Control Implemented on Quantum Computers: Effects of Noise, Non-Determinism, and Entanglement

Kip Nieman and Keshav Kasturi Rangan and Helen Durand*

Department of Chemical Engineering and Materials Science, Wayne State University, Detroit, MI 48202. The first two authors contributed equally on this work.

E-mail: helen.durand@wayne.edu

Abstract

Quantum computing has advanced in recent years to the point that there are now some quantum computers and quantum simulators available to the public for use. In addition, quantum computing is beginning to receive attention within the process systems engineering community for directions such as machine learning and optimization. A logical next step for its evaluation within process systems engineering is for control, specifically, for computing control actions to be applied to process systems. In this work, we provide some initial studies regarding the implementation of control on quantum computers, including the implementation of a single-input/single-output proportional control law on a quantum simulator with noise, evaluation of potential impacts of non-determinism on theory for advanced control laws, and discussion of consequences of the way that entanglement works for next-generation manufacturing communication objectives.

1 Introduction

In modern and next-generation manufacturing processes, control systems are critical in maintaining and improving operation. Advances in computers have allowed for implementation of increasingly sophisticated algorithms to meet these needs; however, the limits of current computing strategies continue to be an obstacle to the development of model or optimization-based control algorithms for increasingly complicated processes. One might ask whether quantum computation could hold any potential for outperforming classical computation methods for control, as there are some computational problems where quantum computing has shown a potential for outperforming classical algorithms according to metrics like function evaluations.[1] However, it is currently unknown whether quantum computing could provide any benefits for control systems. It is our premise, however, that the answer to this question could benefit from the development of a deeper understanding of how control and quantum computing interact. Specifically, the unique nature of quantum computing creates new complexities that need to be understood from a control-theoretic standpoint. This includes "noise" (which can be considered to be deviations in qubit behavior compared to what is intended) which on today's quantum computers can introduce errors into computations.[2,3] Another challenge is that some quantum computing algorithms (even if implemented on perfect/noiseless circuits) do not provide an answer with certainty (but only in probability), with the result that the algorithm may need to be run a number of times to see what the most likely answer is.[1] These types of challenges, foreign to the implementation of control today, lead us to suggest that examining relationships between control and quantum computers could provide a deeper understanding of the way that quantum computers interact with control objectives and could provide part of the path toward answering the question of whether or not quantum computers hold any benefits for use in control.

In the process systems engineering field, quantum computers (primarily the D-Wave quantum annealer, though with some analysis of an IBM quantum computer in[4]) have been utilized for fault detection and for optimization related to energy systems.[4–7] Applications of quantum computers for control to date have included, for example, the application of fuzzy logic control,[8] reinforce-

ment learning,[9–11] and model predictive control or MPC (an optimization-based and model-based control law[12–16]) implemented on a quantum annealer.[17] However, this is a relatively limited set of investigations that does not provide enough data for deeply understanding benefits and limitations of control implemented on quantum computers.

This work marks our group's first step in the direction of fleshing out such relationships between control theory and design and quantum computation in greater depth, and thus takes a preliminary look at implementing control algorithms on quantum computers and at analyzing how related quantum properties interact with control algorithms and theory. This work is organized as follows. The first section gives an overview of some quantum computing topics which underlie the rest of the work. Next, the impacts of several phenomena in quantum computing on control are examined. The first is "noise" in the quantum computing hardware. The closed-loop simulation of a single input/single output system controlled using a simple controller on a quantum simulator with noise utilizing a quantum Fourier transform (QFT)-based addition algorithm is used to analyze the impacts of noise in the hardware on the behavior of a closed-loop dynamic system. Next, we examine the effects of non-determinism in an algorithm on a quantum computer for control using an advanced control law known as Lyapunov-based economic model predictive control (LEMPC)[18] represented by a lookup table. A modified version of Grover's search algorithm (a quantum computing algorithm for finding an element known to be present in an unordered list, with a certain probability) is then applied (inefficiently but still non-deterministically) to select control inputs from the lookup table, given a sensor measurement. We are then able to draw control-theoretic conclusions for LEMPC applied via this non-deterministic algorithm, providing a preliminary look at how non-determinism in quantum computing algorithms can impact the associated control theory. Finally, we discuss implications of the quantum phenomenon of entanglement for communication for next-generation manufacturing.

2 Principles of Quantum Mechanics and Quantum Computing

Here we define several important terms related to quantum mechanics and computing that will be utilized throughout this work, adopted from notation and quantum mechanical frameworks discussed in.[1] Quantum computers are devices that take advantage of the principles of quantum mechanics to perform operations on quantum particles in a manner that causes their measurement at the end of a series of operations to reveal answers to questions which can be programmed. The fundamental unit of quantum information is a qubit, which like traditional bits, can take values of 0 or 1 (written as $|0\rangle$ and $|1\rangle$ using "bra-ket" notation or $[1\ 0]^T$ and $[0\ 1]^T$ using vector notation). However, a unique feature of qubits due to the manner in which they take advantage of quantum mechanics is that qubits can also be in a "superposition" of the states, which is written as a linear combination of the basis states $|0\rangle$ and $|1\rangle$ as $c_1 |0\rangle + c_2 |1\rangle$. Alternately, the state of a qubit can be written in vector notation as $[c_0\ c_1]^T$. The values of $|c_1|^2$ and $|c_2|^2$ represent the probabilities that the qubit will be in state $|0\rangle$ or state $|1\rangle$ upon measurement of its state, respectively, where $|c_1|^2 + |c_2|^2 = 1$. With slight abuse of notation, $|c|$ represents the modulus of a complex number if the argument of $|\cdot|$ is a complex scalar.

An $n \times n$ matrix U is unitary if for its Hermitian transpose U^+, $UU^+ = U^+U = I_n$. Matrices describing the dynamics of quantum systems must be unitary. The tensor product of two vectors $a = [a_0, \ldots, a_m]^T$ and $b = [b_0, \ldots, b_n]^T$ is defined by $a \otimes b = [a_0 b_0, a_0 b_1, \ldots, a_n b_m]^T$. The tensor product of two matrices A and B is the matrix formed by multiplying each component of A by the matrix B. Quantum mechanical systems can be assembled (in the sense that more than one particle can be present in a distinct superposition of states which will each collapse with a certain probability to a given state upon measurement as described above) by taking the tensor product of the state vectors of each individual particle. The state of a multi-qubit system, $|\Psi\rangle$, is represented via a vector $[c_0, c_1, \ldots, c_{n-1}]^T$, where c_i, $i = 0, \ldots, n-1$, are complex numbers for which $\sum_{i=0}^{n-1} |c_i|^2 = 1$, where each of the coefficients in the vector are related to the probability of observing the qubit

system in one of n possible states upon measurement. Before the system state is measured, it is in a superposition of some number of individual states represented by each of the nonzero entries of the vector. When measured, the probability that it is found in the i-th state is given by $|c_i|^2$. The state of the system can also be written as a linear combination of each of the basis states $|\Psi\rangle = c_0 |x_0\rangle + c_1 |x_1\rangle + ... + c_{n-1} |x_{n-1}\rangle$, where $|x_i\rangle = [0, \ldots, 1, \ldots, 0]^T$, which represents a vector with a 1 at the i-th position (indexed from $i = 0$ to $i = n-1$). The matrix describing the evolution of the assembled system is the tensor product of the matrices describing the transitions within each individual system which comprises the assembly.

An "entangled" state is one which cannot be represented as the tensor product of individual state vectors. Important examples of two-qubit entangled states are the Bell states or Einstein, Podolsky, and Rosen (EPR) states, such as $|\beta_{00}\rangle$ and $|\beta_{01}\rangle$, which are defined by[19] as:

$$|\beta_{00}\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}} \tag{1}$$

$$|\beta_{01}\rangle = \frac{|01\rangle + |10\rangle}{\sqrt{2}} \tag{2}$$

An important aspect of these states is that, since the qubits are entangled, the measurement of one qubit is tied to the state of the other qubit. For example, consider the state $|\beta_{00}\rangle$. If the first qubit is measured in the $|0\rangle$ state, the second qubit will be in the same state. Likewise, if the first qubit is measured in the $|1\rangle$ state, the second qubit will be in the same state.

Quantum logic gates are elements in a quantum circuit that transform qubit(s) between multiple quantum states. Logic gates in a computer, like qubits, can also be represented using vectors and matrices. For example, the NOT gate can be represented by the matrix [0 1; 1 0], because multiplying this matrix by $|0\rangle$ gives back $|1\rangle$, and vice versa, which is similar to its action in classical circuits. Gates in a quantum computer must be reversible, and therefore must be represented by unitary matrices which act on qubits. An important unitary operation for placing qubits in a superposition

of states is denoted by the "Hadamard matrix" H and is represented as follows:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{3}$$

An additional qubit can be used to determine when a gate is activated, forming a controlled gate. In this case, if the input state of the control bit is $|0\rangle$, the gate will not be executed so the output will be identical to its input. Otherwise, when the control bit is in the $|1\rangle$ state, the controlled gate will operate on its qubits. It is also possible to apply multiple control qubits to a quantum gate. In the case of applying three control qubits, for example, the quantum circuit can be designed in a way where the controlled gate only activates for a particular combination of control bits (for example, the controlled gate only activates when the control bits are in the state $|111\rangle$).

3 Implementing Closed-loop Control on a Simulated Noisy Quantum Computer

In this section, we provide the first of three studies relating control and quantum computing. Given that today's quantum computers are still unable to achieve noiseless operation,[20] it therefore would be expected that on today's quantum computers, noise would also impact the computation of control actions. Because of this, we look at a simple controlled system using a simulation of a controller on a quantum simulator with noise. This simulation uses the quantum Fourier transform (QFT)-based addition algorithm. This section is an extended version of work in.[21]

3.1 Implementing Closed-loop Control on a Simulated Noisy Quantum Computer: Preliminaries

Here, we present the preliminaries required specifically for the discussion regarding simulating closed-loop control on a noisy quantum computer; specifically, we discuss the Quantum Fourier Transform (QFT) and QFT-Based Addition. The quantum Fourier transform (QFT) algorithm

is used in a variety of quantum algorithms such as addition and Shor's factoring algorithm.[1,19] It is the quantum analog of the discrete Fourier transform and when implemented on a quantum state $|\Psi\rangle = \sum_{j=0}^{N-1} \hat{a}_j |j\rangle$, for complex scalars $\hat{a}_j$, $j = 0, \ldots, N-1$, it returns $\sum_{k=0}^{N-1} \hat{b}_k |k\rangle$ with $\hat{b}_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \hat{a}_j e^{2\pi i jk/N}$.[22]

The Quantum Fourier Transform has been used in performing addition on quantum computers,[23] and the addition circuit is reflected in Fig. 1. Specifically, the circuit in Fig. 1 adds two integers $\bar{a}$ and $\bar{b}$. The binary representations of these integers are reflected in the figure; for example, the binary representation of $\bar{b}$ is considered to be $n-1$ bits long with an additional 0 appended at the n-th (most significant) bit, and the resulting n qubits are labeled $|b_i\rangle$, $i = 0, \ldots, n-1$ in the figure. The circuit in Fig. 1 uses QFT (consisting of a combination of Hadamard and controlled rotation gates) and inverse QFT ($QFT^\dagger$). The output of each qubit of this circuit represents one of the bits of the sum of $\bar{a}$ and $\bar{b}$. To provide more specifics, consider a positive integer $\bar{a}$; the binary representation of $\bar{a}$ (denoted by a) consists of a_i, $i = 0, \ldots, n-2$, and is placed in a set of $n-1$ qubits (where $n-1$ is the number of bits in the binary representation of $\bar{a}$, with the most significant bit being a_{n-2} and the least significant being a_0). An additional 0 is added as the most significant bit so that $a_{n-1} = 0$, without changing the value of a when it is rewritten in the decimal system. The set of gates labeled "QFT" in Fig. 1 are a series of controlled Z_k gates ($Z_k = \text{diag}(1, e^{\frac{2\pi i}{2^k}})$, where $\text{diag}(x)$ represents a diagonal matrix with the elements of the vector x on the diagonal) and Hadamard gates, followed by swap gates (with H representing the Hadamard gate, swap gates represented by cross-hairs, and the notation used for describing a series of controlled Z_k gates defined in Fig. 2, where the open circles on a circuit represent control qubits which dictate whether a gate which they control is executed by taking values of $|0\rangle$ (no execution) or $|1\rangle$ (gate execution)).

3.2 Closed-loop Control using a Simulated Noisy Quantum Computer

We utilize a simulation study with a simple system ($\dot{x} = x + u$, which can be readily stabilized using the control law $u = -2x$ when that controller is implemented on a classical computer) to elucidate relationships between quantum computing-implemented control and noise in quantum computers.

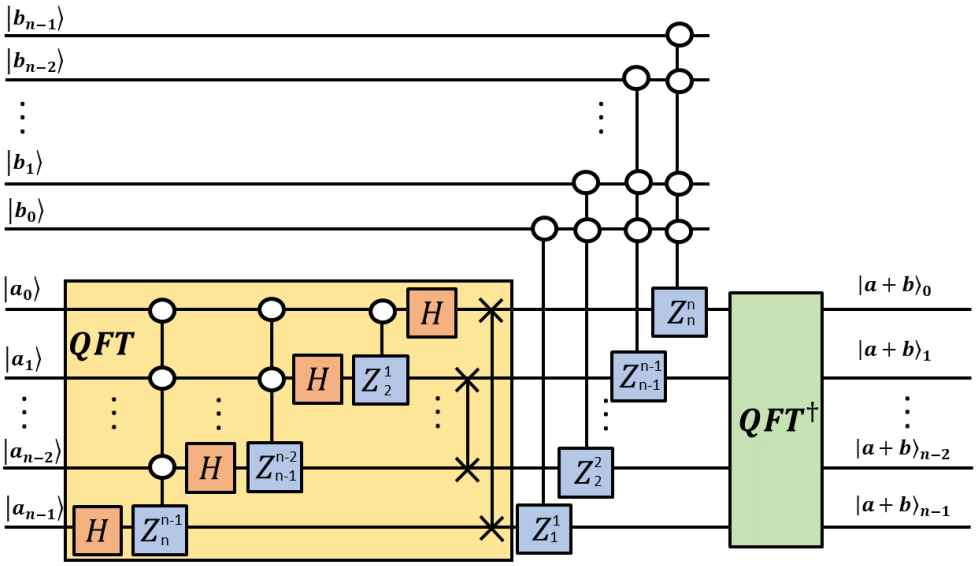

Figure 1: Quantum addition circuit.

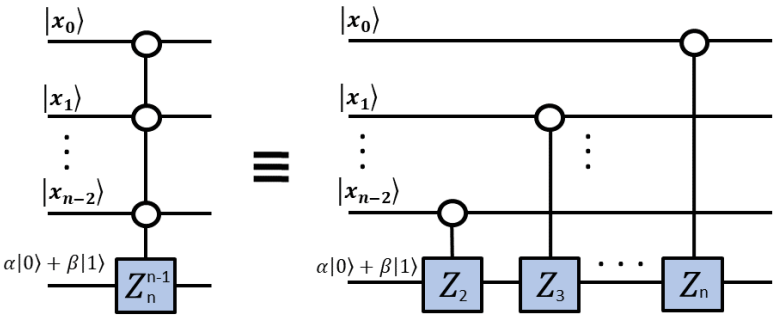

Figure 2: Z_n^{n-1} circuit definition. On the left is simplified notation for the full circuit on the right.

The control loop considered is shown in Fig. 3. In this simulation study, the control action $u = -2x$ is computed on a quantum simulator from IBM (the qasm_simulator) using the quantum Fourier transform code for addition of two binary numbers (x and x to form $2x$), with the QFT-based addition code largely based on.[24] Since this simulation is performed using a quantum simulator rather than an actual quantum computer, a noise model is required to simulate noise that would be present when performing the calculations on a quantum computer. This model is added to the quantum simulator to simulate noise in quantum circuitry, so that there is randomness in a single computation of u (or "shot" as it is called in the IBM Quantum Experience) despite that QFT transform-based addition is a deterministic algorithm if a quantum computer has no noise. We then compare the case with a single "shot" used in a computation of $2x$ with the case that multiple shots are used, and also compare with the result when a classical computer is utilized.

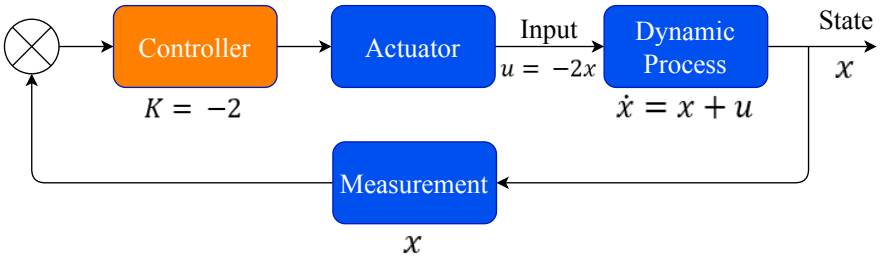

Figure 3: Simple control law for a linear process

3.2.1 Noise Model for Quantum Computing-Implemented Controller

The 1-qubit and 5-qubit quantum computers available for free use on IBM's Quantum Experience inherently have noise but do not have enough qubits to enable a meaningful computation of the process input, $u = -2x$. For this reason, a quantum simulator (the IBM qasm_simulator in Qiskit) is used to emulate the action of a quantum computer. The simulator used here is a 32 qubit system and can be used to simulate quantum circuits both ideally (i.e., in the absence of noise) as well as in the presence of noise by importing noise models into the simulation.

The noise model on the quantum simulator can be derived using the noise from an actual device (such as the 5-qubit quantum computer ibmq_manila) or via custom noise modeling, where different types of error models and their parameters can be specified by the user. Utilizing the approximated

noise from an actual quantum device would be preferable for studying control on quantum computers. However, since the noise model is related to the actual quantum computer, it does not execute on qasm_simulator when the number of qubits used on a circuit in the qasm_simulator exceeds the number of qubits in the quantum computer. Therefore, a custom noise model was used in the control simulations.

