CPS: SMALL: Learning How to Control - A Meta-Learning Approach for the Adaptive Control of Cyber-Physical Systems

1.0 Introduction:

Cyber-physical systems may be seen as consisting of two types of fabrics; a physical fabric whose dynamics govern mass and energy flows and a cyber fabric whose dynamics govern information flows. The Internet-of-Things (IoT) enabled manufacturing system depicted in Fig. 1 is an example of a particularly important class of CPS. The physical fabric for this system is woven from a heterogeneous mix of machines that carry and process materials across the factory floor. The cyber fabric is a heterogeneous mix of wired and wireless communication networks that ensure digital data streams from factory sensors and machines can be used by cloud-based planners and edge devices to manage the physical fabric's workflows in a safe and efficient manner. The problem addressed in this project is concerned with *learning how to control* both fabrics, physical and cyber, in a coordinated and scalable manner.

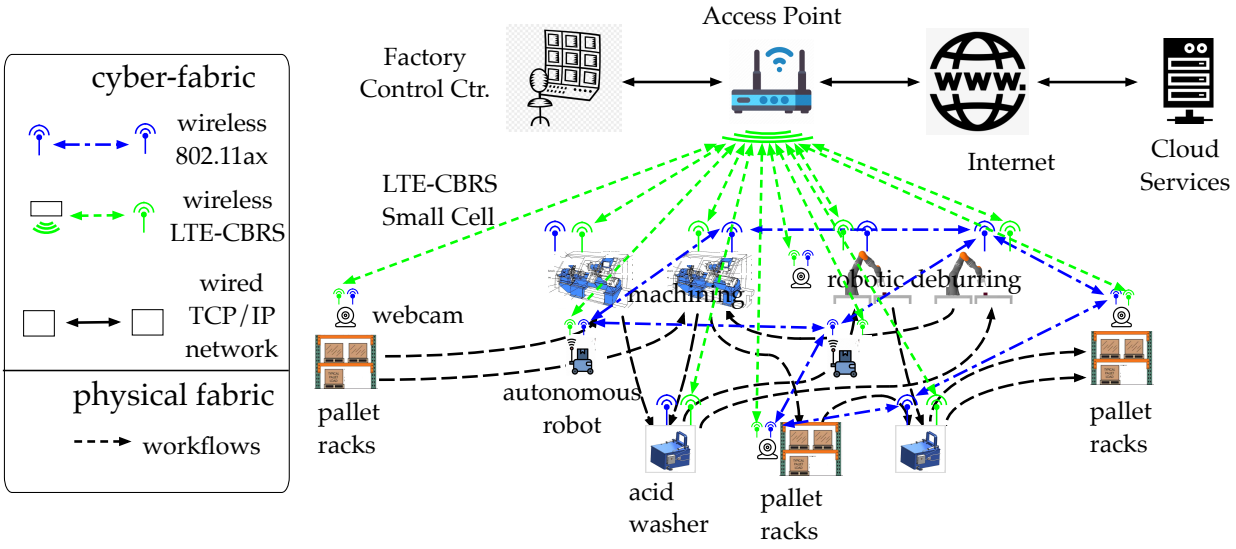

Figure 1: IoT-enabled Manufacturing as a Cyber-Physical System

The IoT-manufacturing system depicted in Fig. 1 is a complex CPS with a great deal of modeling uncertainty. The physical and cyber fabrics are open to environmental forcing that can shift in an abrupt and unpredictable manner. These shifts may be due to changes in customer work orders that change factory floor workflows. These shifts may also be caused by environmental changes that increase interference in the cyber fabric's wireless communication networks. The dynamics of both fabrics are coupled since congestion in the physical fabric may engender congestion in the cyber fabric and vice versa. IoT manufacturing's inherent complexity and uncertainty stand as major obstacles to U.S. manufacturer's adoption of IoT technologies [14]. So while the worldwide rate of investment in IoT technologies by the manufacturing sector stands at 6.5%. This rate is highest in China and Europe with the U.S. lagging behind [13]. This disparity may be addressed through the development of methods that manage the uncertainty and complexity of these particular CPS systems. This project proposes doing that by developing a meta-learning framework [23] that *learns how to control* complex CPS performing a variety of tasks in an uncertain environment.

Prior work has often used *deep reinforcement learning* (DRL) [64] to learn how to control CPS

fabrics [59, 76, 85]. This prior work, however, has focused on idealized applications in stationary environments. Moving to real-life CPS with nonstationary disturbances requires an adaptive approach that is difficult to realize using DRL. The difficulties stem from fundamental criticisms of deep learning [61] with regard to its need for huge data sets and the lack of transparency of the resulting models. Deep learning is *data hungry* since it requires extremely large training sets to achieve acceptable performance. Its data sets are usually processed *offline*, which is clearly unsuitable for real-time adaptive control. Moreover, it is difficult to *interpret* how a deep neural network's parameters are related to fundamental features in the CPS. This makes it difficult to transfer knowledge between different tasks in a way that ensures controlled performance is both optimal and robust. When deep learning is used in reinforcement learning, we find impressive results demonstrating near human levels of control [64]. But these successes are not *robust* to environmental change [37]. In other words, DRL does not have the robust stability to reliably manage the complex CPS found in the IoT-manufacturing application shown in Fig. 1.

