Multi-Agent Proximal Policy Optimization Applications in Low-Dropout Regulator Design

Thang Quoc Nguyen*, Lihong Zhang* Octavia A. Dobre*, Trang Hoang†, Trung Q. Duong*‡
* Memorial University, Canada, e-mail: {nqthang, lzhang, odobre, tduong}@mun.ca
† Ho Chi Minh City University of Technology (HCMUT), VNU-HCM, Vietnam, e-mail: hoangtrang@hcmut.edu.vn
‡ Queen's University Belfast, U.K, e-mail: trung.q.duong@qub.ac.uk)

***Abstract*—In analog and mixed-signal integrated circuits (ICs), low-dropout regulators (LDOs) are crucial for maintaining a stable power supply throughout the IC. As such, designing LDOs with both time and quality efficiency has attracted substantial research interest. This paper presents an implementation of multi-agent proximal policy optimization (MAPPO) in both separated-parameter and parameter-sharing configurations to address the challenges of multi-objective, multi-variable LDO design. Our experiments show that parameter-sharing MAPPO outperforms both separated-parameter MAPPO and single-agent PPO in exploration and convergence, benefiting from cooperative learning via parameter sharing, which accelerates the identification of optimal design configurations. In summary, our findings indicate that parameter-sharing MAPPO efficiently manages complex specifications and variables.**

***Index Terms*—proximal policy optimization (PPO), multi-agent proximal policy optimization (MAPPO), low-dropout regulator (LDO), multi-agent reinforcement learning (MA-RL).**

I. Introduction

In analog and mixed-signal integrated circuits (ICs), low-dropout regulators (LDOs) play a critical role in power management by providing a stable, "clean" power supply to various components within System-on-Chip (SoC) architectures [1]. The ideal LDO output voltage should be robust against process, supply voltage, and temperature variations (PVT), exhibiting minimal noise and maintaining stability under varying load conditions and environmental changes. Furthermore, LDOs should minimize the dropout voltage, defined as the difference between the input and output voltages. These characteristics make LDOs critical in diverse applications, including biomedical sensors [2], cameras, memory devices [3], and satellite systems [4].

Designing LDOs is a complex task that typically involves three stages: topology selection, sizing, and layout. The sizing stage, in particular, is time-consuming and highly reliant on designer expertise due to inherent non-linearities and intricate trade-offs among performance specifications [5]. Consequently, significant research effort has been directed towards automating the sizing process for LDOs and other analog circuits.

Traditional approaches to automated sizing often employ evolutionary algorithms, such as genetic algorithms (GA) [6] and particle swarm optimization (PSO) [7]–[9]. However, these heuristic methods can be susceptible to local optima, particularly in high-dimensional design spaces with numerous local optima. Bayesian optimization (BO) offers an alternative [10], [11], but its computational cost scales cubically with the number of samples [5], potentially limiting its applicability in large-scale problems. Additionally, BO's sample efficiency can be a concern as the number of design variables increases.

Recently, reinforcement learning (RL) has emerged as a promising technique for addressing optimization challenges in IC design. Using neural networks as function approximators, RL algorithms can effectively model complex relationships and learn from experience. Previous studies [12], [13] have explored the use of deep deterministic policy gradient (DDPG) method for LDO sizing problem. However, these studies often simplify the design problem by assuming a constant reference voltage, neglecting the design of the bandgap reference (BGR) circuit that generates this voltage in practical LDO implementations. This simplification may stem from the limitations of single-agent RL in handling the interactions between interdependent circuit blocks.

This study aims to overcome these limitations by employing a multi-agent reinforcement learning (MARL) approach, specifically Multi-Agent Proximal Policy Optimization (MAPPO). MAPPO has demonstrated superior performance in multi-agent environments, including telecommunications [14] and game playing [15]. Recent work [16] has also shown its potential for analog circuit design. However, a comprehensive comparison between MAPPO and single-agent PPO in this context is lacking, as is an investigation into the impact of different multi-agent settings, such as parameter-sharing and separated-learning, on MAPPO's performance. This research addresses these gaps by systematically evaluating the performance of MAPPO under different multi-agent configurations and comparing it to single-agent PPO in the challenging task of LDO and BGR co-design.

This article presents the following main contributions:

1) Introduces a multi-agent framework for LDO design, enabling the decomposition of the complex optimization problem.
2) Employs MAPPO with both parameter-sharing and separated-parameter schemes to optimize LDO sizing.
3) Demonstrates the superior performance of parameter-sharing MAPPO over separated-parameter MAPPO and single-agent PPO through a comparative analysis.

