Real-Time EMT Model Tuning Through Co-Simulation of Optimization Algorithms: Case Studies of a Microgrid Controller

Thomas Holliday, Renuka Loka, Mohanad Elsayed,
Mahmoud Kabalan, Mohamed M. Zakaria Moustafa
Center for Microgrid Research,
University of St. Thomas,
St. Paul, MN 55105,
holl7906@stthomas.edu, loka3402@stthomas.edu, elsa8648@stthomas.edu,
mahmoud.kabalan@stthomas.edu, mohamed.moustafa@stthomas.edu

Thomas Bozada
U.S. Army Engineer,
Research and Development Center
Champaign, IL, USA

***Abstract*—Metaheuristic search algorithms (MHSAs) are widely applied to optimization problems in control systems. Their adaptability and search capability make them effective for controller tuning, particularly in nonlinear or time-variant environments. In hardware-in-the-loop (HIL) and real-time simulation (RTS) contexts, these features are invaluable for automating controller optimization. Yet, integration of MHSAs with RTS platforms for autonomous online closed-loop optimization remains limited. This paper introduces a framework that co-simulates MHSAs and online RTSs to enable real-time, optimization of virtual models. Typhoon HIL libraries allow the Pygad genetic algorithm (GA) to execute, evaluate, and optimize electromagnetic transient (EMT) simulations in real time. The approach is demonstrated by tuning the proportional and integral (PI) gains of a secondary controller in a virtual microgrid (MG). Several case studies confirm its effectiveness, achieving a 98.96% reduction in settling time without overshoot. The results demonstrate the potential of MHSA–RTS integration for controller enhancement across diverse optimization tasks.**

***Index Terms*—Metaheuristic Search Algorithm, Real-Time Simulation, Genetic Algorithm, Controller Optimization, Secondary Control, Co-Simulation**

I. Introduction

Metaheuristic search algorithms (MHSAs) are widely used computational optimization methods with a broad range of applications in control engineering. Although various analytical techniques exist for controller tuning, some control system applications require faster response and better stability than traditional methods can provide. In such cases, MHSAs have been successfully employed to enhance both the speed and robustness of controllers, outperforming classical approaches such as the Ziegler–Nichols method [1]. Numerous studies have analyzed the performance of various MHSAs—including genetic algorithms (GAs), imperialist competitive algorithms (ICAs), particle swarm optimization (PSO), and ant colony optimization (ACO)—for proportional integral derivative (PID) controller tuning [1], [2], [3]. Across these studies, MHSAs consistently demonstrate significant improvements in PID settling time relative to the Ziegler–Nichols approach. Among the algorithms studied, GAs often achieve the best performance. However, outcomes are highly dependent on the choice of objective function [1].

The use of MHSAs extends beyond controller tuning to a variety of power system optimization tasks. For example, power factor optimization is demonstrated in [4], where the GA adjusts the capacitance of multi-step capacitor banks based on measured power factor in both simulations and a physical testbed, resulting in significant improvements in reactive power. In this approach, the GA adjusts several variables in a function for calculating power factor. Once the predicted power factor is within the objective function's constraints, the variables' values are applied to the system [4]. Similarly, in [5], energy consumption in a dual-motor system model is optimized using both GA and PSO, with final results validated through a hardware-in-the-loop (HIL) setup. Bai et al. proposed a heuristic optimization method for power flow in wind-based distributed energy resource (DER) systems, employing an artificial bee colony algorithm to run and evaluate simulations in MATPOWER [6]. Collectively, these studies indicate that MHSAs are effective for a range of power system optimization problems. However, despite their demonstrated capabilities, MHSAs have not yet been applied to evaluate and optimize real-time simulations (RTSs).

Automation of RTSs is primarily discussed within the HIL testing literature. Prior studies have shown that automating HIL tests significantly reduces the testing time [7], [8]. However, achieving accurate results from HIL configurations

Effort sponsored by the U.S. Government under Other Transaction for Prototype Agreement number W9132T239C007 between UNIVERSITY OF ST. THOMAS and the Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Government.

This project was made possible in whole or in part by a grant from the Minnesota Department of Commerce through the Minnesota Renewable Development Account, which is financed by Xcel Energy ratepayers.