This project will address DRL's robust stability problem through a novel learning model that we call a *behaviorally ordered abstraction* or BOA. BOAs are based on the fact that when a dynamical system's orbits are closed and bounded, then its trajectories can be written as the sum of its *steady-state* and *natural* response [28]. A BOA is a pair of models that separately capture these two responses. In particular, a BOA has a *moment-matching* (MM) model [3] capturing the plant's steady-state response and a *model-following* (MF) system capturing the plant's natural response. The remarkable thing is that both models have a linear structure that can be efficiently learned in a data driven manner through regression [78] or dynamic mode decomposition algorithms [74]. This means that BOA models will address deep learning's issues with regard to being data-hungry and using offline training. Moreover, the BOA's parameters can be concretely interpreted in terms of the system's natural and steady-state response, thereby addressing issues regarding a deep neural network's interpretability. The final useful BOA property is that its MF-model is the augmented plant of a generalized regulator used in formulating a robust control problem [98]. This means that BOAs provide a framework for ensuring a DRL control policy has *robust stability*.

The BOA-models used in this project represent a novel approach to learning control that we introduced in [48] to robustly control hopping robots. This project will explore how one can use these models to robustly control complex CPS found in IoT manufacturing systems. In particular, this project will develop a *meta-learning* [23] framework based on BOAs that *learn how to control* complex CPS performing a number of different tasks. The resulting meta-learning algorithms will be experimentally evaluated using a hardware testbed based on the IoT-manufacturing system shown in Fig. 1. The proposed testbed will be an IoT-enabled CPS whose physical fabric consists of robots following scheduled routes determined by a central planner receiving real-time robot data streams over a cyber fabric formed by a 5MHz WiFi network.

The remainder of the project description is organized as follows. Section 2 is the *Research Description*. Section 3 is the project's *Evaluation Plan*. Section 4 is the Project's *Management Plan*. *Broader Impacts* and *results form prior NSF research* are in sections 5 and 6, respectively

2.0 Research Description:

This section describes the research to be performed by the project. Section 2.1 defines *behaviorally ordered abstractions (BOAs)*. Section 2.2 describes *preliminary work*. Section 2.3 describes the proposed *research activities*. Section 2.4 is the subsection on *CPS Research Focus*.

2.1 Behaviorally Ordered Abstractions (BOAs): Machine learning (ML) is often posed as an optimization problem selecting models that minimize a chosen loss function with respect to a finite amount of data generated by a system. One may also, however, think of ML as selecting models

for *coordinate-free concepts* that are used to understand and manage what a given system is doing. These "concepts" are then represented as low-dimensional manifolds in a high dimensional data space [55]. The complexity of finding such manifolds depends greatly on the *features* used to encode the data. The reason why deep learning is data hungry is because it is trying to learn these features directly from the data. This approach makes sense for image classification [43], but this project is concerned with learning how to control *dynamical systems* and for such systems the "concepts" of interest are the system's *basic limit sets* [12]. These basic sets are *fundamental features* for all compact dynamical systems. By structuring our models in terms of these particular features, one can greatly reduce the sample complexity of learning these models.

Basic limit sets are invariant sets that form natural features for homogeneous dynamical systems with closed and bounded (i.e. compact) orbits. For such systems there always exists a *limit set* that is invariant, attracting, and is formed from the union of mutually disjoint *basic limit sets* [12]. When there are a countable number of basic sets, then one can classify any orbit of the system as being either *recurrent* or *nonrecurrent* [7]. Recurrent orbits are contained within a single basic set and they represent what we commonly think of as the system's *steady-state response*. A nonrecurrent trajectory connects two different basic sets in the sense that the trajectory approaches different basic sets as $t \to \infty$ and as $t \to -\infty$. Any nonrecurrent trajectory converges asymptotically to a recurrent orbit and the difference between the nonrecurrent and recurrent orbits forms what we commonly think of as the system's *natural response*. In traditional signals/systems terminology, this means that the trajectories of any homogeneous system with compact orbits can be decomposed as the sum of its natural and steady-state responses [28]. For convenience, we refer to this as the system's *behavioral decomposition*. BOA models are based on this behavioral decomposition.

To formally define a BOA, we consider a *plant*, Σ_0, with two types of inputs and two types of outputs. The plant inputs are *disturbances*, $w(t)$, and *controls*, $u(t)$. The plant outputs are the system *loss*, $z(t)$, and the system *observations*, $y(t)$. To take advantage of behavioral decompositions, we assume the disturbance, $w(t)$, is the output of another dynamical system, Σ_g, called the *generator*. The plant is the process we wish to control and the generator is the process representing the dynamics of the external environment that the plant is embedded within. We also assume that the control, $u(t)$, is generated by a *stabilizing base controller*, Σ_c, that ensures the plant's states remain closed and bounded. This base controller could have been obtained previously through a number of methods. For instance, it could have been the policy learned using DRL. But the stabilizing control policy could have also been generated through model predictive control methods (as done in [9]). For whatever base control policy, Σ_c, the main requirement is that it is "stabilizing" with respect to some generator, Σ_g. Our main problem is then to learn how to *robustly stabilize* the control policy in the presence of perturbations to the environment, Σ_g or to the physical plant, Σ_0. Since we don't have an accurate prior model for Σ_g or Σ_0, we must use ML methods to learn how to stabilize the prior base policy, Σ_c.

