optica

Deep reinforcement learning control of white-light continuum generation

Carlo M. Valensise,[1,*] Alessandro Giuseppi,[2] Giulio Cerullo,[1] and Dario Polli[1]

[1]*IFN-CNR, Department of Physics, Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milan, Italy*
[2]*University of Rome "La Sapienza," Department of Computer, Control and Management Engineering (DIAG), Via Ariosto 25, 00185 Rome, Italy*
**Corresponding author: carlo.valensise@polimi.it*

Received 10 November 2020; revised 6 January 2021; accepted 18 January 2021 (Doc. ID 414634); published 10 February 2021

White-light continuum (WLC) generation in bulk media finds numerous applications in ultrafast optics and spectroscopy. Due to the complexity of the underlying spatiotemporal dynamics, WLC optimization typically follows empirical procedures. Deep reinforcement learning (RL) is a branch of machine learning dealing with the control of automated systems using deep neural networks. In this Letter, we demonstrate the capability of a deep RL agent to generate a long-term-stable WLC from a bulk medium without any previous knowledge of the system dynamics or functioning. This work demonstrates that RL can be exploited effectively to control complex nonlinear optical experiments.

https://doi.org/10.1364/OPTICA.414634

In recent years, a strong synergy has developed between photonics and the computational tools collectively referred to as artificial intelligence (AI) [1], bringing mutual benefits to both disciplines. On one hand, photonic technologies are increasingly employed to improve and speed up data collection, at the basis of every AI application. On the other hand, AI provides robust analytical and predictive tools adapted to a wide variety of photonic contexts: nonlinear spectroscopy [2,3], quantum optics [4], supercontinuum generation in optical fibers [5], and image propagation in diffuse media [6]. In this work, we deepen the link between the two fields by applying reinforcement learning (RL) [7] to automate white-light continuum (WLC) generation, one of the central problems in nonlinear optics. We demonstrate how RL is able to locate, in a multidimensional space and in an unsupervised fashion, a set of parameters able to guarantee a broadband and long-term stable WLC. Previously, machine learning has been used in combination with evolutionary algorithms such as genetic algorithms [8] to tailor the supercontinuum produced in an integrated-photonic chip [9] or to automatically perform mode-locking on a fiber laser [10].

WLC generation is a complex third-order nonlinear process that finds numerous applications in ultrafast optics and spectroscopy. Experimentally, WLC generation is quite simple: a moderately energetic (0.5–3 μJ) femtosecond pulse is tightly focused in a plate of a transparent material (such a sapphire, CaF_2, YAG, YVO_4) with few-millimeter thickness. By suitably controlling the pulse energy, the focused beam divergence and the position of the nonlinear plate with respect to the focal plane, one obtains the formation of a filament and an explosive spectral broadening, resulting in a spectrum that extends to both the blue and the red of the driving pulse spectrum, with a moderate spectral energy density of 10–20 pJ/nm [11]. Due to its spectral extension, high spatial beam quality, and outstanding shot-to-shot energy stability, WLC is employed for seeding of optical parametric amplifiers [12], as the probe pulse in broadband transient absorption spectroscopy [13,14], and in the generation of octave-spanning spectra for characterization of the carrier–envelope phase of few-optical-cycle pulses [15,16]. WLC generation involves the complex combination of several nonlinear optical processes, such as spatial self-focusing, temporal self-phase modulation, self-steepening, and space–time focusing [17,18], as well as group velocity dispersion and plasma generation through multi-photon ionization. The complexity of the underlying spatiotemporal nonlinear optical processes prevents a complete theoretical description of WLC generation and has led to the development of empirical procedures for optimization of WLC properties (bandwidth, energy, stability). Often, a variation in the parameters of the driving laser, due to either fluctuations or change in operating regime (such as, e.g., a different repetition rate) degrades the quality of the WLC, calling for a time-consuming manual adjustment procedure.

