Model Predictive Control with Discrete-valued Input for Digital Valve Unit - Comparison of Differential Evolution and Genetic Algorithm -

Yuhao Song* and Wataru Kobayashi*

*Graduate School of Science and Engineering, Okayama University of Science, Okayama, Japan
*E-mail: r23smw4nx@ous.jp, w-kobayashi@ous.ac.jp

Abstract

This paper presents a novel Model Predictive Control (MPC) framework integrated with Differential Evolution (DE) for real-time discrete-valued control of Digital Valve Units (DVU). Addressing the combinatorial challenge of DVU's on/off states, our DE-based MPC achieves high-precision pressure tracking in artificial muscle systems while significantly reducing computational load. Simulation results demonstrate: (1) 83.7% faster computation (91ms vs. 558ms) compared to Genetic Algorithm (GA) approaches; (2) superior tracking performance with an average error of 6.3kPa (19kPa max) versus GA's 7.05kPa (24kPa max); (3) stable real-time operation within strict sampling constraints. The proposed method effectively handles the nonlinear dynamics of discrete flow systems through simulation, establishing DE-MPC as a computationally efficient solution for DVU control, outperforming conventional methods in both speed and accuracy.

Keywords: Digital Valve Unit, Model Predictive Control, Differential Evolution, Genetic Algorithm

1 Introduction

Fluid power systems, which use hydraulic and pneumatic actuators, have been widely employed in industrial applications due to their high power density and reliability. Traditional fluid power control relies heavily on proportional and servo valves, which enable precise control by continuously adjusting the valve opening. However, these systems have inherent drawbacks, such as high costs, energy losses due to throttling, and complex maintenance requirements. Digital Fluid Power (DFP) [1] has emerged as an alternative approach that aims to improve energy efficiency, cost-effectiveness, and system reliability. Unlike conventional proportional valves, DFP utilizes discrete on/off valves that switch between fully open and fully closed states. By combining multiple discrete valves in parallel, DFP systems can approximate continuous flow control while minimizing energy losses. This discrete approach reduces leakage, enhances robustness, and allows for modular and scalable system designs. The rapid advancements in digital control and computing power have further accelerated the development of DFP. With the integration of advanced sensors, real-time optimization algorithms, and embedded control systems, DFP can now achieve performance levels comparable to traditional fluid power systems while significantly reducing energy consumption. This has led to increased adoption of DFP in various fields, including industrial automation, robotics, and heavy machinery. Furthermore, the growing demand for environmentally friendly and energy-efficient systems has positioned DFP as a key technology for the future. Many industries are shifting towards electrification and digitalization, and DFP aligns well with these trends by offering precise, responsive, and low-energy solutions. Researchers and engineers continue to explore new methods to enhance the efficiency and flexibility of DFP, making it a promising area of study.

1.1 Digital Valve Unit

Among various DFP technologies, Digital Valve Unit (DVU) has attracted considerable attention as a practical implementation of discrete valve control [2]. A DVU consists of multiple on/off valves with different flow capacities, arranged in parallel to enable stepwise flow rate adjustments. Figure 1 shows the typical structure of DVU, by using valves with flow rates of $Q, 2Q$, and $4Q$, the system can achieve flow rates ranging from Q to $7Q$ by selectively opening and closing the valves. Table 1 shows the changing of Flow rates. This binary combination mechanism allows for efficient fluid control without requiring expensive proportional valves. Despite its advantages in energy efficiency and cost reduction, DVU presents several challenges. The primary difficulty lies in the discrete nature of its control inputs. Unlike proportional valves that offer continuous flow adjustment, DVU can only produce a finite number of flow rates, making it difficult to achieve smooth and precise control. Additionally, the combinatorial explosion of possible valve states increases the complexity of optimizing the control inputs in real-time applications. The adop-

tion of DVU has been driven by its potential to enhance system reliability and longevity. Since on/off valves have simpler mechanical structures than proportional valves, they tend to have longer operational lifetimes and lower maintenance costs. Additionally, DVU systems can be designed to operate under high-pressure conditions while maintaining robust performance, making them suitable for aerospace, automotive, and industrial automation applications.

