Symbolic Kolmogorov-Arnold Networks as a Constant-Time Alternative to LTV-MPC for Embedded Control of Non-Minimum Phase Systems

Adilkhan Salkimbayev
Student, Undergraduate
Kazakh-British Technical University
Almaty, Kazakhstan
a_salkimbayev@kbtu.kz

Timur Samigulin
Scientific Advisor, PhD
Kazakh-British Technical University
Almaty, Kazakhstan
t.samigulin@kbtu.kz

***Abstract*—Model Predictive Control (MPC) is the gold standard for constrained multi-variable control, yet its computational burden often precludes deployment on low-cost embedded hardware. In this work, we propose a Symbolic Kolmogorov-Arnold Network (KAN) as a computationally efficient explicit controller. We benchmark both approaches on a Johansson Quad-Tank system exhibiting Non-Minimum Phase (NMP) dynamics. Experiments conducted via Hardware-in-the-Loop (HIL) on an STM32H7 microcontroller demonstrate that the Symbolic KAN approximates the MPC policy with high fidelity while reducing inference time by up to five orders of magnitude (19ms with stochastic jitter up to 82ms to 1µs), reaching hardware limitations of edge devices.**

***Index Terms*—Model Predictive Control, Kolmogorov-Arnold Networks, Symbolic Regression, Embedded AI, Non-Minimum Phase Systems, Explainable AI.**

I. Introduction

Control of non-minimum phase (NMP) systems remains a significant challenge due to the inherent inverse response dynamics [1] characterized by the presence of a non-minimum phase zero in the right half-plane (RHP). Model Predictive Control (MPC) has become the industrial standard for such systems, offering a framework to handle constraints while providing theoretical guarantees on closed-loop stability and optimality [2], [3]. However, the requirement for online quadratic programming (QP) solving creates a bottleneck for embedded deployment.

While Bemporad [4] proposed Explicit MPC to move computation offline, the exponential growth of the solution complexity with respect to the prediction horizon (N) creates a prohibitive memory footprint for complex dynamics. For instance, Alvarado et al. [5] successfully implemented MPC on a quadruple-tank system, but were constrained to short prediction horizons and low sampling frequencies due to the hardware limitations of the era and the complexity of the parametric solution.

While modern primal-dual solvers like OSQP [6] mitigate the solution time compared to earlier methods, they remain iterative and non-deterministic, creating the latency spikes observed in this study. Thus, a gap remains for a controller that offers the optimality of MPC with the constant-time inference of a PID law.

Recent advances in Scientific Machine Learning (SciML) propose replacing heavy solvers with Neural Networks [7]. Some of the more promising attempts at such control have been introduced by Raissi [8], Brunton et al. [9] and Chen et al. [10]. However, such propositions involved the usage of standard Multi-Layer Perceptrons (MLPs), which suffer from a lack of interpretability ("Black Box" problem) [11], making them risky for safety-critical applications [12].

This paper introduces a methodology using Kolmogorov-Arnold Networks (KANs) [13] to bridge this gap. By exploiting the KAN's interpretability and capacity to learn spline-based activation functions, we extract a symbolic control law that is both computationally efficient and mathematically auditable.

II. Mathematical Modeling

The system under test is the Quadruple-Tank process described by Johansson [14]. The nonlinear dynamics are governed by Bernoulli's principle and Torricelli's equations:

$$\begin{aligned}
\frac{dh_1}{dt} &= -\frac{a_1}{A_1}\sqrt{2gh_1} + \frac{a_3}{A_1}\sqrt{2gh_3} + \frac{\gamma_1 k_1}{A_1}v_1 \\
\frac{dh_2}{dt} &= -\frac{a_2}{A_2}\sqrt{2gh_2} + \frac{a_4}{A_2}\sqrt{2gh_4} + \frac{\gamma_2 k_2}{A_2}v_2 \\
\frac{dh_3}{dt} &= -\frac{a_3}{A_3}\sqrt{2gh_3} + \frac{(1-\gamma_2)k_2}{A_3}v_2 \\
\frac{dh_4}{dt} &= -\frac{a_4}{A_4}\sqrt{2gh_4} + \frac{(1-\gamma_1)k_1}{A_4}v_1
\end{aligned} \tag{1}$$