979-8-3315-8138-1

also requires automating the underlying RTSs. The benefits of automated RTS testing are highlighted in [9], where predefined tests are automatically simulated and compared against programmable logic controller (PLC) data, resulting in substantially faster execution times. Similarly, relay testing and data analysis are automated in an RTDS-based HIL setup [10], again using predefined test cases. While hard-coded scenarios are effective when parameter values are known in advance, such approaches lack flexibility and become inefficient for optimization-driven testing. Despite the growing importance of optimization and automation in RTS applications, the literature lacks studies addressing the integration of online closed-loop MHSAs for optimizing internal simulation parameters in RTSs. This paper addresses this gap through three key contributions:

- A framework for online, closed-loop co-simulation between MHSAs and RTSs for model validation and controller optimization.
- Optimization and validation of a PI-based secondary controller using a virtual microgrid model.
- An adaptive and scalable co-simulation architecture enabling cross-platform integration through application programming interfaces (APIs).

II. System Description

The Center for Microgrid Research (CMR) at the University of St. Thomas operates a 480 V, 60 Hz, 3 Phase 4 wire microgrid. The CMR Microgrid can be reconfigured to operate with different combinations of DERs in either grid-connected or islanded modes. For this paper, the configuration shown in Fig. 1 is employed, which includes a photovoltaic (PV) array, an energy storage system (ESS), a diesel generator (DiG), a load bank, and utility connections. Active and reactive power at the point of common coupling (PCC), denoted P_{PCC} and Q_{PCC}, are measured and compared against operator-defined setpoints P_{sp} and Q_{sp}. The PI-based secondary controller uses these measurements to calculate individual active and reactive

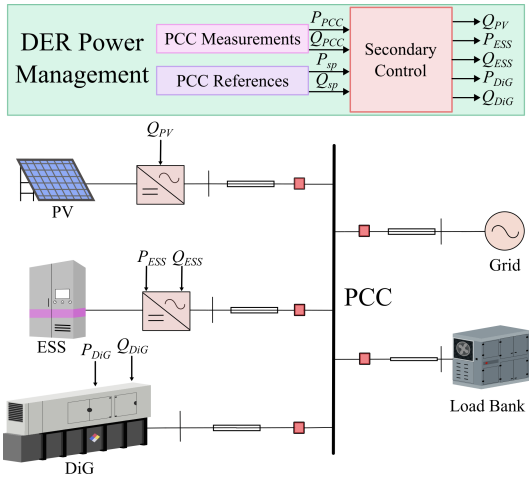

Fig. 1. Schematic of the CMR MG with secondary control.

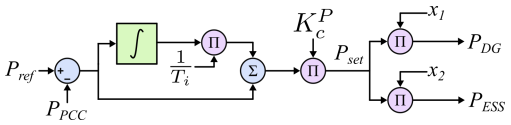

Fig. 2. Diagram of the proportional power sharing secondary control.

power setpoints for each DER, including Q_{PV}, P_{ESS}, Q_{ESS}, P_{DiG}, and Q_{DiG}.

The secondary controller enhances power sharing among DERs when the CMR Microgrid operates in grid-connected mode and stabilizes voltage and frequency during islanded operation. Fig. 2 presents a schematic of the secondary control for active power in grid-connected operation, where x_1 and x_2 represent the gains that determine the proportion of the load supplied by each DER. The integral time constant is denoted by T_i, and the PI controller gain is given by K_c. For additional details regarding the development, validation, and application of the secondary controller, see [11]. Using the proposed co-simulation framework for real-time optimization, the secondary control's PI controllers can be tuned to produce desired dynamic responses in the CMR Microgrid.

III. Methodology

The CMR's Microgrid, shown in Fig. 1, is modeled in Typhoon HIL. Distribution line impedances and internal DER parameters are configured to match real world measurements obtained from the CMR Microgrid. A supervisory control and data acquisition (SCADA) panel is employed during model development to provide operator control while the model executes in real time on a Typhoon HIL 606 simulator. For automated testing, the SCADA panel can be replaced with Typhoon HIL's Python-based TyphoonTest IDE.

Traditionally, TyphoonTest IDE is employed to automate case studies using Python. However, existing examples demonstrate hard-coded cases [12]. While this approach is suitable when parameter values are known in advance, its static nature renders parameter tuning for optimization and validation highly inefficient. To overcome this limitation, an MHSA is co-simulated with the RTS to compute and test different parameter values in real time. The Python script uses the Typhoon HIL API to initialize the model with parameters generated by the MHSA, execute the simulation, repeat the scenario with updated parameters, acquire data, and terminate the simulation. Fig. 3 illustrates the sequence of steps performed by the Python script within the RTS. Real-time scripted control not only improves run-to-run consistency but is also essential for algorithm-driven optimization.

