as minimum transmit energy most likely need to be considered within the FOM. A proper environment design should focus on limiting the addition of terms to the FOM such that the overall objective is easier to define.

B. Reinforcement Learning Algorithm

We now proceed to describe the PPO RL algorithm, the advocated method for training the agent for WO. Here, we provide an overview of the method, illustrating the cores ideas, their usage in this particular setting and why it is a sensible algorithmic choice. For a more detailed discussion, the reader is referred to [19]. In addition, we discuss our perceived difference between an optimizer and a designer when RL is used. In addition, we also provide a small discussion about other global optimization methods and RL.

Proximal Policy Optimization Algorithm: PPO is a policy gradient method that optimize a parametrized *stochastic* policy $\pi_\theta(\cdot)$ with respect to a modified version of the expected return function

$$L(\theta) = \hat{\mathbb{E}}_t\big[\pi_\theta(a_t|s_t)\hat{A}_t\big] \tag{3}$$

by stochastic gradient descent. Here, θ represents the parameters of the policy, $\hat{\mathbb{E}}_t[\cdot]$ denotes the empirical average over a finite batch of samples and $\hat{A}_t$ is an estimator of the advantage function at timestep t. Without discussing the details, we need to interpret $\hat{A}_t$ as a measure of the *quality* of an action given a particular state [19]. Note that this method searches the space of policies through θ, and thus avoids assigning values to state-action pairs. This property alleviates the exploration challenges appearing when dealing with continuous variables. Typically, the policy $\pi_\theta(\cdot)$ is implemented through an artificial neural network architecture or by a function including domain knowledge. In this work, we consider the former for the example in the next section.

In addition to have the properties of policy gradient methods, PPO also leverages ideas of trust-region policy optimization (TRPO) [20] to achieve data efficiency and reliable performance while using only first-order optimization. This leads to simpler-to-implement methods and a lower sample complexity. The cost function optimized by the used PPO version is given by

$$J(\theta) = \hat{\mathbb{E}}_t\big[\min\big\{r_t(\theta)\hat{A}_t, \mathrm{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\big\}\big] \tag{4}$$

Here,

$$\min\{a,b\} = \begin{cases} a & : a \leqslant b \\ b & : a > b \end{cases}, \ \mathrm{clip}(a,b,c) = \begin{cases} b & : a < b \\ a & : b \leqslant a \leqslant c \\ c & : a > c \end{cases}$$

and $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{\mathrm{old}}}(a_t|s_t)$ measures deviations from the previous policy. Further, the $\min$ and the clip avoid excessively large policy updates and enforce a *pessimistic* lower bound for performing updates. Besides (4), other extra terms are typically added such as an entropy term to control the randomness of the policy and promote exploration.

In summary, PPO allows for a good compromise between having few "moving parts" to be tuned, ease of implementation and sample complexity by aiming to find a new not-too-different policy from the old one. In addition, as it is general, it can be used in a model-free setting, i.e., no information about the environment [cf. T]. It also allows for a mix of discrete and continuous variables in a straightforward manner. In particular, as PPO is considered a time-efficient algorithm, when the policy is small, i.e., its evaluation is not the bottleneck, and the environment is "fast" (not a heavy simulation or no physical system required), due to its simplicity, PPO is considered a good algorithmic choice. When the latter is not the case, i.e., slow environment, alternatives such as DQN [21] or SAC [22] could be better options.

C. Optimizer or Designer

Typically, an optimizer selects a FOM (cost function) and runs an algorithm to find a solution that maximizes/minimizes the FOM. This requires that every time the FOM is modified, e.g., a change in the minimum required transmit energy, the (optimization) procedure needs to be run again, possibly from scratch. When the optimization procedure can be done off-line even if it takes long time, this tends to not be a problem. However, for the inset of such approaches in low latency applications, a different strategy might be needed. This is when what we referred to as a *designer* comes into play.

Here, a designer is an "agent", as the one discussed in the RL framework, that is endowed with a policy $\pi_\theta(\cdot)$ capable of providing solutions to problems with a parametrized FOM, i.e., $\mathrm{FOM}(\phi)$. Here ϕ denotes the parameters that give shape to the FOM. Instances of these problems are those where, for example, the minimum values for operational requirements are defined in ϕ.