To tackle the nonlinear dynamics within a convex optimization framework, we employ Successive Linearization. At each time step k, the nonlinear model (1) is approximated by an Affine Linear Time-Varying (LTV) structure:

$$x_{k+1} = A_k x_k + B_k u_k + d_k \tag{2}$$

$$y_k = C x_k \tag{3}$$

where d_k represents the linearization offset. The Jacobian matrix A is defined as:

$$A = \begin{pmatrix} -\frac{a_1}{A_1}\sqrt{\frac{g}{2h_1}} & 0 & \frac{a_3}{A_1}\sqrt{\frac{g}{2h_3}} & 0 \\ 0 & -\frac{a_2}{A_2}\sqrt{\frac{g}{2h_2}} & 0 & \frac{a_4}{A_2}\sqrt{\frac{g}{2h_4}} \\ 0 & 0 & -\frac{a_3}{A_3}\sqrt{\frac{g}{2h_3}} & 0 \\ 0 & 0 & 0 & -\frac{a_4}{A_4}\sqrt{\frac{g}{2h_4}} \end{pmatrix} \tag{4}$$

Matrix B is given by Johansson as:

$$B = \begin{pmatrix} \frac{\gamma_1 k_1}{A_1} & 0 \\ 0 & \frac{\gamma_2 k_2}{A_2} \\ 0 & \frac{(1-\gamma_2)k_2}{A_3} \\ \frac{(1-\gamma_1)k_1}{A_4} & 0 \end{pmatrix} \tag{5}$$

Matrix C is defined as:

$$C = \begin{pmatrix} k_c & 0 & 0 & 0 \\ 0 & k_c & 0 & 0 \\ 0 & 0 & k_c & 0 \\ 0 & 0 & 0 & k_c \end{pmatrix} \tag{6}$$

Where k_c had been set to 1 $\frac{V}{cm}$ for direct conversion.

Although the HIL setup utilizes the full measurement matrix in (6), the underlying system's structural observability was verified for the standard partial output configuration ($y = [h_1, h_2]^T$), satisfying Kalman rank controllability and observability conditions consistent with [14].

III. Control Architecture

The control loop consists of an Extended Kalman Filter (EKF) [15] for state estimation and a Gain-Scheduled Controller (LTV-MPC/Symbolic KAN).

A. Stochastic Estimation (EKF)

While the standard Quadruple-Tank process typically relies on partial observability ($y = [h_1, h_2]^T$), for this HIL benchmark we assumed full state availability ($y = x$) to decouple the control law performance from observer convergence dynamics. The EKF was implemented with an identity measurement matrix ($H = I_4$) to filter Gaussian sensor noise injected by the plant simulation, ensuring that both KAN and MPC operated on comparable, clean state estimates.

The optimal control problem was first solved using the CVXPY [16] extension for Python, which was then converted via CVXpygen framework to the OSQP solver [6], a successor of CVXGEN [17]. The cost function minimizes tracking error and control effort:

$$J = \sum_{k=0}^{N-1} (\|x_k - x_{ref}\|^2 Q + \|u_k\|^2 R) \tag{7}$$

In matrix form, this equation becomes:

$$J = \sum_{k=0}^{N-1} (\|x_k - x_{ref}\|^T Q \|x_k - x_{ref}\| + \|u_k\|^T R \|u_k\|) \tag{8}$$

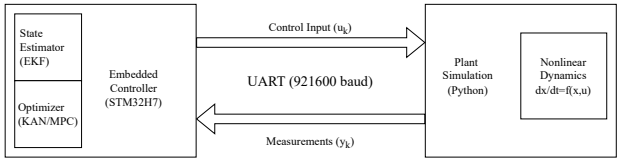

Fig. 1. **HIL Setup.** The diagram describes the process undergoing in Synchronized Time.