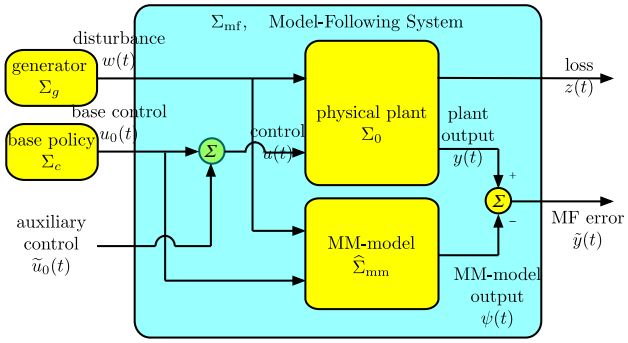

Figure 2: The MF System, Σ_{mf}, captures the natural dynamics of the plant, Σ_0, under auxiliary control, $\tilde{u}_0(t)$, with respect to a given generator, Σ_g, and base control policy Σ_c.

A behaviorally ordered abstraction (BOA) for the plant, Σ_0, with respect to generator, Σ_g, and base controller, Σ_c, is a pair of state-based systems, $(\widehat{\Sigma}_{\text{mm}}, \widehat{\Sigma}_{\text{mf}})$, called the moment-matching or MM-model and model-following or MF-model, respectively. The MM-model, $\widehat{\Sigma}_{\text{mm}}$, captures the

plant's steady-state response with respect to the generator and base control policy. This means that the MM-model's output, $\psi(t)$, asymptotically converges to the steady-state response of the plant's observation, $y(t)$. The MF-model, $\widehat{\Sigma}_{\rm mf}$, is a state-based model for a *model-following* MF-system shown in Fig. 2. The MF-system is a classical model-following system [66] often used in model reference adaptive control (MRAC) [69]. The difference from conventional MRAC is that our MF-system uses the MM-model, $\widehat{\Sigma}_{\rm mm}$, as the reference model. Our model, $\widehat{\Sigma}_{\rm mf}$, for the MF-system has an *auxiliary control input*, $\tilde{u}(t)$, injected on top of the base control, $u_0(t)$. The output of the MF-system is a model following error , $\tilde{y}(t) = y(t) - \psi(t)$, describing how closely the plant's observation , $y(t)$, is tracking the MM-model's output, $\psi(t)$. The MF-system therefore captures how the auxiliary inputs, $\tilde{u}(t)$, excite those controllable plant states that are observable from the observation, $y(t)$. Since the MF-system subtracts out the MM-model's response, we see that the MF-model, $\widehat{\Sigma}_{\rm mf}$, is capturing the plant's *natural response* whereas the MM-model, $\widehat{\Sigma}_{\rm mm}$, is capturing the plant's *steady-state response*.

Both components of a BOA can be learned in an efficient and online manner. Let us take a closer look at the two models to justify this assertion. Let the plant, Σ_0, and generator, Σ_g, be

$$\Sigma_0 \; : \; \left\{ \begin{array}{rcl} \dot{\mathbf{x}}(t) & = & \mathbf{F}(\mathbf{x}(t), w(t), u_0(t)) \\ y(t) & = & \mathbf{G}(\mathbf{x}(t)) \end{array} \right. \qquad \Sigma_g \; : \; \left\{ \begin{array}{rcl} \dot{\omega}(t) & = & \mathbf{S}(\omega(t)) \\ w(t) & = & \mathbf{L}\omega(t) \end{array} \right.$$

Following [3], a moment matching MM-model, $\widehat{\Sigma}_{\rm mm}$, has the state space realization model

$$\begin{array}{rcl} \dot{\xi}(t) & = & \mathbf{S}(\xi(t)) + \mathbf{\Delta}(w(t) - \mathbf{L}\xi(t)) \\ \psi(t) & = & [\mathbf{G} \circ \mathbf{\Pi}]\,(\xi(t)) \end{array} \tag{1}$$

where $\xi(t)$ is the MM-model state and $\mathbf{\Delta}$ is a matrix parameter. The *moment map*, $\mathbf{G} \circ \mathbf{\Pi}$, lifts the MM-model's state into the plant's state space and is chosen in such a way [3] that the MM-model's steady-state output, $\psi(t)$, equals the plant's steady-state output, $y(t)$.

Learning the MM-model has two parts; learning the state dynamics and the moment map. If the inputs, w and u_0, are bounded, then we can use a Fourier analysis of w to find the disturbance signal's fundamental harmonics and then use the periods of these harmonics to construct a set of harmonic oscillators, thereby fixing the $\mathbf{S}$ matrix in equation (1). This approach reduces the MM-model's state equations to that of a Luenberger observer that is estimating the "state" of the generator, Σ_g. Note that the MM-model's state equations are linear so that the plant's nonlinearities are pushed into the moment map, $\mathbf{G} \circ \mathbf{H}$. Learning this memoryless map is a classical machine learning problem that can be solved by training linear or kernel-based support vector machines (SVM). In other words, the MM-model can be trained as a data-driven regression problem [78]. Such regressions can also be solved in an online manner using recursive least-squares estimators. Moreover, the SVM's VC-dimension is given the size of the moment map model, thereby allowing one to use existing frameworks to manage the model's complexity [6].

