



UNIVERSITÀ POLITECNICA DELLE MARCHE
Repository ISTITUZIONALE

Encrypted MPC Design using Orthonormal Functions to Control Distributed Electrical Drive Systems

This is the peer reviewed version of the following article:

Original

Encrypted MPC Design using Orthonormal Functions to Control Distributed Electrical Drive Systems / Baldi, M., Brasili, E., Chiaraluce, F., El Mechri, R., Ippoliti, G., Orlando, G.. - (2026), pp. 1004-1009. (34th Mediterranean Conference on Control and Automation, MED 2026 Ancona, IT 23 - 26 June 2026) [10.1109/MED70602.2026.11598112].

Availability:

This version is available at: 11566/362873 since: 2026-09-14T07:52:20Z

Publisher:

Institute of Electrical and Electronics Engineers Inc.

Published

DOI:10.1109/MED70602.2026.11598112

Terms of use:

The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. The use of copyrighted works requires the consent of the rights' holder (author or publisher). Works made available under a Creative Commons license or a Publisher's custom-made license can be used according to the terms and conditions contained therein. See editor's website for further information and terms and conditions.

This item was downloaded from IRIS Università Politecnica delle Marche (<https://iris.univpm.it>). When citing, please refer to the published version.

Publisher copyright:

IEEE - Postprint/Author's Accepted Manuscript

©2026 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. To access the final edited and published work see 10.1109/MED70602.2026.11598112

(Article begins on next page)

Encrypted MPC Design using Orthonormal Functions to Control Distributed Electrical Drive Systems

M. Baldi¹, E. Brasili¹, F. Chiaraluce¹, R. El Mechri¹, G. Ippoliti¹ and G. Orlando¹

Abstract—This study presents an encrypted Model Predictive Control (MPC) framework for linear constrained systems, complemented by experimental validation across a networked communication architecture for electric drive systems. The functionality of the encrypted controller is twofold: it safeguards communication links and prevents the exposure of sensitive internal control parameters. The processing latencies of the encrypted scheme we propose are shown to be negligible, ensuring compatibility with the strict real-time constraints of the electrical drive control system.

Index Terms—Encrypted Control; Networked Control Systems; Model Predictive Control, Laguerre functions.

I. INTRODUCTION

The proliferation of the Internet of Things (IoT), cyber-physical systems (CPS), and smart grids necessitates a paradigm shift in control engineering to address emerging complexities in connectivity and security. Contemporary control architectures increasingly leverage cloud and networked infrastructures; however, the transition to these distributed platforms necessitates a trade-off between computational scalability and cybersecurity resilience. Accordingly, the development of cyber-resilient control systems for networked and cloud-based platforms has become a focal point of recent investigation [1]. In this context, Partially Homomorphic Encryption (PHE) has emerged as a compelling solution [2], [3], as it enables the execution of specific arithmetic operations—such as addition or multiplication—directly on ciphertexts. Furthermore, PHE techniques are reasonably fast, unlike Fully Homomorphic Encryption (FHE) techniques, which have latencies that are poorly compatible with real-time control systems. Consequently, PHE turns out to be a technique particularly suited to the realization of real-time control algorithms in the encrypted domain, avoiding the need for the controller to process unencrypted data.

The proposed architectural framework is organized as follows: state measurements are initially encrypted at the sensor layer. The obtained ciphertexts are subsequently transmitted to a networked controller, which computes the control law directly in the encrypted domain. The resulting encrypted command is then forwarded to the actuator, where it is decrypted and applied to the plant. Because the data remains

encrypted throughout both transmission and computation, the system is inherently resilient against eavesdropping and various external cyber-adversaries. The controller operates exclusively on encrypted data, a design choice that substantially strengthens the security posture of the networked control system. Specifically, this work introduces an encrypted Model Predictive Control (MPC) framework tailored for linear systems subject to input constraints. By doing so, the foundational concepts established in [2], [3] are extended to encompass constrained dynamical systems [4].

