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

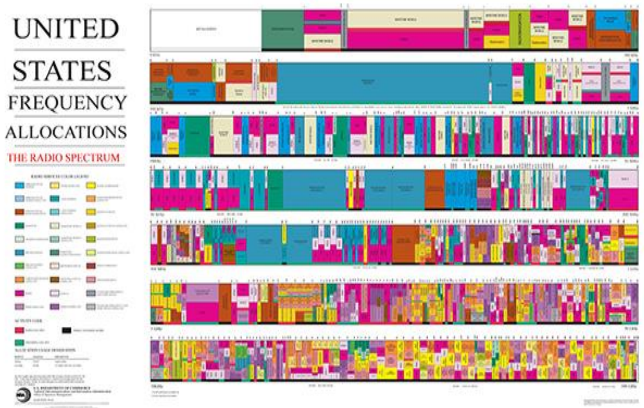
TNO innovation for life

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**

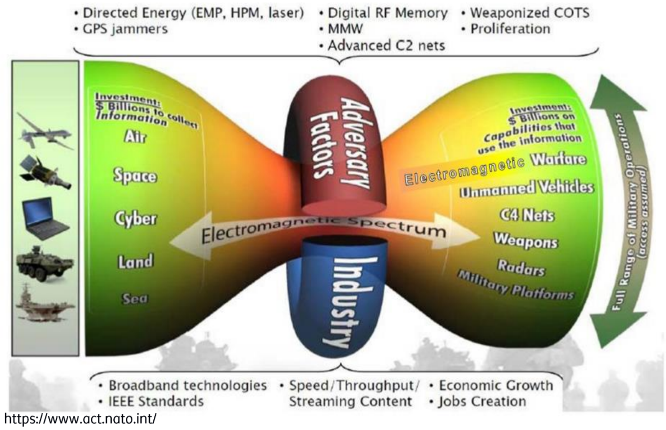
Radar Adaptivity

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



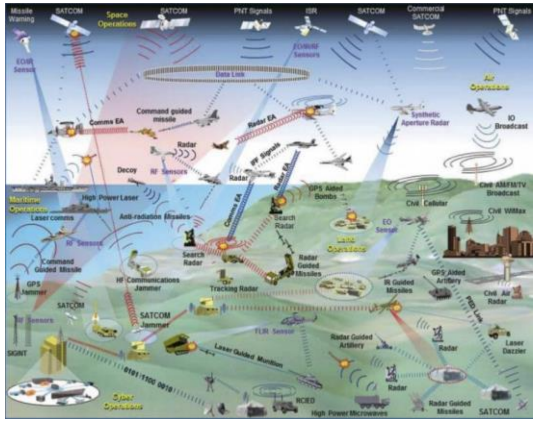
https://ntia.gov/

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 <u>policy</u> to

act given the environment's state.

Enabling Radar Waveform Adaptivity

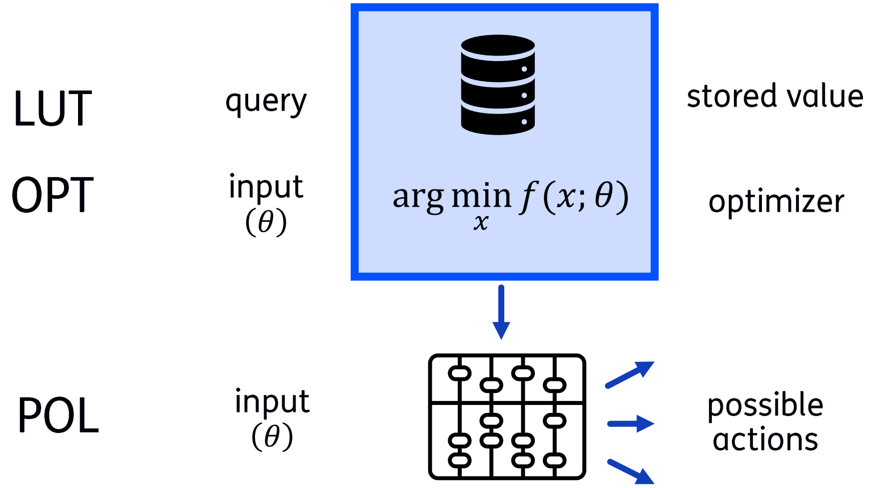
A **policy**

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



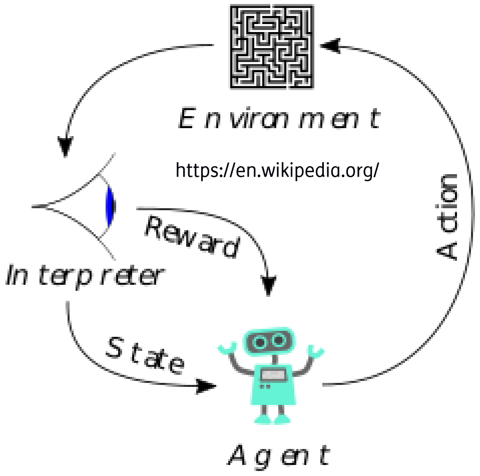
How can we learn a policy for radar waveform design?