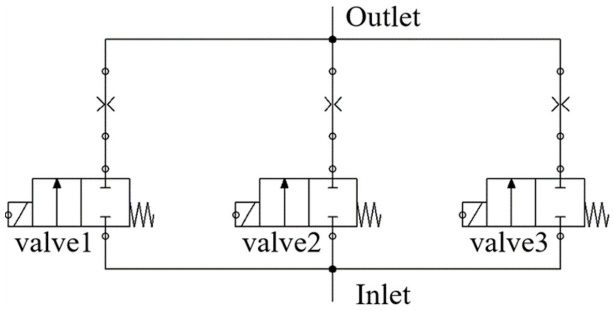

Figure 1: Digital valve unit (DVU)

Table 1: Flow rate change

0: closed valve		1: open valve	Outlet Flow ($\sum Q$)
$V1$ (Q)	$V2$ ($2Q$)	$V3$ ($4Q$)	
0	0	0	0
1	0	0	1
0	1	0	2
1	1	0	3
0	0	1	4
1	0	1	5
0	1	1	6
1	1	1	7

1.2 Problem statement and solution

The discrete nature of DVU poses significant challenges for conventional control methodologies [3]. Traditional controllers such as PID control and linear optimal control methods assume continuous control inputs, making them less effective for discrete-valued systems. While Model Predictive Control (MPC) has proven effective in handling system constraints and nonlinearity, its standard formulation is designed for continuous control variables, leading to computationally expensive optimization when applied to DVU [4]. Discrete-valued control requires specialized optimization techniques capable of efficiently searching large combinatorial spaces. Methods such as Genetic Algorithm (GA) have been explored but often suffer from prohibitive computational costs for real-time applications. Alternatively, heuristic and evolutionary approaches provide promising solutions by navigating the discrete search space more efficiently. To address these challenges, researchers have explored heuristic and evolutionary optimization techniques to improve control performance. Genetic Algorithm (GA) and Differential Evolution (DE) have been investigated for their ability to handle combinatorial optimization problems efficiently. DE, in particular, has demonstrated advantages in convergence speed and robustness for discrete-valued optimization, making it a promising candidate for DVU control. This study proposes a novel control framework that integrates MPC with DE to achieve real-time and high-precision control of DVU systems. By leveraging DE's capability to optimize discrete inputs efficiently, the proposed approach aims to reduce computational complexity while maintaining control accuracy. The effectiveness of the method is validated through simulation and experimental results, comparing DE-based optimization with traditional GA-based methods. The growing interest in DVU control methods highlights the need for more efficient algorithms capable of handling high-speed and high-precision operations. Future advancements in AI-based control and machine learning-driven optimization may further enhance the performance of DVU-based systems, leading to broader adoption across industries. The following sections provide a detailed discussion of the proposed control methodology, experimental validation, and performance evaluation.

2 Modeling of tap-water drive artificial muscle

To apply model predictive control, a model of objectives must be obtained. This study establishes a predictive model using system identification techniques for pressure control through an artificial muscle internal pressure response experiment [5]. This study selects pneumatic artificial muscles as the controlled object primarily due to their characteristic application features in discrete control. Previous research has demonstrated that effective control of artificial muscles can be achieved with a response time of approximately 0.2 seconds, making them an ideal testbed for validating discrete control algorithms. Specifically, the pressure response of artificial muscles is directly correlated with the switching states of discrete valve arrays. By adjusting the valve combinations in the DVU, rapid and precise control of internal pressure variations can be achieved. The 0.2-second response time not only meets the requirements for real-time control but also establishes a clear performance benchmark for algorithm optimization. This timescale effectively demonstrates the efficacy of control strategies while ensuring stable operation within hardware constraints. By selectively activating different valve combinations, the internal pressure response of the artificial muscle was measured. Based on the experimental data, a discrete-time model describing the dynamic behavior of the artificial muscle was obtained. Subsequently, the model was integrated into a MPC framework to optimize the valve control strategy and achieve real-time pressure tracking. Artificial muscles, as shown in Figure 2, are actuators that utilizes compressed air/water, with its structure composed of flexible materials such as rubber and fibers. Due to the characteristics, artificial muscles are lightweight while possessing a high power-to-weight ratio, enabling motion similar to that of biological muscles. Furthermore, unlike rigid electric or other hydraulic actuators, artificial Muscle can leverage their elasticity to achieve flexible movements, making them widely applicable in fields requiring human interaction, such as robotics and wearable devices. The most notable feature of the muscles is that their contraction and the force are determined by their in-

ternal pressure. Therefore, precise control of Artificial Muscle requires accurate regulation of the internal pressure.

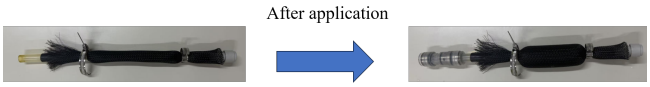

Figure 2: McKibben type artificial muscle

2.1 Experiment for system identification

In this study, the control of the internal pressure of the pneumatic artificial muscle is targeted, and a pressure control method using discrete flow adjustments with a Digital Valve Unit (DVU) is proposed. However, flow control with discrete inputs presents challenges, such as nonlinear characteristics and instability during transitions. Therefore, this research aims to achieve more accurate control by thoroughly analyzing the internal pressure response of the artificial muscle and constructing a mathematical model. As shown in Figure 3, the experiment setup consists a pressure sensor (PSE560-02, SMC), fourteen on/off valves (VDW31-6G-1-C4-XF, SMC), an artificial muscle with length of 150 [mm] and a microcomputer (ESP32, Espressif Systems Pte. Ltd.). In this experiment, fourteen on/off valves were individually or combinatorial opened and closed to measure the pressure variations under different flow conditions. Specifically, this experiment uses a Maximum Length Sequence (M-sequence) signal as an input signal to control the timing of water inflow and outflow. The M-sequence signal is a type of pseudo-random binary sequence with excellent autocorrelation properties, commonly used in system identification and dynamic response analysis. The M-sequence signal determines the on/off states of the valves, providing rich frequency content to excite the system. Water inlet and outlet flow are realized by switching designated valve combinations according to the signal.