TABLE I
Experimental Parameters for MP and NMP Configurations

Parameter	Symbol	MP	NMP
Valve Ratios	γ_1, γ_2	$0.70, 0.60$	$0.43, 0.34$
Pump Gains	k_1, k_2	$3.33, 3.35$	$3.14, 3.29$
Initial State (cm)	x_0	$[12.4, 12.7, 1.8, 1.4]^T$	$[12.6, 13.0, 4.8, 4.9]^T$
Target State (cm)	x_{ref}	$[10.0, 10.0, 2.0, 2.0]^T$	

B. The Student: Symbolic Kolmogorov-Arnold Network

Two Symbolic KAN is structured to approximate the optimal control law $u_{opt} = \pi(x_{est}, x_{ref})$. To accommodate the distinct dynamics of the Non-Minimum Phase (NMP) and Minimum Phase (MP) regimes, we employ a Gain-Scheduled architecture where the active network is switched based on valve ratios γ: $\gamma_1 + \gamma_2 > 1$ triggers minimum phase and $0 < \gamma_1 + \gamma_2 < 1$ triggers NMP (non-minimum phase).

IV. Experimental Results

A. Setup & Training

Simulation was performed using the HIL methodology. The Nucleo-H753ZI (Cortex-M7, 480MHz, DP-FPU) communicates with the PC via UART at 921,600 baud. The simulation setup is described in Fig. 1.

Both controllers were compiled using identical toolchains and aggressive compiler optimization (-Ofast) to ensure a fair execution-time comparison.

The KANs were trained offline using the PyKAN framework [13] on a dataset generated by the LTV-MPC. To overcome the computational bottlenecks of standard B-spline grids, we adopted initialization and grid-adaptation strategies derived from FastKAN [18] and AF-KAN [19]. These optimizations, combined with the L-BFGS optimizer [20] being chosen over standard SGD [21], accelerated convergence and facilitated the extraction of compact symbolic formulas.

The symbolic extraction process followed a three-stage pipeline. First, the network was trained with strong entropy regularization ($\lambda = 10^{-3}$) to suppress non-essential activation functions. Second, a magnitude-based pruning filter masked any nodes with an activation contribution below $\epsilon = 10^{-3}$. Finally, the raw mathematical expressions were extracted as-is directly into the main Python loop for testing and then inserted into the C implementation post testing verification.

Procedure described above enabled the full neural network to converge to 4.84% Mean Absolute Error (MAE) on the test set.

The resulting tracking performance is illustrated in Fig. 2.

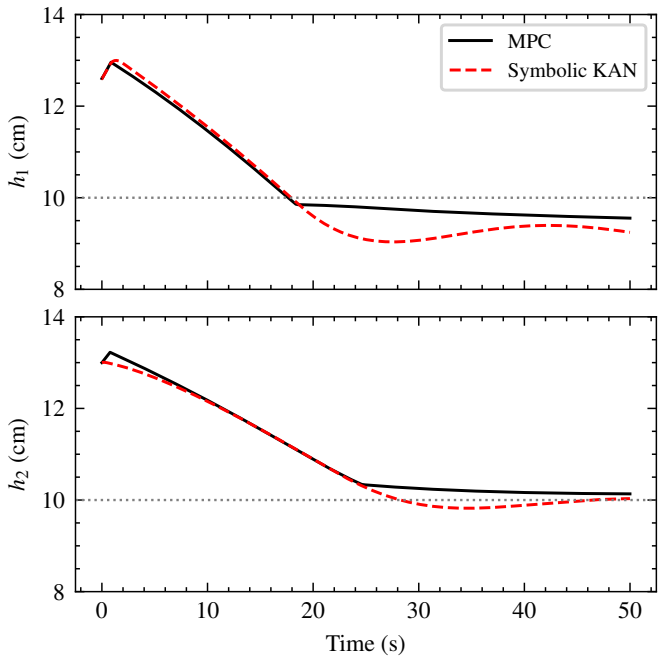

Fig. 2. **Control Performance.** Levels of tanks 1 and 2 throughout the duration of the control loop are shown. MPC exhibits a physics-based, predictive behavior. KAN displays awareness of non-linearity, having approximated the control law.

The simulation employed constants for both MP and NMP systems provided by Johansson, which are shown in Table I.

The simulation's length had been set to 50 seconds and discretized into intervals of 3 milliseconds each, yielding a total of 16666 data points. The simulation ran in Synchronized Time. Runtime was directly calculated in the main program loop; the calculated result was retranslated to the Python plant simulation.