RL is a powerful method for the solution to optimization problems that can be formalized as Markov decision processes (MDPs) [19]. The solution is determined by an agent that observes, at each discrete temporal step t, the state s_t of an environment E and is then able to decide actions a_t that affect the environment evolution. After taking an action, a reward $r(s_t, a_t, s_{t+1})$ is given to the agent, reflecting the quality of the action taken given the current state and the subsequent one. The agent's objective is to determine an optimal policy π to perform the best actions so as to maximize the expected cumulative reward. Initially the agent has no *a priori* knowledge of the internal functioning or dynamics of the environment, and the optimal policy is determined by the agent through direct experience of the behavior of the environment. RL agents are currently at the center of intensive research, especially for what concerns solutions based on deep learning. Deep neural networks (NNs) [20] are powerful mathematical tools that allow the agent to understand complex environments, and learn how to obtain good rewards (see Supplement 1). Deep RL agents were demonstrated to be particularly successful in complex tasks such as playing video games [21,22] and traditional games [23]. Moreover, deep RL is

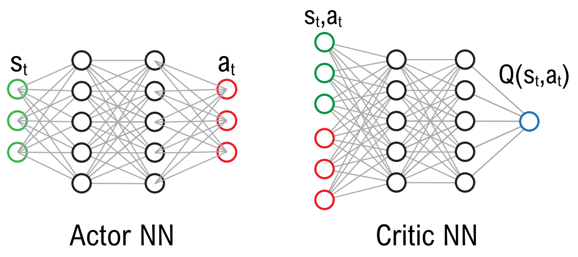

Fig. 1. Scheme of the actor and critic NN of the RL agent. The actor NN takes as input the state and outputs an action; the critic NN takes as input the state–action pair (s_t, a_t) and outputs the state–action value function. See Supplement 1 for a full description of the two NNs.

used for control tasks such as robotics and autonomous driving [24], with recent contributions focusing on critical features for real-world application, namely, safety [25] and constrained control [26]. In this work, we propose and demonstrate that deep RL is a powerful tool also within the nonlinear optics research field. In particular, we train a deep RL agent to control and optimize a strongly nonlinear process such as WLC generation.

A broad and stable WLC spectrum requires the optimization of at least three degrees of freedom, namely, the energy of the pump pulse, the numerical aperture of the focused beam, and the position of the nonlinear plate with respect to the beam waist. All these quantities are continuous, making the task suitable, from a deep RL standpoint, to be solved through an "actor–critic" architecture [27]. In general, deep learning-based actor–critic solutions use a pair of NNs trained with different purposes (Fig. 1): the actor network approximates the policy $\pi(s)$, the mapping between states and actions, while the critic network approximates the so called state–action value function $Q(s, a)$ [7], a quantity that, for each state–action pair (s, a) approximates the cumulative reward that can be obtained starting in (s, a) and following the policy π thereafter (see Supplement 1). Several training processes for the two networks have been proposed in the last few years. One of the first and most widely used deep RL solutions for the actor–critic setting is the deep deterministic policy gradient (DDPG) [24]; to solve the WLC generation problem, we employed its recent evolution called twin delayed DDPG (TD3) [28], which represents the state of the art in the field [29].

The optical setup used for WLC generation is sketched in Fig. 2. It starts with a fiber-based ytterbium laser system (Coherent Monaco) providing pump pulses with ≈300 fs duration at 1030 nm, at a variable repetition rate up to 50 MHz, which we

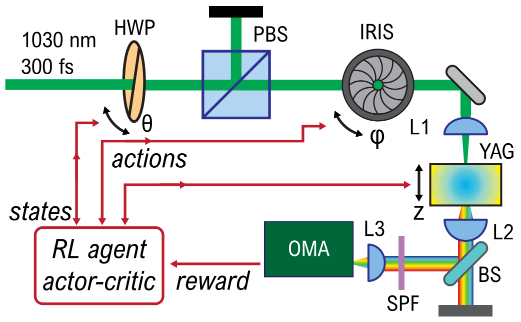

Fig. 2. Optical setup. HWP, half-wave plate; PBS, polarizing beam splitter; L1, focusing lens (focal length 5 cm); L2, collimating lens (focal length 2 cm); BS, beam splitter T:R-90:10; SPF, short-pass filter; L3, focusing lens (focal length 3 cm); OMA, optical multichannel analyzer; RL, reinforcement learning agent observing the state of the environment, taking actions on it, getting rewards in turn.