Model Predictive Control (MPC) is a prominent optimal control paradigm characterized by its ability to explicitly incorporate system constraints. Over the past decade, it has achieved widespread adoption and significant advancements in both industrial and academic sectors [5], [6]. Notwithstanding its high performance, the practical utility of conventional MPC is often constrained by the necessity of solving a complex optimization problem online, which imposes a significant computational burden. This requirement frequently precludes the deployment of MPC in resource-constrained cyber-physical systems with stringent real-time requirements. Consequently, cloud-based and distributed control architectures have emerged as a viable solution, delegating intensive computations to high-performance remote infrastructures [7], [8]. Furthermore, the use of the forward shift operator in MPC to parameterize the future control signal poses significant challenges for high-performance applications. Specifically, rapid sampling rates and complex dynamics require a large number of parameters, which frequently results in excessive online computational demands. Drawing upon the framework established in [9], this study employs a parameterization of the control trajectory utilizing discrete orthogonal polynomial functions, i.e. a Laguerre network of suitable order. This parameterization facilitates a parsimonious representation of the future control signal, thereby reducing the total number of parameters required to model the control trajectory while inherently promoting a smoother input response.

The objective of this work is twofold: first, to verify the performance of the encrypted MPC scheme through its implementation on a practical networked control system utilizing a Raspberry Pi and a DC motor; second, to assess the real-time processing latency of the resulting control system. The paper is structured as follows: Section II details the MPC controller design, while Section III covers its encryption. Section IV describes the experimental platform. Section V evaluates execution overhead across various key sizes. Finally, Section VI concludes the paper.

*This work was partially supported by project SERICS (PE00000014) under the Italian Ministry of University and Research National Recovery and Resilience Plan, funded by the European Union - Next Generation EU.

¹Authors are with Dipartimento di Ingegneria dell'Informazione, Università Politecnica delle Marche, 60131 Ancona, Italy. m.baldi@univpm.it, e.brasili@pm.univpm.it, f.chiaraluce@univpm.it, r.elmechri@staff.univpm.it, g.ippoliti@univpm.it, g.orlando@univpm.it

II. MPC USING LAGUERRE FUNCTIONS

Building upon the methodology proposed in [9], this study derives an MPC formulation that utilizes a set of discrete-time Laguerre functions to provide a high-fidelity, reduced-parameter approximation of the predicted future control sequence.

A. Control signal trajectory described by Laguerre functions

Following the approach proposed in [9], a set of discrete orthonormal functions is used to represent the trajectory of the future control signal difference over a moving horizon window. Specifically, Laguerre functions are selected as the orthonormal basis for this purpose. Accordingly, for a set of Laguerre functions, $\ell_1(k), \ell_2(k), \dots, \ell_N(k)$ [9]:

$$\Delta u(k_i + k) \approx \sum_{m=1}^N \gamma_m(k_i) \ell_m(k) \quad (1)$$

where $\Delta u(k_i + k) = u(k_i + k) - u(k_i + k - 1)$, with k_i the initial time of the moving horizon window and k the future sampling instant. Here, N denotes the number of expansion terms and γ_m , $m = 1, 2, \dots, N$, are the corresponding coefficients depending on k_i . As N increases, the expansion converges to the sequence $\Delta u(k_i)$, $\Delta u(k_i + 1)$, $\Delta u(k_i + 2)$, $\dots$, $\Delta u(k_i + N_c)$, where N_c is the control horizon in the conventional design [9].

As shown in [9], the discrete Laguerre functions satisfy the following difference equation:

$$\mathcal{L}(k+1) = \Gamma \mathcal{L}(k), \quad (2)$$

where

$$\mathcal{L}(k) = [\ell_1(k) \ \ell_2(k) \ \dots \ \ell_N(k)]^T, \quad (3)$$

and

$$\Gamma = \begin{bmatrix} \alpha & 0 & 0 & \dots & 0 \\ b & \alpha & 0 & \dots & 0 \\ -\alpha b & b & \alpha & \dots & 0 \\ \alpha^2 b & -\alpha b & b & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ (-1)^{N-2} \alpha^{N-2} b & (-1)^{N-3} \alpha^{N-3} b & \dots & b & \alpha \end{bmatrix}. \quad (4)$$

where $b = 1 - \alpha^2$ with $0 \leq \alpha < 1$ the scaling factor. The scaling factor of the Laguerre polynomials modulates the decay rate of the control increments, effectively parameterizing the control horizon and shaping the closed-loop transient dynamics.