A variety of MHSAs are available, but a GA is selected due to its robustness and widespread use in PID tuning. Pygad, an open-source GA Python library, provides standard GA functionalities [13]. Table I presents the Pygad parameters used for optimization in this research. Parameter values are determined through trial tests to identify configurations that achieve optimized results most efficiently, even prior to full algorithm convergence. Setting the population size close to the number of parents selected for crossover and the number

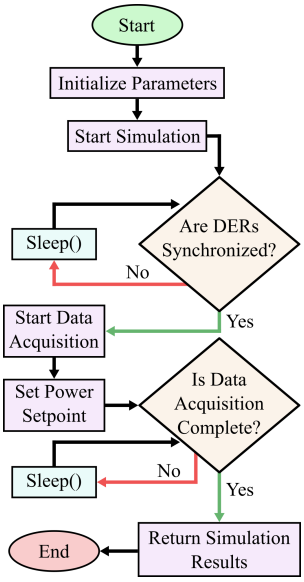

Fig. 3. Flowchart showing actions taken by the python script each time the simulation is called.

of solutions kept per generation ensures higher fitness solutions are preserved from generation to generation while still allowing for exploration. Each PI gain is considered to be a gene, and both share the same minimum and maximum values. With these population and gene definitions, 50 generations of populations suffice for discovery of the highest fitness genes by the GA. The adaptive mutations parameter increases the magnitude of gene value changes for earlier generations, allowing for exploration of the gene-space early on and refinement later. Single point crossover and tournament parent selection maintains diversity in the proceeding generations while maintaining genes with high fitness. During the trial tests, convergence rarely occurred with other mutation types, crossover types, and parent selection methods. Optimal values are also rarely achieved when the possible gene values are not given an appropriate range. Increasing the PI gains beyond 0.5 did not result in a significant reduction in settling time, so 0.5 is used as the maximum value for the gains. Given the nature of MHSAs, optimal or near-optimal gene values can often be identified before convergence is complete. To facilitate this, a gene logging function records the genes and cost values for each simulation, enabling early detection of promising solutions.

Most MHSAs rely on an objective function to quantitatively evaluate the success of a test. In this paper, the secondary controller's PI controllers are tuned to minimize settling time. However, faster settling can also induce instability or overshoot. To balance these factors, a multi-objective function is employed to assess stability, settling time, and overshoot simultaneously. Stability is evaluated based on whether the DERs maintain synchronization with the grid and whether the active power remains within a specified maximum error, δ, at the end of the simulation. If these criteria are not reached, a significant cost penalty, C_p, is added to the final cost C:

TABLE I
PYGAD CONFIGURATION

Pygad Parameter	Value
Number of Generations	50
Number of Parents for Crossover	10
Population Size	14
Number of Genes	2
Solutions Kept per Generation	10
Min and Max Values for Genes	0, 0.5
Mutation Type	Adaptive
Crossover Type	Single Point
Parent Selection Method	Tournament

otherwise $C_p = 0$. Settling time, T_s, is calculated as the first timestep in a moving window where the active power consistently remains within a defined tolerance. Both metrics are formalized in the following equations:

$$N = \left\lceil \frac{t_d}{\Delta t} \right\rceil \tag{1}$$

$$\delta = |P_{sp}|\,\varepsilon \tag{2}$$

$$T_s = \min\{t_i \big| \, |P_{PCC}(t_j) - P_{sp}| \geq \delta \, \forall \, j \in [i, i+N], \; t_i \geq T_{\min}\} \tag{3}$$

where N is the number of timesteps in the moving window; t_d is the window's length in seconds; Δt is the timestep in seconds; ε is the fractional error; t_i is the timestep at index i; $P_{PCC}(t_j)$ is the active power during the window defined by j; and T_{min} is the minimum settling time.