Reinforcement Learning

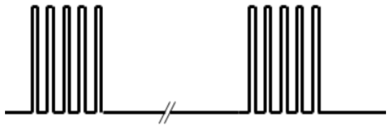
A way to **learn policies** for particular problems is by **reinforcement learning** (RL) algorithms.

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**.

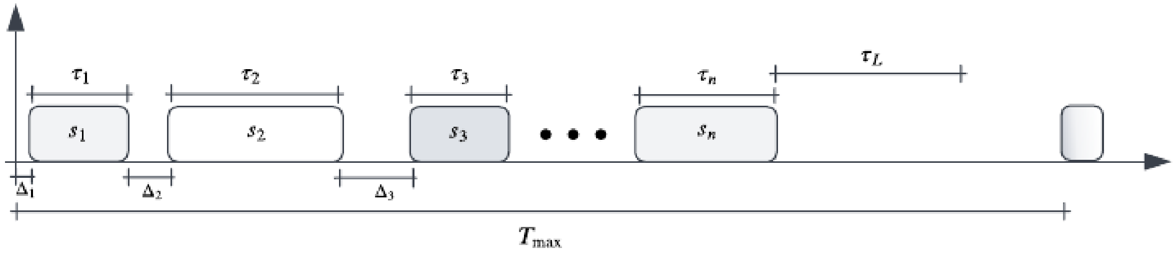
This by interacting with the environment

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

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 :: $\{s_1(t), \ldots s_K(t)\}$
- **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** :: $k \in \mathcal{K} \coloneqq \{1 \ldots, K\}$

Action Space

Reinforcement Learning For Waveform Optimization

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

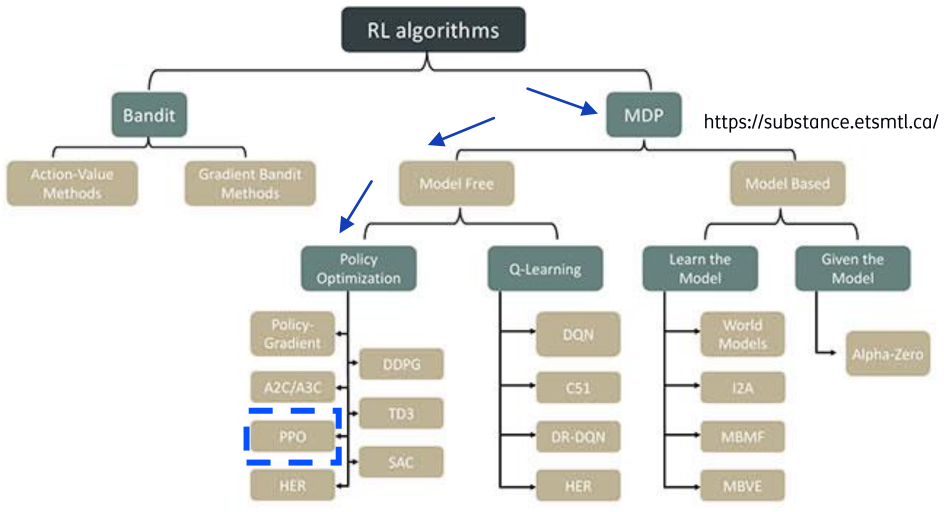
Based on what we want

- **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].



https://substance.etsmtl.ca/

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:

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

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

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

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

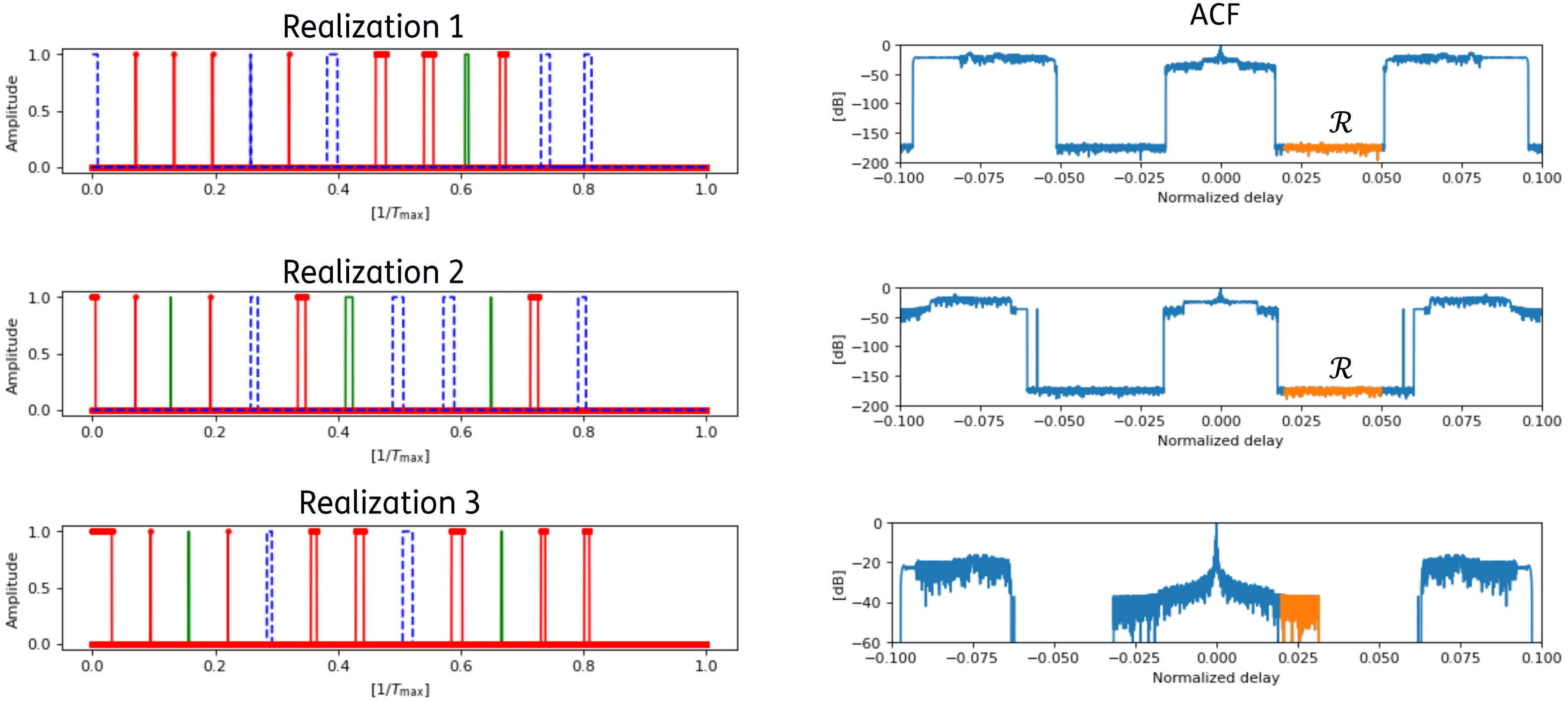
$$\mathcal{D} := \left[\frac{10}{B}, \frac{T_{\max}}{15}\right]$$

$$N = 32$$

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

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

$$\mathcal{R} := [0.020{,}0.05]$$



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

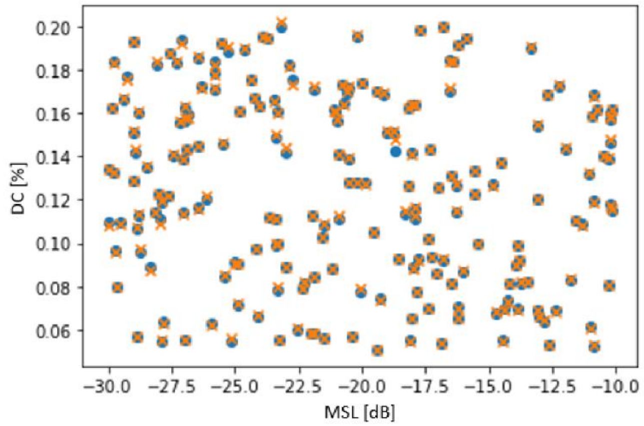
State Space: $\boldsymbol{s_t} = [s_{t-1}, a_{t-1}]; s_0 \coloneqq [\mathrm{DC}_0, \mathrm{MSL}_0]$

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)

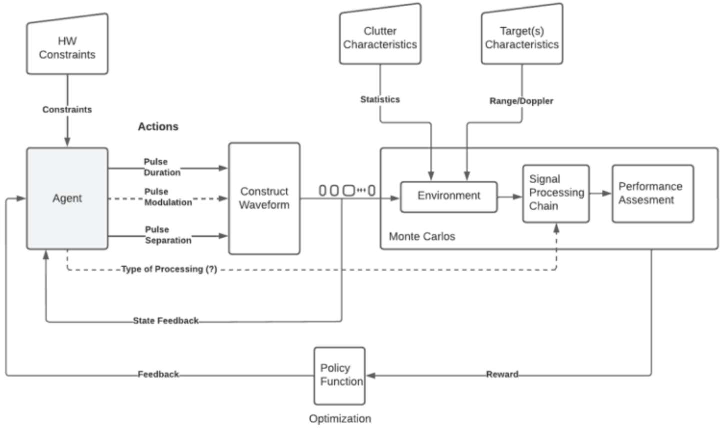


Policy evaluation tend to be fast
(small models)

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