The remaining sections of this paper are organized as follows: Section II introduces LDO and MAPPO. Section III

Identify applicable funding agency here. If none, delete this.

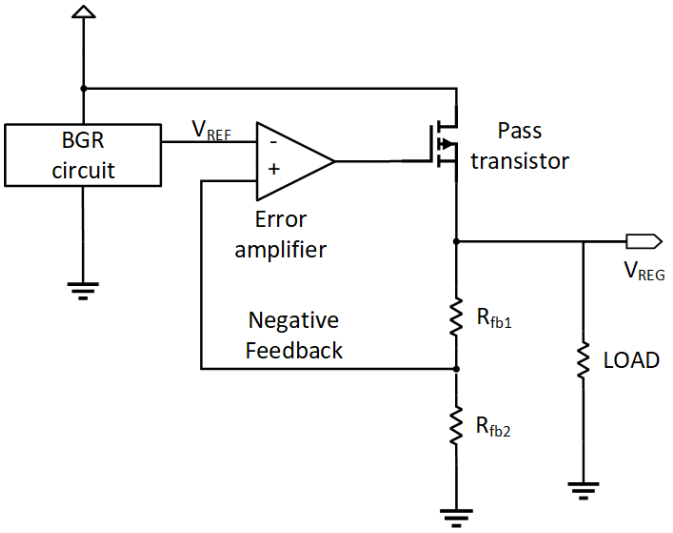

Fig. 1. Schematic of LDO circuit

details the optimization framework. Section IV discusses the optimization results. Section V concludes the paper.

II. Methodology

A. Low-Dropout Regulator (LDO)

Fig. 1 illustrates a typical LDO architecture. The circuit comprises several key components: a pass element, an error amplifier, and a BGR circuit. The pass element, commonly an N- or P-channel MOSFET, serves as a variable resistor responsible for the regulation of the output voltage.

To maintain a stable output voltage, the LDO utilizes a negative feedback loop. The output voltage is compared with a reference voltage generated by the BGR circuit, and any difference is amplified to adjust the gate voltage of the pass transistor. This dynamic adjustment ensures precise output voltage regulation despite fluctuations in load current or power supply voltage.

Under steady-state conditions, the LDO exhibits a linear relationship between the output V_{REG} and its reference voltage V_{REF}: $V_{REG} \approx \left(1 + \frac{R_{fb1}}{R_{fb2}}\right) V_{REF}$. This linear relationship demonstrates the LDO's ability to maintain a consistent output voltage despite any environment affect.

To ensure the LDO's output voltage quality, several key metrics are evaluated, as shown in Table I. The first six rows cover overall LDO performance, including power supply rejection ratio (PSRR) and phase margin at both maximum (10 mA) and minimum (10 μA) loads. The last four rows list performance metrics for the BGR sub-block.

In analog IC design, multi-objective optimization is typically addressed by converting multiple objectives into a Figure of Merit (FoM), often represented as a weighted sum prioritizing key objectives. Similarly, in reinforcement learning, agents aim to maximize reward functions. Thus, in optimization tasks, the objective can be directly defined as the agents' reward when applying RL for optimization.

TABLE I
Specifications of LDO circuit

Metrics	Constraint
Dropout voltage	$\leq$ 200 mV
Load regulation	$\leq$ 36 mV
$PSRR_{1kHz}$	$\leq$ -30 dB
$PSRR_{1MHz}$	$\leq$ -20 dB
Phase margin	$\geq 60°$
Quiescent current	$\leq$ 200 μA
BGR's temperature coefficient	$\leq$ 10 $ppm/°C$
V_{REF}	$\approx$ 0.9 V
BGR's $PSRR_{1kHz}$	$\leq$ -70 dB
BGR's $PSRR_{1MHz}$	$\leq$ -20 dB

B. Multi-Agent Proximal Policy Optimization (MAPPO)

A detailed description of the MAPPO is provided in Algorithm 1. In contrast to value-based reinforcement learning algorithms, MAPPO is hypothesized to be robust against the issue of value function overestimation.