Overshoot O_s is calculated by determining the magnitude error between the active power and the setpoint, and is expressed as follows:

$$O_s = \max_{t \geq 0} \left| P(t) - P_{sp} \right| \tag{4}$$

Cost is defined as the weighted, normalized summation of settling time and overshoot; their weights are W_T and W_O respectively. Settling time and overshoot are normalized with the normalization parameters T_{norm} and O_{norm}, which are the maximum acceptable unit values of settling time and overshoot. Additionally, the cost penalty for instability is added . Depending on its configuration, the MHSA either maximizes fitness, F, or minimizes the cost defined by the objective function. Because Pygad is inherently designed for fitness maximization, objective functions originally expressed for minimization must first convert the cost into a corresponding fitness value, as follows:

$$C = W_T \frac{T_s}{T_{norm}} + W_O \frac{O_s}{O_{norm}} + C_p \rightarrow -C = F \tag{5}$$

In the GA configuration used in this research work, the algorithm systematically generates populations of simulations, referred to as population members. Each population member contains a set of genes, K_c and T_i, along with an associated fitness value. After constructing the population, the fitness of each population member is evaluated. Population members exhibiting the highest fitness are selected as parents for crossover,

determining the genes of the subsequent generation. A subset of the fittest population members is retained in the population to preserve advantageous traits, while all remaining population members are replaced through crossover or mutation. This process is illustrated in Fig. 4.

IV. Simulation Results and Validation

4,441 population members are generated by the GA to optimize the secondary controller's active power response. The generation of each population member takes on average 62.2 seconds, which totals 76.73 hours of runtime. In each simulation, a step change from 0 kW to 15 kW at the PCC is requested while in grid-connected operation. All DERs are available in the MG. The active load is maintained at a constant 45 kW, while the PV is set to supply a constant 5 kW—the secondary controller does not adjust the PV's active power setpoint. Initially, the requested power at the PCC is 0 kW and 0 kVAR, requiring the DiG, ESS, and PV to collectively supply 45 kW, with 11.4 kW and 28.6 kW provided by the DiG and ESS, respectively. Once steady state is reached, -15 kW is requested at the PCC, and the MG begins supplying the grid. During this step change, the goal is to achieve the shortest settling time and minimal overshoot. The fitness values corresponding to each K_c and T_i pair across all 4,441 simulations are shown in Fig. 5. The highest fitness K_c and T_i pair has respective values of 0.0794 and 0.002. These values correspond to population member 2,689 (46.46 hours of runtime), but similar fitness is achieved with parameters found as early as population member 10 (10.36 minutes of runtime). The untuned gains are 0.1 and 1 respectively. Additionally, each case study includes a comparison with the Ziegler-Nichols method described in [14]. The Ziegler-Nichols K_c and T_i gains for active power are 2.272 and 0.0156 respectively and 5.45 and 0.0073 for reactive power.

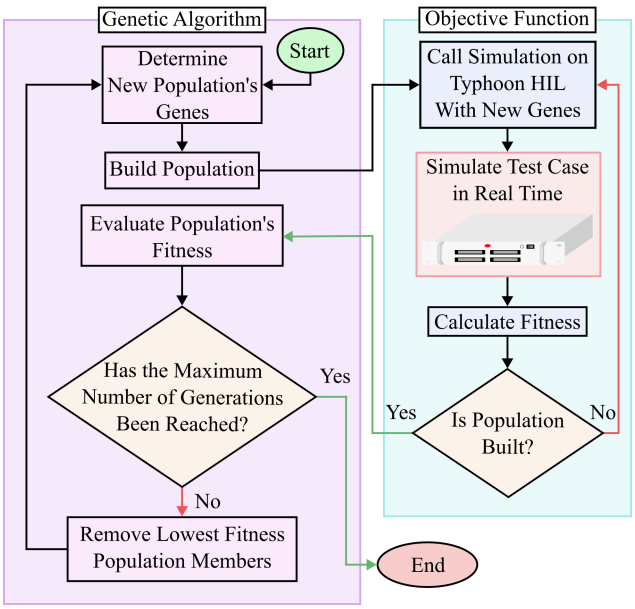

Fig. 4. Flowchart showing the sequence of the GA and objective function.

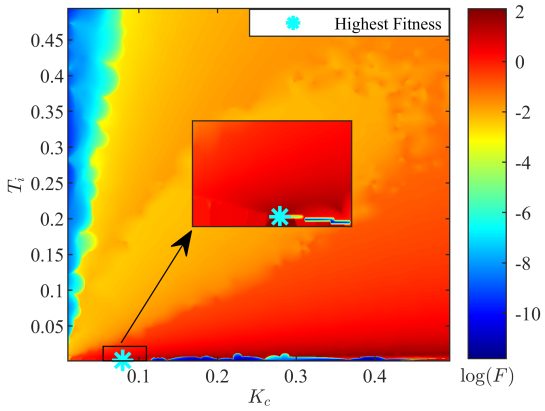

Fig. 5. Contour map of PI gains and their associated fitness on a log scale. Warmer colors indicate higher fitness.