The initial condition is:

$$\mathcal{L}(0)^T = \sqrt{1 - \alpha^2} \cdot [1 \ -\alpha \ \alpha^2 \ -\alpha^3 \ \dots \ (-1)^{N-1} \alpha^{N-1}]. \quad (5)$$

B. State-space model with embedded integrator

The plant under consideration is a DC motor, characterized by a linear single-input single-output (SISO) dynamical model. Consequently, the control law is developed for a SISO system represented by the following discrete-time state-space formulation:

$$\begin{aligned} x_m(k+1) &= A_m x_m(k) + B_m u(k), \\ y(k) &= C_m x_m(k), \end{aligned} \quad (6)$$

where u , y , and $x_m \in \mathbb{R}^{n_1}$ are the manipulated input, the process output, and the state-variable vector, respectively. To facilitate the design of a controller with an embedded integrator, the original model (6) is augmented by adopting the control increment $\Delta u(k) = u(k) - u(k-1)$ as the new input. According to the methodology in [9], the augmented system is constructed as follows. Let

$$\Delta x_m(k) = x_m(k) - x_m(k-1). \quad (7)$$

By defining the augmented state vector as:

$$x(k) = \begin{bmatrix} \Delta x_m(k) & y(k) \end{bmatrix}^T, \quad (8)$$

the augmented state-space representation is formulated as follows:

$$\begin{aligned} \underbrace{\begin{bmatrix} \Delta x_m(k+1) \\ y(k+1) \end{bmatrix}}_{x(k+1)} &= \underbrace{\begin{bmatrix} A_m & o_m^T \\ C_m A_m & 1 \end{bmatrix}}_A \underbrace{\begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix}}_{x(k)} \\ &+ \underbrace{\begin{bmatrix} B_m \\ C_m B_m \end{bmatrix}}_B \Delta u(k), \quad (9) \\ y(k) &= \underbrace{\begin{bmatrix} o_m & 1 \end{bmatrix}}_C \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix}. \end{aligned}$$

where $o_m = \underbrace{[0 \ 0 \ \dots \ 0]}_{n_1}$ represents an all-zero row vector of size n_1 . The resulting triplet (A, B, C) constitutes the augmented model, which serves as the basis for the subsequent predictive control design.

From (1), for a single-input system, the future control trajectory at time m is given by

$$\Delta u(k_i + m) = \sum_{i=1}^N \ell_i(m) \gamma_i = \mathcal{L}(m)^T \zeta, \quad (10)$$

where $\zeta^T = [\gamma_1, \gamma_2, \dots, \gamma_N]$. Consequently, the predicted future state variable at time $k_i + m$, denoted by $x(k_i + m | k_i)$, is expressed as [9]

$$x(k_i + m | k_i) = A^m x(k_i) + \Sigma(m)^T \zeta, \quad (10)$$

where the matrix is defined as $\Sigma(m)^T = \sum_{i=0}^{m-1} A^{m-i-1} B \mathcal{L}(i)^T$.

Analogously, based on the predicted state variables, the plant output at future sample m is given by

$$y(k_i + m | k_i) = CA^m x(k_i) + \sum_{i=0}^{m-1} CA^{m-i-1} B \mathcal{L}(i)^T \zeta. \quad (12)$$

Under this formulation, the predicted state and output variables are parameterized by the Laguerre coefficient vector ζ . Consequently, ζ serves as the decision variable to be optimized during the design process.

C. Unconstrained control law derivation

Assume that at time k_i , the reference signal R_s remains constant throughout the prediction horizon N_p ,

$$R_s^T = \begin{bmatrix} \underbrace{1 \ 1 \ \cdots \ 1}_{N_p} \end{bmatrix} r(k_i). \quad (11)$$

Incorporating the set-point into the cost function requires a redefinition of the state variable $x(k_i + m | k_i)$. Within the augmented model for SISO systems, the output matrix C is characterized as (see Eq. 9)

$$C = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \end{bmatrix}, \quad (12)$$

with the state vector defined as $x(k) = [\Delta x_m(k)^T, y(k)]^T$. The reference vector $x_r(k_i)$ is introduced, where $x_r(k_i)$ is given by

$$x_r(k_i) = \begin{bmatrix} 0 & 0 & \cdots & 0 & r(k_i) \end{bmatrix}^T, \quad (13)$$

where the dimension of $x_r(\cdot)$ matches that of the augmented state vector $x(\cdot)$, with the number of zero elements corresponding to the dimension of $\Delta x_m(k)$. This construction ensures that $r(k_i) = Cx_r(k_i)$. Furthermore, since the reference signal $r(k_i)$ is assumed to be invariant over the prediction horizon, the inclusion of the set-point signal within the optimization window suggests a redefined state variable $x(k_i + m | k_i)$ of the form

$$x(k_i + m | k_i) = \begin{bmatrix} \Delta x_m(k_i + m | k_i)^T \\ y(k_i + m | k_i) - r(k_i) \end{bmatrix}. \quad (14)$$