While there exist numerous types and sources of quantum errors,[25–28] the custom noise model in this work considers only a depolarizing error in the controlled phase gate used in the simulations. When this is performed, the error parameter must be specified. To select a reasonable error parameter, an actual quantum device (in this case, the 5-qubit ibmq_manila) was selected and a noise model for this device was approximated using the NoiseModel.from_backend command in the open source software development kit Qiskit.[29] Then simulations of a controlled-Z gate (which represents a controlled rotation of 180 degrees) with two qubits initially in the $|0\rangle$ state were conducted using the captured noise model and custom noise models, where the error parameter in the custom noise model was tuned to ensure that the results approximately match.

A controlled-Z gate was used to find the error parameter because the basis gates in ibmq_manila do not include the H or controlled rotation gates (which are the gates used in the QFT circuit). However, the simulator includes a controlled-NOT (CNOT) gate, which in combination with 2 H gates is the equivalent of a controlled rotation of a specific amount (180 degrees) as shown in Fig. 4.[30] Therefore, because rotation operations are utilized in the QFT circuit, the controlled-Z gate developed from a combination of a CNOT gate and 2 H gates was assumed to be a reasonable benchmark for determining the depolarizing error parameter for other controlled rotation gates in the QFT algorithm.

The depolarizing error parameter was set to 0.05 in the noise model based on a comparison in August 2021 of the probabilities of seeing different results after performing the controlled-Z representation from Fig. 4 on the qasm_simulator using the noise model generated from ibmq_manila. An example of the results from a run (consisting of multiple iterations or shots) of the two simulators is shown in Fig. 5. As shown in Fig. 5, the results from the noisy simulations in both cases are similar, so that setting the depolarizing error parameter to 0.05 for a controlled rotation gate

in the QFT algorithm was assumed to be reasonable. However, as also shown in Fig. 5, the noise model from the backend is not necessarily constant with time; therefore, we will later investigate the impact of different depolarizing error parameters.

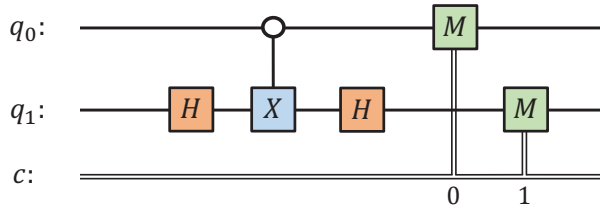

Figure 4: Controlled-Z gate using 2 H gates and a CNOT gate. M stands for measurement of a qubit.

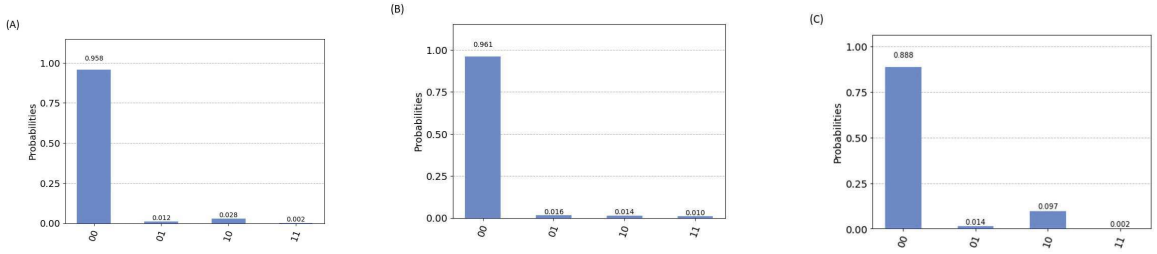

Figure 5: Comparison of probabilities of different results after 1024 shots when (A) the simulation uses a depolarizing error parameter of 0.05, and (B) when it uses the noise simulation derived from ibmq_manila (an example from August 21, 2021), and (C) when it uses the noise simulation derived from ibmq_manila (an example from April 8, 2022).

3.2.2 Results from Quantum Computing-Implemented Control

A controller implemented using QFT-based addition on a quantum simulator requires state measurements from the process in order to calculate control inputs. The process state measurements are simulated on a classical computer, so they are decimal numbers with high precision (many decimal places). Since the quantum algorithm works with binary numbers, the state measurements must be modified before being supplied to the quantum algorithm. The modification procedure involves

truncating the state measurement to 2 decimal places by multiplying the process state by 100 and taking the floor of the consequent value. The resulting binary number is supplied to the quantum algorithm by representing each digit as a qubit in state $|0\rangle$ or $|1\rangle$. After the quantum algorithm is performed and the qubits are measured, the resulting binary number represents the process input, $u = 2x = x + x$, which must be converted back to an integer. To do this, the binary number is converted to an integer and then divided by 100.

When multiple shots of the QFT-based addition algorithm are run, the full procedure for obtaining the result of $x+x$ is obtained by running the Result.get_counts function in Qiskit, and the number returned by performing "max" on the result with the operator.itemgetter() function in Python is taken to be the control action. The negative sign in the full control law (i.e., $u = -2x = -(x+x)$) is accounted for by taking the negative of the value of $2x = x + x$ found by this procedure. Ten hours of operation of the process were simulated, with the initial condition $x(0) = 7.4$, using the Explicit Euler numerical integration method with an integration step of 0.001 time units and a sampling period of 0.1 time units. Simulations described below were performed using one of two systems (either a Mac running Python 3.8.8 with a 3.1 GHz Dual-Core Intel Core i5 and 8 GB 2133 MHz LPDDR3, or an Intel(R) Xeon(R) CPU E3-1240 v5 at 3.50GHz, 3.50 GHz with 32.0 GB of RAM and running Windows 10, with Python 3.9.7) with the Python Integrated Development Environment (IDE) Spyder.

To demonstrate the impact of noise in quantum computers on the results, simulations have been executed on a quantum simulator using the depolarizing error noise model described above, and the results are compared against those from a classical digital computer and QFT-based addition. The results from computing $x + x$ via classical addition and QFT-based addition in the absence of noise are the same since QFT-based addition is a deterministic algorithm. The state and input trajectories for the three simulations are compared in Fig. 6, with the noisy quantum simulation being run with 254 shots, and in Fig. 7 with the noisy quantum simulation being run with 1 shot. The figures show that the results obtained using the classical computer and simulator for the noise-free case match (the input trajectories computed were the same in both cases). In both the case that 1 shot is used and when 254 shots are used, the state and input trajectories associated with

the noisy quantum computer are not overlaid with those in the absence of noise.

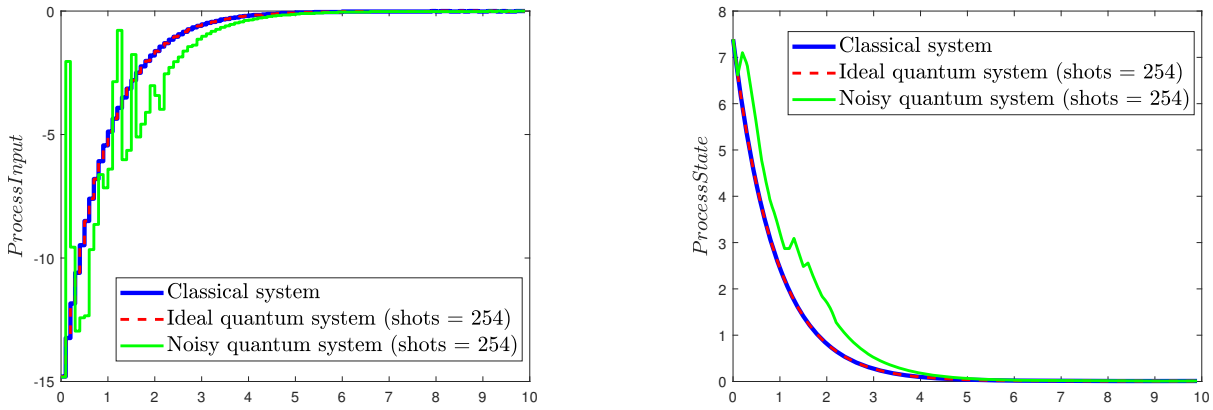

Figure 6: Comparison between the input (left) and state (right) trajectories when the control input is computed on a classical computer ("Classical system"), on a quantum simulator with 254 shots and no noise ("Ideal quantum system"), and on a quantum simulator with 254 shots and noise ("Noisy quantum system"), for $x(0) = 7.4$. The x-axis is time units.

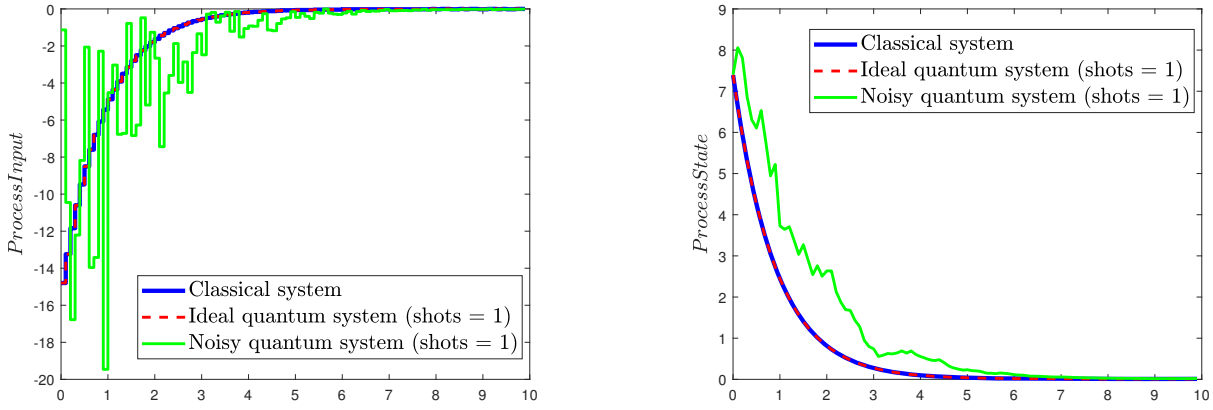

Figure 7: Comparison between the input (left) and state (right) trajectories when the control input is computed on a classical computer ("Classical system"), on a quantum simulator with 1 shot and no noise ("Ideal quantum system"), and on a quantum simulator with 1 shot and noise ("Noisy quantum system"), for $x(0) = 7.4$. The x-axis is time units.

An important step toward the future development of more advanced quantum computing-implemented control laws is understanding why the results for this simple system are what they are. The following subsections will therefore seek to probe this simulation, analyzing what impacts the following different aspects of the simulation have on the results, in particular closed-loop stability in the sense of driving the closed-loop state toward the origin: 1) the state measurement magnitude

and number of shots; 2) the implementation of the QFT-based addition algorithm; and 3) the error parameter in the depolarizing error model.

Remark 1. *We note that though this section focuses only on control actions which can be written as some multiple of the state of a system, this class of control actions also encompasses the results of MPC for the linear unconstrained tracking (quadratic objective function) case, where an explicit solution given in the form $-Kx$, where K is a matrix.*[31]

Results from Quantum Computing-Implemented Control: Impacts of State Measurement Magnitude and Number of Shots

In the QFT-based addition algorithm implemented, both operands have the same number of digits in binary form. The algorithm used in this section checks this number of bits before performing the addition algorithm and only performs addition for the required number of bits. For control design, this has a special effect: as the closed-loop state moves toward the origin, the magnitude of the measured value of x decreases, meaning that the size of the binary representation of x also decreases. In computing $x + x$ with QFT-based addition, this means that in the implementation of the algorithm utilized, as the state is driven toward the origin, the QFT-based addition code is based on smaller numbers of qubits.

An impact of this is suggested by Figs. 6-7, which suggest that the size of the state measurement and the number of shots used may have an impact on how closely the results from the noisy computer match those obtained deterministically (though because the impact of the noise is different each time that the simulation is run, this should be understood in terms of general trends and not necessarily every specific simulation). Specifically, Figs. 6-7 suggest that, in accordance with the expectation from Fig. 5 that control inputs might not be "correct" if determined via a single shot, but may be more likely to be "correct" if more shots are performed and the control action is taken to be the result from applying "max" on Result.get_counts (the use of this specific operation is emphasized because there may be multiple binary strings which appeared the same number of times when the simulation was run for a given number of shots), the state trajectory when 254 shots are used by the quantum computer more closely matches the no noise and classical cases than the state trajectory

when 1 shot is used. A contributing factor for not observing an exact overlay of all the plots in Fig. 6 is that the initial condition is 7.4, which after scaling by 100 and truncation to two digits as described above to prepare the number for QFT-based addition, is 740 (1011100100). In binary, this number is 10 qubits long, and with an extra 0 added to the beginning of the number to account for its potential growth by one more qubit during QFT-based addition (Section 3.1), it is a qubit register of length 11. If every one of the 11 qubits could potentially be corrupted by noise, 2048 possible process inputs would exist, which is much larger than the 254 shots used. This suggests that 254 shots may not be sufficient to cause a binary string with the maximum occurrences over multiple shots to match the "correct" (or deterministic) result that would have been evaluated by a classical computer.

Two additional observations of interest are a consequence of randomness introduced to the control action evaluated by the quantum computer after each run as a consequence of noise applied via the controlled phase ("cp") gates. This raises the question of how to verify safety of a controller when it would be difficult to know whether every potential case that might occur was visited in a simulation case study on the computer. Another interesting observation is that as the closed-loop state moves closer to the origin (steady-state), the size of the qubit register required to represent its state decreases in the implementation of QFT-based addition used based on[24] (for example, when the closed-loop state is 0.01, the number 1 would be fed to the QFT-based algorithm, which requires only one qubit to represent it and thus 2 qubits in the operation of computing $x + x$). When 254 shots are used and only 2 qubits are needed to represent each binary possibility, there exist only 4 possible values that the process control input can take, which implies that it may be more likely, when the depolarizing error parameter is 0.05, that the input with the maximum occurrences over the 254 shots matches the deterministic result. This suggests that control algorithms on quantum computers may be able to adapt the number of shots required based on the state measurement (e.g., to require less shots when the state measurement is closer to the origin), depending on how they are implemented/coded. It also indicates that the deviation variable form of the state and input may be useful for implementation of control laws on quantum computers, as it may enable the state measurements provided to a controller to approach smaller values as the system is stabilized that

may require less bits to represent, and this may also enable less shots to be required to attempt to likely obtain the desired control action.

To further investigate the impact of the size of the qubit register needed for addition on the agreement between the state trajectories under the deterministic input policies and under the input computation affected by noise, an additional set of simulations using a classical computer, noiseless quantum simulator, and the quantum simulator with noise, where both quantum systems were simulated with 1 shot, 60 shots, and 254 shots for $x(0) = 0.74$, was also analyzed. The reason for this change in the initial condition is that we would like to test, in accordance with the above paragraphs, whether increasing the number of shots compared to the number of possible inputs that could be computed given the state measurement has an impact on whether the deterministic control action is likely to be selected on the noisy computer. If we consider the prior initial condition of 7.4, it would be necessary to add additional shots beyond 254 to test this, which will increase the computational load required for running the simulation. If instead the initial condition is changed to 0.74 (so that the initial condition can be represented in binary as 1001010 after correcting for 2 decimal places), then only $7 + 1$ qubits (which includes the extra 0 at the beginning of the binary representation) are needed to perform the computations at the initial condition. This means that there are less total possibilities (256) of inputs (if all qubits could be corrupted) for the same number of shots, so that potentially the "correct" solution may distinguish itself more significantly without needing to increase the computational load. The results are shown in Figs. 8-10. As shown in the figures, the number of shots in this case, combined with the size of the state measurements in binary and the specific implementation used based on that in,[24] is able to eliminate the difference between the inputs applied when the classical and no-noise quantum simulators are used compared to when the noisy quantum simulator is used. The noise model affects the simulation differently each time so that there is a potential for results to differ between runs of the simulation, but the overall trend suggests that tuning the number of shots may be a means for tuning the likelihood of "correctness" of the inputs applied to the process, and might be a desirable tuning mechanism for attempting to trade off control output accuracy (and thereby potentially closed-loop stability) with computation time for the control law execution.

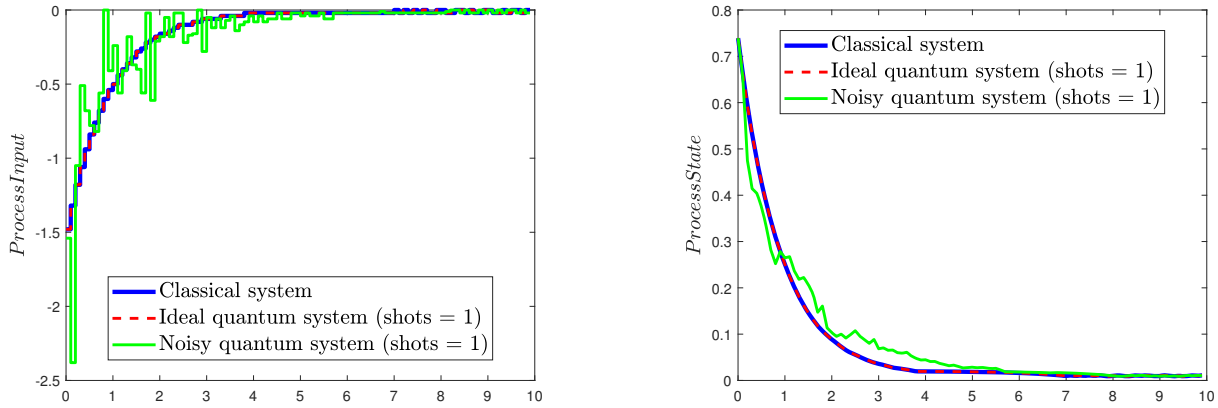

Figure 8: Comparison between the input (left) and state (right) trajectories when the control input is computed on a classical computer ("Classical system"), on a quantum simulator with 1 shot and no noise ("Ideal quantum system"), and on a quantum simulator with 1 shot and noise ("Noisy quantum system (1 shot)") for $x(0) = 0.74$. The x-axis is time units.