To strictly validate the controller's logic despite the latency spikes, the HIL framework utilized a 'Stop-and-Wait' synchronization protocol. The Python physics engine pauses integration until the UART receives the control vector u_k and the measured execution time Δt_{exec} from the MCU. This effectively decouples the logical correctness of the control law from the real-time constraints, allowing us to profile the worst-case execution time (WCET) without inducing simulation artifacts.

B. Closed-Loop Performance

Fig. 2 illustrates the closed-loop tracking performance under the NMP configuration.

As evidenced by the HIL simulation, Symbolic KAN managed to mimic the control policy applied by the MPC at every discretized time step, managing to approximate the non-linear curve of $\sqrt{h}$ present in Johansson's equations via smooth spline basis functions.

To test the robustness of Symbolic KAN-based control, we added a disturbance at $t = 22.5s$ and executed a new HIL simulation, comparing the disturbance rejection of both competing systems. The simulation plot is shown in Fig. 3.

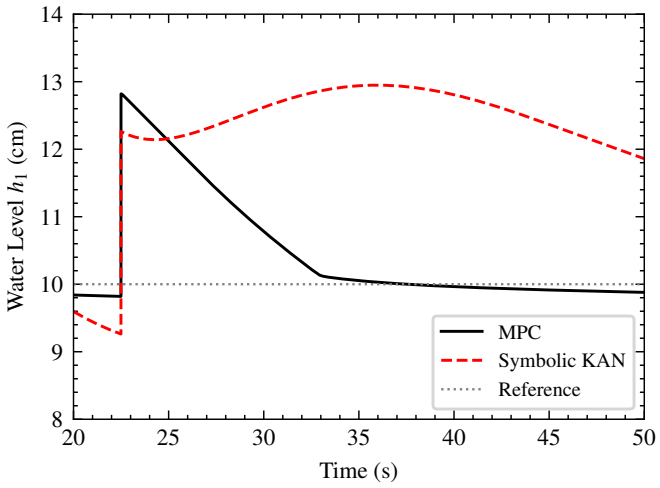

Fig. 3. **Disturbance Rejection.** MPC executes a strict, immediate return to the target state by deterministically solving a long-horizon ($N = 30$) quadratic problem. In contrast, KAN smooths the response to the disturbance, decreasing the level logarithmically.

The resulting plots display the behaviors of both systems in detail; KAN approximated the control policy to account for non-linear behavior of the process, thus gradually lowering the water level against the sudden disturbance; whereas LTV-MPC followed the target state cleanly by iteratively solving a quadratic problem and cutting the voltage of the first pump to reach the target state in least possible time.

Therefore, the MPC had exhibited a stricter degree of disturbance rejection due to its lengthy look-ahead horizon. Meanwhile, KAN managed to mimic the operating principle and repeat the control pattern with high accuracy while invoking far fewer mathematical operations.

Quantitatively, the disparity in tracking performance is negligible. Over the full 50-second trajectory, the LTV-MPC achieved a Root Mean Square Error (RMSE) of 1.83 cm, while the Symbolic KAN achieved 1.90 cm—a deviation of only 3.8%. In steady state ($t > 40s$), the KAN maintained the setpoint with a precision of $\pm 0.65cm$, compared to $\pm 0.44cm$ for the MPC. This confirms that the symbolic distillation preserves the high fidelity of the MPC's nonlinear policy while delivering the computational benefits of $O(1)$ inference.

Furthermore, empirical analysis of the quadratic cost function $J = x^T Qx + u^T Ru$ confirms the convergence of the Symbolic KAN control law (Fig. 4). Although the Non-Minimum Phase dynamics induce a transient energy increase during the inverse response phase (a necessary physical condition to reverse the flow gradient), the KAN successfully steers the system state to the equilibrium manifold, ensuring asymptotic convergence to the setpoint.