The design problem thus can be tackled by considering the parameters ϕ as part of the states of the environment; that is, as user-provide information [cf. (2)]. As can be expected, the increased of dimensionality of the state vector would lead to a larger experience pool, i.e., more episodes, to provide a policy capable to deal with all different FOM instances. However, while this effort can be devoted off-line (during policy optimization), the evaluation of the policy could be orders of magnitude faster than solving the optimization problem for a particular instance (even if other global optimization method is used). As it is shown in an example in Section IV, this characteristic makes RL a strong candidate for bringing full radar adaptivity one step closer to real-time implementation.

D. Other Global Optimization Methods

Global optimization has a long history and several families of methods. Among the most popular methods are those based on evolutionary arguments [23], e.g., genetic algorithms (GA), and those based on sequential design strategies such as in Bayesian optimization literature, e.g., [24]. However, several of these methods require i) intensive empirical hyper-parameter tuning, ii) tailored-made extensions for variable dimensional optimization problems, and iii) their extension to deal with the design problem is challenging. Despite these issues, the fact that they are optimization methods implies that they can be used *within* the RL framework to find the optimal policy.

Note that by itself, GAs have found extensions to deal with ii) through variable-length genome algorithms (VLGA), these extensions require the modification of the genetic operators. These adaptations are not trivial as they are problem dependent and are a subject of research in their own, see, e.g., [25] and reference therein. Similar issues arises for other evolutionary algorithms such as particle swarm optimization, see, e.g., [26].

On the other hand, while Bayesian optimization (BO) methods are highly competitive methods for problems with a small (fixed) number of tunable parameters and expensive black-box FOMs, they tend to struggle on high dimensional problems and large sample budgets. These make its stand-alone usage unattractive for the WO problem. However, as mentioned before, modifications of classical BO methods have been recently shown to be competitive to optimize policies for RL applications, see, e.g., [24]. That is, BO, as well as GAs, can find application within RL, as RL encompasses concepts beyond solely optimization.

In the following, we instantiate our discussion of this section and present a couple of examples to illustrate the usage of the described RL techniques for the WO problem.

IV. Illustrative Examples

To demonstrate the capabilities of the discussed framework for the WO problem, in this section, we explore a series of examples that highlight the characteristics of the RL-based approach for WO.

A. Optimization of MSL with DC Constraints

Our first example consists on designing a pulsed waveform with the objective of

(i) achieving a particular duty cycle DC_0 while
(ii) maintaining the MSL of a region of interest $\mathcal{R}$ at a desired level MSL_0.

To achieve these goals we introduce the following FOM

$$\mathrm{FOM} = \frac{|\mathrm{DC} - \mathrm{DC}_0|}{\mathrm{DC}_0} + \max\left\{\frac{\mathrm{MSL} - \mathrm{MSL}_0}{|\mathrm{MSL}_0|}, 0\right\} \quad (5)$$

whose first term is the normalized error with respect to the targeted duty cycle and the second promotes achieving the targeted MSL. Note that due the lower clipping (with the $\max\{,\}$) designs with lower MSL than targeted are not preferred over those meeting the goal. Here, a waveform with $\mathrm{FOM} = 0$ meets the design requirements.

As reward function R to optimize the policy, we consider the following function

$$R(n) = \begin{cases} 0 & \forall\, n : n < N \text{ and } T_{\mathrm{design}} < T_{\max} \\ -\mathrm{FOM} & \text{otherwise} \end{cases} \quad (6)$$

Note that here we have opted for not providing any intermediate reward to the agent. That is, the agent receives a reward (or penalization) only when the limits on the time and/or number of pulses budget have been reached. This choice has been made to avoid computing the FOM every time the agent makes a decision and to resemble a black-box-function global optimization method.

For the pulse modulations, we consider a dictionary consisting on variations of the widely-used chirp signal. The three variants ($K = 3$) considered are: 1) up chirp , 2) down chirp and 3) up-down chirp. This selection is arbitrary and any other pulse modulations could have been considered. Here, we opt for variations of the chirp due to their pervasiveness in current radar systems and to ease the exposition. The parameters of the waveform are constrained as: $\mathcal{T} := [1/B, T_{\max}/15]$, $\mathcal{D} := [10/B, T_{\max}/15]$ and $N = 32$. And the given MSL and DC targets are $-40\mathrm{dB}$ and 10%, respectively. Here, we have selected the normalized (with respect to the total waveform