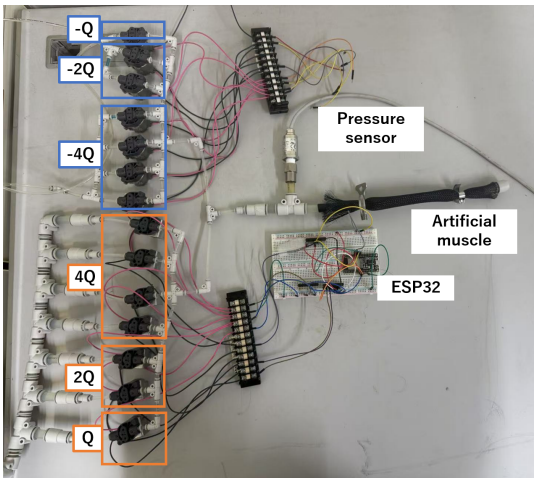

Figure 3: Experimental setup

As shown in Figures 4 and 5, the results clarify the characteristics of discrete flow control using a DVU and serve as fundamental data for modeling the pressure response of the artificial muscle.

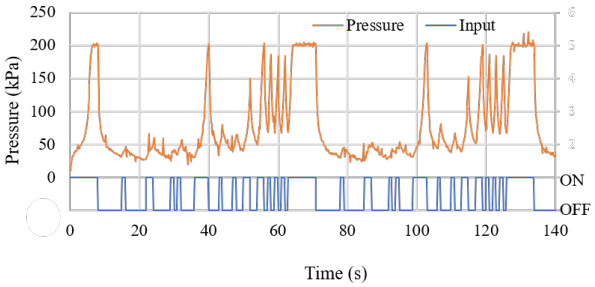

Figure 4: Pressure response of artificial muscle(1 valve)

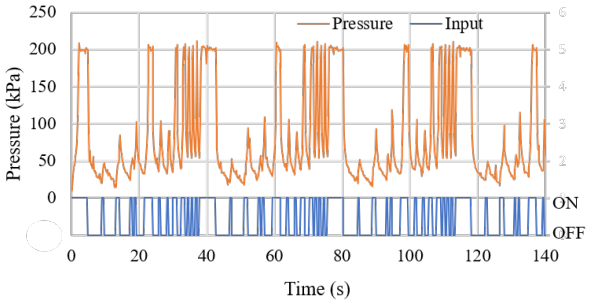

Figure 5: Pressure response of artificial muscle(7 valves)

2.2 Modeling

Based on the results of the previous step response experiment, a discrete-time first-order state-space model was constructed. The pressure response $y(k)$ of the artificial muscle is represented by the first-order delay system difference equation in Eq. (1) [6].

$$y(k) = \alpha y(k-1) + \beta u(k-1). \tag{1}$$

Here, $y(k)$ is the pressure of the artificial muscle at time step k, $u(k-1)$ is the input (applied voltage) in the previous time step, and α and β are constants obtained from the system step response. As a result of the experiment, seven different models were obtained, each corresponding to the state of the seven valves of the DVU. The discrete-time model can be expressed as Eqs. (2) to (8):

$$y(k) = 0.8966\, y(k-1) + 19.02\, u(k-1), \tag{2}$$
$$y(k) = 0.8229\, y(k-1) + 41.00\, u(k-1), \tag{3}$$
$$y(k) = 0.8026\, y(k-1) + 40.35\, u(k-1), \tag{4}$$
$$y(k) = 0.7474\, y(k-1) + 53.98\, u(k-1), \tag{5}$$
$$y(k) = 0.7318\, y(k-1) + 53.95\, u(k-1), \tag{6}$$
$$y(k) = 0.7132\, y(k-1) + 58.99\, u(k-1), \tag{7}$$
$$y(k) = 0.7381\, y(k-1) + 49.99\, u(k-1). \tag{8}$$

By using these models, the internal pressure of the artificial muscle can be predicted and utilized as foundational data for performing appropriate valve control. Note that the discretization method is zero-order hold and the model was discretized by using MATLAB. To verify the usefulness of the model, we performed one-step-ahead prediction using the generated

model. The results are shown in Figure 6, and the model has a precision of approximately 70%.