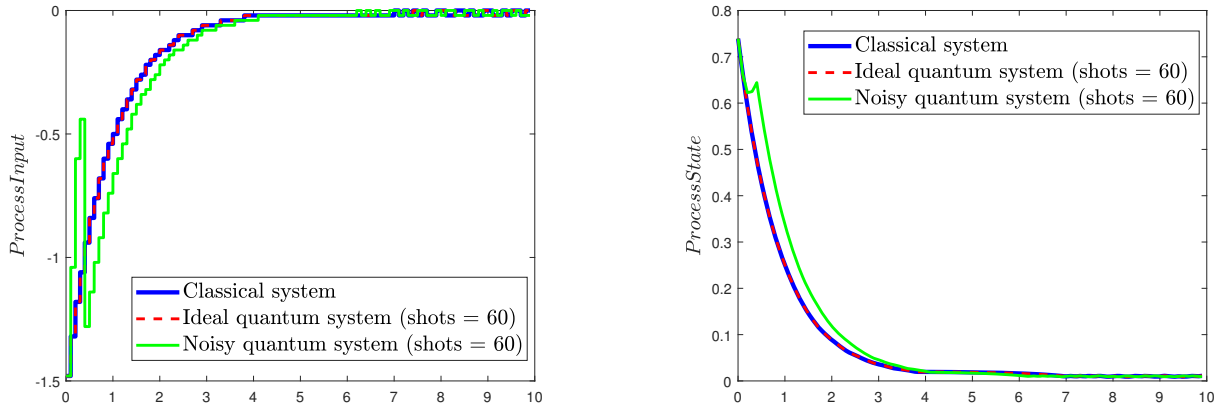

Figure 9: Comparison between the input (left) and state (right) trajectories when the control input is computed on a classical computer ("Classical system"), on a quantum simulator with 60 shots and no noise ("Ideal quantum system"), and on a quantum simulator with 60 shots and noise ("Noisy quantum system (60 shots)") for $x(0) = 0.74$. The x-axis is time units.

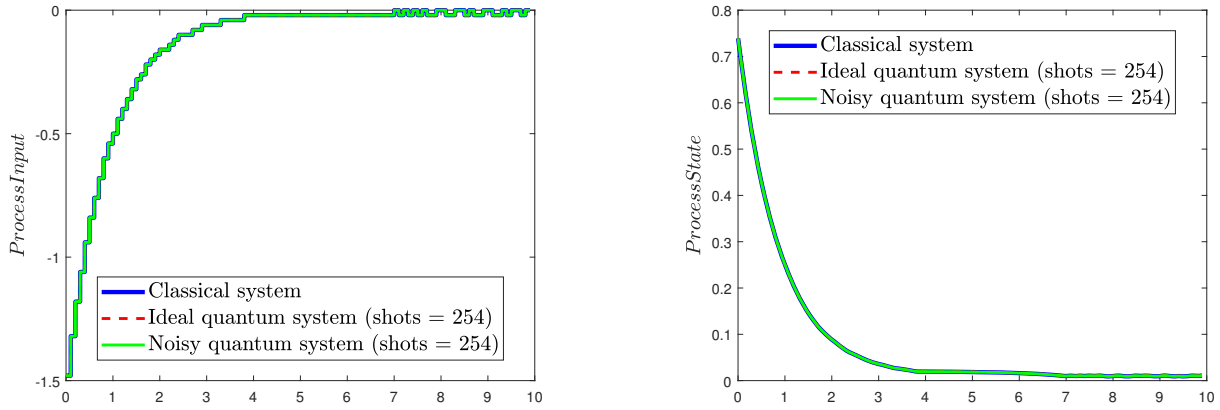

Figure 10: Comparison between the input (left) and right (right) trajectories when the control input is computed on a classical computer ("Classical system"), on a quantum simulator with 254 shots and no noise ("Ideal quantum system"), and on a quantum simulator with 254 shots and noise ("Noisy quantum system"), for $x(0) = 0.74$. The x-axis is time units.

In the process states of Fig. 8, an evaluation from a single shot of a quantum simulator does not provide the same result as a deterministic input computation as a consequence of the depolarizing noise model. We can attempt to tune the number of shots to see how many might be needed to improve the chances of the quantum and classical input computations matching. This threshold can be a certain percentage value greater than the number of possible outcomes based on the number of qubits that represent the initial quantum state. It is important to note that we do not want the number of shots to be very large, as it will affect the computational effort required to repeatedly perform the same computation.

Results from Quantum Computing-Implemented Control: Effects of Implementation of QFT-based Addition Algorithm

In the prior section, it was noted that a specific implementation of QFT-based addition had been used that, for control, shrinks the size of the qubit register required for performing the addition with time. This algorithm appears to achieve this in a fashion that impacts the computation time of the control laws and has implications for closed-loop stability with reduced computation time toward the origin. However, one of the curiosities of the simulations in the prior sections is that all simulations converged to a neighborhood of the origin, even when a single shot was used and the closed-loop state was initialized at 7.4. This raises the question of why that might be, and how

that should inform future exploration of closed-loop stability for control implemented on quantum computers.

Analyzing Fig. 8 provides some insight into a potential reason that the closed-loop state trajectory may converge to a neighborhood of the origin even with a single shot applied. For example, in Fig. 8, it can be seen that the closed-loop state does not monotonically decrease toward the origin. This suggests that at certain sampling periods an "incorrect" control action is applied that drives the closed-loop state away from the origin. However, subsequent control actions eventually begin to drive the state back toward the origin. This suggests that perhaps there is more than one control action that might be stabilizing (i.e., there is a potential that even if the noise causes a control action to be selected that is different from that of the classical controller, does not necessarily mean that it is unable to drive the closed-loop state closer toward the origin).

Hence, an analysis is performed assuming the worst case scenario where every bit could be corrupted, so that all control actions that could be computed from a given implementation of QFT-based addition are possible. The list of all possibilities is analyzed to determine which would drive the closed-loop state toward the origin from a given state. This can aid in analyzing the extent to which even "incorrect" inputs might drive the closed-loop state toward the origin, so that computing an input "incorrectly" may not be catastrophic from a safety standpoint. This would enable a controller implemented even on a noisy quantum computer to still stabilize the process as seen in Figs. 6-10.

In performing this analysis, we will look at two implementations of QFT-based addition: the first implementation of the QFT-based addition, as described above, considers a decrease in the size of the qubit register as the closed-loop state measurement approaches the origin or steady-state and is represented by fewer qubits in binary over time (Implementation 1). In the second scenario the size of the qubit register does not decrease as the closed-loop state approaches the origin even when the values are small by using 0's to maintain the number of bits in the binary representation of each state constant (Implementation 2). For both cases, we will estimate the proportion of all possible control actions for a given state (assuming all qubits could be corrupted and considering the size of the quantum register in the given QFT-based addition implementation) which would cause $\dot{x}$ to be

negative. The specific manner in which these calculations are done is as follows: the process state is initially evaluated from either $x(0) = 7.4$ or $x(0) = 0.74$, the initial state in integer form is first scaled up by a factor of 100 i.e., $100x$, following which the floor of the number is determined and finally converted to binary form. The size of the resulting integer in binary ('Qubit Register Size' in Table 1) is then recorded, and 1 is added to it to reflect the total number of bits which could be obtained in a computation of u according to $x + x$. If Implementation 1 is used, the qubit register size is taken to be the size just described. If Implementation 2 is used, the qubit register size is taken to be the size just described at the initial condition (8 qubits if the initial state is 0.74, and 11 qubits if it is 7.4). Then, the number of possible inputs ('No. Possible Inputs' in Table 1) is computed as 2^n, where n is the number of qubits in the register.

For an input $u = -2x$ to be stabilizing (i.e., to drive the process state, x, toward the steady-state) when the state measurement is positive, $\dot{x} = x + u$, $x + u$ must be negative. Therefore, considering the scaling, if u is greater than (ceil($100x$)) ('ceil($|100x|$)' in Table 1), its negative will be stabilizing. Thus, a rapid method for estimating the set of process inputs which could cause the process state trajectory, $\dot{x}$, to be negative at a given state is obtained by noting all of the integers between (ceil($100x$)) and 2^n ('Floor of No. Stabilizing Inputs' in Table 1). Once the total number of inputs for a given state measurement that could result in $\dot{x}$ being negative (or zero) is estimated in this fashion, the percentage of the total number of inputs that are stabilizing, among all of those which could be computed for the given qubit register size, is determined by dividing the number of stabilizing process inputs by the total number of possible process inputs that can be applied and multiplying by 100 ('% Stabilizing Inputs' in Table 1). If instead the state measurement is negative, it is necessary to cause $\dot{x}$ to be positive to drive the closed-loop state back toward the origin, and so the same analysis would hold, except for the absolute value of x. This analysis is depicted in Table 1. In this analysis, only the proportion of possible stabilizing inputs are estimated and not the likelihood of these control actions. To further clarify how the analysis is performed in Table 1, consider the case where the state of the system, $x(0)$, is 0.74 (implying that the scaled value is 74) and where 8-qubit registers are used to represent the process state to which QFT-based addition is applied. Hence, the total number of possible process inputs that can be applied is $2^8 = 256$. Since

the scaled process state is 74, the number of stabilizing process inputs is 182 (i.e. the floor of the difference between the maximum number of potential process inputs and the absolute value of the ceiling of the scaled process state).

Table 1: Estimate of the percentage of possible control actions that are stabilizing for the system represented using quantum devices of different qubit register sizes and initial states. The states listed for 254 shots with a shrinking register are from a different run of the simulation than is shown in Fig. 6 and thus may provide different results due to the noise model.

Case	Time (No. Sampling Periods Completed)	Actual State, x	ceil($\lvert 100x \rvert$)	Qubit Register Size	No. Possible Inputs	Floor of No. Stabilizing Inputs	% Stabilizing Inputs
1	0	0.74	74	8	256	182	71.09
2	17	0.1313	14	5	32	18	56.25
3	36	0.02487	3	3	8	5	62.50
4	79	0.01	1	2	4	3	75.00
5	0	0.74	74	8	256	182	71.09
6	17	0.1312	14	8	256	242	94.53
7	36	0.0249	3	8	256	253	98.83
8	79	0.01	1	8	256	255	99.61
9	0	7.4	740	11	2048	1308	63.87
10	17	1.53	153	9	512	359	70.12
11	36	0.02487	3	3	8	5	62.50
12	79	0.01	1	2	4	3	75.00
13	0	7.4	740	11	2048	1308	63.87
14	17	0.105	11	11	2048	2037	99.46
15	36	-0.416	42	11	2048	2006	97.95
16	79	0.145	15	11	2048	2033	99.27

Table 1 shows the manner in which the estimated percentage of stabilizing process inputs depends on the initial process state ($x(0)$) and the size of the qubit registers used in evaluating the process inputs (i.e., Implementation 1 or 2). It is notable from comparing, for example, Cases 1 and 2 with 9 and 10 in Table 1, that many possible process inputs are stabilizing regardless of the implementation used and also regardless of the initial value of the process state (i.e., the absolute difference between the initial process state and the steady-state/origin). In addition, the percentages of the inputs that are estimated to be stabilizing according to the method used does not appear to be a monotonic

function of the process state near the origin (e.g., Cases 9-11 in Table 1), though the method used has some approximation that has potential to influence this conclusion. In addition, the analysis assumes continuous control action implementation, but the results in the figures come from sample-and-hold implementation, so that this also indicates that the results in Table 1 have limitations for fully describing the situation simulated. However in the continuous implementation context (which can be considered to occur in the limit of an infinitely small sampling period), Table 1 indicates that by forcing the algorithm to consider a fixed number of potential process inputs that can be applied to the process, as the process moves toward the steady-state the percentage of stabilizing inputs significantly increases according to the estimation method used because it becomes less challenging for many values of u in the input range to exceed (ceil$(100x)$) for small x. As can be observed in Cases 5-8 and 13-16 in Table 1, the percentage of stabilizing inputs increases to close to 100% as the states near the steady-state of the process.

Table 1 provides an indication that there are a large number of control actions in the set of all possible inputs at a given state measurement, regardless of the implementation method and initial condition of the process state, that could drive the closed-loop state toward the origin at a given sampling time. Though this provides some insight into why the closed-loop state may be driven toward the origin even in the single-shot cases as in the input trajectory shown in Fig. 8, it does not provide insights into the frequency with which the various possible inputs are computed over multiple shots to determine the control input, u, at a certain instant in time. This forms another important part of understanding how often a stabilizing input might be selected for a given state measurement using the noisy computer simulation. To give an indication of frequency, the quantum simulator with noise and a depolarizing error parameter set to 0.05 with the initial condition $x(0) = 0.74$ was run with 254 shots and Implementation 1, and results for the frequency of occurrence of each possible input over 254 shots obtained for a particular run are depicted in Fig. 11 (frequencies can change between runs due to the random component of the noise simulation). These pie charts reflect the frequency with which different possible inputs appear among the 254 shots at different sampling periods throughout the simulation, where larger slices in the pie charts reflect that an input was computed more frequently. The largest sections of all three pie charts

correspond to the input selected at each sampling time, and also correspond to the process input that would be evaluated by a classical or a quantum computer not subject to any noise (since with 254 shots and an initial condition of 0.74, the results from the classical simulations and noisy computer were overlaid). The frequency of each occurrence is shown in these charts to be more than any other process input value and increases as the system moves closer to the steady-state. However, due to the rounding of the state measurement, when the process state is sufficiently close to the steady-state, the closed-loop state oscillates in a region near the origin.

To see the impact of the implementation strategy on the frequency with which an input appears, Fig. 12 shows the same simulation, but in the case that a fixed qubit register size is used throughout the simulation. The behavior of the frequencies in this case is starkly different from that observed with Implementation 1. For example, Fig. 12 reflects that even as the closed-loop state approaches the steady-state (Cases 6-8 in Table 1), none of the computed inputs "stands out" as much from the rest as in Fig. 11.

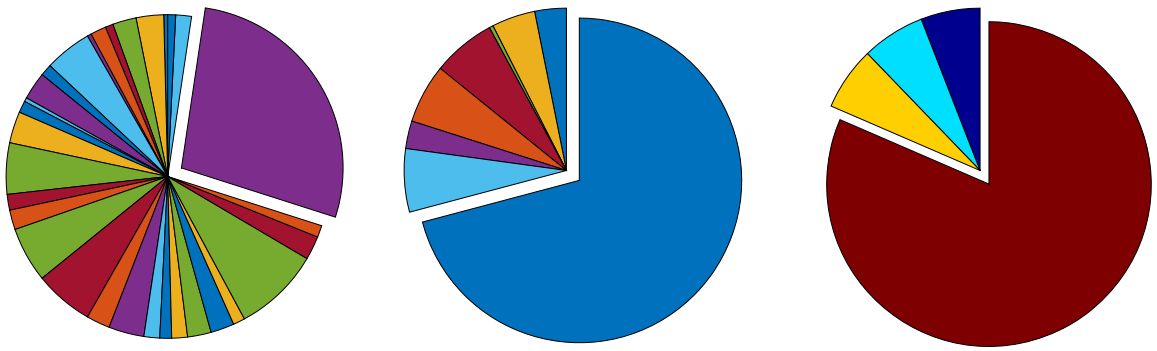

Figure 11: Distribution of process inputs evaluated at the 17^{th} (left), 36^{th} (middle), and 79^{th} (right) sampling periods over 254 shots for an initial state of $x(0) = 0.74$.

Results from Quantum Computing-Implemented Control: Impacts of the Error Parameter Value

In all of the above simulations, the value of the error parameter in the depolarizing noise model was set to 0.05. In this section, we analyze the impacts of increasing this value. The depolarizing error model was used as discussed previously, but here with different error parameters. Fig. 13 depicts the closed-loop state under Implementation 1 for the 32-qubit quasm_simulator with depolarizing error parameters ranging from 0.05 to 1. For all of the error parameters shown in this

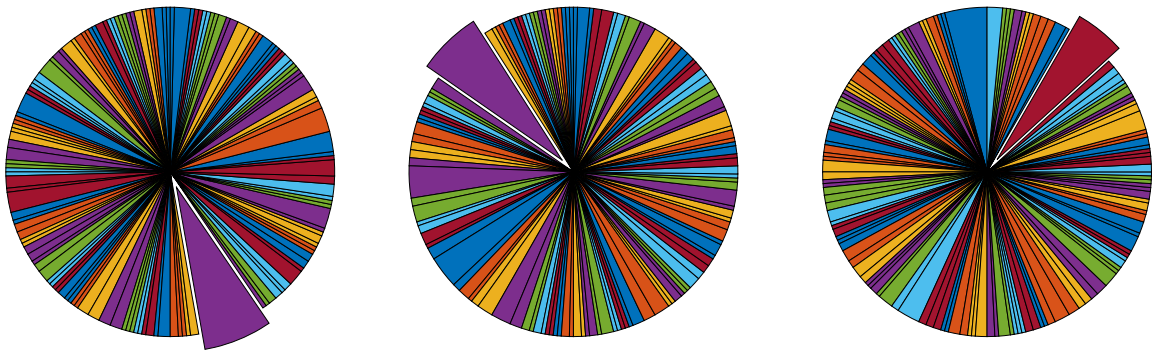

Figure 12: Distribution of process inputs evaluated at the 17^{th} (left), 36^{th} (middle), 79^{th} (right) sampling periods over 254 shots for an initial state of $x(0) = 0.74$ and where the number of qubits used to represent the process state, x, is constant. The slides of the pie that are emphasized show that there are cases where some inputs are more probable than others, but the pieces of the pie emphasized for being more probable are not significantly larger than some of the other pieces in some of the pies.

figure, the closed-loop state is eventually driven toward a neighborhood of the origin. Part of the reason for this is that though the frequency of selection of the deterministic input (as determined by the classical computer) might be decreased when the depolarizing error parameter is increased, the type of analysis from Table 1 continues to hold because it was independent of input frequencies, i.e. many inputs are stabilizing. This contributes to the fact that as the closed-loop state approaches the origin over time even if the inputs selected are not necessarily the ones that a classical computer would have selected, the combined effects of the rate of stabilization to destabilization, percentage of stabilizing control inputs and range of possible control inputs evaluated at any given time result in the stabilizing behavior observed in Fig. 13. The sign handling also contributes to the stabilization (i.e., that the sign is applied after the addition is performed and is not part of the quantum algorithm), and the sample-and-hold implementation can also play a role. This indicates that the dynamics, implementation strategy on a quantum computer, and physical parameters of the quantum computer can work together to demonstrate behaviors of interest in a system under quantum computing-implemented inputs, but that a further rigorous stability analysis should be performed in future work to better clarify the mechanism by which the various simulated controllers with noise assumed in the computing devices are stabilizing. Despite the need for this future work for further clarification, these results suggest some promise for noisy quantum computers to be effectively used

for next-generation control algorithms. Additionally, to validate that the process state can reach the origin when the error parameter is 1 (which is not captured in the timeframe of the left image in Fig. 13) the process was run for a longer duration (200 sampling periods) using the initial condition of 7.4, 254 shots, and an error parameter of 1 on the quasm_simulator. The results of this analysis are shown in the right image in Fig. 13, where the process state is seen to reach a neighborhood of the steady-state.