C. Computational Latency

To demonstrate rigorous evaluation of the embedded inference latency, we adopted methodology consistent with the MLPerf Tiny benchmark suite [22], utilizing on-chip DWT cycle counters. We extracted runtime performance of both

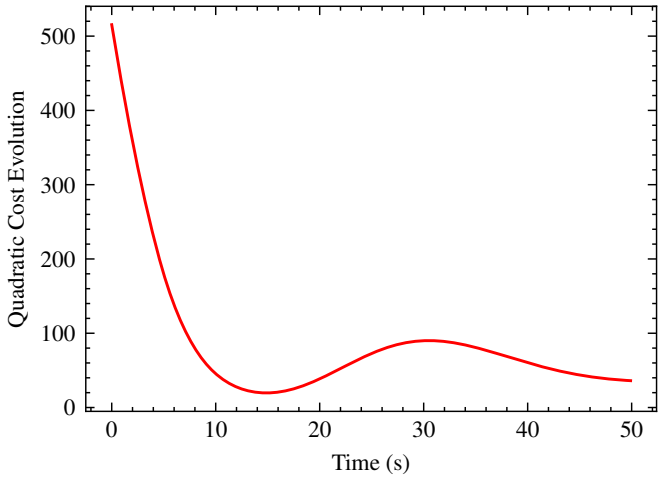

Fig. 4. **Evolution of quadratic cost function J during the NMP control task.** The transient increase in energy (steps 100-300) reflects the inverse response dynamics required to reverse the flow gradient. Subsequent asymptotic convergence to zero confirms the stability of the Symbolic KAN control law.

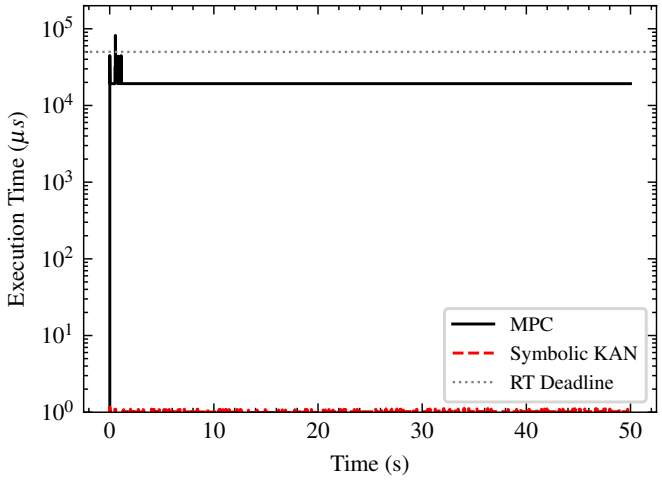

Fig. 5. **Runtime Performance.** Execution latency distribution ($N = 16666$). RT Deadline defined at 20 Hz. While the MPC baseline (black) averages 19ms, stochastic solver iterations cause spikes up to 81ms. The Symbolic KAN (red) demonstrates strictly deterministic $O(1)$ inference at 1.0μs, exceeding MPC by 4 orders of magnitude on average and by 5 orders of magnitude worst-case.

control loops, plotting the distribution of latency across the entire simulation's runtime. Runtime Performance plot is given in Fig. 5. Log scale was chosen for the plot due to 4 orders of magnitude separating the execution cycles of competing control systems.

Most notably, while the LTV-MPC solver is mathematically deterministic, its iterative nature results in non-deterministic execution timing. Depending on the active set changes, the solver required up to 81.8ms to converge, violating the real-time deadline set at 20 Hz. Inconsistent load on the CPU led to stochastic jitter, which in turn caused runtime oscillations.

On the other hand, Symbolic KAN executed in deterministic, microseconds-long window without stutters or stochastic runtime oscillations. Such difference in cycle time demonstrates that Symbolic KAN converts the control law into raw mathematical expressions with high, verifiable accuracy.

Beyond temporal performance, the Symbolic KAN reduced the firmware memory footprint by a factor of 11.8 (Table II). While the OSQP solver and matrix libraries required 221.70 KB of Flash memory, the compiled Symbolic KAN occupied only 18.74 KB.

TABLE II
COMPUTATIONAL RESOURCE COMPARISON (STM32H7)

Metric	LTV-MPC	Symbolic KAN	Improvement
Inference Time (Avg)	19,981 μs	1.04 μs	$\mathbf{19,212}\times$
Inference Time (Max)	81,820 μs	1.094 μs	$\mathbf{74,790}\times$
Flash Memory (Code)	221.70 KB	18.74 KB	$\mathbf{11.8}\times$
Maximum Achievable Update Rate	50 Hz	1 MHz	$\mathbf{20,000}\times$
Determinism	No	Yes	–
Deadline violation	Yes	No	–

The memory footprint reduction is particularly significant for the concept of "Smart Sensors." With a code size of only 18.74 KB, a Symbolic KAN controller could theoretically be embedded directly into the firmware of a smart valve or pump driver (often running on Cortex-M0+ with less than 32KB Flash), enabling decentralized optimal control at the edge without requiring a central PLC.