The objective function for minimizing the output tracking error is defined as [9]:

$$J = \sum_{m=1}^{N_p} x(k_i + m | k_i)^T Q x(k_i + m | k_i) + \zeta^T R_L \zeta, \quad (17)$$

with $Q = C^T C$ and R_L denotes a symmetric positive-definite matrices.

For the unconstrained case, the optimal parameter vector is derived by substituting (10), with the state redefined as (14), into (17) and solving for the stationary point of the cost functional:

$$\zeta = -\Theta^{-1} \Phi x(k_i). \quad (15)$$

with

$$\Theta = \left(\sum_{m=1}^{N_p} \Sigma(m) Q \Sigma(m)^T + R_L \right), \quad (16)$$

$$\Phi = \left(\sum_{m=1}^{N_p} \Sigma(m) Q A^m \right). \quad (17)$$

Once the optimal parameter vector ζ is obtained, the receding horizon control law is implemented as

$$\Delta u(k_i) = \mathcal{L}(0)^T \zeta. \quad (18)$$

D. Constrained control for SISO systems

While Quadratic Programming (QP) is essential for Multi-Input Multi-Output (MIMO) systems due to its effectiveness in handling active constraints in high-dimensional spaces, it entails considerable computational cost. In standard MPC, constraints are enforced over the entire prediction horizon, leading to a large number of constraints even in Single-Input Single-Output (SISO) systems. However, according to the receding horizon principle, only the first control input of the optimal sequence is applied. If a constraint on u or Δu is active at the first sampling instant, it effectively determines the optimal solution, while later constraints often have limited influence on the immediate control action. Following [9], a simplified approach for SISO systems consists of imposing constraints only on the first sample of the horizon and neglecting the remaining ones. This assumption enables an analytical solution to the constrained problem, significantly reducing computational complexity without substantially affecting closed-loop performance.

E. Constraints on control increments

Assume that the constraints are formulated as $\Delta u^{\min} \leq \Delta u(k) = \mathcal{L}(0)^T \zeta \leq \Delta u^{\max}$. The optimization of the cost function at sampling instant k_i results in three possible outcomes:

- 1) $\zeta = -\Theta^{-1} \Phi x(k_i)$, if $\Delta u^{\min} < \mathcal{L}(0)^T \zeta < \Delta u^{\max}$. The optimal parameter vector is the global minimizer of the quadratic cost function (17).
- 2) If $\mathcal{L}(0)^T \zeta \leq \Delta u^{\min}$ the constrained optimal solution ζ yields $\Delta u(k_i) = \Delta u^{\min}$. Instead of evaluating ζ , the control increment is directly saturated at $\Delta u(k_i) = \Delta u^{\min}$ as the optimal solution.
- 3) Correspondingly, if the unconstrained solution violates the upper bound such that $\mathcal{L}(0)^T \zeta \geq \Delta u^{\max}$, the control increment is constrained to $\Delta u(k_i) = \Delta u^{\max}$.

Once $\Delta u(k_i)$ is obtained, the actual control signal at sampling instant k_i is updated according to $u(k_i) = u(k_i - 1) + \Delta u(k_i)$.

F. Control input constraints

Subject to the input constraints $u^{\min} \leq u(k) \leq u^{\max}$, the optimal solution is determined by one of three cases:

- 1) $\zeta = -\Theta^{-1} \Phi x(k_i)$ if $u^{\min} < u(k_i - 1) + \mathcal{L}(0)^T \zeta < u^{\max}$, (the optimal solution satisfies the constraints).

- 2) If $u(k_i - 1) + \mathcal{L}(0)^T \zeta \leq u^{\min}$ then $u(k_i) = u^{\min}$,
- 3) If $u(k_i - 1) + \mathcal{L}(0)^T \zeta \geq u^{\max}$ then $u(k_i) = u^{\max}$.