Let us now examine how one trains the second BOA model, $\widehat{\Sigma}_{\rm mf}$. The MF-system, $\Sigma_{\rm mf}$, is a model-following system built around the MM-model, $\widehat{\Sigma}_{\rm mm}$. In other words, the MF-system characterizes how well the plant is tracking the steady-state behavior predicted by the MM-model. Locally, this tracking behavior can be modeled by a linear state-based system, $\widehat{\Sigma}_{\rm mf}$. Our approach for learning the MF-model is based on the Koopman decomposition [5]. In particular, we inject a test signal, $\tilde{u}(t)$, on top of the base control input, $u_0(t)$. The test signal is a sequence of exponentially decaying impulses that excite the natural dynamics of the plant about the steady-state response. We then use delay embeddings [32] of the MF-error, $\tilde{y}(t)$, as a surrogate for the con-

trollable/observable states of the MF-system. This approach allows us to use algorithms such as *dynamic mode decomposition* with control (DMDc) [74] to identify the MF-model, $\widehat{\Sigma}_{\text{mf}}$. The DMDc algorithm is also solving a regression algorithm, though its implementation is more involved since it takes the singular value decomposition (SVD) of the data matrix. Nonetheless, the regressive nature of the problem suggests that the DMDc algorithm can also be realized in an online manner.

It may be surprising, at first glance, that the BOA's linear structure is capable of capturing a nonlinear plant's behavior. The reason for this is that the MF-model captures the dynamics *about* one of the plant's basic limit sets. These dynamics are inherently hyperbolic since the center manifold lies in the basic limit set and so the plant's natural dynamics are locally approximated by linearizations. This means that the plant's nonlinearities are isolated to its behavior on a basic limit set; behaviors that are captured by the MM-model's moment map. Since training that moment map is basically a regression, we can again get away with linear modeling structures. Another reason why linearly structured models can be used is because we are using them in a feedback loop. In other words, one does not need to use a high fidelity *open loop* plant model because the feedback signals act to reduce the *closed loop* system's sensitivity to modeling error.

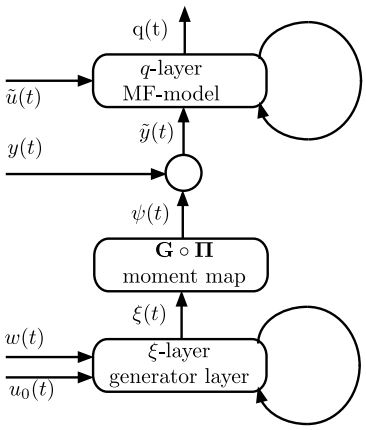

Figure 3: BOA as a multi-layered recurrent network

Let us now consider how a BOA differs from deep neural networks. Deep neural networks consist of numerous layers capturing fundamental "features" in the data set. Deep learning trains these layers through the backpropagation algorithm and places no assumptions on what these features will be. As a result deep learning yields small biases but that performance is achieved after training on very large data sets. Deep learning is well suited for image classification applications where the fundamental features need to be learned [43]. This project, however, focuses on dynamical systems whose behaviors can be characterized in terms of naturally occurring invariant sets that allow us to decompose a system's response into its natural and steady-state responses. These two responses are invariant concepts that we take as the dynamical system's features. Training BOAs is efficient because we have fixed the type of features we are trying to learn, rather than attempting to relearn what those concepts should be.

Even though we have described BOAs without referring to traditional neural network modeling methods, it is possible to view the BOA as a specialized multi-layered recurrent neural network. Fig. 3 illustrates an alternative view of the BOA model consisting of three layers. The bottom ξ-layer captures the features of the generators, Σ_g. The MM-model's moment map is captured in the middle layer of Fig. 3. Finally the MF-model is captured by the top q-layer in the figure. Because the q-layer and ξ-layer are modeling state-space dynamics, these are actually recurrent layers. In this regard, one may therefore view the BOA as a specialized recurrent neural network whose layers are trained in a manner that allows them to be concretely interpreted with respect to the behavioral decomposition described above.

The last important feature we wish to emphasize regarding BOA modeling is that it provides a powerful and flexible way of formulating *robust* control systems. In particular, the BOA's MF-model may be viewed as a model for the augmented plant of the generalized regulator shown in Fig. 4. Generalized regulators are canonical feedback structures used to formulate robust optimal control problems [98]. The augmented plant is formed by augmenting the original plant, Σ_0, with dynamical systems that characterize how we want the controlled plant to behave. A state feedback controller forces the plant to track the steady-state response predicted by the MM-model.