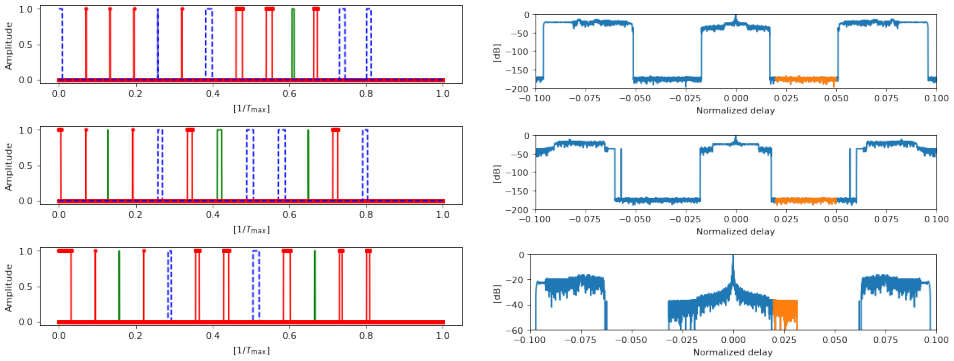


Fig. 3. Waveforms generated with the optimized policy for $\mathcal{R} := [0.020, 0.05]$. (top) Deterministic policy. (Middle and Bottom) Stochastic policy. (Right column) ACF with $\mathcal{R}$ highlighted. (Left column) Transmission times and modulations: (red) down- (blue) up- (green) up-down chirp.

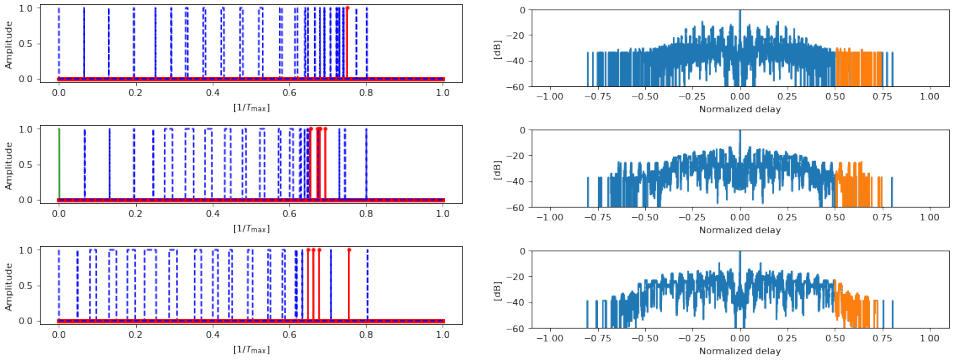


Fig. 4. Waveforms generated with the designer policy for $\mathcal{R} := [0.5, 0.75]$. (Right column) ACF with $\mathcal{R}$ highlighted. (Left column) Transmission times and modulations: (red) down- (blue) up- (green) up-down chirp.

length) range region of interest as $\mathcal{R} := [0.020, 0.05]$. An illustration showing the optimized waveform and its autocorrelation (ACF) function is shown in Fig. 3 (top). This waveform achieves a DC $= 10.64\%$ and a MSL < -40dB. After the policy has been optimized, we proceed to explore its available design space. In Fig. 3 (middle-bottom), we show two random waveforms generated by the optimized policy. Notice that despite that in the temporal domain difference seems to be minor, the order in which modulations are used indeed offers certain degree of variability. This is expected as it seems that most of the suppression in the ACF comes from the time distribution of the waveform. Further investigation of how to increase the diversity in the time parameters, by modifying the stochasticity of the final policy, is topic for future work.

B. Designer for MSL and DC Constraints

We now shift our attention towards the design formulation of the problem. To do so, we endow the agent with extra observations [cf. $\mathcal{E}$ in (2)]. The extra observations are given to the agent at the beginning of each episode (game) and the DC_0 and MSL_0, respectively. That is, every time that the agent will attempt to design a waveform, its state (list of actions in this case) will contain the vector $\boldsymbol{e} := [\text{DC}_0, \text{MSL}_0]^\top$ added to it.