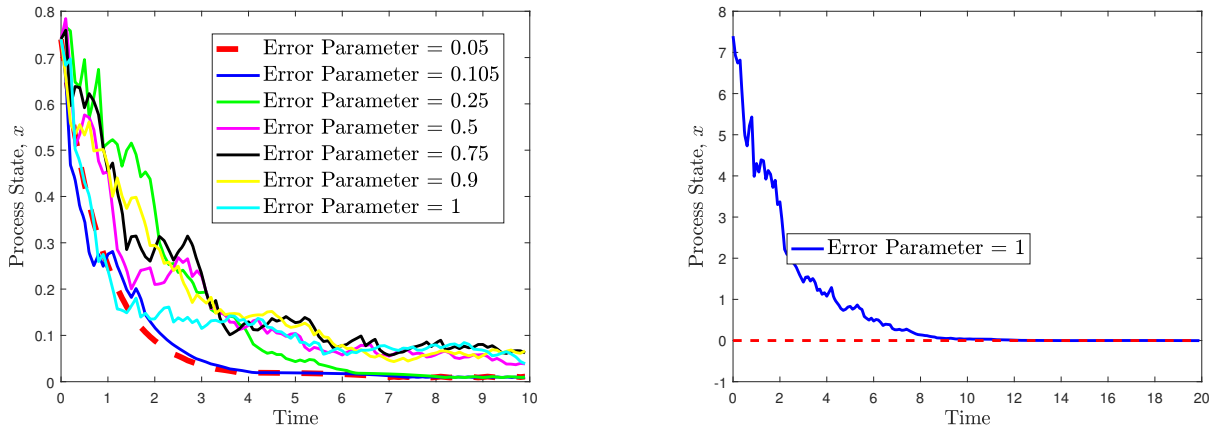

Figure 13: Comparison of process states for various error parameters (left). The figure on the right repeats the case with an error parameter of 1 on a longer timescale.

To show the difference in how inputs are selected when the error parameter is 1 compared to when it is 0.05, the two histograms in Fig. 14 compare the distribution of all possible occurrences of the process input when the process state is $x = 0.01695$ (close to the steady-state, such that there are only four possible values of the floor of the scaled value of this state measurement). These figures show a significant difference in the frequency with which each potential input appeared with 254 shots. In both cases, a control input of -0.02 was applied, taking the state to 0.01663, but Fig. 14 shows that this input, in the case where the error parameter is 1, was selected by only a small margin as the most frequent input (because the various possibilities were close to being equiprobable).

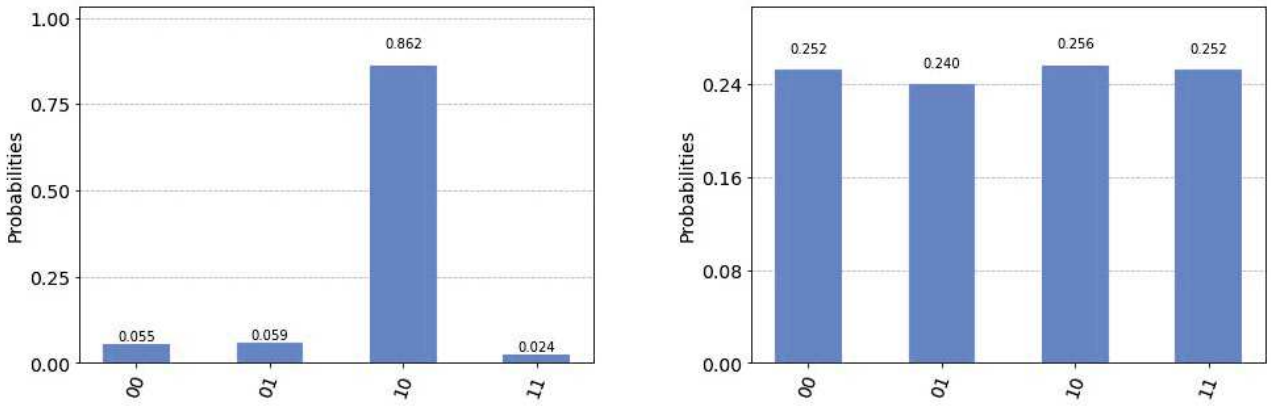

Figure 14: Histogram showing the frequency with which each potential value of $x+x$ was computed at a state of 0.01695 with 254 shots and an error parameter of 0.05 (left) and 1 (right).

4 Algorithmic Non-Determinism Considerations for Control on Quantum Computers

In this section, we provide the second of three studies relating control and quantum computing. While the previous section focuses on understanding how noise inherent to quantum computing hardware may affect control, this section is an initial examination of the consequences of non-determinism introduced by algorithms. It is important to understand these effects from a control-theoretic standpoint, as it must be ensured that a controller will keep a process in a safe region of operation.

To provide control theory relating non-determinism in an algorithm to control laws, we consider an advanced control algorithm known as Lyapunov-based model predictive control (LEMPC) that has strong control-theoretic properties.[18] To achieve our objective related to quantum computing, we consider the control actions computed by the LEMPC to be represented as a lookup table containing all of the possible measured states for a given sensor's precision and the corresponding process inputs (rounded also with finite precision). We will use an algorithm inspired by one of the flagship algorithms for learning quantum computing (Grover's algorithm) to search the table for an input when supplied a state. Because this algorithm is non-deterministic, even if the quantum computer is noiseless, the input provided to the process will be the desired value from the lookup

table with some probability and other inputs will also have a chance of being applied instead.

Since the lookup table needs to represent the states and inputs in binary form to enable discussion related to quantum computing, rounding needs to be applied to the states and the inputs. The standard theory of LEMPC[18] is developed such that the controller handles sufficiently small plant/model mismatch. However, to leave no doubt as to how non-determinism (as opposed to rounding) impacts control theory for the intersection of control and quantum computation, we develop explicit relationships between closed-loop stability results under LEMPC and the rounding of the states and inputs. This makes the impacts of the number of digits after the decimal that are kept when rounding the states and inputs for placement in the lookup table explicit.

The goal of this is in preparation for the discussion of the theoretical results for LEMPC with the non-deterministic lookup table search policy which will follow after the lookup table search policy on the quantum computer is described. Specifically, our goal in this section is to provide a preliminary look at what non-determinism in control input selection due to an algorithm on a quantum computer used for implementing control might entail for control theory. To do this, it is necessary that the control theory in the absence of the non-determinism due to the quantum computing algorithm be clearly delineated. This section therefore first provides the control theory for LEMPC implemented as a lookup table with rounded state measurements from the process, where the optimization calculations impose no limits on precision but the calculated inputs are rounded before being supplied to the process, when the lookup table is searched with a deterministic algorithm as would be expected on, for example, a classical computer.

Following this, to provide initial indications of how control theory might intersect with quantum computing algorithms that do not provide an answer with certainty even in the absence of noise, we develop a non-deterministic policy for searching the lookup table where the LEMPC measurements and control action pairs are placed, using a quantum computer. This algorithm is inspired by the Grover's search policy that can improve the number of function evaluations for searching an unordered list for the element that causes a function to evaluate to 1 when it is known that only a single element from the list causes the function to evaluate to 1 instead of 0;[1] however, because this is not directly applicable to searching a lookup table, we implement a series of controlled Grover's

search blocks in a quantum circuit. This algorithm then becomes no longer computationally nor size-efficient compared to using a classical computer for the search, but this is not problematic for the intended goal of this study, which is to look at how non-determinism inherent in the calculation of a control action would impact the associated control theory. We close the discussion with some insights into that control theory, which are facilitated by the strong control-theoretic properties of LEMPC when implemented on classical computers.

4.1 Preliminaries for LEMPC

We begin with preliminary information for formalizing the discussion in the following sections.

4.1.1 Notation

The notation $|x|$ represents the Euclidean norm and x^T represents the transpose of a vector x. The function $\alpha : [0, a) \to [0, \infty)$ is a class $\mathcal{K}$ function if $\alpha(0) = 0$, and if it is continuous and strictly increasing. The symbol Ω_ρ represents a level set of a scalar-valued function $V(x)$, where $\Omega_\rho := \{x \in \mathbb{R}^n : V(x) \leq \rho\}$. Set subtraction is denoted using '/' (i.e., $A/B := \{x \in \mathbb{R}^n : x \in A, x \notin B\}$). A sampling time is represented as $t_k := k\Delta, k = 0, 1, 2, ...$, where Δ is called the sampling period. Finally, $\mathbb{R}$, $\mathbb{I}$, $\mathbb{N}$ represent the set of all real, integer, and natural numbers, respectively.

4.1.2 Class of Systems and Lyapunov-based Controller

In this work, the following class of systems is considered:

$$\dot{x} = f(x(t), u(t), w(t)) \tag{4}$$

where $x \in X \subset \mathbb{R}^n$ is the state vector, $u \in U \subset \mathbb{R}^m$ is the input vector, and $w \in W \subset \mathbb{R}^z$ is the disturbance vector. f is a locally Lipschitz on $X \times U \times W$. W and U are defined as $W := \{w \in \mathbb{R}^z : |w| \leq \theta, \theta > 0\}$ and $U := \{u \in \mathbb{U}^m : |u| \leq u^{\max}\}$. In this work, we consider the nominal system ($w \equiv 0$) which is stabilizable through the application of an asymptotically stabilizing feedback control law $h(x)$, a sufficiently smooth Lyapunov function $V(x)$, and class $\mathcal{K}$

functions $\alpha_i(\cdot), i = 1, 2, 3, 4$, where, $\forall x \in D \subset \mathbb{R}^n$ (D is an open neighborhood of the origin):

$$\alpha_1(|x|) \leq V(x) \leq \alpha_2(|x|) \tag{5a}$$

$$\frac{\partial V(x)}{\partial x} f(x, h(x), 0) \leq -\alpha_3(|x|) \tag{5b}$$

$$\left| \frac{\partial V(x)}{\partial x} \right| \leq \alpha_4(|x|) \tag{5c}$$

$$h(x) \in U \tag{5d}$$

Ω_ρ is defined as the stability region of the nominal closed-loop system under the Lyapunov-based controller $h(x)$ and is chosen so that $x \in X, \forall x \in \Omega_\rho$. We require that $h(x)$ satisfies the following relation:

$$|h(x_1) - h(x_2)| \leq L_h |x_1 - x_2|, \forall x_1, x_2 \in \Omega_\rho \tag{6}$$

where L_h is a positive constant. Because V is a sufficiently smooth function and f is locally Lipschitz, we can say the following $\forall x_1, x_2 \in \Omega_\rho, u, u_1, u_2 \in U$, and $w \in W$:

$$|f(x_1, u_1, w) - f(x_2, u_2, 0)| \leq L_x |x_1 - x_2| + L_u |u_1 - u_2| + L_w |w| \tag{7a}$$

$$\left| \frac{\partial V(x_1)}{\partial x} f(x_1, u_1, w) - \frac{\partial V(x_2)}{\partial x} f(x_2, u_2, 0) \right| \leq L_x' |x_1 - x_2| + L_u' |u_1 - u_2| + L_w' |w| \tag{7b}$$

$$|f(x, u, w)| \leq M \tag{8}$$

where $L_x, L_x', L_u, L_u', L_w, L_w'$, and M are positive constants.

4.1.3 Lyapunov-based Economic Model Predictive Control

Lyapunov-based economic model predictive control was developed in[32] with the following formulation:

$$\min_{u(t) \in S(\Delta)} \quad \int_{t_k}^{t_{k+N}} L_e(\tilde{x}(\tau), u(\tau)) d\tau \tag{9a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = f(\tilde{x}(t), u(t), 0) \tag{9b}$$

$$\tilde{x}(t_k) = x(t_k) \tag{9c}$$

$$\tilde{x}(t) \in X, \, \forall \, t \in [t_k, t_{k+N}) \tag{9d}$$

$$u \in U, \; \forall \, t \in [t_k, t_{k+N}) \tag{9e}$$

$$V(\tilde{x}(t)) \leq \rho_e, \;\; \forall \, t \in [t_k, t_{k+N})$$

$$\text{if} \;\; \tilde{x}(t_k) \;\in\; \Omega_{\rho_e} \tag{9f}$$

$$\frac{\partial V(\tilde{x}(t_k))}{\partial x}(f(\tilde{x}(t_k), u(t_k), 0) \leq \frac{\partial V(\tilde{x}(t_k))}{\partial x} \; f(\tilde{x}(t_k), h(\tilde{x}(t_k)), 0) \tag{9g}$$

$$\text{if } x(t_k) \notin \Omega_\rho / \Omega_{\rho_e}$$

where $\tilde{x}$ represents the state prediction from Eq. 9b initialized from the state measurement (Eq. 9c). Eqs. 9d and 9e are state and input constraints, respectively. The LEMPC formulation involves optimization of a cost function (Eq. 9a), which is selected to represent some economic measure of the process. $u(t) \in S(\Delta)$ in Eq. 9a indicates that $u(t)$ is a piecewise-constant input vector consisting of N sections (where N is the length of the prediction horizon), each held for a single sampling period Δ (state measurements are available at every sampling time). The formulation is applied in a receding horizon fashion (i.e., only the first of the calculated inputs is applied to the process, and the optimization is performed again at each subsequent sampling time using updated state measurements from the process).

Additionally, the LEMPC formulation contains two constraints that activate depending on the location of the state within state space. The first (Eq. 9f) allows the controller to optimize the cost function freely as long as the state prediction remains within a region $\Omega_{\rho_e} \subset \Omega_\rho$. The second (Eq. 9g) activates when the state leaves this region and enters $\Omega_\rho / \Omega_{\rho_e}$ and is designed to drive the state back into Ω_{ρ_e}. The region Ω_{ρ_e} is defined based on an upper bound on disturbances (designated as θ) in a way where the state cannot leave Ω_ρ under the controller of Eq. 9.

4.1.4 Rounding Error

Rounding can be represented using a mapping from $\mathbb{R}$ to $\mathbb{Q}_{b,\gamma,\delta}$, which represents the space of rounded numbers, applied in the following way:[33]

$$\mathbb{Q}_{b,\gamma,\delta} := \{-b^{\gamma},\ -b^{\gamma}+b^{-\delta},\ -b^{\gamma}+2b^{-\delta},\ ...,\ 0,\ ...,\ b^{\gamma}-2b^{-\delta},\ b^{\gamma}-b^{-\delta}\} \tag{10}$$

where $b, \gamma, \delta \in \mathbb{N}, b \geq 1$ represent values that effectively represent the basis, magnitude, and resolution respectively. The values of b and γ must be selected such that the entire range of considered numbers (states or inputs in this case) falls within $[-b^{\gamma}, b^{\gamma})$. In addition, smaller values of δ will give a closer rounded approximation.

Given this approximation, the largest difference (e.g., obtained through rounding) between the actual values and the rounded values is $\frac{1}{2}b^{-\delta}$. If values of b_x, γ_x, and δ_x are selected to represent the states and b_u, γ_u, and δ_u are selected for the inputs, this applies bounds on the differences between the actual state $x(t_k)$ and rounded state measurement ($\bar{x}(t_k)$) and between a value of an input computed by a controller ($\bar{u}(t_k)$) and its rounded value ($\bar{\bar{u}}(t_k)$) as follows:

$$|x(t_k) - \bar{x}(t_k)| \leq \frac{1}{2} b_x^{-\delta_x} := \delta_1 \tag{11}$$

$$|\bar{\bar{u}}(t_k) - \bar{u}(t_k)| \leq \frac{1}{2} b_u^{-\delta_u} := \delta_2 \tag{12}$$

where δ_1 and δ_2 are defined for convenience and are used in the following sections.

4.2 Lyapunov-based Economic Model Predictive Control with Rounded States and Inputs

The objectives of this section are to: 1) develop a theory for LEMPC that explicitly accounts for rounding of states and inputs so that the closed-loop stability properties of an LEMPC implemented as a lookup table and searched deterministically are well-characterized; and 2) to provide an indication of how the number of digits after the decimal place impacts the closed-loop stability

results under LEMPC (which can be important for considering quantum computers due to the limited number of qubits available today). This theory is a preliminary result needed for discussing control-theoretic principles related to the non-deterministic search of the lookup table using a quantum computing algorithm following the description of that algorithm. This section first presents the formulation and implementation of the equivalent of a lookup table-based LEMPC (i.e., the control actions computed via the implementation strategy to be described would be the same as those that would be available from a lookup table implementation of LEMPC). Subsequently, the control theory is clarified.