Algorithm 1: MAPPO

Initialize actors and critics parameters of all agents.
Initialize data buffer D.
for step $= 1, 2, ..., step_{max}$ **do**
  for $i = 1$ **to** B **do**
    Initialize state of actors and critics.
    Trajectory $\tau = []$.
    for $t = 1$ **to** T **do**
      Each agent executes action a_t, get reward r_t and next state s_{t+1}.
      $\tau = \tau \cup [s_t, a_t, r_t, s_{t+1}]$.
      Compute $\hat{A}$ and $\hat{R}$.
      Split τ into small data chunks.
      for c **in** chunks **do**
        $D = D \cup \left[\tau[c], \hat{A}[c], \hat{R}[c]\right]$.
      end
    end
  end
  Select a random mini-batch b from D.
  Update $L(\phi)$ and $L(\theta)$ using b.
end

Similar to single-agent PPO, MAPPO employs an actor-critic architecture. The actor network learns a policy to select actions that optimize the cumulative reward, while the critic network evaluates the quality of those actions and guides the actor's learning process. The MAPPO framework operates on the principle of centralized training with decentralized execution, integrating the benefits of both centralized and decentralized methods. During centralized training, each critic has access not only to its local observations but also to shared state information and actions from all agents, facilitating more cooperative policy development. The centralized critic

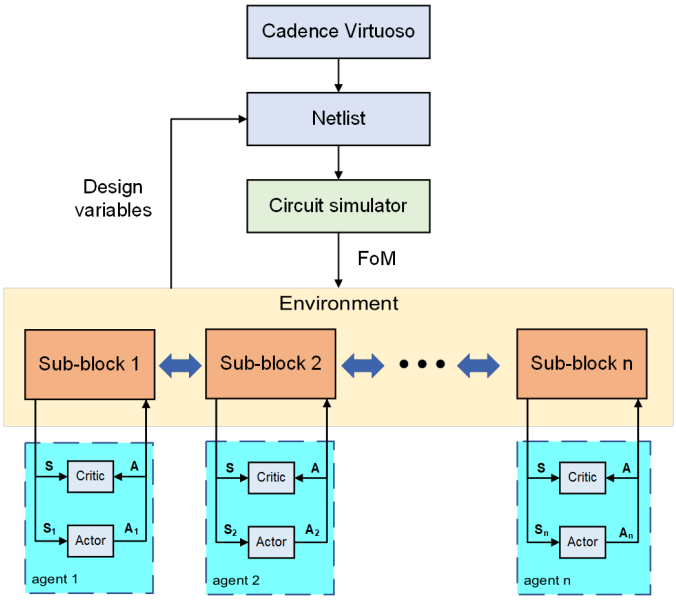

Fig. 2. Implemented multi-agent optimization framework

is trained to minimized the loss function shown in Eq. (1):

$$L(\phi) = \frac{1}{Bn}\sum_{i=1}^{B}\sum_{k=1}^{n}\max\left[(V_\phi(s_i^k) - \hat{R}_i)^2, (L_{\phi,k}^{CLIP})^2\right], \quad (1)$$

where $L_{\phi,k}^{CLIP} = \text{clip}(V_\phi(s_i^k), V_{\phi old}(s_i^k) - \epsilon, V_{\phi old}(s_i^k) + \epsilon)$, V_ϕ is the value function, $\hat{R}_i$ is the discounted return, B is batch size and n is the number of agents.

In decentralized execution phase, global information and critics are not required; each agent bases its actions solely on its local observations, thus reducing extraneous information in decision-making step. In each iteration, the actors utilize the trained decentralized policy to sample trajectories. These trajectories are aggregated to compute advantage values and to maximize the following objective function shown in Eq. (2):

$$L(\theta) = \frac{1}{Bn}\left[\sum_{t=1}^{B}\sum_{k=1}^{n} L_{t,\theta,k}^{CLIP} + \sigma\sum_{i=1}^{B}\sum_{k=1}^{n} S[\pi_\theta(s_t^k)]\right], \quad (2)$$

where $L_{t,\theta,k}^{CLIP} = \min\left(r_{t,\theta}^k \hat{A}_t^k,\ \text{clip}(r_{t,\theta}^k, 1-\epsilon, 1+\epsilon)\right)$, S is the entropy function which encouraging exploration, and σ is entropy coefficient.

In cooperative multi-agent RL, agents can learn with separate or shared parameters. While separate policies allow for individual learning, parameter sharing enhances efficiency when agents are homogeneous [17]. By learning from collective experiences, agents accelerate training and reduce the number of parameters, leading to more effective outcomes [18]. This study will confirm these benefits in IC sizing, demonstrating improved training efficiency and parameter optimization.