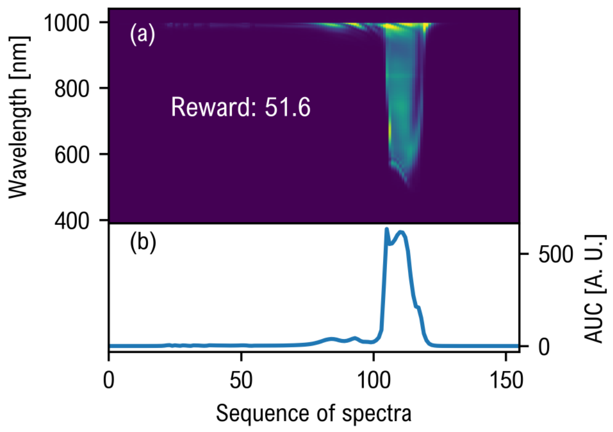

Fig. 3. Example of transition between two states. (a) Spectra collected during the transition reported along with the (b) corresponding area under the curve. The sum of the values in (b) divided by the number of acquired spectra (proportional to the length of the dynamics) gives the scalar reward.

fixed at 2 MHz, and a pulse energy up to 80 μJ, which we fixed at ≈1.25 μJ. A motorized rotary stage (Thorlabs, PRM1Z8) varies the orientation angle ϑ of a half-wave plate (HWP), followed by a polarizing beam splitter (PBS), so as to finely control the power reaching the crystal. The aperture of an iris (set at an angle ϕ) is controlled by a second motorized rotary stage (Thorlabs, K10CR1) to vary the numerical aperture of the pump beam focused on a 6 mm thick YAG crystal via lens L1 with 5 cm focal length. The position (z) of the crystal with respect to the laser focus can be varied by a third motorized actuator (Thorlabs, MT1-Z8). After the interaction, lens L2 with 2 cm focal length collimates the generated WLC, which is then filtered by a short-pass filter (Thorlabs, FESH1000) to reject the fundamental beam and the long-wavelength lobe that is not detectable by the silicon spectrometer employed (Ocean Insight, Flame VIS-NIR). To reduce the light intensity, a beam splitter (BS) with a transmittance/reflectance ratio of 90/10 is placed before the spectrometer. No multiple filamentation or crystal damage was observed with this experimental configuration. Python programming language was used to control the actuators [30] and read the spectrometer [31].

To express the WLC generation problem as a MDP, we define the state of the system as the vector $s = (\vartheta, \phi, z)$. The actions correspond to movements of the actuators to absolute positions, and thus are expressed in a similar way: $a = (\rightarrow \vartheta, \rightarrow \phi, \rightarrow z)$. Each action taken by the agent is evaluated with respect to the goal of generating a broad WLC spectrum, by computing a scalar reward $r(s_t, a_t, s_{t+1})$. During the movement of the three actuators, a series of output spectra is recorded by the spectrometer (set at 3 ms integration time). This batch of spectra enables a better estimation of the action–value function since it does not depend on only the last position, but contains information about the entire movement [Fig. 3(a)]. Next, each spectrum is normalized and integrated, obtaining a sequence of values [Fig. 3(b)] that are finally averaged to get a single scalar value to be used as a reward for the action.