Results provided above establish two important formulations.

The first is that the load on the CPU caused by Symbolic KAN comprises only about 0.033%, approximately 3000 times less than MPC, leaving more than 99.5% for background processes. Given constrained compute, implementation of MPC is further held back by cache memory storage requirements and sampling bandwidth.

Moreover, KAN's reduced CPU load and lowered cache memory consumption preserves the control loop within the healthy limits of hardware implementation. Such policy preserves consistency and maximizes the service life of computing hardware involved in the process.

Second is that while MPC's runtime has to be adjusted depending on the number of degrees of freedom of a given system, there is no set ceiling on the applications of Symbolic KAN, as it is capable of deriving mathematical expressions of any order. This suggests that Symbolic KANs offer a viable data-driven alternative for nonlinear plants where analytical derivation of robust controllers (e.g., H_∞ or Sliding Mode) is mathematically intractable or computationally prohibitive. By shifting the complexity to the offline training phase, KANs demonstrate feasibility of advanced control for resource-constrained hardware.

D. Safety Audit: The "Sign Flip"

During symbolic extraction for minimum phase system configuration, the KAN initially identified a control law with a negative gain on the error term for Pump 2 (Fig. 6), creating a positive feedback loop. Due to the interpretable nature of KANs, this was manually corrected to negative feedback prior to C deployment.

Such precedent of sign alteration establishes a need for end-to-end human supervision for further deployment of learning-based control and mitigation of danger before direct

Raw Symbolic Output

```
... + 53.32934*x_5 −50.5462 * x_6 - 0.32*x_7 + ...
```

Detected Anomaly: $u_{pump2} \propto -\mathbf{50.55} \cdot x_{error}$
(Positive Feedback Loop Identified via Symbolic Regression)

Fig. 6. **The "White Box" Safety Audit.** Symbolic extraction (top) revealed a critical sign inversion. The highlighted term (-50.55) was flagged and manually corrected to validate negative feedback stability.

deployment of Neural Network-based control, rendering black-box methods like TinyML [23] unsuitable for safety-critical applications due to their lack of verifiable stability guarantees and interpretability.

A potential solution would be to train an interpretable neural network directly on the microcontroller managing the plant, as presented by Essahraui et al. [24] However, lack of direct human control over KAN's learning process and hardware limitations restricting the learning capabilities of Symbolic KAN present a heavy challenge to all possible applications.

V. Discussion

A. Determinism

For the entirety of the 21st century, MPC had been established as a universally accepted robust methodology in Embedded Control, known for its deterministic output and disturbance rejection. While LQR (Linear Quadratic Regulator) suffered from a fundamental lack of physical constraints, MPC provided a bounded alternative, addressing the largest concern surrounding LQR.

Experimentations done in this paper, however, reveal one of the glaring weaknesses of MPC architecture: while MPC is capable of Synchronized Time control, the iterative nature of the Quadratic Problem solver makes it impossible for MPC to reach a lengthy prediction horizon without stochastic runtime oscillations. Active set changes influence the complexity of the solution determined at every iteration, which can compromise the deadline set for real-time control and lead to heavy system lags. Therefore, MPC enforces a dilemma for automation engineers: on one hand, reduction of the look-ahead horizon in order to prioritize fast runtime, on the other hand, acceptance of the longer execution cycle for maintenance of a longer horizon.

Finally, MPC requires an baseline measure of drastically faster compute, endangering legacy industrial equipment. While small-scale robotization managed to integrate MPC, opting to reconcile the compromise set by the architecture, large-scale industrial processes remain averse to it due to significantly increased computational expenses and difficulty of maintenance compared to standard PID control.

Symbolic KAN introduces an alternative: by learning MPC's control policy and mapping out gravity-induced nonlinear gradients present in Johansson's Quadruple-Tank process, it reached both the constant-time computation and enforced robust nonlinear control, combining both the MPC's reliability and verifiability and PID's constant-time inference.