This augmented plant, therefore has three inputs; disturbances $w(t)$, from the generator, Σ_g, the base control from the policy, Σ_c, and auxiliary control inputs, $\tilde{u}(t)$. The augmented plant has two outputs; a loss signal, $z(t)$, that measures the controlled system's performance and the MF-error signal, $\tilde{y}(t)$, that is used by the feedback controller to generate the auxiliary input, $\tilde{u}(t)$. The controller shown in Fig. 4 is designed from the MF-model, $\widehat{\Sigma}_{\rm mf}$. It is a state-based feedback control that uses delay embeddings [32] of the MF-error, $\tilde{y}$, to reconstruct the MF-system's controllable states. Because the MF-model is a linear state space realization, there are a number of control synthesis methods we can use to manage the closed-loop sensitivity at z to disturbances, w. In essence, this control architecture is regulating the system's MF-error so the steady-state output is tracking that of the MM-model's prediction of the steady-state behavior.

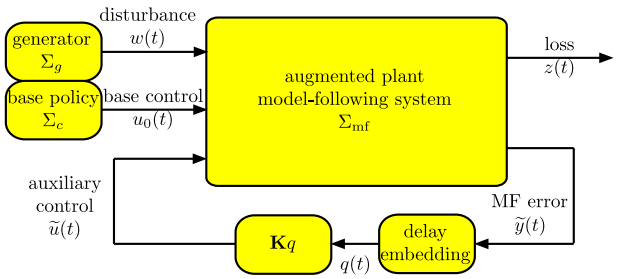

Figure 4: **BOA-based Generalized Regulator:** The MF-system acts as the augmented plant of a generalized regulator. Delay-embedding of the MF-error signal, $\tilde{y}(t)$ forms the MF-system's controllable/observable states which are used by state feedback control to regulate the MF-error, $\tilde{y}(t)$.

Note that we are proposing to first "learn" a BOA for the open-loop plan and then we synthesize the associated controller. This is similar to the strategy used in *model-based* DRL [68]. It is widely acknowledged that model-based DRL is more sample efficient than model-free DRL [64] since one doesn't have to relearn the plant dynamics for each new task the system performs. What model-based DRL, however, does not provide are guarantees on the robustness of the learned control policy. The BOA models can provide such guarantees on robust stability and performance when we embed them in the generalized regulator shown in Fig. 4. In particular, rather than seeing BOA-based control as an alternative to DRL, we should think of it as providing a way to robustly stabilize an existing base control policy. That base policy could have been learned using DRL or it could have been synthesized using model-predictive control [9]. In this regard BOA-based control may provide a mature theory for robustly stable DRL-based control policies.

2.2 Preliminary Results for BOA-based Control: This subsection describes prior work demonstrating that training BOA models for control can be done in a sample efficient and online manner. This prior work used BOAs to control information flows in wireless communication networks [20] and to control hopping robots traversing changing terrain [48]. This prior work suggests that BOAs provide a good basis for robustly controlling the cyber and physical fabrics of CPS.

2.2.1 BOA-based Flow Control in Wireless Networks:. Preliminary results presented at EMSOFT 2021 [20] demonstrated that one could use BOA-based methods to learn how to control the information flows in a wireless network forming the cyber fabric of an IoT-enabled system. The generalized regulator for this application is shown in Fig. 5 where the plant is a mesh wireless network connecting machines on the factory floor. The base control, $u_0(t)$, is a primal-dual flow controller [57] used in wired TCP/IP networks [31]. This base control is realized as a distributed policy in which users generate flow rates and a network manager generates " prices" to keep the total link traffic below a known capacity limit. The network is subject to multiuser interference because of the network's wireless (RF) nature. This means that network links may randomly drop packets whose subsequent retransmission increases the flow rate. Fig. 5 shows packet drops as external disturbances, $w(t)$. The loss, $z(t)$, is the network's QoS measured by the average utilization of all network flows.

We used a simulation of the wireless network to learn a BOA whose MF-model captured the error between the MM-model's predicted QoS and the plant's actual QoS. As discussed above, delay embeddings of the MF error were used to reconstruct the system's state vector, $q(t)$, and we then used a linear quadratic regulator (LQR) for the feedback control gain, $\mathbf{K}$, shown in the block diagram. The top plot in Fig. 5 shows the MF-error for a simulation run. In the first half of this run, the BOA was trained in an online manner with the auxiliary control, $\tilde{u}(t)$, set to zero. The plot shows an initial transient that stabilizes to a steady-state error varying between ± 0.5. At the end of the training period, the LQR control gain is synthesized from the BOA's MF-model and we then turn on the auxiliary control. Fig. 5's plot shows that the controlled system was very effective in regulating the MF-error well below what was observed using the base control policy alone. These results therefore demonstrated that BOA-models could be trained in a fast online manner to regulate the information flows in an IoT system's wireless cyber fabric.