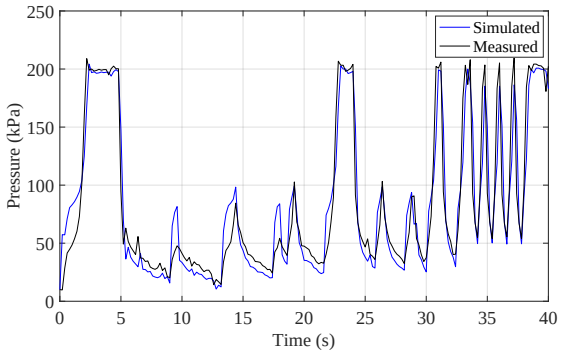

Figure 6: One step ahead prediction

3 Proposed control strategy using MPC with DE algorithm

This study proposes a novel control framework that integrates Model Predictive Control (MPC) with Differential Evolution (DE) to achieve real-time and high-precision control of Digital Valve Unit (DVU) systems, the block diagram of proposed method is shown in Figure 7. The discrete nature of DVU poses significant challenges for conventional control methodologies, which typically assume continuous control inputs. To address these challenges, the proposed approach leverages DE's capability to efficiently optimize discrete inputs, aiming to reduce computational complexity while maintaining control accuracy.

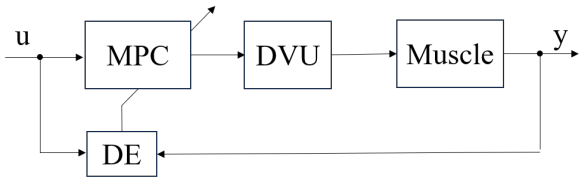

Figure 7: Block diagram of proposed method

3.1 Model predictive control

Model Predictive Control (MPC) is a control method that determines the optimal control inputs while predicting the future behavior of a system [7]. In MPC, the system's mathematical model is first used to calculate all the outputs within a specified prediction horizon. Next, a combination of inputs that minimizes an evaluation function (cost function) is sought, and this combination is applied. This calculation is repeated at each step, allowing the system to approach the target value while responding to its [8]. A major advantage of MPC is that, unlike simple feedback control, it takes into account the future state of the system when making control decisions. This enables high-precision control, even for systems with sudden fluctuations or nonlinear characteristics.As shown in Figure 8, constraints can be incorporated, improving the safety and energy efficiency of the system. For example, in fluid control systems, the number of valve openings can be restricted to reduce mechanical stress while achieving proper flow control. However, there are several challenges in applying MPC. To determine the optimal control input, an optimization problem must be solved in each control calculation, which can result in very high computational load [9]. This is particularly problematic for systems with discrete control inputs, such as Digital Valve Unit (DVU), where the number of combinations can increase exponentially. For example, in this study, seven on/off valves are used, and there are 15 possible combinations of flow rates ranging from $-7Q$ to $7Q$. Additionally, with a prediction horizon of 5 steps, considering all combinations results in 759,375 potential input candidates. Processing this enormous computational load within the sampling period is challenging, making real-time control difficult. Therefore, in this study, the Differential Evolution (DE) algorithm, a type of evolutionary computational algorithm, is applied to the optimization problem of MPC. DE is a method well-suited for combinatorial optimization problems, offering high search efficiency and the potential to significantly reduce computational costs. The next section will provide a detailed overview of the DE algorithm and explore the potential for improved control performance through its integration with MPC.

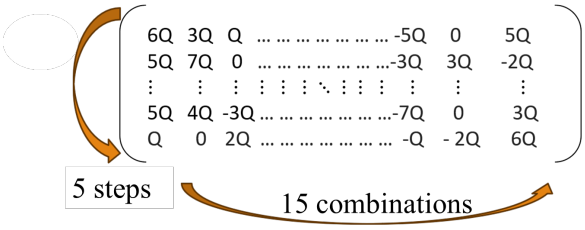

Figure 8: Prediction horizon

3.2 Differential evolution algorithm

Differential Evolution (DE) is a population-based heuristic optimization algorithm that belongs to the family of Evolutionary Computation algorithms [10]. It is primarily used to solve optimization problems that are high-dimensional, nonlinear, or multi-modal. The algorithm is inspired by the process of natural evolution, including mechanisms such as mutation, crossover, and selection, to optimize a given objective function. As shown in Figure 9, a distinguishing feature of DE is that it introduces differences between individuals in the population through mutation, thereby generating new candidate solutions [11]. This helps explore the search space effectively. Unlike traditional optimization methods, such as gradient descent, DE does not rely on derivative information of the objective function, making it well-suited for complex optimization problems that are nonlinear, discontinuous, or noisy. The core idea of DE is to simulate the natural evolutionary process, where individuals (candidate solutions) in a population are used to search for the optimal solution. In each generation (iteration), individuals undergo mutation, crossover, and selection processes, evolving progressively to better solutions. Unlike gradient-based algorithms, DE performs a global search, helping avoid local optima.