III. ENCRYPTED CONTROL SCHEME

Homomorphic encryption allows a third party to perform computation on encrypted data without disclosing information on unencrypted data. In this scenario, there is an interest in not revealing the control values to a remote controller. Through Fully Homomorphic Encryption (FHE) it is possible to perform both additions and multiplications in the encrypted domain. Unfortunately, however, current FHE schemes have a high computational cost, and are therefore unsuitable for real-time control. Consequently, following an approach already proposed in the literature [2], we rely on Partially Homomorphic Encryption (PHE) cryptosystems, which allow for either multiplications or additions in the encrypted domain, but with less computational overhead. In particular, we consider the RSA cryptosystem, which is a public key cryptosystem, and as such can be described by the three functions KeyGen, Enc, Dec:

- KeyGen(λ): on input a security parameter λ , outputs a public key (e, n) and a private key d , where
 - n is the public modulus, which is the product of two large primes p and q , $\log_2(n) \approx \lambda$;
 - e is the public exponent, a positive integer such that $e < (p-1)(q-1)$, $\gcd(e, (p-1)(q-1)) = 1$;
 - d is the private exponent, a positive integer such that $e \cdot d \equiv 1 \pmod{(p-1)(q-1)}$.
- Enc(m): on input an integer plaintext $m < n$ outputs the ciphertext $c = m^e \pmod{n}$;
- Dec(c): on input an integer ciphertext $c < n$ outputs the plaintext $m = c^d \pmod{n}$.

RSA is homomorphic with respect to multiplications; in fact, given two plaintexts m_1, m_2 :

$$\begin{aligned} \text{Dec}(\text{Enc}(m_1) \cdot \text{Enc}(m_2)) &= \text{Dec}(m_1^e \cdot m_2^e \pmod{n}) \\ &= \text{Dec}((m_1 \cdot m_2)^e \pmod{n}) \\ &= (m_1 \cdot m_2)^{e \cdot d} \pmod{n} \\ &= m_1 \cdot m_2 \end{aligned}$$

The security of RSA is based on the hardness of the factorization problem, which given an integer asks to find its unique prime factorization. In fact, an adversary able to factorize n would be able to find p and q , and consequently find the secret key d . Although the factorization problem can be solved in polynomial time with a quantum computer—which is why post-quantum cryptosystems are already being standardized, basing their security on different mathematical problems and aiming to replace RSA to defend us even from attackers equipped with quantum computers—RSA is still widely used in both commercial and industrial applications. The security parameter suggested for RSA by the National Institute of Standards and Technology (NIST) is 2048 bits.

In this work, RSA is employed to encrypt the MPC controller's internal parameters, its reference, and its input,

such that, by exploiting the multiplicative homomorphism of RSA, the multiplications required to find the optimal solution are performed in the encrypted domain. Given that the system operates on vector-valued quantities and considering that RSA cannot simultaneously satisfy both multiplicative and additive homomorphism, it was therefore necessary to define the controller computations as follows. The variables introduced in the previous sections are redefined as

$$\begin{aligned} \zeta &= \zeta_1 + \zeta_2, \Theta^{-1}\Phi = \Psi = [\Psi_1 \ \Psi_2] \\ x(k_i) &= \begin{bmatrix} x_1(k_i) \\ x_2(k_i) \end{bmatrix}, \Delta u(k_i) = -\Delta u_1(k_i) - \Delta u_2(k_i). \end{aligned}$$

Accordingly, the encrypted forms of the equations for eqs. (15) and (18) are given by

$$\text{Enc}(\zeta_1) = \text{Enc}(\Psi_1) \cdot \text{Enc}(x_1(k_i)) \pmod{n} \quad (19)$$

$$\text{Enc}(\zeta_2) = \text{Enc}(\Psi_2) \cdot \text{Enc}(x_2(k_i)) \pmod{n} \quad (20)$$

$$\text{Enc}(\Delta u_1(k)) = \text{Enc}(\mathcal{L}(0)^T) \cdot \text{Enc}(\zeta_1) \pmod{n} \quad (21)$$

$$\text{Enc}(\Delta u_2(k)) = \text{Enc}(\mathcal{L}(0)^T) \cdot \text{Enc}(\zeta_2) \pmod{n} \quad (22)$$

$$\Delta u(k_i) = -\text{Dec}(\text{Enc}(\Delta u_1(k_i))) - \text{Dec}(\text{Enc}(\Delta u_2(k_i))). \quad (23)$$