4.2.1 Formulation of LEMPC with Rounded States and Inputs

The traditional LEMPC implementation strategy (Eq. 9) can maintain the closed-loop state in Ω_ρ when there is sufficiently small plant/model mismatch (θ), which means that it can account for rounding errors if they are sufficiently small. However, for the purposes of this discussion regarding the impacts of non-determinism in the control algorithm on closed-loop stability and feasibility of LEMPC, this loose notion that the plant/model mismatch introduced by rounding can be accounted for is not sufficient for clearly delineating how closed-loop stability results are related to the size of the computer available for storing the lookup table (i.e., how small δ_1 and δ_2 must be), nor is it sufficient for clarifying which aspects of the theory for the non-deterministic search policy are related to the rounding compared to being related to the non-deterministic policy itself. In this section, we seek to disentangle such considerations by providing a clear theory for the case with rounding only.

We seek the control theory for an LEMPC that would be implemented as a lookup table in finite precision. This can be obtained through the following series of steps: first, the sensor measures the actual process state at a sampling time (denoted by $x(t_k)$) and rounds it to $\bar{x}(t_k)$. This rounded state measurement is then used in solving the following LEMPC on a classical computer, where we assume no limits on precision in the computation of the control action $\bar{u}(t)$ using this controller for

simplicity of presentation:

$$\min_{\bar{u}(t)\in S(\Delta)} \quad \int_{t_k}^{t_{k+N}} L_e(\tilde{x}(\tau),\bar{u}(\tau))d\tau \tag{13a}$$

$$\text{s.t.} \quad \dot{\tilde{x}} = f(\tilde{x},\bar{u},0) \tag{13b}$$

$$\tilde{x}(t_k) = \bar{x}(t_k) \tag{13c}$$

$$\tilde{x} \in X,\, \forall\, t \in [t_k, t_{k+N}) \tag{13d}$$

$$\bar{u} \in U,\ \forall\, t \in [t_k, t_{k+N}) \tag{13e}$$

$$V(\tilde{x}(t)) \leq \rho_e,\ \ \forall\, t \in [t_k, t_{k+N})$$

$$\text{if}\ \ V(\bar{x}(t_k))\ \ \leq \rho_e \tag{13f}$$

$$\frac{\partial V(\bar{x}(t_k))}{\partial x}(f(\bar{x}(t_k),\bar{u}(t_k),0)$$

$$\leq \frac{\partial V(\bar{x}(t_k))}{\partial x}\, f(\bar{x}(t_k),h(\bar{x}(t_k)),0) \tag{13g}$$

$$\text{if}\ \ V(\bar{x}(t_k))\ \ > \rho_e$$

where $\tilde{x}(t_k)$ represents predictions of the state based on $\bar{x}(t_k)$ and Eq. 13b. After $\bar{u}(t_k)$ is obtained, because it is assumed that there were no impacts of finite precision in the LEMPC of Eq. 13, $\bar{u}(t_k)$ may need to be represented using infinite precision. Therefore, to enable it to be implemented in a lookup table using finite precision, it is rounded to $\bar{\bar{u}}(t_k)$ before being implemented on the process.

4.2.2 Implementation Strategy for LEMPC with Rounded States and Inputs

The following implementation strategy (see Fig. 15) summarizes the procedure described in the prior section for using LEMPC with rounded states and inputs:

1. At time t_k, the process state $x(t_k)$ is measured using a sensing device.

2. The rounded measured process state $\bar{x}(t_k)$ is obtained from $x(t_k)$ and provided to the LEMPC.

3. The LEMPC calculations are performed with infinite precision. If $x(t_k) \in \Omega_{\rho_e}$, go to Step 3a. Otherwise, go to Step 3b.

(a) The controller solves an optimization problem using $\bar{x}(t_k)$ to determine the control action $\bar{u}(t_k)$ that would maximize a cost function while maintaining state predictions in Ω_{ρ_e}.

(b) The controller calculates the control action $\bar{u}(t_k)$ that would drive the closed-loop state towards the origin.

4. The calculated control action $\bar{u}(t_k)$ is rounded and designated as $\bar{\bar{u}}(t_k)$ and then provided to the process.

5. $t_k \leftarrow t_{k+1}$. Go to Step 1.

This implementation strategy can be used to develop every rounded state measurement and rounded control action pair in a lookup table and is equivalent then to searching this lookup table with a deterministic algorithm.

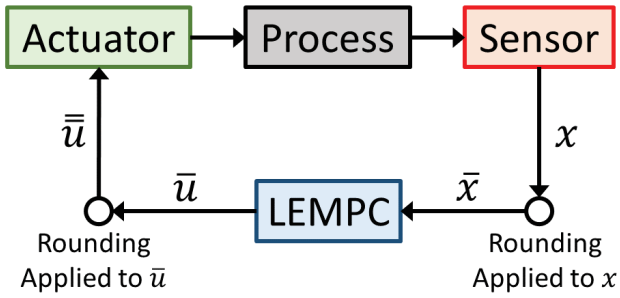

Figure 15: Implementation strategy for LEMPC with rounded states and inputs.

4.2.3 Stability and Feasibility Analysis for LEMPC with Rounded States and Inputs

In this section, we demonstrate that the closed-loop state of the system of Eq. 4 is maintained within Ω_ρ at all times under the implementation strategy in Section 4.2.2. We start by stating two propositions. The purpose of Proposition 1 is to give an upper bound on the difference between the state x_a of the actual process (under the rounded input $\bar{\bar{u}}$ and in the presence of plant/model mismatch represented by $w(t)$, but initialized from the non-rounded state measurement $x(t_0)$) and

the state x_b of the nominal system in Eq. 13b under the non-rounded input $\bar{u}$ computed using the rounded state measurement $\bar{x}(t_0)$.

Proposition 1. *Consider the systems:*

$$\dot{x}_a = f(x_a(t), \bar{\bar{u}}(t_0), w(t)) \tag{14a}$$

$$\dot{x}_b = f(x_b(t), \bar{u}(t_0), 0) \tag{14b}$$

with initial states of $|x_a(t_0) - x_b(t_0)| \leq \delta_1$, $x_a(t_0), x_b(t_0) \in \Omega_\rho$, *where* $|\bar{\bar{u}}(t_0) - \bar{u}(t_0)| \leq \delta_2$. *There exists a class* $\mathcal{K}$ *function* $f_W(\cdot)$ *that satisfies the following equations* $\forall x_a, x_b \in \Omega_\rho$ *and* $\forall w \in W$*:*

$$|x_a(t) - x_b(t)| \leq f_W(t - t_0, \delta_1, \delta_2) \tag{15}$$

where

$$f_W(\tau, \delta_1, \delta_2) := \left(\delta_1 + \frac{L_w\theta + L_u\delta_2}{L_x}\right) e^{L_x\tau} - \frac{L_w\theta + L_u\delta_2}{L_x} \tag{16}$$

Proof 1. *To prove Proposition 1, we begin by integrating Eqs. 14a and 14b. The difference between the resulting equations is:*

$$x_a(t) - x_b(t) = x_a(t_0) - x_b(t_0) + \int_{t_0}^{t} [f(x_a(s), \bar{\bar{u}}(t_0), w(s)) - f(x_b(s), \bar{u}(t_0), 0)]ds \tag{17}$$

Applying the triangle inequality and Eq. 11 gives:

$$|x_a(t) - x_b(t)| \leq \delta_1 + \int_{t_0}^{t} |f(x_a(s), \bar{\bar{u}}(t_0), w(s)) - f(x_b(s), \bar{u}(t_0), 0)|ds \tag{18}$$

Within the integral, by adding and subtracting $f(x_b(s), \bar{\bar{u}}(t_0), 0)$ *and using Eq. 7a, we obtain:*

$$|x_a(t) - x_b(t)| \leq \delta_1 + \int_{t_0}^{t} L_x|x_a(s) - x_b(s)| + \int_{t_0}^{t} L_w|w(s)|ds + \int_{t_0}^{t} L_u|\bar{\bar{u}}(t_k) - \bar{u}(t_k)|ds \tag{19}$$

Applying Eq. 12 and $|w| \leq \theta$ and integrating the corresponding terms, gives:

$$|x_a(t) - x_b(t)| \leq \delta_1 + \int_{t_0}^{t} L_x |x_a(s) - x_b(s)| ds + L_w \theta (t - t_0) + L_u \delta_2 (t - t_0) \tag{20}$$

Applying the Gronwall-Bellman inequality[34] gives:

$$|x_a(t) - x_b(t)| \leq \delta_1 + (L_w \theta + L_u \delta_2) t + \int_{t_0}^{t} [\delta_1 + (L_w \theta + L_u \delta_2) s](L_x) e^{\int_s^t L_x d\tau} ds \tag{21}$$

Simplifying this gives Eq. 16.

Proposition 2 is gives a bound on the difference between the Lyapunov function of two different states ($V(x)$ and $V(\hat{x})$), given that the states are in Ω_ρ.

Proposition 2. [32] *For the Lyapunov function $V(\cdot)$ of Eq. 4, we can find a quadratic function $f_V(\cdot)$ such that:*

$$V(x) \leq V(\hat{x}) + f_V(|x - \hat{x}|) \tag{22}$$

$\forall x, \hat{x} \in \Omega_\rho$ with

$$f_V(s) = \alpha_4(\alpha_2^{-1}(\rho)) s + M_v s^2 \tag{23}$$

where M_v is a positive constant.

Theorem 1 is formulated to provide conditions for closed-loop stability of the system given in Eq. 4 under the implementation strategy in Section 4.2.2.

Theorem 1. *Consider the system in Eq. 4 under the LEMPC implementation strategy in Section 4.2.2 based on a controller $h(x)$ that satisfies the conditions in Eq. 5, and where $\delta_1 > 0$ and $\delta_2 > 0$ are defined by Eqs. 11-12. Let $\rho_s < \rho_{\min} < \rho_e < \rho$, $\rho' < \rho'' < \rho$, $N \geq 1$, $\Delta > 0$, $\epsilon_w > 0$, $\rho_{samp2} < \rho (\rho_{samp2} := \max\{V(x(t)) : \bar{x}(t_k) \in \Omega_{\rho_e},\ t \in [t_k, t_{k+1}),\ u \in U,\ w \in W\})$, and $\epsilon'_x > 0$ satisfy*

$$\rho_e + f_V(f_W(\Delta, \delta_1, \delta_2)) \leq \rho' \tag{24}$$

$$\max\{\rho', \rho_e\} + f_V(\delta_1) \leq \rho'' \tag{25}$$

$$L'_u\delta_2 + 3L'_xM\Delta + L'_x\delta_1 + L'_w\theta - \alpha_3(\alpha_2^{-1}(\rho_s)) \leq -\epsilon_w/\Delta \tag{26}$$

$$L'_xM\Delta - \alpha_3(\alpha_2^{-1}(\rho_s)) \leq -\epsilon'_x/\Delta \tag{27}$$

$$\rho_{samp2} + \max\{f_V(\delta_1), f_V(f_W(\Delta, \delta_1, \delta_2))\} \leq \rho \tag{28}$$

If $x(t_0) \in \Omega_{\rho_e}$, $\bar{x}(t_0) \in \Omega_{\rho_e}$, *and* $N \geq 1$ *where*

$$\rho_{\min} = \max\{V(x(t)) : x(t_k) \in \Omega_{\rho_s},\ t \in [t_k, t_{k+1}),\ u \in U,\ w \in W\} \tag{29}$$

then $x(t) \in \Omega_\rho$ *for* $t \geq 0$ *and* $\bar{x}(t_k) \in \Omega_\rho$ *at every sampling time* $t_k \geq 0$.

Proof. This proof consists of two parts. In Part 1, we demonstrate the recursive feasibility of a solution to Eq. 13. In Part 2, we prove that $x(t)$ and $\bar{x}(t)$ are maintained within Ω_ρ for $t, t_k \geq 0$ under the implementation outlined in Section 4.2.2.

Part 1. To demonstrate feasibility at t_0, we show that $h(\cdot)$ implemented in sample-and-hold satisfies all constraints in the LEMPC formulation (Eq. 13). Since $h(\cdot)$ is defined with the condition that $h(\cdot) \in U$ (Eq. 5), Eq. 13e holds. Eq. 13g trivially holds under $h(\tilde{x}(t_p))$, $t \in [t_p, t_{p+1})$, $p = k, \ldots, k+N-1$. Eq. 13f is feasible under $h(\cdot)$ in sample-and-hold if: (1) $\tilde{x}(t_p) \in \Omega_{\rho_s}$ and (2) $\tilde{x}(t_p) \in \Omega_\rho/\Omega_{\rho_s}$. To demonstrate (1), if $\tilde{x}(t_p) \in \Omega_{\rho_s}$, then by the definition of ρ_{min} (Eq. 29), $\tilde{x}(t) \in \Omega_{\rho_{min}} \subset \Omega_{\rho_e}, t \in [t_p, t_{p+1})$. To demonstrate (2), the trajectory of the Lyapunov function along the predicted state trajectory $\tilde{x}(t)$ under $h(\tilde{x}(t_p))$ is:

$$\begin{aligned}\frac{\partial V(\tilde{x}(t))}{\partial x} f(\tilde{x}(t), h(\tilde{x}(t_p), 0) =& \frac{\partial V(\tilde{x}(t))}{\partial x} f(\tilde{x}(t), h(\tilde{x}(t_p)), 0) + \frac{\partial V(\tilde{x}(t_p))}{\partial x} f(\tilde{x}(t_p), h(\tilde{x}(t_p)), 0) \\ &- \frac{\partial V(\tilde{x}(t_p))}{\partial x} f(\tilde{x}(t_p), h(\tilde{x}(t_p)), 0)\end{aligned} \tag{30}$$

for $t \in [t_p, t_{p+1})$, where the equality follows from adding and subtracting $\frac{\partial V(\tilde{x}(t_p))}{\partial x} f(\tilde{x}(t_p), h(\tilde{x}(t_p)), 0)$ to/from $\frac{\partial V(\tilde{x}(t))}{\partial x} f(\tilde{x}(t), h(\tilde{x}(t_p)), 0)$. Applying Eqs. 5b and 7b, we obtain:

$$\frac{\partial V(\tilde{x}(t))}{\partial x} f(\tilde{x}(t), h(\tilde{x}(t_p)), 0) \leq L_x^{'} |\tilde{x}(t) - \tilde{x}(t_p)| - \alpha_3(|\tilde{x}(t_p)|) \tag{31}$$

Next, Eqs. 5a and 8 can be applied to give:

$$\frac{\partial V(\tilde{x}(t))}{\partial x} f(\tilde{x}(t), h(\tilde{x}(t_p)), 0) \leq L_x^{'} M\Delta - \alpha_3(\alpha_2^{-1}(\rho_s)) \tag{32}$$

If Eq. 27 holds and if $\tilde{x}(t_p) \in \Omega_\rho/\Omega_{\rho_s}$, then $\frac{\partial V(\tilde{x}(t))}{\partial x} f(\tilde{x}(t), h(\tilde{x}(t_p)), 0) \leq -\epsilon_x'/\Delta$ for $t \in [t_p, t_{p+1})$. This implies that $V(\tilde{x}(t)) \leq V(\tilde{x}(t_p))$ for all $t \in [t_p, t_{p+1})$ under $h(\tilde{x}(t_p))$ so that $\tilde{x}(t) \in \Omega_{\rho_e}$ for $t \in [t_p, t_{p+1})$ if $\tilde{x}(t_p) \in \Omega_{\rho_e}/\Omega_{\rho_s}$. Therefore, if $\tilde{x}(t_p) \in \Omega_{\rho_e}/\Omega_{\rho_s}$, Eq. 13f is feasible using $h(\tilde{x}(t_p))$, $t \in [t_p, t_{p+1})$, $p = k, \ldots, k+N-1$. Furthermore, since $X \subset \Omega_\rho$ and $h(\tilde{x}(t_p))$, $t \in [t_p, t_{p+1})$, $p = k, \ldots, k+N-1$ maintains $\tilde{x}(t)$ in Ω_ρ by the proof just provided, Eq. 13d is also satisfied such that the LEMPC of Eq. 13 is feasible at every sampling time if $\bar{x}(t_k) \in \Omega_\rho$, which will be demonstrated in Part 2.

Part 2. To demonstrate that $x(t) \in \Omega_\rho$ and $\bar{x}(t_k) \in \Omega_\rho$ for $t, t_k \geq 0$ when $x(t_0)$ and $\bar{x}(t_0) \in \Omega_{\rho_e}$, it is necessary to consider the potential mismatch between $x(t_k)$ and $\bar{x}(t_k)$ in four cases as follows: Case 1) $x(t_k) \in \Omega_{\rho_e}$ and $\bar{x}(t_k) \in \Omega_{\rho_e}$; Case 2) $x(t_k) \in \Omega_{\rho_e}$ and $\bar{x}(t_k) \in \Omega_\rho/\Omega_{\rho_e}$; Case 3) $x(t_k) \in \Omega_\rho/\Omega_{\rho_e}$ and $\bar{x}(t_k) \in \Omega_{\rho_e}$; and Case 4) $x(t_k) \in \Omega_\rho/\Omega_{\rho_e}$ and $\bar{x}(t_k) \in \Omega_\rho/\Omega_{\rho_e}$.

Case 1: When $\bar{x}(t_k) \in \Omega_{\rho_e}$, Eq. 13f is activated. From Proposition 1 (Eq. 15), the following bound can be placed on the norm of the difference between the predicted state in the LEMPC and the actual state:

$$|\tilde{x}(t) - x(t)| \leq f_W(\Delta, \delta_1, \delta_2) \tag{33}$$

for $t \in [t_k, t_{k+1})$. Proposition 2 (Eq. 22) can be used to bound $V(x(t))$ and $V(\tilde{x}(t))$ for $t \in [t_k, t_{k+1})$ as follows:

$$V(x(t)) \leq V(\tilde{x}(t)) + f_V(|\tilde{x}(t) - x(t)|) \tag{34}$$

Using Eqs. 11 and 12 and the fact that $V(\tilde{x}(t)) \leq \rho_e$, Eqs. 33 and 34 can be combined to give:

$$V(x(t)) \leq \rho_e + f_V(f_W(\Delta, \delta_1, \delta_2)) \tag{35}$$

If Eq. 24 holds, Eq. 35 gives that $V(x(t)) \leq \rho' < \rho$, $\forall t \in [t_k, t_{k+1})$. To demonstrate that this implies

that the next rounded state measurement is also in Ω_ρ, Proposition 2 (Eq. 22) can be used to write:

$$V(\bar{x}(t_{k+1})) \leq V(x(t_{k+1})) + f_V(\delta_1) \tag{36}$$

Combining Eqs. 35 and 36 gives:

$$V(\bar{x}(t_{k+1})) \leq \rho_e + f_V(f_W(\Delta, \delta_1, \delta_2)) + f_V(\delta_1) \leq \rho' + f_V(\delta_1) \tag{37}$$

from Eq. 24. When Eq. 25 holds, Eq. 37 gives that $V(\bar{x}(t_{k+1})) \leq \rho'' < \rho$. This implies that $x(t) \in \Omega_\rho$ for $t \in [t_k, t_{k+1})$ and $\bar{x}(t_{k+1}) \in \Omega_\rho$ if $x(t_k) \in \Omega_{\rho_e}$ and $\bar{x}(t_k) \in \Omega_{\rho_e}$.