The training procedure for a TD3 deep learning agent is divided into episodes, each lasting for T_e time steps. During the training, all the transitions (s_t, a_t, s_{t+1}, r_t), composed by the initial state, the action taken, the next state, and the reward received, are stored in the so-called replay-buffer, which gathers all the acquired knowledge of the system. At the beginning of the training process, usually the environment is explored by performing random actions, so that the agent may gather some starting data. After a few episodes of random exploration, a sufficient amount of transitions is present in the replay-buffer, and the agent can start the training of its actor and critic NNs. As the agent trains, its knowledge on the system increases. As customary in RL solutions, to maintain active the exploration of the environment and improve the learned policy to obtain higher rewards, the actions produced by the actor network are randomly perturbed. This ensures that the agent experiences possibly more rewarding states that are not considered within its (current) policy, and avoids being attracted towards locally optimal policies.

After each episode is completed, the system is randomly reset to a new initial condition. To assess the improvements of the learned policy after some training episodes, the agent undergoes some evaluation episodes in which its actions are not perturbed. This evaluation procedure allows to determine whether the agent has completed the training (i.e., is able to obtain sufficiently high rewards) or if it needs to undergo a new training phase. We refer the reader to Supplement 1 for a more detailed description of the training procedure.

Figure 4 reports an exemplary complete experiment of our RL agent controlling WLC generation. We selected $T_e = 50$ time steps for each episode, and used the first four episodes for random sampling. After that, three evaluation episodes are then performed to check the agent knowledge just after the random initialization. Clearly, at this point the agent is not able to produce positive rewards (i.e., to generate WLC). During the first exploration phase (time steps 350–450), some positive rewards are observed by the agent, also due to the random perturbation of its actions that causes the controller to try new strategies. However, it is only during the third exploration period (time steps 600–700) that the policy is consolidated to systematically obtain positive rewards. In fact, during the third and final set of evaluations (time steps 700–850), the reward obtained by the agent is always positive and reaches high values, meaning that the WLC is constantly generated. Note that, despite following a deterministic policy, the obtained reward varies in intensity among the three last evaluations. This is due to the different starting point of each evaluation and to the strong nonlinear nature of the WLC generation process, which determines different spectra even for slight variations in the actuators' positions, as shown in Fig. 5.

Figure 6 reports the details of the beginning of an evaluation episode, once the correct policy is determined by the agent. The system starts in a random state, and the actor NN is used to predict the next position given the current state. Already after four steps the system converges to a stable state for WLC generation. In Fig. 6(d), the spectra corresponding to the first four steps are sequentially reported, to highlight the onset of WLC. Notably, the final spectrum is not the most intense of the series; this behavior is commonly found in RL solutions, as the agent aims at maximizing the cumulative reward in the future. The strong nonlinearity of the underlying process does not guarantee that all transitions with high rewards are reproducible; therefore, the agent moves towards states that give lower but more constant rewards.

To assess the long-term quality of the generated WLC, after the agent convergence to a stable WLC generation configuration, its activity was blocked and a sequence of $N = 7000$ spectra ($\{\mathbf{S}\}_N$) was acquired during 40 min, at a rate of ≈3 spectra per second. To evaluate the overall stability we computed the auto-correlation between the recorded spectra as a function of the delay τ:

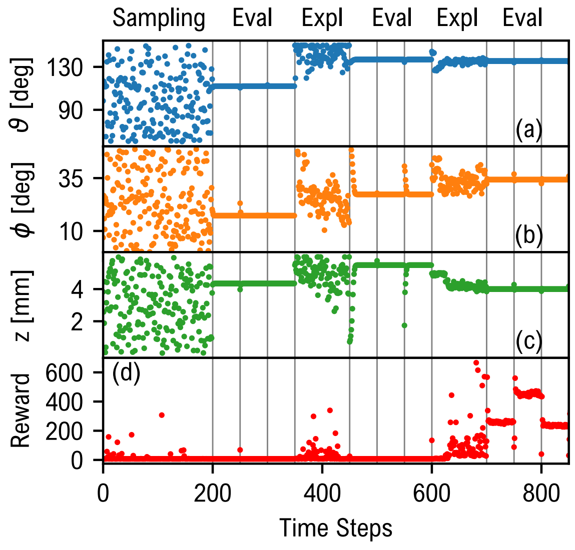

Fig. 4. Example of training of RL agent for WLC generation. (a)–(c) All positions of the actuators during the whole experiment. The different phases of the experiments are highlighted by vertical gray bars. (d) Reward corresponding to each transition. The experiment is concluded when the policy is able to provide a stable solution for WLC generation.