This simple reformulation enables the separation of multiplication operations from addition operations. The encrypted multiplications (eqs. (19) to (22)) are executed on the *master* unit, while the additions (23) are performed, in the domain of plaintext data, on the *slave* unit—the plant-side unit responsible for signal encryption and decryption. It is important to emphasize that, since the parameters Ψ_1, Ψ_2 and $\mathcal{L}(0)^T$ are computed entirely offline, they are preloaded in encrypted form into the master unit to enable the computation of the control law. Comprehensive details regarding the system components and their respective functionalities are presented in Section IV.

A. Conversion from real value to plaintext

An issue that has been neglected in Section III for simplicity, is that of encoding the control system variables into the plaintext space. RSA requires the input to the encryption function to be a positive integer smaller than the public modulus n . Control system variables take real values that can be both positive and negative. Therefore, before encrypting the variables that must be sent to the controller, a fixed-point encoding is necessary. This is performed by scaling the values by a power of 10 corresponding to the required precision, and then rounding to the nearest integer. If the value is negative, a sign reversal is performed. Upon receiving the values back from the controller, the values are scaled back and the sign is inverted when needed.

IV. EXPERIMENTAL TESTBED

The DC motor drive shown in Fig. 1 serves as the experimental testbed to evaluate the proposed controller encryption scheme. The results presented focus on assessing the applicability, quantifying the performance degradation induced by the encryption process, and determining the

TABLE I: Control scheme parameters

MPC Regulator Parameters	
$\alpha = 0.03$	$N_c = 10$
$R = 0.01$	$\Delta u^{min} = -0.03 \quad \Delta u^{max} = 0.03$
$N = 1$	$u^{min} = 0.5 \quad u^{max} = 3.3$
$N_p = 100$	

real-time computational overhead. The experimental platform utilizes a DC motor whose shaft velocity is measured by an integrated tachometric generator ([10]). Control and data exchange between the system's components are managed by two Raspberry Pi ([11], [12]) over an Ethernet link. Both Raspberry Pi units are configured with a real-time Raspbian-Linux environment. The control system tasks are distributed such that one unit (*master*), performs the controller homomorphic multiplications, and the second (*slave*) manages the plaintext addition to compute the control voltage, the cryptographic operations, sensor input/reading, and the actuation of control commands for the DC motor. The system plant consists of a DC motor whose dynamics are assumed to be linear. The system input is the voltage applied to the motor driver, and the output is the velocity feedback provided by the tachometer. A transfer function was derived for the plant through black-box identification, resulting in:

$$P(s) = \frac{46}{s + 42} \quad (24)$$

The continuous-time transfer function has been discretized to find the discrete-time state-space model of Eq. (6). The most suitable controller parameters identified are listed in Table I. The RSA encryption scheme was implemented using key lengths of 512 and 1024 bits, representing security levels that closely approximate practical and realistic settings. This choice allows for the evaluation of control performance under different encryption strengths, providing insight into the trade-offs between computational load, dynamic response and system security. Experimentally, the control loop interval was adjusted to accommodate the increased computational load associated with longer RSA keys. As key length increases, the encrypted operations become more time-consuming, necessitating longer intervals between successive control cycles to ensure proper execution and maintain system stability. For the unencrypted controller, a control loop interval of 20 ms was selected. When using RSA encryption, the interval was increased to 50 ms for 512-bit key and 100 ms for 1024-bit key. Fig. 2 presents a flow diagram describing the encrypted control loop. The slave, utilizing a WaveShare high-precision AD/DA module, reads the rotational velocity from the tachometer. This unit is responsible for updating and encrypting the state variable, and subsequently transmitting the encrypted value to the master across the network. The master exploits homomorphic properties to create a multi-signal encrypted representation of the control input, which is then returned to the slave. Finally, the slave decrypts the received signals to yield the control

input, which is then converted into a voltage command by the AD/DA board and applied to the motor driver circuit. All control and encryption logic deployed in the testbed system was coded in C-language. Encryption/decryption and operations over encrypted data were performed using the OpenSSL library, taking advantage of its implementation of the RSA cryptosystem and integer modulo operations.