Case 2: When $\bar{x}(t_k) \in \Omega_\rho / \Omega_{\rho_e}$, Eq. 13g is activated. When Eq. 13g is activated and $\bar{x}(t_k) \in \Omega_\rho / \Omega_{\rho_e}$, Eq. 5b gives:

$$\frac{\partial V(\bar{x}(t_k))}{\partial x} f(\bar{x}(t_k), \bar{u}(t_k), 0) \leq \frac{\partial V(\bar{x}(t_k))}{\partial x} f(\bar{x}(t_k), h(\bar{x}(t_k)), 0) \leq -\alpha_3(|\bar{x}(t_k)|) \tag{38}$$

The trajectory of the Lyapunov function along the actual closed-loop state trajectory under $\bar{u}(t_k)$ (the input before rounding) in the presence of disturbances (denoted by x_1) would be:

$$\begin{aligned} \frac{\partial V(x_1(t))}{\partial x} f(x_1(t), \bar{u}(t_k), w(t)) =& \frac{\partial V(x_1(t))}{\partial x} f(x_1(t), \bar{u}(t_k), w(t)) - \frac{\partial V(\bar{x}(t_k))}{\partial x} f(\bar{x}(t_k), \bar{u}(t_k), 0) \\ &+ \frac{\partial V(\bar{x}(t_k))}{\partial x} f(\bar{x}(t_k), \bar{u}(t_k), 0) \end{aligned} \tag{39}$$

where the equality follows from adding and subtracting $\frac{\partial V(\bar{x}(t_k))}{\partial x} f(\bar{x}(t_k), \bar{u}(t_k), 0)$ to/from $\frac{\partial V(x_1(t))}{\partial x} f(x_1(t), \bar{u}(t_k), w(t))$. Combining this with Eqs. 38, 5a, 7b, 8, and 11 and the bound on w and rearranging yields:

$$\frac{\partial V(x_1(t))}{\partial x} f(x_1(t), \bar{u}(t_k), w(t)) \leq L_x' M \Delta + L_x' \delta_1 + L_w' \theta - \alpha_3(\alpha_2^{-1}(\rho_s)) \tag{40}$$

if x_1 remains in Ω_ρ. Finally, the trajectory of the Lyapunov function along the closed-loop state

trajectory under $\bar{\bar{u}}(t_k)$ in the presence of disturbances is:

$$\begin{aligned}\frac{\partial V(x(t))}{\partial x} f(x(t), \bar{\bar{u}}(t_k), w(t)) =& \frac{\partial V(x(t))}{\partial x} f(x(t), \bar{\bar{u}}(t_k), w(t)) - \frac{\partial V(x_1(t))}{\partial x} f(x_1(t), \bar{u}(t_k), w(t)) \\ &+ \frac{\partial V(x_1(t))}{\partial x} f(x_1(t), \bar{u}(t_k), w(t))\end{aligned} \tag{41}$$

for $t \in [t_k, t_{k+1})$, where the equality comes from adding and subtracting $\frac{\partial V(x_1(t))}{\partial x} f(x_1(t), \bar{u}(t_k), w(t))$ to/from $\frac{\partial V(x(t))}{\partial x} f(x(t), \bar{\bar{u}}(t_k), w(t))$. Applying Eqs. 7b and 40:

$$\frac{\partial V(x(t))}{\partial x} f(x(t), \bar{\bar{u}}(t_k), w(t)) \leq L'_u \delta_2 + 3L'_x M\Delta + L'_x \delta_1 + L'_w \theta - \alpha_3(\alpha_2^{-1}(\rho_s)) \tag{42}$$

for $t \in [t_k, t_{k+1})$ and $x_1 \in \Omega_\rho$. If Eq. 26 holds, then $\frac{\partial V(x(t))}{\partial x} f(x(t), \bar{\bar{u}}(t_k), w(t)) \leq -\epsilon_w/\Delta$. This indicates that $V(x(t)) \leq V(x(t_k))$ for $t \in [t_k, t_{k+1})$ when $\bar{x}(t_k) \in \Omega_\rho/\Omega_{\rho_s}$ and $x(t_k) \in \Omega_{\rho_e}$. To demonstrate that $\bar{x}(t_{k+1}) \in \Omega_\rho$ in this case, Proposition 2 gives:

$$V(\bar{x}(t_{k+1})) \leq V(x(t_{k+1})) + f_V(\delta_1) \leq \rho_e + f_V(\delta_1) \tag{43}$$

If Eq. 25 holds, then $\bar{x}(t_{k+1}) \in \Omega_\rho$. For x_1 to have been in Ω_ρ, $V(x_1(t)) \leq V(x(t)) + f_V(|x_1(t) - x(t)|) \leq \rho_e + f_V(f_W(\Delta, \delta_1, \delta_2))$ (since $|x_1(t) - x(t)|$ will also meet Eq. 15, $x(t_k) \in \Omega_{\rho_e}$, and $V(x(t)) \leq V(x(t_k))$ for $t \in [t_k, t_{k+1})$); Eq. 24 then guarantees that $x_1(t) \in \Omega_\rho$, $\in [t_k, t_{k+1})$.

If instead $x(t_k) \in \Omega_{\rho_s}$, then Eq. 29 demonstrates that $x(t) \in \Omega_{\rho_{\min}} \subset \Omega_{\rho_e} \subset \Omega_\rho$ for $t \in [t_k, t_{k+1})$. Then if Eq. 43 and Eq. 29 hold, $\bar{x}(t_{k+1}) \in \Omega_\rho$ also. This demonstrates that $\bar{x}(t_{k+1}) \in \Omega_\rho$ and $x(t) \in \Omega_\rho$, $t \in [t_k, t_{k+1})$, if $x(t_k) \in \Omega_{\rho_e}$ and $\bar{x}(t_k) \in \Omega_\rho/\Omega_{\rho_e}$.

Case 3: If $x(t_k) \in \Omega_\rho/\Omega_{\rho_e}$ and $\bar{x}(t_k) \in \Omega_{\rho_e}$, the same proof as for Case 1 holds and $x(t_{k+1}) \in \Omega_\rho$ and $\bar{x}(t_{k+1}) \in \Omega_\rho$.

Case 4: If $x(t_k) \in \Omega_\rho/\Omega_{\rho_e}$ and $\bar{x}(t_k) \in \Omega_\rho/\Omega_{\rho_e}$, the same proof as for Case 2 holds through Eq. 42, which indicates that $x(t_{k+1}) \in \Omega_\rho$ except that $V(x(t_k)) \in \Omega_{\rho_{samp1}}$ (because either $V(x(t_{k-1})) \in \Omega_{\rho_e}$, with $V(x(t_{k-1})) \in \Omega_{\rho_{samp2}}/\Omega_{\rho_e}$ or $V(x(t_k)) \leq V(x(t_{k-1}))$, so $V(x_1(t)) \leq V(x(t)) + f_V(f_W(\Delta, \delta_1, \delta_2)) \leq \rho_{samp2} + f_V(f_W(\Delta, \delta_1, \delta_2)) \leq \rho$ from Eq. 28). Since $V(\bar{x}(t_k)) \leq V(x(t_k)) + f_V(\delta_1) \leq \rho_{samp2} + f_V(\delta_1) \leq \rho$ when Eq. 28 holds, and $V(x(t_{k+1})) \leq V(x(t_k))$, then

$V(\bar{x}(t_{k+1})) \leq V(x(t_{k+1})) + f_V(\delta_1) \leq V(x(t_k)) + f_V(\delta_1) < \rho$ also, which indicates that $\bar{x}(t_{k+1}) \in \Omega_\rho$.

At every sampling time, one of the four cases is true. Because each of the four cases kept $x(t) \in \Omega_\rho$, $t \in [t_k, t_{k+1})$, and $\bar{x}(t_{k+1}) \in \Omega_\rho$, applying this recursively shows that this is true for all $t \geq 0$. □

Remark 2. *The role of ρ' and ρ'' in Theorem 1 is to account for the fact that the rounded value of $x(t)$ could be in a larger level set of V than the actual value of $x(t)$.*

Remark 3. *Eqs. 24-26 place requirements on δ_1 and δ_2, in particular the requirement that they must be sufficiently small, which implies that more digits after the decimal are needed. This would imply that more bits/qubits would be needed to represent a number.*

4.3 Non-deterministic Quantum Computing Algorithm: Grover's Algorithm Inspired Circuit

In this section, a control algorithm is encoded in a quantum circuit to demonstrate how a quantum-encoded control algorithm can yield results with some probability. The circuit consists of a series of Grover algorithm gates controlled by a set of n qubits as shown in Fig. 16. The output of one Grover's algorithm block is fed into the next Grover block. The process state measurement will be supplied as the control qubits and the corresponding control action is outputted from the final Grover's algorithm block in the circuit.

This circuit assumes that a complete lookup table has been found *a priori*, which consists of all possible states and control actions. Each input and state is an n length binary number. During operation, the measured process state would be supplied (as binary numbers) to the circuit as the control bits. Each Grover's algorithm block is designed to pair with one (and only one) of the process states in the table. If a Grover's algorithm block is activated by the set of control qubits, the output of the Grover's algorithm block will be in a quantum state that will most likely collapse to the desired process input. All of the other Grover's algorithm blocks will not be activated by a set of control qubits, meaning that the quantum state leaving the gate will be unchanged from the quantum state entering the block. This means that the state leaving the last Grover's algorithm

block can then be measured, and it will most likely collapse to the binary number containing the desired control actions. The correct control action is found by extracting the middle n qubits.

Overall, this algorithm is not efficient or practical for several reasons. Given that most processes will have many possible states, creating a full lookup table of states and inputs would be cumbersome. Additionally, this means that the circuit will become large because its size is tied to the number states and control actions. The resulting controller, given its size and complexity, would likely be slower than traditional search algorithms. Despite these flaws, this control structure is still useful for studying and understanding how non-deterministic inputs can be generated by a quantum computing algorithm will affect controller performance.

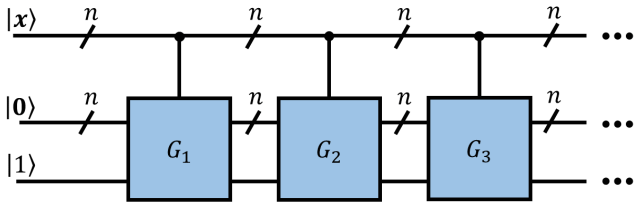

Figure 16: Circuit of a series of quantum gate blocks that represent Grover's algorithm controlled by a set of n qubits.

4.3.1 Grover's Search Algorithm

In general, Grover's search algorithm[1] is a quantum computing algorithm that searches the space consisting of $n-$length binary digits denoted as $\boldsymbol{x} = \{x_1, x_2, ..., x_i, ...x_n\}$ for a unique string x^*. The function f is a black-box function that is defined as:

$$f(x_i) = \begin{cases} 1 & x_i = x^* \\ 0 & x_i \neq x^* \end{cases}$$

The algorithm outputs the location of x^* within $\boldsymbol{x}$ that yields $f(x) = 1$. By letting each element of an unstructured database be associated with an element in $\boldsymbol{x}$, it is possible to use this algorithm

to search for the location of an element within the database.

The quantum circuit representing Grover's algorithm is shown on the left in Fig. 17 and contains a set of n top qubits, each initialized in state $|0\rangle$, and a bottom control bit initialized in state $|1\rangle$. After first placing the qubits in a superposition using Hadamard gates, unitary gates known as "phase inversions" (denoted as U_f) and "inversion about the mean" (denoted as $-I + 2A$) are performed repeatedly in succession to the result in a state that, once measured, has a high probability of collapsing to the location of the bit string that is being searched for.

The U_f phase inversion unitary operation that is performed on the qubits can be represented as a modified identity matrix, where the 2×2 matrix formed by the two rows and columns associated with the bit string being searched for are flipped (i.e., the 1's along the diagonal are displaced to be located above and below the diagonal). The $-I + 2A$ inversion about the mean unitary operation is constructed from the identity matrix I and a matrix A, which contains all elements with values of $(2^n)^{-1}$. Both of these matrices can be applied up to $\sqrt{2^n}$ times to result in a system state that, once measured, has the highest probability of collapsing to reveal the location of which of the 2^n binary strings is the one for which the function f evaluates to 1.

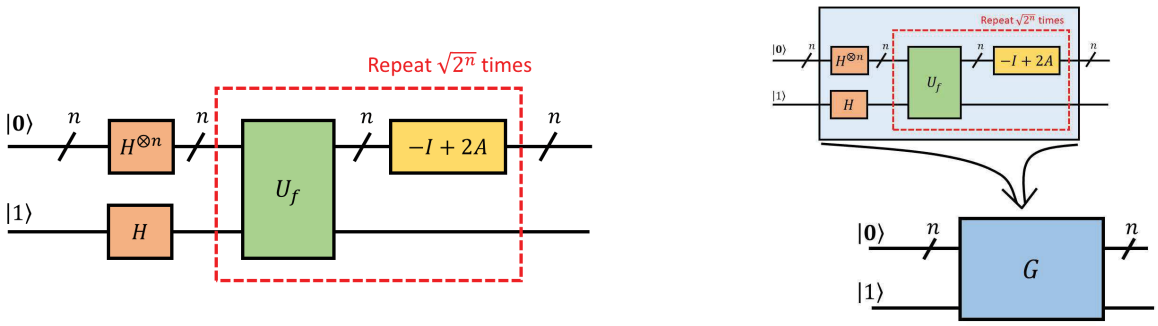

Figure 17: Grover's search algorithm circuit layout (left) and the simplified notation used in this work (right).

For the circuits presented later in this work, the standard Grover's algorithm circuit is represented as a single block as shown on the right in Fig. 17. Fig. 18 demonstrates a Grover's search algorithm gate controlled by n qubits, which are supplied to the circuit in state $|x\rangle$.

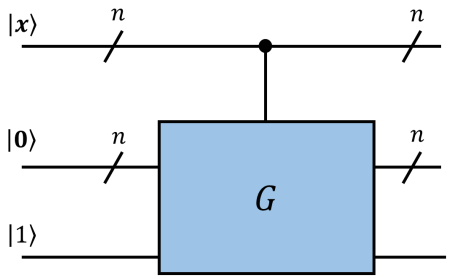

Figure 18: A controlled Grover's algorithm block, where the top qubits $|x\rangle$ represent the control qubits.

4.3.2 Simulation of Modified Grover's Search Algorithm

This section describes our modification of Grover's algorithm that will be used to relate quantum computing and control theory. A MATLAB simulation was created to simulate the Grover's algorithm inspired circuit for a 3-qubit ($n = 3$) case using matrix and vector multiplication and tensor products. It is assumed that a lookup table of all possible measured process states and corresponding control inputs has been generated and converted to binary numbers. For the 3-bit simulation, Table 2 has been assumed for illustration (not related to an LEMPC). Since there are $2^n = 2^3 = 8$ possible states, Table 2 consists of 8 items, which means that 8 controlled Grover's algorithm blocks are required to implement the circuit as shown in figure 16. In these simulations, the $-I + 2A$ inversion about the mean unitary operation was applied once in each Grover's algorithm block, though more could be applied to increase the probability of measuring the desired control input.

Table 2: Table of measured states and the corresponding appropriate control actions, converted to binary numbers.

Measured State	Control Input
000	001
001	100
010	111
011	010
100	110
101	011
110	000
111	101

Since $n = 3$, seven total qubits are fed into the circuit. The top 3 qubits are supplied the measured state and the bottom 4 qubits are given the state $|0001\rangle$. Each of the 2^n controlled Grover's algorithm blocks then tests for one of the possible measured states in sequence. Only one of these blocks activates with the supplied combination of control bits to alter the state of the bottom qubits to (with a probability) give the desired control action when measured. The other gates do not modify the state of the qubits. At the end of the circuit, the states of the qubits leaving the system can then be measured, and the state of qubits 4, 5, and 6 can be taken as the appropriate control input to apply to the process.

The results of a simulation for the state measurement of $|000\rangle$ is shown in Fig. 19. The plot consists of a distribution of the percent probabilities of the resulting state from the circuit collapsing to a particular state when measured. It can be seen that the highest probabilities occur for the $|0000010\rangle$ and $|0000011\rangle$ states, which both have the desired control input string of 001 in positions 4 through 6 (which matches the control input listed for a state of 000 in Table 2). The percent probabilities of these outcomes are both 39.06%, which means that the total probability of ending up with the desired control input string is 78.13%. The other input strings also have a possibility of being measured, each with a percent probability of 1.56%, for a total probability of calculating an incorrect process input of 21.88%.

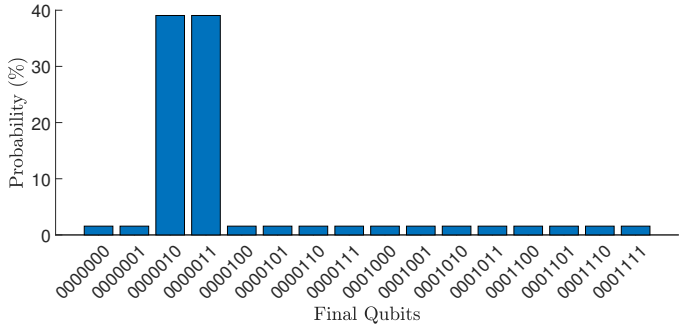

Figure 19: Probability distribution of the quantum register based on the output of the circuit shown in Fig. 16.