III. Optimization Process

A. Optimization framework overview

Figure 2 presents the LDO sizing optimization framework, divided into the circuit and algorithm blocks. In the circuit block, the LDO schematic with variable design parameters is drawn in Cadence Virtuoso, generating a netlist file that is then simulated using Spectre. The algorithm block implements three reinforcement learning algorithms—PPO, parameter-sharing MAPPO, and separated-parameter MAPPO—coded in Python. Identical hyperparameters are applied across all algorithms to ensure a fair comparison.

The LDO under examination is a circuit with 28 design variables. Within the scope of MAPPO investigations, these variables are organized into four sub-components. Each agent is assigned a single sub-component, thus managing only seven variables each.

The output actions generated by the agents, governed by the activation function of the neural network, fall within a normalized range of [-1, 1]. To translate these outputs into practical sizing values for the IC components, denormalization is applied following the regulation shown in Eq. (3), which is introduced by [13]:

$$\boldsymbol{A} = \frac{1}{2}\left[\hat{\boldsymbol{A}} \odot (\boldsymbol{A}_{max} - \boldsymbol{A}_{min}) + (\boldsymbol{A}_{max} + \boldsymbol{A}_{min})\right], \quad (3)$$

where $\hat{\boldsymbol{A}}$ is the normalized action, $\boldsymbol{A}_{max}$ and $\boldsymbol{A}_{min}$ are the upper bound and lower bound of the design space respectively.

After denormalization, elements of the action vector are assigned to their corresponding design variable in the netllist for the upcoming circuit performance simulation step. After simulation, the FoM value is sent to the algorithm block for evaluation.

B. Reward

Following the reward formulation in [19], in this work, the reward associated with each performance metric is defined in Eq. (4).

$$r_i = \min\left(\frac{o_i - o_i^*}{o_i + o_i^*}, 0\right). \quad (4)$$

where o_i is simulated performance metric and o_i^* is the target value of o_i. This reward structure can encourage the agent to achieve performance metrics that meet or exceed the predefined targets.

Furthermore, certain performance metrics may require special consideration due to their criticality in specific applications. In the design of BGR and LDO circuits, PSRR is of paramount importance. Additionally, for LDOs, the stability of the feedback loop, often characterized by phase margin, is crucial. To prevent these critical metrics from degrading to unacceptable levels, specialized reward functions are introduced, as shown in Eq. (5) and Eq. (6).

$$r_{PSRR} = \begin{cases} -1 & \text{if } PSRR \geq 0\ dB \\ \min\left(\frac{o_i - o_i^*}{o_i + o_i^*}, 0\right) & \text{if } PSRR < 0\ dB \end{cases} \quad (5)$$

$$r_{PM} = \begin{cases} -1 + \min\left(\frac{o_i - o_i^*}{o_i + o_i^*}, 0\right) & \text{if } PM \geq 45\ dB \\ \min\left(\frac{o_i - o_i^*}{o_i + o_i^*}, 0\right) & \text{else} \end{cases} \quad (6)$$

The overall reward for each agent is then calculated as a weighted sum of the individual rewards associated with each

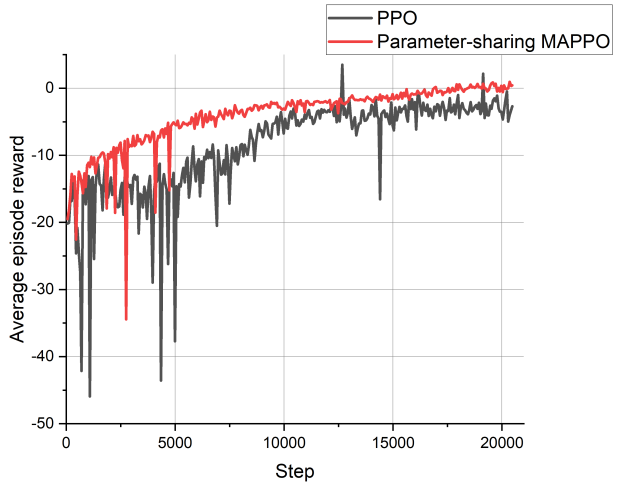

Fig. 3. Average episode reward versus step result for PPO and parameter-sharing MAPPO