Once the policy has been optimized, by providing to the agent the target DC and MSL, a random waveform can be synthesized without having to run another optimization problem. An example of the designed waveforms by our agent is shown in Fig. 4. Here, we have selected as region of interest $\mathcal{R} := [0.5, 0.75]$. In Fig. (4), the waveforms have (DC, MSL)-targets $\{(5\%, 24.9\text{dB}), (10\%, 23.5\text{dB}), (17\%, 13.4\text{dB})\}$, respectively. Finally, to illustrate the performance of the optimized designer policy, in Fig. 5, we show the targeted and the achieved values of a sample of designed waveforms.

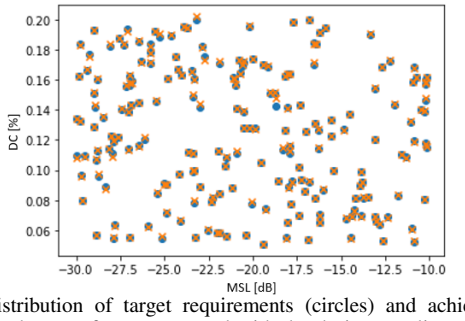


Fig. 5. Distribution of target requirements (circles) and achieved values (crosses) for the waveforms generated with the designer policy.

We remark that though here we opted to design the waveforms with respect to the MSL and DC, the optimization with respect to different regions of interests can be done by following the same procedure, i.e., adding the intervals represented, for example, with a binary mask to the state of the environment.

V. Conclusions and Future Directions

In this paper, we introduced a RL-based framework for waveform optimization. We proposed a particular parametrization to describe the waveform to be optimized and a definition

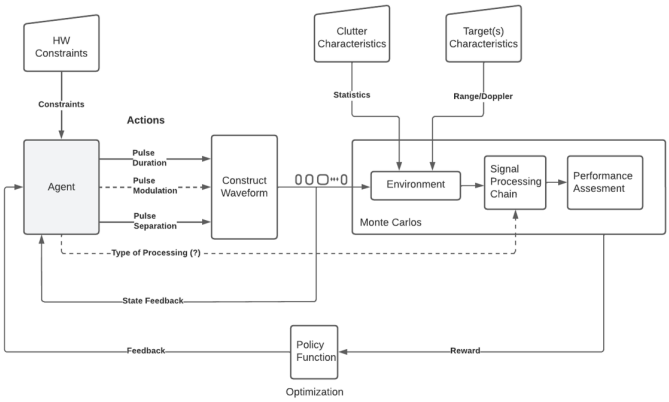


Fig. 6. Illustration of complete action-observation-reward cycle for a simulated radar chain. Here, we consider the possibility of also adapting the processing to the environment.

for the environment explored by the RL agent. In addition, we discussed the implementation aspects, general recommendations and insights of PPO, the chosen RL algorithm. We also introduce the design problem and our perceived difference with the traditional optimization problem setting, as well as the relation of RL with other global optimization methods. Finally, we illustrate the capabilities of the selected RL method for radar waveform optimization and design by considering the problem of learning a policy to select the number of pulses, pulses duration and modulations of a pulsed waveform.

As direct extension, we could consider instances with more complex environments, e.g., including Monte Carlo simulations with specific clutter, jammer and targets signals. Further, it would be of interest to allow the agent to select the type of processing or algorithm, e.g., linear or nonlinear, that best fit the current environment. An illustration of this setting is shown in Fig. 6. These may allow to consider problems where outputs distributions are intractable due to the complexity of the implemented processing chain.

In addition, besides using the presented RL to devise designers, we can also consider the agents as potential *testers*. That is, given a particular chain, potentially adaptive, we can make use of RL to find edge configurations, i.e., parameters with unacceptable FOM values, in an automatic manner.

In our view, the RL ability to i) explore the vast space of parameters in a smarter manner than pure brute force, ii) structure experience-based knowledge through a policy and iii) learn fast-to-evaluate policies may bring full adaptivity a step closer to operational radar systems.

REFERENCES

[1] J. R. Guerci, "Cognitive radar: A knowledge-aided fully adaptive approach," in *2010 IEEE Radar Conference*. IEEE, 2010, pp. 1365–1370.

[2] P. John-Baptiste Jr, "Advancing fully adaptive radar concepts for real-time parameter adaptation and decision making," Ph.D. dissertation, The Ohio State University, 2020.

[3] M. Brandfass, J. Meyer-Hilberg, A. Dallinger, and H. Appel, "Towards cognitive radar via knowledge aided processing for airborne and ground based radar applications," in *2019 20th International Radar Symposium (IRS)*. IEEE, 2019, pp. 1–10.