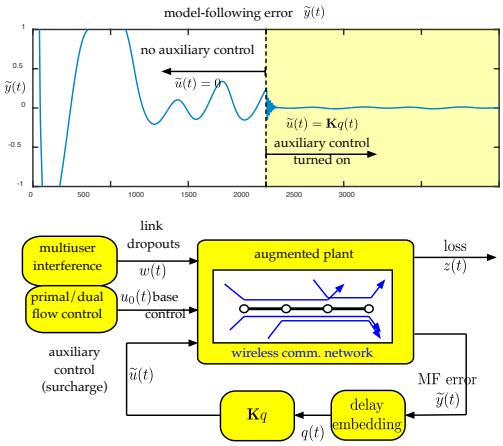

Figure 5: Prior results [20] demonstrated BOA-based control of information flow rates in a cyber-fabric formed from a wireless RF communication network.

2.2.2 BOA-based Adaptive Control of Robotic Hopper: Preliminary work in [48] trained BOAs used to adapt the step-length controller [22] for Raibert's hopper [75] to changes in terrain slope and hopper dynamics. This is an example of BOA-based methods being used to control a particular type of CPS known as a *hybrid system* [49]. We considered a variation of the standard hopper that had an additional spring connected link that would cause the robot's head to *bobble*. We initially trained the BOA on a stiff-headed hopper (large spring constant) as it traversed uneven terrain. The learned BOA was used to synthesize a passivity based control for the MF-system. The resulting controlled system was then tested on the bobble-headed hopper (small spring constant).

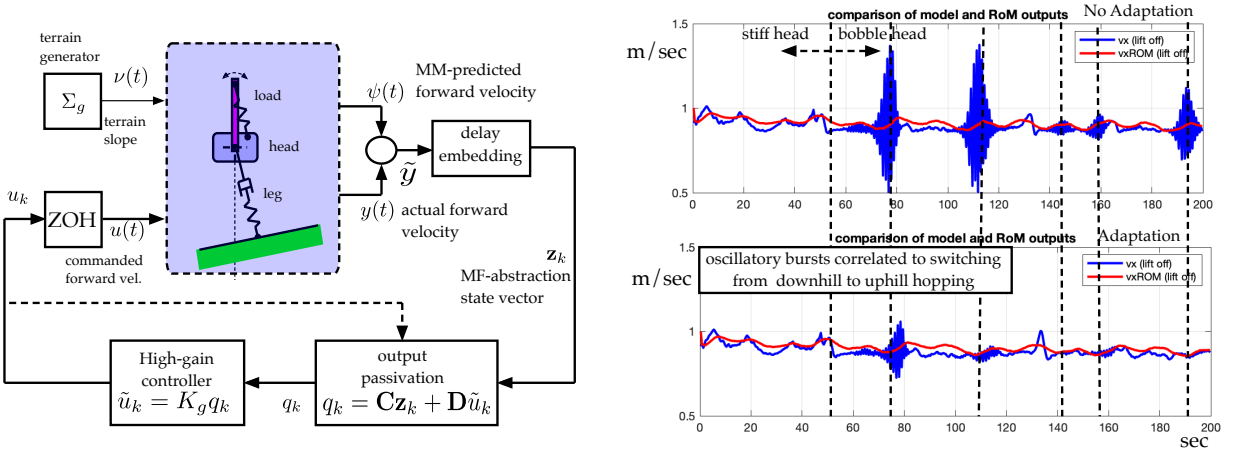

Figure 6: Prior work [48] demonstrated that BOA-based learning could be used to adaptively control Raibert's hopper. The example was initially trained on a stiff-headed hopper (top plot) and then used the MIT rule to adapt that controller to effectively control a bobble-headed hopper (bottom plot).

The left side of Fig. 6 shows the block diagram for the controlled hopper. The top plot on the right side of Fig. 6 shows the hopper's forward velocity with the learned controller and no adaptation. Prior to 50 seconds, we are controlling a stiff-headed hopper and after 50 seconds we switch

to the bobble-headed hopper. The bold vertical dashed lines mark times when the hopper transitions from hopping downhill to hopping uphill. The plot shows that the head begins to bobble every time the hopper changes its hopping direction. The bobble head results in an oscillatory burst in the hopper's forward velocity. While the passivity based control ensures these oscillations do not cause the hopper to fall down, the hopper does stumble awkwardly since its forward velocity clearly exhibits large deviations away from the commanded forward speed.

This preliminary work also demonstrated that a BOA-based passivity controller could adapt to the bobbling head. We demonstrated this adaptive capacity by using the fact that passivity based controllers are high gain controllers. This means that closed loop system performance can be improved by simply increasing the control gain. We used a simple gradient-based adaptation rule to increase the control gain, K_g, every time an oscillatory burst was detected. There are many ways of realizing this gradient based rule, but the simplest was to use the MIT rule [62].

The bottom plot in Fig. 6 shows what happens when the MIT rule was used to adjust the feedback gain. The plot shows that after every oscillatory burst, the subsequent bursts are smaller. If we think of hopping with a stiff head and hopping with a bobble head as two distinct tasks, then the result demonstrates the ability to transfer, or rather "adapt", control knowledge from the stiff-headed task to the bobble-headed task. This is precisely what one is looking for in meta-learning. It suggests that BOAs provide a model for meta-learning that may be better at switching between tasks than previous meta-DRL methods used to control robotic systems [68].