A. Case Study 1

The first case study compares the untuned active power response of the secondary controller, as reported in [11], with the PI gains optimized by the GA. All DERs in the MG are active, with the PV set to supply a constant 25 kW and the load fixed at 40 kW; initially, the requested power at the PCC is 0 kW. At 5 seconds, a -25 kW step change is applied, as illustrated in Fig. 6. With the untuned gains, the MG requires 40 seconds to reach steady state at the requested -25 kW. In contrast, the GA-tuned gains reduce the settling time, T_{S1}, by 98.83%, achieving steady state in just 0.47 seconds without overshoot. Minor oscillations occur immediately before steady state with both the GA gains and Ziegler-Nichols gains. However, oscillations from the GA gains are less significant than those with the Ziegler-Nichols gains. This case proves the framework is successful at optimizing the PI gains.

B. Case Study 2

The second case study demonstrates the grid supplying the MG with both active and reactive power with all DERs available. Additionally, this case tests the efficacy of applying the tuned PI gains to a separate system. The tuned PI gains are determined using simulation data for cases with explicitly

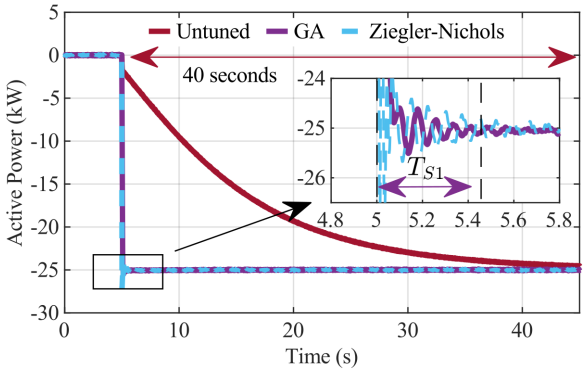

Fig. 6. The CMR Microgrid RTS active power response with untuned gains and tuned gains from the GA and Ziegler-Nichols method with -25 kW requested at the PCC.

active power. Thus, improved settling time and stability cannot be guaranteed.

The load is set to a constant 60 kW and 40 kVAR. Initially, the requested power at the PCC is 0 kW and 0 kVAR, and the MG supplies the entire load. After 5 seconds, as shown in Fig. 7, 30 kW and 15 kVAR are requested at the PCC. Following this request, each DER decreases its power output, allowing the grid to begin supplying the load. With the untuned PI gains, the MG takes 45 seconds to achieve the requested power. However with the GA gains, the active power settling time, $T_{S2,1}$, decreased to 0.47 seconds, which is 98.96% faster than with the untuned gains. As expected, the reactive power settling time, $T_{S2,2}$, is slower at 2.19 seconds, but it shows significant improvement (95.13%) over the settling time with the untuned gains. For both active and reactive power, the overshoot is significant with the Ziegler-Nichols gains. For the GA gains, this is not the case, and both remain stable.

V. Conclusion

This paper presents a novel framework for real-time model optimization using online co-simulated closed-loop RTSs and optimization algorithms. A Python-based MHSA is integrated with the TyphoonTest IDE and Typhoon HIL API to autonomously control, evaluate, and optimize EMT simulations in real time on a Typhoon HIL 606. Python integration allows for flexibility with other MHSAs such as ICAs, PCO, and ACO which can be implemented in future studies. Similarly, this framework can be applied to any RTS model in Typhoon HIL or other RTS platforms with API integrations. Few RTS platforms include API integration such as this, so reproducibility with other setups may be a challenge. However, those with compatible setups may consider this framework a compelling option for online closed-loop co-simulation.

The proposed framework is effectively demonstrated by using a GA to optimize a simulated PI controller in real time. The GA conducts 4,441 simulations, totaling 76.73 hours of runtime, with different PI parameters and, for each one, evaluates the settling time, stability, and overshoot to determine the parameters' fitness. The highest fitness K_c and K_i values are 0.0794 and 0.002 respectively. Multiple case studies validate the GA-derived parameters with the highest fitness, demonstrating a maximum 44.53 second, 98.96%, improvement in settling time in the CMR Microgrid's active power response compared to untuned gains.