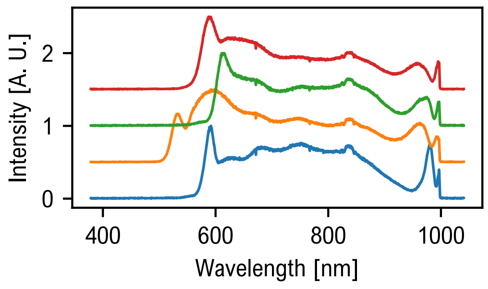

Fig. 5. WLC spectra produced by RL agent after training. Differences in shapes are due to the strong nonlinearity of the process and random nature of parts of the training procedure.

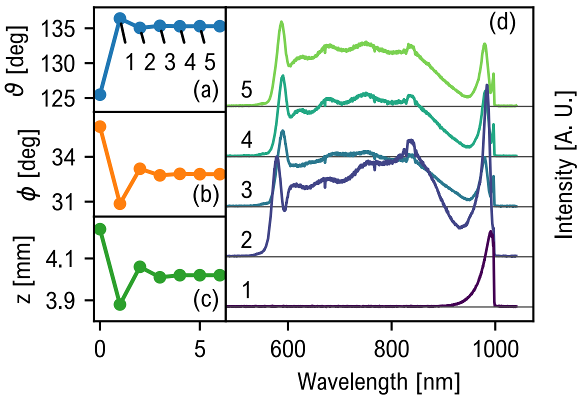

Fig. 6. Example of evaluation episode. (a)–(c) Positions of the three actuators. (d) Spectra corresponding to the first transitions of the episode reported sequentially.

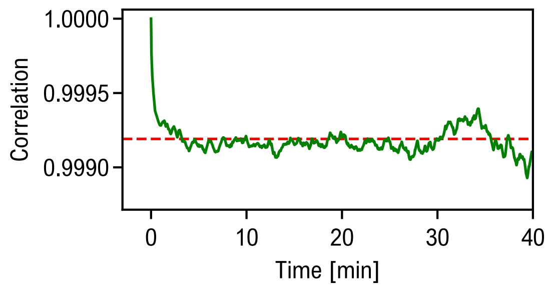

Fig. 7. Correlation function for 7000 WLC spectra sampled in 40 min. See Eq. (1).

$$C(\tau, \{\mathbf{S}\}_N) = \frac{1}{n-t} \sum_{i=0}^{n-1-\tau} \mathbf{S}_i \cdot \mathbf{S}_{i+\tau}. \tag{1}$$

The results are shown in Fig. 7. Very high correlation values (>99,9%) along the whole sampling period are observed, thus confirming the good stability reached by the AI-generated WLC.

To conclude, in this work, we demonstrated the capability of RL, currently the state of the art for model-free control, to handle and optimize a complex and strongly nonlinear optical process such as WLC generation. The agent is able to learn an effective policy of control of three degrees of freedom that, correctly combined, allow stable and broadband WLC generation from a YAG crystal. This proof-of-concept result may be helpful in experimental contexts in which careful optimization of light sources is required before performing the actual experiments, such as nonlinear spectroscopy, often requiring broadband and stable pulses, or quantum optics, to control and tailor the spontaneous parametric down conversion process, by which entangled photon pairs are generated. More generally, multi-parametric processes that require precise automated control may exploit RL as a control tool. Photonics can benefit from AI not only to process data after measurements, but also to control complex experimental processes.

Funding. Horizon 2020 Framework Programme (101016923); Ministero dell'Istruzione, dell'Università e della Ricerca (R164WYYR8N); Regione Lombardia (POR FESR 2014-2020).

Disclosures. The authors declare no conflicts of interest.

Supplemental document. See Supplement 1 for supporting content.