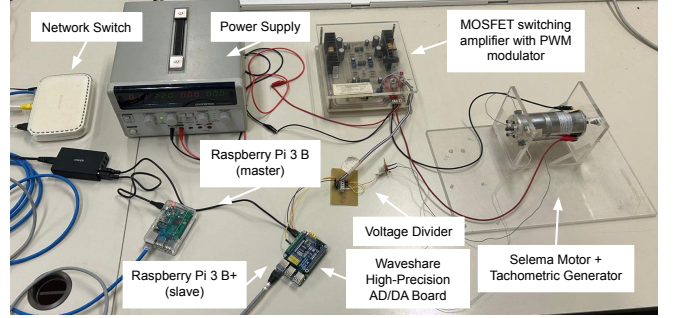


Fig. 1: Experimental setup

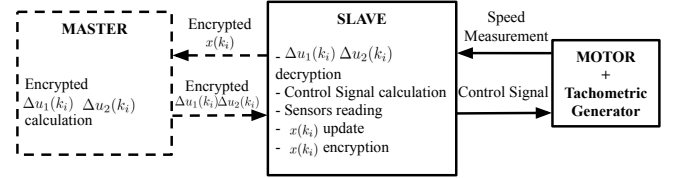


Fig. 2: Master-slave control loop with homomorphic encryption

V. PERFORMANCE ASSESSMENT

Although homomorphic multiplications significantly increase computational load and slow down the control loop, the resulting impact on system performance is moderate. Figures 3 and 4 compare the motor speed responses and control signals for unencrypted control and for encrypted control using 512-bit and 1024-bit RSA keys. As the key length increases, small delays become evident; however, stable and effective regulation is preserved in all cases, demonstrating that secure control can be achieved without major degradation of system dynamics. The observed delays are mainly attributable to the computational and memory limitations of the Raspberry Pi platform, as reflected in the encryption and decryption times reported in Table II, together with the average execution time of the control cycle. By examining the latter, it is easy to observe that increasing the key length by 512 bits introduces approximately 13 ms of delay. This highlights the trade-off between cryptographic strength and real-time constraints, requiring careful selection of the key length to ensure compatibility with the control sampling period. Overall, the experimental results confirm that the proposed encrypted control scheme satisfies deterministic timing requirements when appropriate loop intervals are adopted, even on resource-constrained hardware. To further assess performance, the Integral Absolute Error (IAE) and Mean Squared Error (MSE) were evaluated over the steady-state interval shown in Fig.5, with results summarized in Table

III. The unencrypted controller provides baseline values of $IAE = 2.364$ and $MSE = 1.910$. Applying encryption with a 512-bit key leads to a notable increase in both indices, indicating reduced tracking accuracy. Conversely, the 1024-bit key yields error values closer to the unencrypted case, suggesting that appropriate tuning of the control loop interval can effectively compensate for the increased computational burden.

TABLE II: Average computational times for encryption, decryption and control loop

Key length	Encryption time (ms)	Decryption time (ms)	Control loop time (ms)
no encryption	—	—	18.0432
512	0.4052	2.5771	21.9125
1024	0.8277	13.8231	34.1613

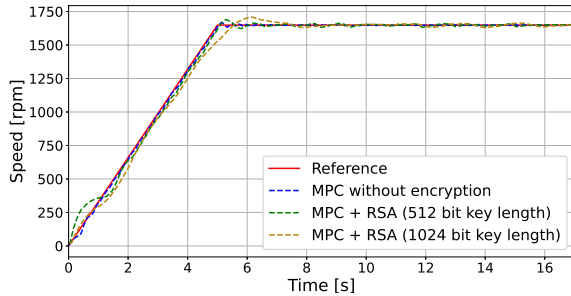


Fig. 3: Motor speed signals comparison: reference speed (red continuous line), unencrypted MPC (blue dashed line), encrypted MPC with RSA (512 bit key length) (green dashed line), encrypted MPC with RSA (1024 bit key length) (yellow dashed line)