B. Design-Time Auditability

Fig. 6 established a critical vulnerability in Neural Network-based control: a possibility of positive feedback, created in the process of model's training.

In a standard Black-Box Neural Network (e.g., TinyML [23]), this positive feedback loop would have been buried in a weight matrix of 10,000 parameters and would not be interpretable. Therefore, changing such derived control law would be next to impossible and would require a new cycle of stricter training, discarding previous efforts.

However, even in the reinstated training cycle, the possibility of artifacts does not vanish, dictated by fundamental limitations of uninterpretable architecture of MLPs.

The implications are severe: improperly configured Neural Network-based control policy would need rigorous testing and would require constant human supervision at all operating stages, dictated by lack of a verifiable mathematical model. This yields drastically increased OPEX and high uncertainty; eventually, such measure could be labeled as highly unreliable and be taken down.

Because KAN is symbolic and fundamentally interpretable, as it decomposes the control manifold in accordance to Kolmogorov-Arnold theorem of multivariable function approximation, its learning trajectory can be directly observed and corrected if deemed necessary. This eliminates the concern about the lack of interpretabilty of Neural Networks, as the process can be constantly supervised and cross-examined with first principles.

As demonstrated by Fig. 6 directly, KANs provide a superior alternative to commonly accepted MLPs even in case of incorrect learning, adhering to safety legislations and concerns.

C. Smoothness Trade-Off

As observed in Fig. 3, the disturbance rejection behaviors of the two controllers differ fundamentally. The LTV-MPC, driven by the quadratic cost function J, executes a "Bang-Bang" style correction, instantly saturating the control input to minimize the error integral over the horizon. While mathematically optimal, this aggressive switching induces high mechanical stress (jerk) on the actuators.

In contrast, the Symbolic KAN exhibits a smoother recovery trajectory. This is an emergent property of the symbolic regression process: by fitting smooth basis functions (splines/sinusoids) to the control policy, the KAN effectively acts as a low-pass filter on the optimal control law. This results in a slightly longer settling time but significantly reduces actuator fatigue, a critical factor for the longevity of industrial pumps and valves.

Therefore, not only does the Symbolic KAN approximate the MPC's control policy with high fidelity, but its inherent smoothness also mitigates mechanical stress, effectively acting as a low-pass filter on the control signal. This reduction in

actuator fatigue directly translates to lower Operational Expenditure (OPEX) at an industrial scale. Unlike the aggressive sharp adjustments of MPC, the KAN extends the service life of physical components while retaining the benefits of nonlinear awareness and $O(1)$ computational predictability.

Based on the results given in this paper, we formulate the principle of 'Industrial Sufficiency', which suggests that mature MCU nodes remain sufficient for a wide class of physics-based high-frequency nonlinear control tasks.

VI. Conclusion

Finally, these results challenge the prevailing trend of solving control complexity by upgrading hardware. We demonstrate that the bottleneck for deploying advanced optimal control on legacy 40nm lithography (e.g., Cortex-M7) is not the hardware's clock speed, but the algorithmic reliance on iterative online solvers. By shifting the computational burden offline via Symbolic KANs, we validate the principle of 'Industrial Sufficiency': that low-cost, mature microcontroller architectures possess ample compute power for high-frequency nonlinear control, provided the control law is appropriately compressed.

Future work will focus on two avenues. First, we aim to deploy this architecture on multi-agent systems to test the scalability of symbolic distillation in distributed control. Second, we propose an "Adaptive Symbolic KAN" framework, where the symbolic coefficients are fine-tuned online using Recursive Least Squares (RLS) to compensate for plant aging (e.g., valve clogging or tank sedimentation), thereby recovering the adaptive properties of MPC that are lost during the offline distillation process.

Crucially, the proposed Adaptive RLS-KAN framework decouples structure learning (offline, non-convex) from parameter estimation (online, convex). By freezing the symbolic basis functions derived by the KAN and updating only the linear coefficients via Recursive Least Squares, the runtime adaptation becomes a series of standard matrix operations (readily available in arm_math.h or BLAS). This ensures that the adaptive mechanism remains deterministic, computationally lightweight ($O(N^2)$), and verifiable, unlike online backpropagation.

The stability and performance conclusions presented herein are empirical, validated through extensive HIL simulation under the tested operating regimes.