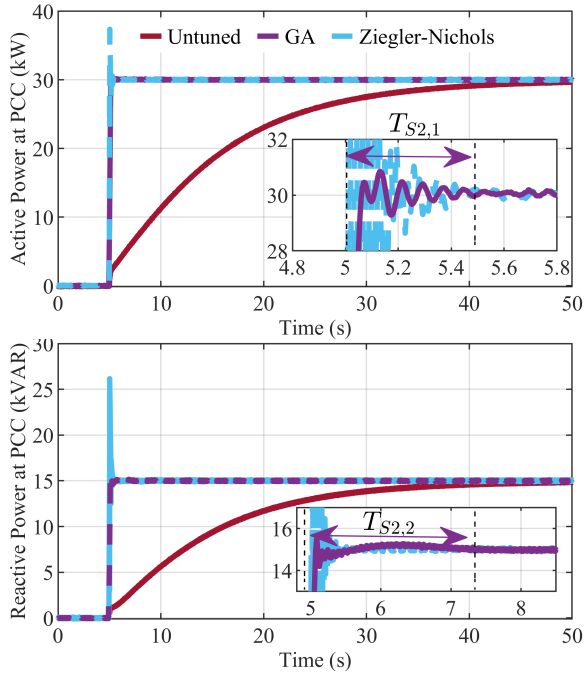

Fig. 7. The CMR Microgrid RTS active and reactive power response with untuned gains and tuned gains from the GA and Ziegler-Nichols method with 30 kW and 15 kVAR requested at the PCC.

References

[1] E. R. Fernandez Cornejo, R. C. Diaz, and W. I. Alama, "Pid tuning based on classical and meta-heuristic algorithms: A performance comparison," in *2020 IEEE Engineering International Research Conference (EIRCON)*, pp. 1–4, 2020.

[2] M. I. Mohamed, G. El-Saady, and A. M. Yousef, "Performance analysis of genetic algorithm and ant colony optimization dependent on pid controller for matrix converter," in *2021 International Conference on Electronic Engineering (ICEEM)*, pp. 1–6, 2021.

[3] J. Qian, S. Shen, L. Shan, L. Lyu, and Z. Qi, "The application of the improved particle swarm algorithm in parameter tuning of partition pid," in *2016 IEEE International Conference on Information and Automation (ICIA)*, pp. 696–701, 2016.

[4] S. Abdelhady, A. Osama, A. Shaban, and M. Elbayoumi, "A real-time optimization of reactive power for an intelligent system using genetic algorithm," *IEEE Access*, vol. 8, pp. 11991–12000, 2020.

[5] S. M. H. A. Nejad and H. Mohammadi, "Hardware-in-the-loop approach for energy consumption optimization in dual-motor gear systems," in *2025 10th International Conference on Technology and Energy Management (ICTEM)*, pp. 1–6, 2025.

[6] W. Bai, I. Eke, and K. Y. Lee, "Heuristic optimization for wind energy integrated optimal power flow," in *2015 IEEE Power & Energy Society General Meeting*, pp. 1–5, 2015.

[7] P. Shende, D. Scarfe, W. Meek, and S. Krishna, "Automating hil test setup and execution," in *2008 IEEE AUTOTESTCON*, pp. 118–121, 2008.

[8] A. Bezbaruah, B. Pratap, and S. B. Hake, "Automation of tests and comparative analysis between manual and automated testing," in *2020 IEEE Students Conference on Engineering & Systems (SCES)*, pp. 1–5, 2020.

[9] H. Eun, E. Jee, D.-H. Bae, Y. Kim, and Y. Lee, "Automated simulation-based integration testing for multiple plcs in a reactor protection system," in *2023 30th Asia-Pacific Software Engineering Conference (APSEC)*, pp. 386–395, 2023.

[10] R. Iracheta-Cortez and N. Flores-Guzman, "Developing automated hardware-in-the-loop tests with rtds for verifying the protective relay performance," in *2016 IEEE 36th Central American and Panama Convention (CONCAPAN XXXVI)*, pp. 1–9, 2016.

[11] M. Elsayed, T. Holliday, R. Loka, M. Kabalan, M. M. Z. Moustafa, and T. Bozada, "Unified secondary control of dispatchable energy sources in utility-connected microgrids," in *2025 North American Power Symposium (NAPS)*, 2025.

[12] Typhoon HIL, "Hil Academy test automation." https://hil.academy/courses/test-automation/, 2022. [Accessed 29-08-2025].

[13] A. F. Gad, "Pygad: An intuitive genetic algorithm python library," *Multimedia Tools and Applications*, vol. 83, p. 58029–58042, Dec 2023.

[14] B. J. G. Ziegler and N. B. Nichols, "Optimum settings for automatic controllers," *Journal of Fluids Engineering*, 1942.