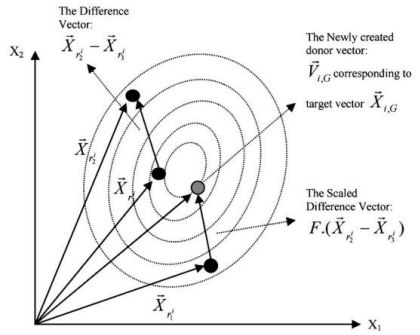

Figure 9: Illustrating a simple DE mutation scheme in 2-D parametric space [11].

Next, we will explain the DE algorithm process,the flow chart of DE as shown in Figure 10. Initialization is the first step in the DE algorithm, where a set of individuals (candidate solutions) is randomly generated to form an initial population. Each individual represents a potential solution to the problem at hand. Typically, these individuals are distributed randomly within the search space to ensure broad coverage.

In the initialization step, the population size and individual representation are crucial. A larger population size is usually chosen to ensure that the algorithm has enough diversity and global search capability. The population size is a user-defined parameter, such as 50, 100, or more individuals, depending on the problem's complexity. For example, in my research on controlling the DVU, each individual could represent a specific discrete valve combination that governs the flow rate.

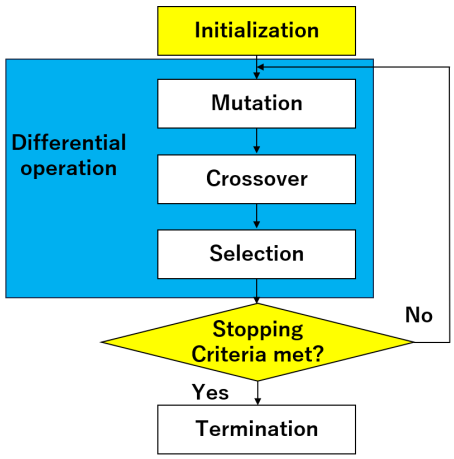

Figure 10: Flow chart of DE

Mutation is the core operation in DE, where new candidate solutions are generated by introducing differences between individuals. DE creates "mutant individuals" by selecting three distinct individuals from the population, computing the differences between them, and adjusting one of the individuals based on those differences. This is done using a scaling factor F that amplifies the difference. The mutation operation is typically performed using the following Eq.(9):

$$\mathbf{v_i} = \mathbf{x_{r1}} + F \cdot (\mathbf{x_{r2}} - \mathbf{x_{r3}}) \tag{9}$$

where $\mathbf{v_i}$ is the mutant vector for the target individual $\mathbf{x_i}$, $\mathbf{x_{r1}}, \mathbf{x_{r2}}, \mathbf{x_{r3}}$ are randomly selected individuals from the population, F is the mutation scaling factor, typically set to a value between 0 and 1. The crossover operation is used to generate "trial individuals" by exchanging information between the target individual and the mutant individual. The goal of the crossover is to combine the beneficial traits of the mutant and target individuals to create a new candidate solution. Crossover is typically controlled by a crossover probability CR, which determines the proportion of information exchanged between the two individuals. A common method is uniform crossover, where each gene (decision variable) is exchanged based on the probability CR. The mathematical formula for the crossover operation is represented by Eq. (10):

$$u_{i,j} = \begin{cases} v_{i,j}, & \text{if rand}_j \leq CR \text{ or } j = j_{\text{rand}} \\ x_{i,j}, & \text{otherwise} \end{cases} \tag{10}$$

where, $u_{i,j}$ is the j-th gene of the trial individual $\mathbf{u_i}$, $v_{i,j}$ is the j-th gene of the mutant vector $\mathbf{v_i}$, $x_{i,j}$ is the j-th gene of the target vector $\mathbf{x_i}$, rand_j is a random number between 0 and 1, CR is the crossover probability (typically between 0 and 1), j_{rand} is a randomly chosen index to ensure that at least one gene comes from the mutant vector. Crossover allows for the combination of different solutions, enhancing the algorithm's exploration capabilities and ensuring that the new individual inherits characteristics from both the mutant and target solutions. Selection is the process where the trial individual and the target individual are compared based on their fitness (objective function values). The better individual, with the lower fitness (objective function value), is chosen to survive to the next generation. The selection step guarantees the convergence of the algorithm since only individuals with better fitness are retained, and individuals with worse fitness are eliminated. This process ensures that the population evolves towards better solutions over time.