REFERENCES

1. K. Goda, B. Jalali, C. Lei, G. Situ, and P. Westbrook, APL Photon. **5**, 070401 (2020).
2. C. M. Valensise, A. Giuseppi, F. Vernuccio, A. D. la Cadena, G. Cerullo, and D. Polli, APL Photon. **5**, 061305 (2020).
3. R. Houhou, P. Barman, M. Schmitt, T. Meyer, J. Popp, and T. Bocklitz, Opt. Express **28**, 21002 (2020).
4. V. Cimini, I. Gianani, N. Spagnolo, F. Leccese, F. Sciarrino, and M. Barbieri, Phys. Rev. Lett. **123**, 230502 (2019).
5. U. Teğin, B. Rahmani, E. Kakkava, N. Borhani, C. Moser, and D. Psaltis, APL Photon. **5**, 030804 (2020).
6. B. Rahmani, D. Loterie, E. Kakkava, N. Borhani, U. Teğin, D. Psaltis, and C. Moser, Nat. Mach. Intell. **2**, 403 (2020).
7. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. (MIT, 2018).
8. D. E. Goldberg and J. H. Holland, Mach. Learn. **3**, 95 (1988).
9. B. Wetzel, M. Kues, P. Roztocki, C. Reimer, P.-L. Godin, M. Rowley, B. E. Little, S. T. Chu, E. A. Viktorov, D. J. Moss, A. Pasquazi, M. Peccianti, and R. Morandotti, Nat. Commun. **9**, 4884 (2018).
10. G. Pu, L. Yi, L. Zhang, and W. Hu, Optica **6**, 362 (2019).
11. M. Bradler, P. Baum, and E. Riedle, Appl. Phys. B **97**, 561 (2009).
12. C. Manzoni and G. Cerullo, J. Opt. **18**, 103501 (2016).
13. S. A. Kovalenko, A. L. Dobryakov, J. Ruthmann, and N. P. Ernsting, Phys. Rev. A **59**, 2369 (1999).
14. U. Megerle, I. Pugliesi, C. Schriever, C. F. Sailer, and E. Riedle, Appl. Phys. B **96**, 215 (2009).
15. M. Kakehata, H. Takada, Y. Kobayashi, K. Torizuka, Y. Fujihira, T. Homma, and H. Takahashi, Opt. Lett. **26**, 1436 (2001).
16. A. Baltuška, T. Udem, M. Uiberacker, M. Hentschel, E. Goulielmakis, C. Gohle, R. Holzwarth, V. S. Yakovlev, A. Scrinzi, T. W. Hänsch, and F. Krausz, Nature **421**, 611 (2003).
17. J. K. Ranka, R. W. Schirmer, and A. L. Gaeta, Phys. Rev. Lett. **77**, 3783 (1996).
18. A. L. Gaeta, Phys. Rev. Lett. **84**, 3582 (2000).
19. E. A. Feinberg and A. Shwartz, eds., *Handbook of Markov Decision Processes* (Springer, 2002).
20. R. Hecht-Nielsen, in *International Joint Conference on Neural Networks* (IEEE, 1989), .
21. V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," arXiv:1312.5602 (2013).
22. V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, Nature **518**, 529 (2015).
23. D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, Nature **529**, 484 (2016).
24. 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:1509.02971 (2019).
25. F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, in *Advances in Neural Information Processing Systems* (MIT, 2017), pp. 908–918.
26. A. Giuseppi and A. Pietrabissa, IEEE Control. Syst. Lett. **4**, 755 (2020).
27. R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, in *Advances in Neural Information Processing Systems* (MIT, 2000), pp. 1057–1063.
28. S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," arXiv:1802.09477 (2018).
29. A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, and Y. Wu, "Stable baselines," 2018, https://github.com/hill-a/stable-baselines.
30. T. Gehring, "Thorlabs APT," 2019, https://github.com/qpit/thorlabs_apt.
31. K. Sunden, "Python seabreeze," 2019, https://python-seabreeze.readthedocs.io/.