4.4 Implications of Non-determinism on Closed-loop Stability

The previous discussions have shown that the LEMPC formulation can account for rounding effects introduced by applying the control formulation on quantum computers. This section begins to examine how the theory would extend to account for non-determinism, as the stability of the system also depends on the probability of obtaining the "desired" solution (i.e., the solution that would be obtained on classical computers). Here, we define the probability of obtaining the expected control action from the modified Grover's algorithm circuit as λ. It is possible to relate this probability to stability of a sampling period (i.e., the probability that the process states remain in Ω_ρ for the duration of at least one sampling period).

First, consider the case where $x(t) \in \Omega_{\rho_e}$ and $\bar{x}(t) \in \Omega_{\rho_e}$. In this case, the theory outlined in the previous section would guarantee that, for LEMPC being performed on a classical computer (where the expected inputs are always applied), $x(t) \in \Omega_\rho$ and $\bar{x}(t_{k+1})$ in Ω_ρ for $t \in [t_k, t_{k+1})$. If, instead, LEMPC in the form of a look-up table were performed on a quantum computer, the algorithm would return the same control action as the classical computer with probability λ. Given this, the probability of maintaining stability for a sampling period is related to the non-determinism in the following way:

$$\mathbf{P}(x(t), \bar{x}(t_{k+1}) \in \Omega_\rho \; \forall \; t \in [t_k, t_{k+1})) \geq \lambda \tag{44}$$

Secondly, consider the case where $x(t_k) \in \Omega_\rho / \Omega_{\rho_e}$ and $\bar{x}(t_k) \in \Omega_\rho / \Omega_{\rho_e}$. Control actions computed by the LEMPC on a classical computer would maintain $x(t) \in \Omega_\rho$ and $\bar{x}(t_{k+1}) \in \Omega_\rho$ for $t \in [t_k, t_{k+1})$ when utilizing LEMPC formulated as a look-up table using the modified Grover algorithm would return the same control action as the classical computer with probability λ. Thus, Eq. 44 still applies for a sampling period.

5 Next-generation manufacturing Communication and Control Related to Quantum Technology

This section discusses the concept of quantum entanglement, which is an important part of quantum mechanics and computing, and its relationship to communication. Quantum entanglement relates the outcomes of measurements on two particles separated at a distance. After being paired, the measurement of one particle has an instant effect on the other particle, no matter the distance. For example, Eq. 1 represents one way in which two particles can be entangled. In this case, when the first qubit is measured collapses to state $|0\rangle$, it can be assured that the state of the second qubit will also be $|0\rangle$ once measured (and, similarly, if the first qubit collapses to $|1\rangle$, so will the second).[1] Given that the state of one particle is transferred instantly at at any range, it could be asked whether this can be utilized in communication applications. If it was possible, it would have significant implications for next generation manufacturing, which involve increased connectivity and integration between many components and involving the transport of large amounts of data.

Having the ability to instantly send information would transform the communication and manufacturing by effectively allowing for greatly increased rates of wireless information transfer (i.e., the number of bits per second), being limited only by the number of qubits implemented. Additionally, such capabilities would allow for the application of extremely distant computational resources for closed-loop control applications. For example, Earth-based computational resources could be utilized for closed-loop control of space-based manufacturing processes, even when communication delays become excessive (the time it takes to relay signals between the earth and Mars, for instance, can represent a time-delay of up to 21 minutes (for a round trip of up to 42 minutes)[35]).

However, it is generally accepted by the no-communication theorem[36–39] that information cannot be transferred in this way, and thus cannot be used for communication purposes. In this section, we review some of the more classical discussions regarding this point and also showcase that several strategies which attempt to add control theory to the strategy to "get around" the limitations do not fix the issue.

5.1 Preliminaries for the Analysis of Communication Delay

First, we outline some preliminaries used in this section.

5.1.1 Deutsch's Algorithm

The Deutsch's algorithm circuit[1] is shown in Fig. 20. This circuit represents a quantum computing algorithm for solving the following problem: given a function f that maps $\{0,1\} \rightarrow \{0,1\}$, determine whether this function has the property that $f(0) = f(1)$ (termed a "constant" function) or that $f(0) \neq f(1)$ (termed a "balanced" function). Whereas in a classical computer, the way to determine this would be to compute both $f(0)$ and $f(1)$ and compare them, in a quantum computer using Deutsch's algorithm, an operation can be performed using two qubits with only one evaluation of the function. The mapping f can be represented via a unitary matrix, denoted here by U_f, that will differ depending on the specific input/output relationship that f has. As shown in Fig. 20, Deutsch's algorithm for an example balanced function receives a top qubit of $|0\rangle$ and a bottom qubit of $|1\rangle$, and it then performs several transformations on the qubits through the matrices H and U_f that cause the top qubit, after the matrix manipulations, to be in state $|0\rangle$ if f is constant, and in state $|1\rangle$ if f is balanced.

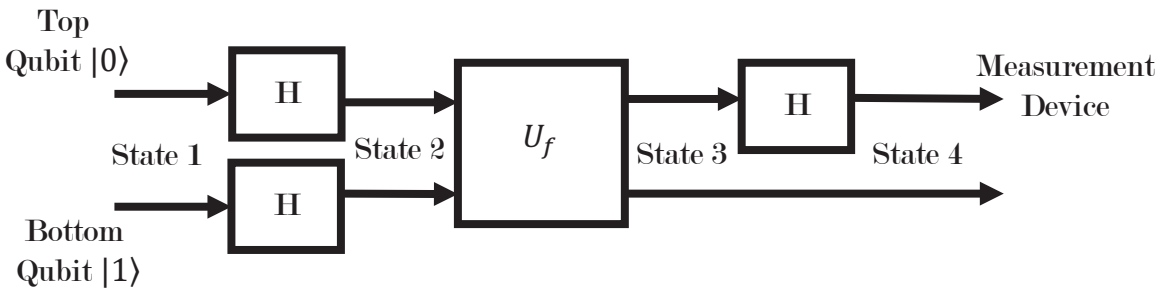

Figure 20: Deutsch's algorithm, redrawn with modifications from.[1]

5.1.2 CHSH game

While it is not believed to be possible to transmit information faster than the speed of light, "non-local games" have been developed which demonstrate means by which entanglement can be used to form a relationship between the actions of two parties who operate independently, at a distance. The CHSH game[40–42] is an example of a game which falls in this category. In the CHSH game (Fig. 21), there are two "players" (traditionally referred to as Alice and Bob) and a "referee." The referee's job is to generate two random numbers $\bar{x}$ and $\bar{y}$ (each either 0 or 1) and send one to Alice and one to Bob. Without communicating with one another, Alice and Bob then need to generate two more numbers (a and b respectively; each a 0 or a 1) and return them to the referee. If $(a+b)$ mod $2 = \bar{x} * \bar{y}$ (where mod signifies the modulo operation), Alice and Bob "win" this game. Their goal is to set a strategy by which each will select a or b based on the value of $\bar{x}$ or $\bar{y}$ that each receives that allows them to win as often as possible. When each is allowed to have half of a Bell pair (which is two entangled qubits), they can develop a strategy for manipulating their qubit and sending their measurement of it to the referee as a and b to win a higher percentage of the time than they would win without the Bell pair.

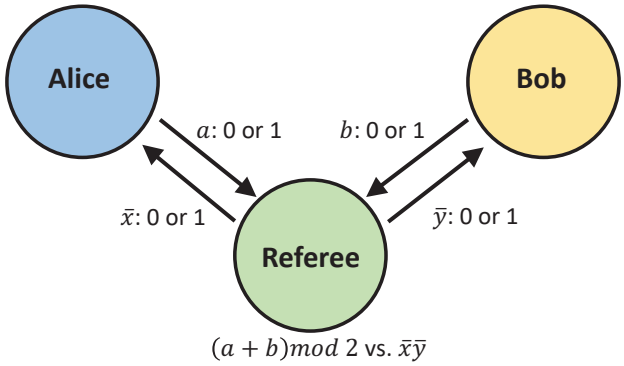

Figure 21: CHSH game flow diagram.

5.2 Deutsch Algorithm-Inspired Strategy

The first strategy considered attempts to manipulate an entangled qubit to force it to communicate information from Alice to Bob. This might be envisioned as a control loop with a remote cloud-based

quantum computer based controller (see Fig. 22), where the process (denoted as the "autonomous agent" in the figure) contains a Sensor Circuit and an Actuator Circuit which aid in manipulating the process state. A Quantum Computing Circuit (presumably located on a Cloud) is used to off-load the computations from the autonomous agent.

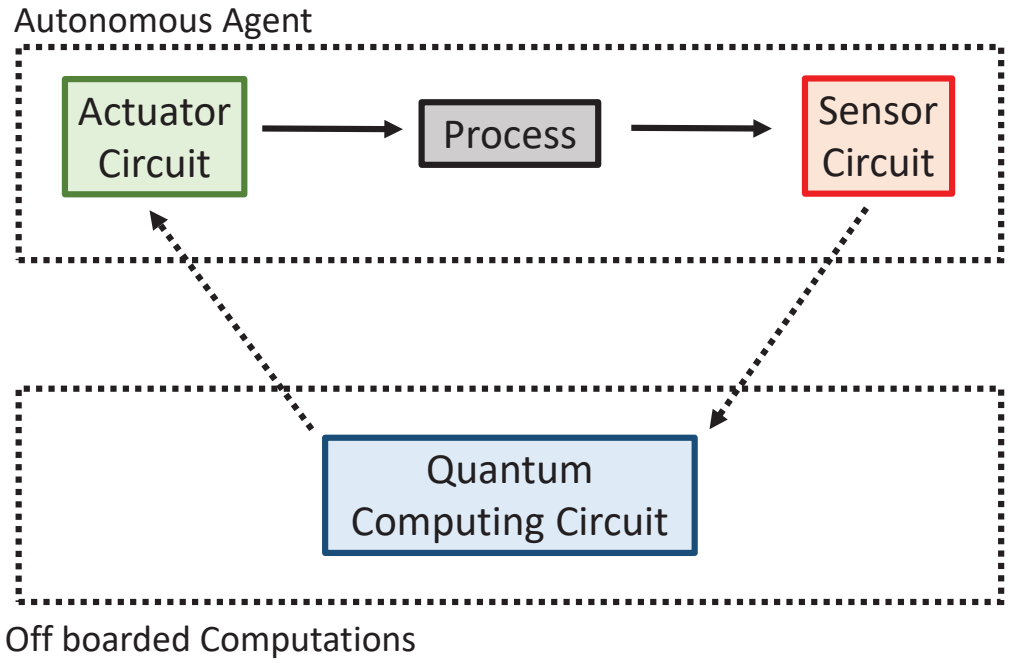

Figure 22: Ideal situation for an autonomous agent.

The specific strategy under consideration in this case is inspired by Deutsch's algorithm (Fig. 20). Specifically, Deutsch's algorithm has the interesting property that it forces the value of a top qubit in a circuit to always take the state $|0\rangle$ for certain structures of U_f (i.e., certain U_f structures are guaranteed to result in a measurement of $|0\rangle$ in the top qubit). We therefore pose the following question: Instead of placing the top qubit in Fig. 20 in state $|0\rangle$, can we instead entangle that top qubit with another qubit, place this other qubit in a remote location, and then force the first qubit value to a known state with Deutsch's algorithm so that we also force the entangled qubit to a certain state? If this was possible, it would transfer information from one qubit to another in a remote location, and potentially provide a framework for fast information transfer for enabling the implementation shown in Fig. 22. However, given that this would involve communication faster

than the speed of light, we can postulate that this should be impossible.

To demonstrate that indeed this does not work, we analyze the circuit concept just described (shown in Fig. 23). This circuit is like that in Fig. 20, with the middle qubit corresponding to the top qubit in Fig. 20 and the bottom qubit corresponding to the bottom qubit in Fig. 20. As in Fig. 20, the qubit corresponding to the top qubit in Fig. 20 has a 50% chance of being measured in state $|0\rangle$ and a 50% chance of being measured in state $|1\rangle$ in State 2. The primary difference compared to Fig. 20 is that the qubit corresponding to the top qubit in Fig. 20 is entangled with the top qubit in Fig. 23. Specifically, these two qubits are entangled as the Bell state $\frac{|01\rangle+|10\rangle}{\sqrt{2}}$. As in Fig. 20, the bottom qubit is assigned a state of $|1\rangle$. The matrix U_f is derived from Deutsch's algorithm; specifically, it represents the case of a balanced function for which $f(0) = 1$ and $f(1) = 0$:

$$U_f = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{45}$$

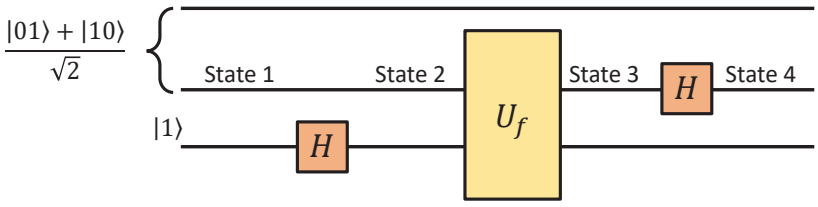

Figure 23: Circuit concept.

The quantum circuit as a whole can be represented using the following matrix:

$$M = (I \otimes H \otimes I)(I \otimes U_f)(I \otimes I \otimes H) = \frac{1}{2}\begin{bmatrix} 1 & -1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 1 & 1 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & 1 \end{bmatrix} \tag{46}$$

The starting state (state 1) is represented as $|\Psi_1\rangle = [0\ \frac{1}{\sqrt{2}}\ \frac{1}{\sqrt{2}}\ 0]^T \otimes [0\ 1]^T = [\,0\ \ 0\ \ 0\ \frac{1}{\sqrt{2}}\ 0\ \frac{1}{\sqrt{2}}\ 0\ \ 0\,]^T$. Multiplying M and $|\Psi_1\rangle$ will then yield the state at the end of the circuit before measurement (state 4): $|\Psi_4\rangle = M\,|\Psi_1\rangle = \frac{1}{2\sqrt{2}}[\,1\ \ -1\ \ -1\ \ \ 1\ \ -1\ \ \ 1\ \ -1\ \ \ 1\,]^T$.

Each coefficient of $|\Psi_4\rangle$, when squared, represents the probability of measuring a particular state for all three qubits after measurement (either $|000\rangle$, $|001\rangle$, $|010\rangle$, $|011\rangle$, $|100\rangle$, $|101\rangle$, $|110\rangle$, or $|111\rangle$, respectively). For example, the probability of obtaining $|000\rangle$ is obtained by squaring the first term ($\left(\frac{1}{2\sqrt{2}}\right)^2 = \frac{1}{8}$). Performing this calculation on all options reveals that there is an equal chance of ending up with any of the 8 possibilities. This means that no information can be sent from the bottom two qubits to the top qubit, and that the resulting measurement of the top qubit cannot be determined any better than a random guess. Therefore, this strategy does not transmit any information and is useless for communication settings.

5.3 Quantum Teleportation Strategy

Though the strategy in Fig. 23 is unsuccessful, a strategy known as quantum teleportation is able to send a quantum state from one individual to another with two bits of classical information transfer, and therefore could be considered for this case. Specifically, the objective of quantum teleportation is to transmit the state of a qubit $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ from Alice (the sender) to Bob (the receiver)

using the aid of a classical information channel.[1,19] The steps to accomplish this are as follows:

1. Two qubits are entangled to become a Bell state. One qubit is given to Alice and the other qubit is given to Bob.

2. Alice interacts a third qubit $|\psi\rangle$ with her entangled qubit.

3. Alice measures her two qubits, causing their states to collapse to $|0\rangle$ or $|1\rangle$.

4. Alice sends Bob the classical bits corresponding to her measurement through a classical communication channel.

5. Depending on the bits sent to Bob by Alice, Bob performs a certain operation on his half of the Bell state, which leaves his qubit in the state $|\psi\rangle$ which Alice originally desired to send Bob.

Because quantum teleportation requires a classical information channel, it does not enable information to be sent faster than the speed of light. Since measurements in a manufacturing control loop are traditionally not quantum states, it is difficult to see the benefit of quantum teleportation itself for transmitting information on the process state or control action between the autonomous agent and the Quantum Computing Circuit because in this strategy, one bit of classical information is encoded in a quantum state which then is transmitted using the entanglement scheme with two bits of classical information.

5.4 State Prediction Strategy

In light of the analysis in Section 5.3, we might ask whether control-theoretic principles might be used to "make up" for the two bits of classical information in quantum teleportation. Specifically, we can ask whether, if $|\psi\rangle$ is encoded to represent the state of the manufacturing process, the Quantum Computing Circuit (Bob) can predict the state of the process (Alice) and use that prediction in place of a measurement of $|\psi\rangle$ to gain auxiliary information needed to perform quantum teleportation without exchanging two bits of classical information. However, not only is this an open-loop

prediction (which defeats benefits of state measurements), but it must also be remembered that one of the steps of quantum teleportation is interaction of $|\psi\rangle$ with the Bell state held by Alice (the manufacturing system). This interaction enables all four of the possible measurements on two qubits to be possible when the sensing circuit performs its measurement (i.e., it is no longer a function only of $|\psi\rangle$ that can be predicted, or of the original Bell state which had a defined relationship between only two qubits so that if one was measured, the state of the other could be known). Therefore, the actions involved in setting up quantum teleportation remove from Bob (the computing system) the possibility of predicting the outcomes of the measurements. This renders physics-based or data-driven modeling ineffective for obtaining $|\psi\rangle$ any better than an open-loop prediction where entanglement is not needed.