The source code, full derivations of controllability/observability, and raw HIL datasets are available at: github.com/IArtemis13l/QuadTank_KAN_and_LTV-MPC (licensed, Apache 2.0).

VII. Acknowledgments

The authors acknowledge the Department of Automation and Control at Kazakh-British Technical University for providing valuable guidance on the theoretical framework and research direction. We also extend our gratitude to the open-source contributors of the PyKAN, CVXPY, and OSQP ecosystems, whose software infrastructure enabled the comparative analysis presented in this work.

References

[1] S. Skogestad and I. Postlethwaite, *Multivariable feedback control: analysis and design*. John Wiley & Sons, 2005.

[2] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[3] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[4] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.

[5] I. Alvarado, D. Limon, D. M. De La Peña, J. M. Maestre, M. Ridao, H. Scheu, W. Marquardt, R. Negenborn, B. De Schutter, F. Valencia *et al.*, "A comparative analysis of distributed MPC techniques applied to the HD-MPC four-tank benchmark," *Journal of Process Control*, vol. 21, no. 5, pp. 800–815, 2011.

[6] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," *Mathematical Programming Computation*, vol. 12, no. 4, pp. 637–672, 2020.

[7] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-based model predictive control: Toward safe learning in control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 269–296, 2020.

[8] M. Raissi, "Deep hidden physics models: Deep learning of nonlinear partial differential equations," *Journal of Machine Learning Research*, vol. 19, no. 25, pp. 1–24, 2018.

[9] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Sparse identification of nonlinear dynamics with control (SINDYc)," *IFAC-PapersOnLine*, vol. 49, no. 18, pp. 710–715, 2016.

[10] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," *Advances in neural information processing systems*, vol. 31, 2018.

[11] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Machine Intelligence*, vol. 1, no. 5, pp. 206–215, 2019.

[12] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané, "Concrete problems in AI safety," *arXiv preprint arXiv:1606.06565*, 2016.

[13] Z. Liu, Y. Wang, S. Vaidya, F. Ruehle, J. Halverson, M. Soljačić, T. Y. Hou, and M. Tegmark, "KAN: Kolmogorov-Arnold Networks," *arXiv preprint arXiv:2404.19756*, 2024.

[14] K. H. Johansson, "The Quadruple-Tank Process: A multivariable laboratory process with an adjustable zero," *IEEE Transactions on Control Systems Technology*, vol. 8, no. 3, pp. 456–465, 2000.

[15] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[16] S. Diamond and S. Boyd, "CVXPY: A python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[17] J. Mattingley and S. Boyd, "CVXGEN: A code generator for embedded convex optimization," in *Optimization and Engineering*, vol. 13, no. 1. Springer, 2012, pp. 1–27.

[18] Z. Li, "Kolmogorov-Arnold Networks are radial basis function networks," *arXiv preprint arXiv:2405.06721*, 2024.

[19] H.-T. Ta and A. Tran, "AF-KAN: Activation Function-Based Kolmogorov-Arnold Networks for Efficient Representation Learning," *arXiv preprint arXiv:2503.06112*, 2025.

[20] S. Wright, J. Nocedal *et al.*, "Numerical optimization," *Springer Science*, vol. 35, no. 67-68, p. 7, 1999.

[21] H. Robbins and S. Monro, "A stochastic approximation method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, 1951.

[22] C. Banbury, V. J. Reddi, P. Torelli, J. Holleman, N. Jeffries, C. Kiraly, P. Montino, D. Kanter, S. Ahmed, D. Pau *et al.*, "MLPerf tiny benchmark," *arXiv preprint arXiv:2106.07597*, 2021.

[23] P. Warden and D. Situnayake, *TinyML: Machine learning with TensorFlow Lite on Arduino and ultra-low-power microcontrollers*. O'Reilly Media, 2019.

[24] S. Essahraui, I. Lamaakal, Y. Maleh, K. El Makkaoui, M. F. Bouami, I. Ouahbi, H. Elmannai, and A. A. Abd El-Latif, "FastKAN-DDD: A novel fast Kolmogorov-Arnold Network-based approach for driver drowsiness detection optimized for TinyML deployment," *PLoS One*, vol. 20, no. 11, p. e0332577, 2025.