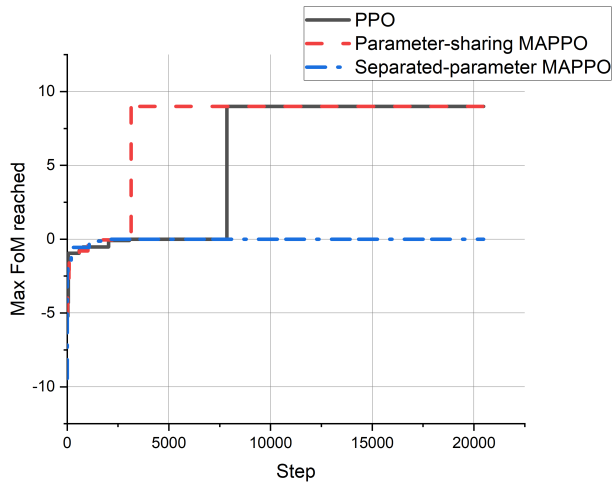

Fig. 4. Comparison of three algorithms in terms of maximum FoM value reached

performance metric: $R = \sum w_i r_i$. In the parameter-sharing MAPPO scenario, all agents share the same reward function, promoting cooperation towards a common objective. Conversely, in the separated-sharing scenario, each agent possesses a unique reward function.

C. Observation

The study defines the observation space for each agent to include the state of 24 MOSFETs in the LDO circuit. The state vector of each transistor S_i is a vector with key electrical characteristics: drain source current I_{DS}, transconductance g_m, drain source conductance g_{DS}, drain source voltage V_{DS}, threshold voltage V_{th}, and saturation voltage V_{dsat}. In MAPPO scenarios, each RL agent observes the six transistors in the state to efficiently capture the dynamics of the LDO system. Before sneding to the agents, state vectors are normalized as described in Eq. (7).

$$\hat{S}_i = \frac{S_i - \mu}{\sigma}, \tag{7}$$

where μ and σ denote the mean and variance of the DC operating point of MOSFETs, respectively.

After normalization, each observation is clipped to [-5, 5], ensuring values remain within a manageable range for the neural network. This process prevents extreme values from impacting training, fostering a more stable learning environment.

IV. Result and Discussion

Fig. 3 compares average episode rewards for single-agent PPO and parameter-sharing MAPPO. Parameter-sharing MAPPO demonstrates stronger early exploration, with rewards steadily increasing from -20 to about -4 within the first 5000 steps. Meanwhile, PPO's rewards oscillate below -10 during this period. Over time, MAPPO shows consistent progress, surpassing a reward of 0 in the last 200 steps, At the same period, PPO fluctuates around -3, exhibiting two peaks at 3.49 and 2.18, and one sharp decline to -16.55 at step 14,450 before stabilizing back around -3, indicating less stable learning.

TABLE II
Comparison with state-of-the-art

	DAC'20 [12]	ICCAD'23 [13]	**This work**
Algorithm	DDPG	DDPG	PPO MAPPO
Employed circuit	LDO	LDO	LDO
Number of specifications	6	6	13
Number of variables	N/A	13	28

Fig. 4 further compares algorithm performance through the maximum FoM achieved during training. Parameter-sharing MAPPO reaches the highest FoM value at 9 within 3147 steps, while PPO requires 7856 steps to reach the same level. Separated-parameter MAPPO lags, only reaching a maximum FoM of 0. These results confirm that parameter-sharing MAPPO offers superior exploration and more efficient convergence, benefiting from shared learning among agents, which accelerates the discovery of an optimal design configuration. Table II provides a comparative analysis of our findings with conventional approaches, which reveals the fact that our proposed scheme outperforms other benchmarks in terms of larger number of handling variables and specifications.

V. Conclusion

This paper presents an implementation of MAPPO in both separate-parameter and parameter-sharing configurations, aimed at addressing the challenges in designing the LDO circuit. Through our experimental analyses, we have demonstrated that the parameter-sharing MAPPO configuration yields superior optimization outcomes. We have also shown that owing to the robust performance of multi-agent reinforcement learning, our proposed scheme can efficiently handle a greater number of specifications and variables compared with conventional approaches.

REFERENCES

[1] T. Limpisawas and W. Wattanapanitch, "A Low-Power Wide-Load-Range Output-Capacitorless Low-Dropout Voltage Regulator With Indirect-Direct Nested Miller Compensation," *IEEE Access*, vol. 10, pp. 67 396–67 412, 2022.

[2] Z. Wang and S. Mirabbasi, "A 0.58-to-0.9-v input 0.53-v output 2.4- μ w current-feedback low-dropout regulator with 99.8

[3] J. S. Kim, K. Javed, and J. Roh, "Design of a Low-Power and Area-Efficient LDO Regulator Using a Negative-R-Assisted Technique," *IEEE Trans. Circuits Syst. II*, vol. 70, no. 10, pp. 3892–3896, 2023.