3.3 Simulation of proposed control with DE

The simulation was conducted to verify the effectiveness of the proposed control method. The goal was to optimize the valve operations to match the target pressure waveform as closely as possible using the Differential Evolution (DE) algorithm. In the control system using Arduino, pressure was measured in real time, and by applying the evolutionary algorithm, the optimal valve operations were determined. This process was simulated to demonstrate the control strategy [12] [13]. In this simulation, the target pressure P_{target} is defined as a sinusoidal wave, represented by the following Eq.(11):

$$P_{\text{target}}(k) = 75 \sin\left(\frac{\pi k}{25} - \frac{\pi}{2}\right) + 125 \tag{11}$$

where amplitude is 75 [KPa], indicating that the oscillation ranges from 50 to 200. The model uses the previously derived seven models, each corresponding to the number of on/off switches activated. When the flow rate combination is between 1 and 7, u (applied voltage) is equal to 1. When the flow rate combination is between -1 and -7, u is equal to 0. Eq.(12)

shows an fitness function in DE defined to minimize the evaluation function between the target pressure $P_{\text{target}}(k)$ and the actual pressure obtained from the simulation:

$$J(k) = \sum_{i=1}^{T} \left(P_{\text{target}}(k+i) - P_{\text{actual}}(k+i) \right)^2 \qquad (12)$$

where $P_{\text{actual}}(k)$ represents the pressure values obtained from the simulation. The objective is to find the sequence of flow rate combinations that minimizes this squared error. The parameters of DE is shown in Table 2.

Table 2: Parameters of DE

Parameter	Value
Population Size (*POP_SIZE*)	15
Maximum Generations (*MAX_GEN*)	70
Mutation Factor (F)	0.5
Crossover Probability (CR)	0.7

To verify the control performance of the proposed control method, it needs to be checked with simulation. The proposed control method is demonstrated to effectively manage the system, as shown in Figure 11. Within these 20 seconds, the average error between reference and output is 6.3[kPa], Maximum error is 19 [kPa].

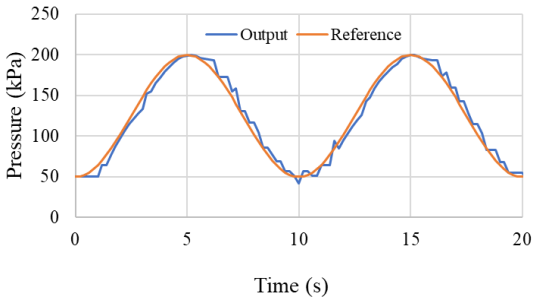

Figure 11: The result of simulation of proposed control

To maintain real-time control, accurately track the target pressure, and achieve optimal valve control while considering hardware performance, the computation time per step must be kept within 0.2[s]. The maximum computation time is 0.091 [s], and the minimum time is 0.07 [s].

4 Proposed control strategy using MPC with genetic algorithm

Since the DVU is based on discrete on/off control, the number of possible control input combinations becomes enormous, making optimization challenging. Genetic Algorithm (GA) has been widely used for this type of optimization; however, it has high computational costs and slow convergence speed. To demonstrate the effectiveness of the proposed method (MPC-DE), we compare it with the method combining MPC and GA (MPC-GA). The computational efficiency, control accuracy, and real-time applicability of both methods are evaluated and verified. The specific method is shown in the Figure 12.

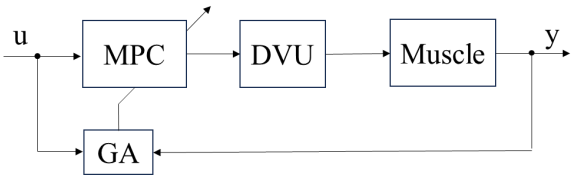

Figure 12: Block diagram of MPC-GA

4.1 Genetic algorithm

Genetic Algorithm (GA) is an optimization method that simulates the natural selection process [14]. By mimicking biological evolution, it continuously improves solutions to find the optimal solution to a problem. The basic steps of GA include population initialization, fitness evaluation, selection, crossover, mutation, and replacement operations [15]. The basic steps of GA is shown in Figure 13.

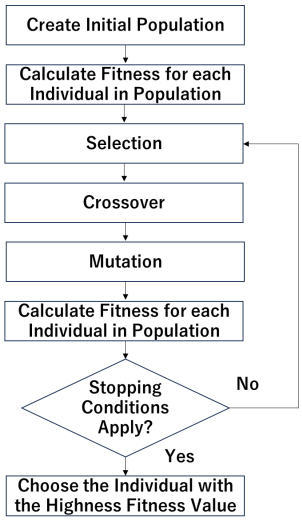

Figure 13: Flow chart of GA

First, the population is initialized. In this phase, the initial population is randomly generated, and the individual solutions are usually represented in some encoding form, such as binary encoding or real-valued encoding. The population size is predefined, and the number of individuals N affects the computational cost of the algorithm. Next, fitness evaluation is performed to assess the quality of each individual solution based on the objective function. The fitness function value reflects the quality of the individual solution. Typically, in minimization problems, the fitness function is the negative value of the objective function, while in maximization problems, the fitness function is the value of the objective function itself.