2.3 Proposed Research Activities

The preliminary work in section 2.2 only demonstrates that it is feasible to use BOA models to robustly stabilize complex CPS. This prior work, however, does not create an engineering system for learning these BOA models, it did not explore the limitations of BOA-based modeling, and it did not directly demonstrate that BOA-based control could indeed be used to improve the robust stability of DRL base control policies. The following research activities (tasks) will address these deficiencies in the prior work

1. **Task 1 - Modeling Nonstationary Environments:** While the prior work considered time-varying disturbances, these generators were stationary in the sense that the number and frequency of the disturbance harmonics did not change. In addition to this, each MM-model may be seen as capturing the steady-state behavior on a specific basic limit set. Complex dynamical CPS may have multiple basic limit sets and this means one must have a way of searching for this basic limit sets. This task will develop algorithms for generating a *BOA database* that captures the full environmental complexity we expect to find in real-life CPS.
2. **Task 2 - Coordinated Online BOA Training:** The preliminary work learned BOAs in a sequential episodic manner. We first identified the harmonics in the generator, Σ_g and then we learned the MM-model. That MM-model was used to construct an MF-system that was then learned by a DMDc algorithm and finally that model for the MF-system was used to synthesize a controller. This episodic approach does not provide an agile way of training BOAs in a nonstationary environment. This task will develop sample efficient online methods that avoid using this episodic approach to training.
3. **Task 3 - BOA-based Control for Multiple Tasks:** CPS such as the IoT manufacturing system in Fig. 1 are complex because they perform a number of different tasks. These tasks may be seen as abrupt changes in the base control policy, Σ_c, or even in the plant, Σ_0. Shifts in the base control may arise due to planned schedule changes in workflows. Shifts in the plant may occur due to automatic reconfiguration of the cyber-fabric or faults in the physical plant. The main challenge in this task involves finding ways to easily adapt previously trained BOA-models and controls to such shifts. This task proposes addressing that challenge by

using *meta-learning* [23] for adapting previously trained BOAs.

4. **Task 4 - Robust Stabilization of DRL Policies:** The prior work involving the robotic hopper demonstrated that one could learn BOAs that would serve as the basis for robustly stabilizing a control policy with respect to passive shifts in the plant dynamics. Such demonstrations, however, do not analytically characterize the robust stability guarantees that we believe BOA-based control will extend to DRL derived base control policies. This task will use *approximate simulation* concepts [19] to derive formal guarantees on using BOA-based control for the robust stability of DRL policies.

The following subsections describe these research activities in greater detail. Fig. 7 shows the Gantt chart for the research tasks and the evaluation tasks.

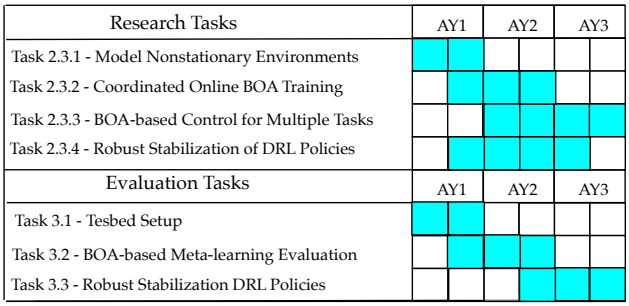

Figure 7: Proposed Project Schedule

2.3.1 Research Task 1 - Modeling Nonstationary Environments: Our prior work assumed the generator dynamic had distinct harmonics that did not change over time. In reality, we expect the generator to change due to environmental shifts and this means we will need to develop methods for training BOAs when the external environment changes. This task proposes using unsupervised learning methods to build a *database* of BOA models that can be used in switching between different controllers for the plant. Unsupervised learning takes unlabeled data and groups it into clusters of "similar" elements and then identifies "keys" that serve as a canonical example of a given cluster's elements. We propose storing this clustered data in a relational database using the given "keys" for context sensitive data retrieval. The algorithm to be developed here will serve as the database' *storage engine*. We will use MySQL for the database' *search engine*. This task therefore will develop software implementing a *BOA database* that will be later used in the project's evaluation tasks (section 3).

Critical research issues in building the *BOA database storage engine* revolve around the particular clustering algorithm being used and the type of similarity metric used to form clusters. We propose using a similarity metric that places two BOAs in the same clusters if their MF-errors, $\tilde{y}(t)$, are close to each other. Such norms on $\tilde{y}(t)$ should be useful similarity measures since we already know that the plant is an *approximate simulation* [19] of the MM-model. While there are numerous clustering algorithms available, we propose to start by using graph-clustering techniques [89] since they are known to scale well for image segmentation [79] but also because we've had success using them to identify basic sets of complex dynamical systems [47]. We also intend to compare graph-clustering against competing methods; KNN algorithms, hierarchical clustering, and barcode methods [17]. The main deliverable from this task will be software components for the BOA database engine as well as analysis characterizing how well the graph-clustering methods are at capturing the nonstationarity we expect to see in real-life CPS applications.

2.3.2 Research Task 2 - Coordinated Online BOA Training:. This task will develop algorithms used for online BOA training. These algorithms will address implementation issues with the episodic training that we used in the preliminary results of section 2.2. This task's activities concern the development of a data-driven online DMDc algorithm and the use of recurrent backpropagation for the coordinated training of the BOAs MM and MF models.