5.5 Stability-based Strategy

If both the Quantum Computing Circuit and the Actuator Circuit of Fig. 22 have an entangled qubit, to each of those agents individually, a measurement of that qubit has a 50% chance of returning $|0\rangle$ and a 50% chance of returning $|1\rangle$, which is no better than a random guess. One might therefore ask whether control actions communicated via entanglement in this manner could be sufficient as long as they are stabilizing. For a nonlinear system, this might mean that they decrease a Lyapunov function over time. At the Actuator Circuit, it would be possible to check whether a control action obtained by measuring an entangled qubit would decrease the Lyapunov function over time. However, even if this control action achieves that, it is again not necessarily the one which would have been computed on the Quantum Computing Circuit (thus it is essentially a random guess and therefore stabilization of a process using such a control action is "lucky"). Therefore, there is no benefit to having a computing circuit at all at that point; it is equivalent to randomly selecting from the set of potential inputs one of them and seeing whether it is stabilizing.

5.6 "Incorporating Hardware" Strategy

The control-theoretic and entanglement-based strategies above were not effective at permitting information transfer between the autonomous and quantum computing systems. Another strategy which might be investigated is one which involves a special design of the hardware on the various circuits in Fig. 22. For example, the Sensor Circuit might consist of $\bar{N}$ qubits entangled with those on the Quantum Computing Circuit. Each of these qubits would correspond to a digit of a binary number representing the measured state, and this correspondence would be hard-wired in both the Sensor Circuit and the Quantum Computing Circuit prior to operation. When the Sensor Circuit receives a measurement of 1 from the process, it would then measure the qubits corresponding to digits of 1 in the binary representation of the state measurement. If there is some way that the Quantum Computing Circuit could determine which qubits were then measured, it could then know the process state regardless of the measurement outcome by noting the measured qubits represent the 1 digits.

For example, if the state can be represented using four digits, the Sensor Circuit and Quantum Computing Circuit would share four Bell state pairs. Then, to send the state 1001 from the Sensor Circuit to the Quantum Computing Circuit, the Sensor Circuit could measure the first and last qubits. If the Quantum Computing Circuit could tell that these two qubits were measured on the sensor, then it would know the binary number that represents the measured process state. The Quantum Computing Circuit could assume that the Sensor Circuit is making regular measurements at some time interval, so it could know when the measurements were taken. However, the Quantum Computing Circuit would be unable to tell which qubits were measured if the measurement collapses the qubits to $|0\rangle$ or $|1\rangle$ and that is the information used to attempt to discern whether a measurement of qubits at the Sensor Circuit took place. This is because measuring any of its qubits would result in the collapsed state of either $|0\rangle$ or $|1\rangle$ with 50% probability, which will be the same result regardless of whether the corresponding Bell state was measured on the sensing circuit. Therefore, this strategy fails at transmitting any information about the state and could not be used in communication strategies.

An alternate "incorporating hardware" strategy could involve the following. Here, we allow n $\bar{N}$-qubit sets to be shared between the Sensor Circuit and Quantum Computing Circuit. When the Sensor Circuit receives a measurement of the process state, it then measures the n sets of entangled qubits. Since each qubit in a set has a 50% probability of measuring $|0\rangle$ or $|1\rangle$, only a portion of the measured sets would have states matching the actual process state. By having a sufficient number of entangled qubit sets (n is large), it is hoped that at least one will correspond to the measurement of the process state. Because the original Bell states were known, this measurement gives the Sensor Circuit knowledge of what states were received by the Quantum Computing Circuit. The Sensor Circuit also knows which qubit set(s) collapsed to the process state when measured.

The Quantum Computing Circuit can then have n circuits, one corresponding to each of the entangled Bell states. When Quantum Computing Circuit measures its entangled qubit sets, it gains knowledge of which states the Sensing Circuit has, though it cannot know which of these matches the actual process state. Instead of trying to determine which state matches the process, the Quantum Computing Circuit can then perform an appropriate series of gates on all of the qubit sets. This leaves the Quantum Computing Circuit with n potential control actions (by ensuring n is large enough, it is hoped that at least one of these is the desired control action) which it wishes to send back to the actuator.

At this point, if it were possible to send the n control actions and the process states they correspond with to the Actuator Circuit, the Actuator Circuit could then select the correct control action by communicating with the Sensor Circuit (as the Sensor Circuit is local and knows the correct process state). In doing this, the number of entangled qubits per set would be doubled to form k $2\bar{N}$-qubit sets, where each set contains a process state-input pair. By letting $k >> n$, it is hoped that at least one of the k $2\bar{N}$-qubit sets collapses to the process state-input pair that corresponds to the actual process state. However, since the entangled qubits collapse randomly, it is not possible for the Quantum Computing Circuit to signify which of the n $2\bar{N}$-qubit sets correctly contains a matching process state-input pair. For this reason the strategy fails, as it cannot communicate control actions to the Actuator Circuit any better than a random guess.

This strategy could have some benefit, as the Quantum Computing Circuit could be designed

to communicate the process state-input pairs to the actuator over classical information channels. Then the actuator could properly select the correct input by communicating with the sensor. Since classical information only travels in one direction using this setup, the communication time would be effectively halved, though with drawbacks such as the redundancy in the hardware that would need to be considered.

5.7 CHSH Game-Inspired Strategy

A final strategy to be discussed is one which questions whether the CHSH game in Section 5.1.2, which allows the results of actions by two agents at a distance to be coordinated without communication such that the two agents can win a game a higher percentage of the time when their coordination includes a Bell pair than when it does not, could aid in setting up communication for next-generation manufacturing. If we consider that the Quantum Computing Circuit in Fig. 22 is one player in the game and the Actuator Circuit is the other, for example, both can execute a strategy on half of an entangled Bell pair to attempt to develop coordinated results. However, unless the two measurements of the Bell States are put together, they appear random to the Quantum Computing Circuit and the Actuator Circuit. The CHSH game is about relationships between the measurements of the Bell states; it is not about the specific values of each individual Bell state. Therefore, unless the Actuator Circuit learns what the Quantum Computing Circuit measured on its Bell state, it will not be able to construct a meaningful control action any more than if it had made a random guess of the control action. It therefore does not permit an instantaneous communication of information of control actions.

6 Conclusion

This work reviewed several topics related to quantum computing and its potential intersections with next-generation manufacturing in control and communication. It is interesting to consider potential applications of the results toward issues such as cybersecurity of control systems. For example, a policy for generating random control actions was developed in;[43] the noise in the quantum hardware

can aid in generating randomness. Also, those results indicate that if inputs which would not be stabilizing were cut from the possibilities at a sampling time on classical or quantum hardware, there would not be a possibility for selecting improper ones from a cybersecurity or quantum computing-implemented control perspective. The results of this work are intended to inspire future work at the intersection of control and quantum computing.

Acknowledgment

Financial support from the Air Force Office of Scientific Research (award number FA9550-19-1-0059), National Science Foundation CNS-1932026 and CBET-1839675, Michigan Space Grant Research Consortium award number 80NSSC20M0124, Wayne State University Grants Boost funding, and Wayne State University is gratefully acknowledged.

We wish to thank Paul M. Alsing of the Air Force Research Laboratory, Information Directorate (AFRL/RI) for useful comments and discussions.

The authors would like to acknowledge the YouTube videos on QFT and QFT-based addition, which can be accessed from https://quantumguru.net/quantumalgorithms/quantumalgorithms.html and were very helpful in preparing this paper and defining the notation in discussing QFT. We would also like to acknowledge the series of seven lecture videos on the state of the field of quantum computing provided by Elias Fernandez-Combarro Alvarez and CERN online, which can be accessed on YouTube within the series "A Practical Introduction to Quantum Computing: From Qubits to Quantum Machine Learning and Beyond" (also accessible from https://home.cern/news/announcement/computing/online-introductory-lectures-quantum-computing-6-november). These were also very helpful in providing an introduction to Qiskit and to the state-of-the-art in quantum computing.

We acknowledge the use of IBM Quantum services for this work. The views expressed are those of the authors, and do not reflect the official policy or position of IBM or the IBM Quantum team.

Literature Cited

(1) Yanofsky, N. S.; Mannucci, M. A. *Quantum Computing for Computer Scientists*; Cambridge University Press, 2008.

(2) Koch, D.; Martin, B.; Patel, S.; Wessing, L.; Alsing, P. M. Demonstrating NISQ era challenges in algorithm design on IBM's 20 qubit quantum computer. *AIP Advances* **2020**, *10*, 095101.

(3) DiVincenzo, D. P.; Loss, D. Quantum computers and quantum coherence. *Journal of Magnetism and Magnetic Materials* **1999**, *200*, 202–218.

(4) Ajagekar, A.; You, F. Quantum computing for energy systems optimization: Challenges and opportunities. *Energy* **2019**, *179*, 76–89.

(5) Ajagekar, A.; You, F. Quantum computing assisted deep learning for fault detection and diagnosis in industrial process systems. *Computers & Chemical Engineering* **2020**, *143*, 107119.

(6) Ajagekar, A.; You, F. A Deep Learning Approach for Fault Detection and Diagnosis of Industrial Processes using Quantum Computing. Proceedings of the International Conference on Systems, Man, and Cybernetics. Toronto, Ontario, Canada, 2020; pp 2345–2350.

(7) Ajagekar, A.; Humble, T.; You, F. Quantum computing based hybrid solution strategies for large-scale discrete-continuous optimization problems. *Computers & Chemical Engineering* **2020**, *132*, 106630.

(8) Rigatos, G. G.; Tzafestas, S. G. Parallelization of a fuzzy control algorithm using quantum computation. *IEEE Transactions on Fuzzy Systems* **2002**, *10*, 451–460.

(9) Dong, D.; Chen, C.; Li, H.; Tarn, T.-J. Quantum reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* **2008**, *38*, 1207–1220.

(10) Lamata, L. Basic protocols in quantum reinforcement learning with superconducting circuits. *Scientific Reports* **2017**, *7*, 1–10.

(11) Wu, S.; Jin, S.; Wen, D.; Wang, X. Quantum reinforcement learning in continuous action space. *arXiv preprint arXiv:2012.10711* **2020**,

(12) Adeodu, O.; Omell, B.; Chmielewski, D. J. On the theory of economic MPC: ELOC and approximate infinite horizon EMPC. *Journal of Process Control* **2019**, *73*, 19–32.

(13) Alessandretti, A.; Aguiar, A. P.; Jones, C. N. On convergence and performance certification of a continuous-time economic model predictive control scheme with time-varying performance index. *Automatica* **2016**, *68*, 305–313.

(14) Amrit, R.; Rawlings, J. B.; Angeli, D. Economic optimization using model predictive control with a terminal cost. *Annual Reviews in Control* **2011**, *35*, 178–186.

(15) Angeli, D.; Amrit, R.; Rawlings, J. B. On Average Performance and Stability of Economic Model Predictive Control. *IEEE Transactions on Automatic Control* **2012**, *57*, 1615–1626.

(16) Griffith, D. W.; Zavala, V. M.; Biegler, L. T. Robustly stable economic NMPC for non-dissipative stage costs. *Journal of Process Control* **2017**, *57*, 116–126.

(17) Inoue, D.; Yoshida, H. Model predictive control for finite input Systems using the D-Wave Quantum Annealer. *Scientific Reports* **2020**, *10*, 1–10.

(18) Heidarinejad, M.; Liu, J.; Christofides, P. D. Economic model predictive control of nonlinear process systems using Lyapunov techniques. *AIChE Journal* **2012**, *58*, 855–870.

(19) Nielsen, M. A.; Chuang, I. *Quantum Computation and Quantum Information, 10th Anniversary Edition*; Cambridge University Press: New York, 2010.

(20) Preskill, J. Quantum computing in the NISQ era and beyond. *Quantum* **2018**, *2*, 79.

(21) Rangan, K. K.; Abou Halloun, J.; Oyama, H.; Cherney, S.; Azali Assoumani, I.; Jairazbhoy, N.; Durand, H.; Ng, S. K. Quantum Computing and Resilient Design Perspectives for Cybersecurity of Feedback Systems. Proceedings of the IFAC Symposium on Dynamics and Control of Process Systems, including Biosystems. Busan, Republic of Korea, in press, 2022.

(22) Team, Q. D. Quantum Fourier Transform. https://qiskit.org/textbook/ch-algorithms/quantum-fourier-transform.html, Accessed: 2022-02-06.

(23) Ruiz-Perez, L.; Garcia-Escartin, J. C. Quantum arithmetic with the quantum Fourier transform. *Quantum Information Processing* **2017**, *16*, 152.

(24) Anagolum, S. DoNew. 2018; `https://github.com/SashwatAnagolum/DoNew`.

(25) Urbanek, M.; Nachman, B.; Pascuzzi, V. R.; He, A.; Bauer, C. W.; de Jong, W. A. Mitigating depolarizing noise on quantum computers with noise-estimation circuits. *Physical Review Letters* **2021**, *127*, 270502.

(26) Nachman, B.; Urbanek, M.; de Jong, W. A.; Bauer, C. W. Unfolding quantum computer readout noise. *npj Quantum Information* **2020**, *6*, 1–7.

(27) Greenbaum, D.; Dutton, Z. Modeling coherent errors in quantum error correction. *Quantum Science and Technology* **2017**, *3*, 015007.

(28) Kabytayev, C.; Green, T. J.; Khodjasteh, K.; Biercuk, M. J.; Viola, L.; Brown, K. R. Robustness of composite pulses to time-dependent control noise. *Physical Review A* **2014**, *90*, 012316.

(29) ANIS, M. S. et al. Qiskit: An Open-source Framework for Quantum Computing. 2021.

(30) Garcia-Escartin, J. C.; Chamorro-Posada, P. Equivalent quantum circuits. *arXiv preprint arXiv:1110.2998* **2011**,

(31) Shaiju, A.; Petersen, I. R. Formulas for discrete time LQR, LQG, LEQG and minimax LQG optimal control problems. *IFAC Proceedings Volumes* **2008**, *41*, 8773–8778.

(32) Heidarinejad, M.; Liu, J.; Christofides, P. D. Economic model predictive control of nonlinear process systems using Lyapunov techniques. *AIChE Journal* **2012**, *58*, 855–870.

(33) Daru, M. S.; Jager, T. Encrypted Cloud-based Control using Secret Sharing with One-time Pads. 2019 IEEE 58th Conference on Decision and Control (CDC). 2019; pp 7215–7221.

(34) Khalil, H. K. *Nonlinear Systems*, 3rd ed.; Prentice-Hall: Upper Saddle River, New Jersey, 2002.

(35) Attwood, J. R. Mars in 2018. *The Journal of The Royal Astronomical Society of Canada* **2018**, *112*, 121–122.

(36) Ghirardi, G.-C.; Rimini, A.; Weber, T. A general argument against superluminal transmission through the quantum mechanical measurement process. *Lettere al Nuovo Cimento (1971-1985)* **1980**, *27*, 293–298.

(37) Hall, M. J. Imprecise measurements and non-locality in quantum mechanics. *Physics Letters A* **1987**, *125*, 89–91.

(38) Ghirardi, G. C.; Grassi, R.; Rimini, A.; Weber, T. Experiments of the EPR type involving CP-violation do not allow faster-than-light communication between distant observers. *EPL (Europhysics Letters)* **1988**, *6*, 95.

(39) Eberhard, P. H.; Ross, R. R. Quantum field theory cannot provide faster-than-light communication. *Foundations of Physics Letters* **1989**, *2*, 127–149.

(40) Clauser, J. F.; Horne, M. A.; Shimony, A.; Holt, R. A. Proposed experiment to test local hidden-variable theories. *Physical Review Letters* **1969**, *23*, 880.

(41) Cui, D.; Mehta, A.; Mousavi, H.; Nezhadi, S. S. A generalization of CHSH and the algebraic structure of optimal strategies. *Quantum* **2020**, 346.

(42) Sikora, J.; Chailloux, A.; Kerenidis, I. Strong connections between quantum encodings, non-locality, and quantum cryptography. *Physical Review A* **2014**, *89*, 022334.

(43) Durand, H. A Nonlinear Systems Framework for Cyberattack Prevention for Chemical Process Control Systems. *Mathematics* **2018**, *6*, 44 pages.

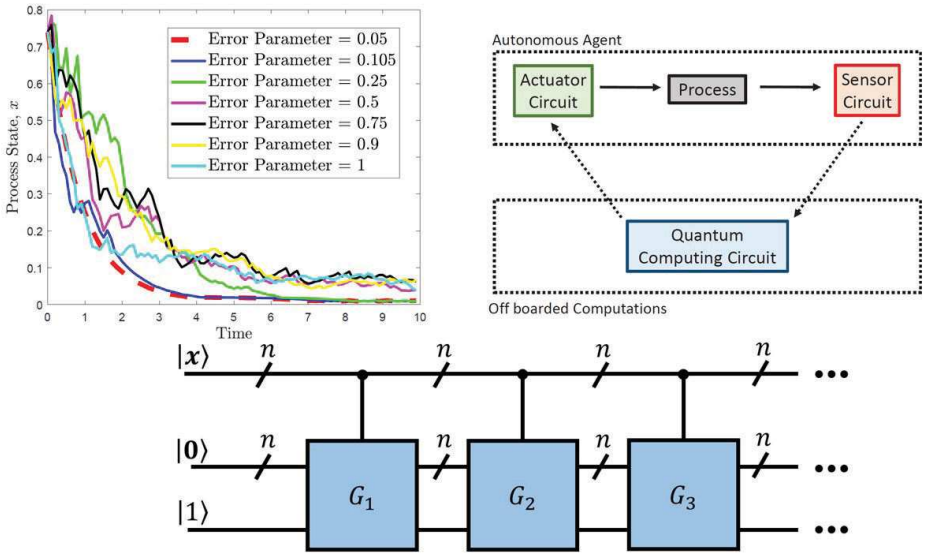

Figure 24: Table of Contents Graphic. In this work, we provide some initial studies regarding the implementation of control on quantum computers, including the implementation of a single-input/single-output proportional control law on a quantum simulator with noise, evaluation of potential impacts of non-determinism on theory for advanced control laws, and discussion of consequences of the way that entanglement works for next-generation manufacturing communication objectives.