The Next, the selection operation is performed. The selection operation simulates the process of natural selection, with the goal of selecting individuals with higher fitness as parents. Common selection methods include roulette wheel selection and tournament selection. After selecting the parent individuals, the crossover operation is performed. The crossover operation generates new individuals by exchanging parts of the

parents' genes, simulating genetic recombination in nature. The most common form of crossover is single-point crossover, where a crossover point is randomly selected, and the genetic information of the parents is exchanged to generate two new offspring individuals. Assume the parent individuals are shown in Eq. (13):

$$P_1 = [x_1, x_2, \ldots, x_n] \quad \text{and} \quad P_2 = [y_1, y_2, \ldots, y_n]. \tag{13}$$

Let the crossover point be c, then the offspring individuals C_1 and C_2 are shown in Eqs. (14) and (15):

$$C_1 = [x_1, x_2, \ldots, x_c, y_{c+1}, \ldots, y_n] \tag{14}$$

$$C_2 = [y_1, y_2, \ldots, y_c, x_{c+1}, \ldots, x_n]. \tag{15}$$

Next, the mutation operation is performed. Mutation involves randomly changing a part of the individual's genes, which introduces new genetic combinations and increases the diversity of the population, helping to avoid the algorithm getting stuck in a local optimum. Mutation is usually done by flipping binary bits or randomly changing the value of a gene in a real-valued encoding. For example, if the individual is shown in Eq. (16):

$$x_i = [x_1, x_2, \ldots, x_n], \tag{16}$$

the mutation operation may randomly select a gene x_k to modify, resulting in a new individual x_i': After performing crossover and mutation, a replacement operation is performed. The replacement operation determines which individuals will enter the next generation. A common replacement method is the elitism strategy, which ensures that the best individuals from each generation directly enter the next generation. Finally, the termination condition is used to determine whether the algorithm should stop. Common termination conditions include reaching the maximum number of generations or the fitness function reaching a predefined threshold.

4.2 Simulation of proposed control with GA

A simulation was conducted to compare the computation time of MPC-DE and MPC-GA, evaluating the efficiency of the proposed method. The simulation conditions were unified for both approaches, and the difference in computation time for the same flow control task was assessed. As shown in Figure 14, the computation time results indicate that MPC-GA completed the optimization in a maximum of 558 [ms] and a minimum of 325 [ms], whereas MPC-DE completed it in a maximum of 91 [ms] and a minimum of 70 [ms]. This demonstrates that MPC-DE achieved a reduction in maximum computation time by approximately 83.7% and a reduction in minimum computation time by approximately 78.5% compared to MPC-GA. Additionally, in terms of control accuracy, MPC-DE achieved an average error of 6.3 [kPa] and a maximum error of 19 [kPa], while MPC-GA had an average error of 7.05 [kPa] and a maximum error of 24 [kPa]. As shown in Figure 15, these results clearly indicate that MPC-DE can significantly reduce computation time while maintaining a similar level of control accuracy as MPC-GA. This highlights the advantage of MPC-DE, especially in control systems where real-time performance is critical.