[4] G. Cui, A. De Maio, A. Farina, and J. Li, *Radar waveform design based on optimization theory*. SciTech Publishing, 2020.

[5] X. Du, A. Aubry, A. De Maio, and G. Cui, "Hidden convexity in robust waveform and receive filter bank optimization under range unambiguous clutter," *IEEE Signal Processing Letters*, vol. 27, pp. 885–889, 2020.

[6] M. Coutino, F. Uysal, and L. Anitori, "Waveform-aware optimal window function design for mismatched filtering," in *2022 IEEE Radar Conference (RadarConf22)*. IEEE, 2022, pp. 01–06.

[7] S. D. Blunt and E. L. Mokole, "Overview of radar waveform diversity," *IEEE Aerospace and Electronic Systems Magazine*, vol. 31, no. 11, pp. 2–42, 2016.

[8] E. J. Hughes and C. M. Alabaster, "Medium prf radar prf optimisation using evolutionary algorithms," in *Proceedings of the 2003 IEEE Radar Conference (Cat. No. 03CH37474)*. IEEE, 2003, pp. 192–197.

[9] L. de Martín and W. van Rossum, "Optimization of pulse intervals for unambiguous doppler recovery with oversampled dictionary," in *2022 IEEE Radar Conference (RadarConf22)*. IEEE, 2022, pp. 1–5.

[10] P. John-Baptiste, G. E. Smith, A. M. Jones, and T. Bihl, "Rapid waveform design through machine learning," in *2019 IEEE 8th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*. IEEE, 2019, pp. 659–663.

[11] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[12] C. E. Thornton, M. A. Kozy, R. M. Buehrer, A. F. Martone, and K. D. Sherbondy, "Deep reinforcement learning control for radar detection and tracking in congested spectral environments," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 4, pp. 1335–1349, 2020.

[13] L. Wang, S. Fortunati, M. S. Greco, and F. Gini, "Reinforcement learning-based waveform optimization for mimo multi-target detection," in *2018 52nd Asilomar Conference on Signals, Systems, and Computers*, 2018, pp. 1329–1333.

[14] A. M. Ahmed, A. A. Ahmad, S. Fortunati, A. Sezgin, M. S. Greco, and F. Gini, "A reinforcement learning based approach for multitarget detection in massive mimo radar," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, no. 5, pp. 2622–2636, 2021.

[15] S. Durst and S. Brüggenwirth, "Quality of service based radar resource management using deep reinforcement learning," in *2021 IEEE Radar Conference (RadarConf21)*. IEEE, 2021, pp. 1–6.

[16] L. Wang, J. Peng, Z. Xie, and Y. Zhang, "Optimal jamming frequency selection for cognitive jammer based on reinforcement learning," in *2019 IEEE 2nd International Conference on Information Communication and Signal Processing (ICICSP)*, 2019, pp. 39–43.

[17] P. Liu, Y. Liu, T. Huang, Y. Lu, and X. Wang, "Decentralized automotive radar spectrum allocation to avoid mutual interference using reinforcement learning," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, no. 1, pp. 190–205, 2021.

[18] Q. Yang, Z. Han, H. Wang, J. Dong, and Y. Zhao, "Radar waveform design based on multi-agent reinforcement learning," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 35, no. 10, p. 2159035, 2021.

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

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

[21] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[22] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[23] A. Slowik and H. Kwasnicka, "Evolutionary algorithms and their applications to engineering problems," *Neural Computing and Applications*, vol. 32, no. 16, pp. 12 363–12 379, 2020.

[24] D. Eriksson, M. Pearce, J. Gardner, R. D. Turner, and M. Poloczek, "Scalable global optimization via local bayesian optimization," *Advances in neural information processing systems*, vol. 32, 2019.

[25] V.-P. Ha, T.-K. Dao, N.-Y. Pham, and M.-H. Le, "A variable-length chromosome genetic algorithm for time-based sensor network schedule optimization," *Sensors*, vol. 21, no. 12, p. 3990, 2021.

[26] P. Kadlec and V. Šeděnka, "Particle swarm optimization for problems with variable number of dimensions," *Engineering Optimization*, vol. 50, no. 3, pp. 382–399, 2018.