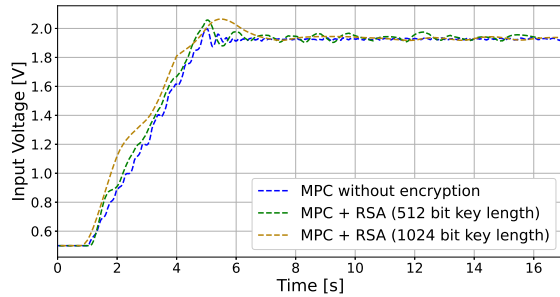


Fig. 4: Control signals comparison: unencrypted MPC (blue dashed line), encrypted MPC with RSA (512 bit key length) (green dashed line), encrypted MPC with RSA (1024 bit key length) (yellow dashed line)

TABLE III: IAE and MSE values for the analyzed configurations

Controller	IAE	MSE
Unencrypted MPC	2.364	1.910
MPC + RSA (512 key length)	9.766	37.302
MPC + RSA (1024 key length)	10.781	33.562

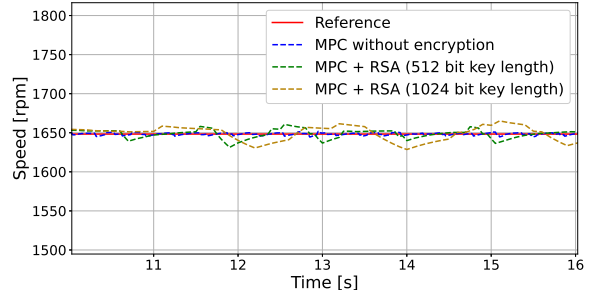


Fig. 5: Zoom of the motor output speed in steady state

VI. CONCLUSION

This study presented an encrypted Model Predictive Control (MPC) framework utilizing Laguerre functions for computational efficiency, integrated with the RSA cryptosystem to ensure data privacy. Experimental results indicate that control performance remains robust despite the non-negligible computational overhead inherent in the homomorphic multiplications required by the encrypted controller.

REFERENCES

- [1] A. A. Cárdenas, S. Amin, and S. Sastry, "Research challenges for the security of control systems," in *Proceedings of the 3rd Conference on Hot Topics in Security*, ser. HOTSEC'08. USA: USENIX Association, 2008.
- [2] K. Kogiso and T. Fujita, "Cyber-security enhancement of networked control systems using homomorphic encryption," in *2015 54th IEEE Conference on Decision and Control (CDC)*, 2015, pp. 6836–6843.
- [3] F. Farokhi, I. Shames, and N. Batterham, "Secure and private control using semi-homomorphic encryption," *Control Engineering Practice*, vol. 67, pp. 13–20, 2017. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0967066117301557>
- [4] M. Schulze Darup, A. Redder, I. Shames, F. Farokhi, and D. Quevedo, "Towards encrypted mpc for linear constrained systems," *IEEE Control Systems Letters*, vol. 2, no. 2, pp. 195–200, 2018.
- [5] K. Zhang, Z. Li, Y. Wang, and N. Li, "Privacy-preserving nonlinear cloud-based model predictive control via affine masking," *Automatica*, vol. 171, p. 111939, 2025.
- [6] N. Li, A. Girard, and I. Kolmanovsky, "Stochastic predictive control for partially observable Markov decision processes with time-joint chance constraints and application to autonomous vehicle control," *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, vol. 141, no. 7, 2019.
- [7] N. Li, K. Zhang, Z. Li, V. Srivastava, and X. Yin, "Cloud-assisted nonlinear model predictive control for finite-duration tasks," *IEEE Transactions on Automatic Control*, vol. 68, no. 9, pp. 5287–5300, 2023.
- [8] A. Sultangazin and P. Tabuada, "Symmetries and isomorphisms for privacy in control over the cloud," *IEEE Transactions on Automatic Control*, vol. 66, no. 2, pp. 538–549, 2021.
- [9] L. Wang, "Discrete model predictive controller design using Laguerre functions," *Journal of Process Control*, vol. 14, no. 2, pp. 131–142, 2004.
- [10] Esa Automation. Selema series 30 motor + tachometric generator specifications. [Online]. Available: <https://www.esa-automation.com/wp-content/uploads/2018/10/DC-series-Motors-63-71.pdf>
- [11] Raspberry Pi. Raspberry pi 3 model b. [Online]. Available: <https://www.raspberrypi.com/products/raspberry-pi-3-model-b/>
- [12] —. Raspberry pi 3 model b+. [Online]. Available: <https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/>