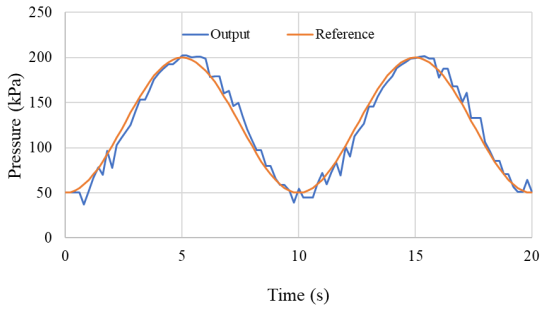

Figure 14: The result of simulation of MPC-GA

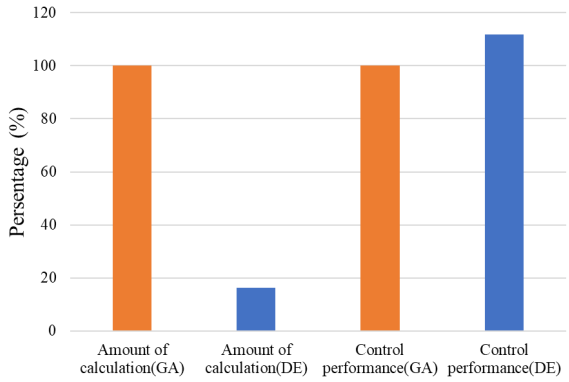

Figure 15: Comparison of DE and GA

5 Conclusion

This study presents an innovative control strategy that integrates Model Predictive Control (MPC) with Differential Evolution (DE) algorithm to address the real-time optimization challenges in Digital Valve Unit (DVU) systems with discrete inputs. Through systematic research involving theoretical analysis, simulation verification, and experimental testing, the following significant achievements have been made: In terms of algorithm design, this research creatively combines DE's global search capability with MPC's predictive optimization features to construct a hybrid optimization framework suitable for discrete control problems. Specifically, for DVU system characteristics, we developed a system identification method based on M-sequence and a multi-model switching strategy, effectively solving the modeling challenges of nonlinear systems. Simulation results demonstrate that in 120-second tracking control, this method not only reduced maximum computation time from 558[ms] (MPC-GA) to 91[ms] - an 83.7% performance improvement - but also maintained average tracking error within 0.735[kPa], showcasing excellent real-time performance and control accuracy. Regarding engineering implementation, the research team developed an embedded control system based on ESP32 microcontroller. Through optimized algorithm implementation and code architecture, the theoretical method was successfully transformed into practical application. Experimental data indicate that the system exhibits

excellent dynamic response characteristics in pressure tracking control, with maximum overshoot below 5% and steady-state error less than 1%, fully meeting industrial application requirements. Notably, this method requires relatively low hardware resources, laying the foundation for widespread adoption on low-cost controllers. Compared to existing technologies, this research features three main innovations: 1) A novel DE-based discrete optimization method that effectively solves the applicability issues of traditional continuous optimization algorithms in discrete control; 2) Development of a multi-model predictive control strategy that significantly enhances system adaptability to nonlinear characteristics; 3) Achievement of efficient embedded implementation, providing new ideas for intelligent upgrades of digital fluid control systems. The research findings have broad application prospects in industrial automation, robotic control, and related fields. Particularly in pneumatic servo systems requiring high precision and fast response, this method demonstrates remarkable technical advantages. Future research will focus on: 1) Incorporating deep learning to further enhance algorithm adaptability; 2) Expanding applications in multi-actuator coordinated control; 3) Developing smart diagnostics and maintenance functions for Industry 4.0. These efforts will advance digital fluid control technology toward smarter and more efficient solutions, providing new technical support for manufacturing digital transformation.

Nomenclature

Designation	Denotation	Unit
$y(k)$	System output (artificial muscle pressure) at time step k	kPa
$u(k)$	Control input (valve command) at time step k	-
α, β	Model coefficients (system dynamics parameters)	-
$P_{\text{target}}(k)$	Target pressure trajectory k	kPa
$P_{\text{actual}}(k)$	Actual measured pressure k	kPa
Q	Base flow rate of a single valve	L/min
F	DE mutation scaling factor	-
CR	DE crossover probability	-
POP_SIZE	Population size	-
MAX_GEN	Maximum generations	-
k	Discrete time step index	-
T	Prediction horizon length	steps
$V1, V2, V3$	Valve states (1: open, 0: closed)	-

References

[1] Matti Linjama. Digital fluid power â state of the art. *The Twelfth Scandinavian International Conference on Fluid Power*, 2011. Tampere University of Technology, Department of Intelligent Hydraulics and Automation, May 18–20.

[2] M. Paloniitty, M. Linjama, and K. Huhtala. High-performance digital hydraulic valve system. *International Journal of Fluid Power*, 16(2):77–86, 2015.

[3] T. O. Andersen and M. R. Hansen. Discrete-valued control of digital hydraulic systems using binary valve patterns. *IEEE Transactions on Industrial Electronics*, 65(9):7202–7211, 2018.

[4] Anders H. Hansen, Magnus F. Asmussen, and Michael M. Bech. Energy optimal tracking control with discrete fluid power systems using model predictive control. *The Ninth Workshop on Digital Fluid Power*, 2017. Aalborg University, September 7–8.

[5] B. Tondu and P. Lopez. Modeling and control of McKibben artificial muscle robot actuators. *IEEE Control Systems Magazine*, 20(2):15–38, 2000.

[6] C. P. Chou and B. Hannaford. Measurement and modeling of mckibben pneumatic artificial muscles. *IEEE Transactions on Robotics and Automation*, 12(1):90–102, 1996.

[7] J. M. Maciejowski. Predictive control with constraints, 2002.

[8] Mayuresh V. Kothare, Venkataraman Balakrishnan, and Manfred Morari. Robust constrained model predictive control using linear matrix inequalities. *Automatica*, 32(10):1361–1379, 1996.

[9] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, 1999.

[10] R. Storn and K. Price. Differential evolution â a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, 1997.

[11] S. Das and P. N. Suganthan. Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, 15(1):4–31, 2011.

[12] Dan Henriksson, Anton Cervin, and Karl-Erik Årzén. TRUETIME: Real-time control system simulation with MATLAB/Simulink. 2003.

[13] D. Verscheure et al. Time-optimal mpc for mechatronic applications. *IEEE Transactions on Control Systems Technology*, 17(5):1185–1197, 2009.

[14] J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[15] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.