[4] A. Privat, P. W. Davis, H. J. Barnaby, and P. C. Adell, "Total Dose Effects on Negative and Positive Low-Dropout Linear Regulators," *IEEE Trans. Nucl. Sci.*, vol. 67, no. 7, pp. 1332–1338, 2020.

[5] T. Q. Nguyen, T. Hoang, L. Zhang, O. A. Dobre, and T. Q. Duong, "A Survey on Smart Optimisation Techniques for 6G-oriented Integrated Circuits Design," *Mobile Netw. Appl.*, vol. 28, p. 2227–2244, 2024.

[6] T. Hoang, T. Q. Nguyen, and H. Phan-Tran-Minh, "Novel GA-OCEAN Framework for Automatically Designing the Charge-Pump Circuit," *IEEJ Trans. Electr. Electron. Eng.*, p. e24129, 2024.

[7] T. Hoang, T. N. Quoc, L. Zhang, and T. Q. Duong, "Novel Methods for Improved Particle Swarm Optimization in Designing the Bandgap Reference Circuit," *IEEE Access*, vol. 11, pp. 139 964–139 978, 2023.

[8] R. Rashid and N. Nambath, "Area Optimisation of Two Stage Miller Compensated Op-Amp in 65 nm Using Hybrid PSO," *IEEE Trans. Circuits Syst. I, Exp. Briefs*, vol. 69, no. 1, pp. 199–203, 2022.

[9] M. Fayazi, M. T. Taba, E. Afshari, and R. Dreslinski, "AnGeL: Fully-Automated Analog Circuit Generator Using a Neural Network Assisted Semi-Supervised Learning Approach," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 70, no. 11, pp. 4516–4529, 2023.

[10] C. Chen, H. Wang, X. Song, F. Liang, K. Wu, and T. Tao, "High-Dimensional Bayesian Optimization for Analog Integrated Circuit Sizing Based on Dropout and gm/ID Methodology," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 11, pp. 4808–4820, 2022.

[11] S. Zhang, W. Lyu, F. Yang, C. Yan, D. Zhou, and X. Zeng, "Bayesian optimization approach for analog circuit synthesis using neural network," in *Proc. Design, Automat. Test Europe Conf. Exhibit*, Florence, Italy, 2019, pp. 1463–1468.

[12] H. Wang, K. Wang, J. Yang, L. Shen, N. Sun, H.-S. Lee, and S. Han, "GCN-RL Circuit Designer: Transferable Transistor Sizing with Graph Neural Networks and Reinforcement Learning," in *Proc. ACM/IEEE Design Autom. Conf.*, San Francisco, CA, USA, Jul. 2020, pp. 1–6.

[13] Z. Li and A. C. Carusone, "Design and Optimization of Low-Dropout Voltage Regulator Using Relational Graph Neural Network and Reinforcement Learning in Open-Source SKY130 Process," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design*, San Francisco, CA, USA, Oct. 2023, pp. 01–09.

[14] D. Guo, L. Tang, X. Zhang, and Y.-C. Liang, "Joint optimization of handover control and power allocation based on multi-agent deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 13 124–13 138, 2020.

[15] C. Yu, A. Velu, E. Vinitsky, J. Gao, Y. Wang, A. Bayen, and Y. Wu, "The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games," in *Proc. Conf. Neural Inf. Process. Syst.*, New Orleans, LA, USA, 2022.

[16] J. Bao, J. Zhang, Z. Huang, X. Feng, Z. Bi, X. Zeng, and Y. Lu, "Multiagent Based Reinforcement Learning (MA-RL): An Automated Designer for Complex Analog Circuits," *IEEE J. Technol. Comput. Aided Design*, pp. 1–1, 2024.

[17] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *Proc. Auton. Agents Multiagent Syst Workshops*. São Paulo, Brazil: Springer, May 2017, pp. 66–83.

[18] F. Christianos, G. Papoudakis, M. A. Rahman, and S. V. Albrecht, "Scaling multi-agent reinforcement learning with selective parameter sharing," in *Proc. Int. Conf. Mach. Learning*. PMLR, 2021, pp. 1989–1998.

[19] K. Settaluri, A. Haj-Ali, Q. Huang, K. Hakhamaneshi, and B. Nikolic, "Autockt: Deep reinforcement learning of analog circuit designs," in *Proc. Design, Automat. Test Europe Conf. Exhibit.*, Grenoble, France, 2020, pp. 490–495.