Our preliminary work in section 2.2 used the DMDc algorithm [74] to train MF-models. The DMDc algorithm is usually implemented in a batch manner. Converting the DMDc algorithm into an online algorithm should be possible since it is basically solving a regression problem.

The most computationally intensive part of the algorithm is the computation of the data matrix' singular value decomposition (SVD). Recursive updating of an SVD can be done through matrix perturbation methods. We believe this can be done using a recent recursive algorithm used for low-rank modifications of the SVD of "thin" data matrices [4].

This fast SVD algorithm will also be useful in addressing another issue faced by the online training of MF-models. Prior work in section 2.2 assumed the auxiliary test inputs, $\tilde{u}(t)$, were scalar valued. Clearly many of the CPS systems we are interested in controlling will have vector-valued test inputs and this means we must identify the "best" direction along which these vector-valued test inputs should be injected. The "best" direction is the one that is maximally coupled to the system's internal states and it may be determined from an SVD of the system's controllability matrix. Recent work in [63] developed data-driven controllability tests and we propose using these data-driven tests along with the fast SVD algorithm to track those test input directions maximizing the degree of controllability defined with respect to the system's gramians [60]. Our exponentially decaying test signals will then be injected along those "best" directions and the resulting output will be used in the online DMDc algorithm described in the preceding paragraph. The main deliverable from this part of the task will be an online DMDc algorithm that automatically selects its test auxiliary signals to maximize the degree of controllability.

This task will also develop methods that avoid the sequential episodic manner in which BOAs were trained in the preliminary work. While one can avoid this sequential update strategy by simply separating the timescales at which the MM and MF-models are trained, this approach requires some prior assumptions about time scale separation that may difficult to justify for complex CPS. We will therefore consider an alternative approach based on the earlier observation that every BOA may be viewed as a multilayer recurrent network (see. Fig. 3). This observation suggests that some variation on classical backpropagation [77] may be used. The main wrinkle with this is that our BOAs are *recurrent* networks and this means we would either use backpropagation through time (BPTT) [93,94] or recurrent backpropagation (RBP) [2,73]. At this point we are leaning toward starting with RBP because BPTT requires an unfolding of the recurrent network into a feedforward structure. We worry that this unfolding may not scale well for our current systems and so our initial impulse is to start using RBP for training the BOA. The main issues for RBP will be the stability of the learning process. But because our models have a linear structure, we believe we can get useable bounds on RBP stability using classical diagonal dominance arguments [80] on Lyapunov-like functions that verify the learning process is *contractive* [56]. This direction is similar to what recently appeared in [8] for recurrent networks. The main deliverable from this part of the task will be algorithms that allow coordinated online training of the MM and MF models of a BOA with analytical guarantees on learning convergence rates that can be used to help control how the RBP learning methods are implemented.

2.3.3 Research Task 3 - BOA-based Control for Multiple Tasks: DRL or BOA-based learning should all be able to generalize beyond their training data in the sense they still effectively regulate the plant against small perturbations of the generator provided the base control policy and the plant doesn't change. This assumption is unrealistic for the class of CPS found in IoT-Manufacturing. In reality, the cyber fabric of the IoT system will change due to shifts in network topology or shifts in load balancing strategies. In the IoT factory's physical fabric, there will be similar shifts that may greatly modify the dynamics in that fabric. In particular, we attribute all of these shifts to changes in the *task* that the system is performing. So while we expect well trained models (BOA or DRL) to generalize with respect to *in-distribution* data (i.e. changes in the environment), we do not expect similar robustness with respect to *out of distribution* data (i.e. changes in the base policy or plant). Since such "out of distribution" shifts will occur in IoT-enabled CPS,

this task will explore the use of *meta-learning* to *adapt* a previously trained BOA control system so it works well when the system performs a task that was not in the original training data.

Meta-learning [23] or "learning to learn" [86] has emerged as a powerful way to generalize between tasks. Meta-learning extends classical machine learning by parameterizing the model in terms of *task parameters* and *meta parameters*. These two types of parameters are selected by two different types of learning agents. The *meta-learner* is an agent that selects meta parameters which minimize the model's loss when it is averaged over all tasks the system might perform. The meta parameter represents a family of models and so one may think of the meta-learner as configuring how the classical machine learning problem is to be solved. The second agent solves a classical machine learning problem and is called the *task-learner*. For a given meta parameter, the task learner selects task parameters minimizing the model's loss on the system's *current* task.

One may formalize the preceding description as a bilevel optimization problem [23] of the form,

$$\begin{aligned} \theta^* &= \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}^{\text{meta}}(\phi_i^*(\theta), \theta; \mathcal{D}_i^{\text{test}}) \\ \phi_i^*(\theta) &= \arg\min_{\phi} \mathcal{L}^{\text{task}}(\phi, \theta : \mathcal{D}_i^{\text{train}}) \end{aligned} \tag{2}$$

The upper problem is solved by the meta-learner agent and selects a meta-parameter, θ^*, minimizing a meta loss function, $\mathcal{L}^{\text{meta}}(\theta, \phi)$, averaged over a test data set, $\mathcal{D}^{\text{test}}$, containing representatives of all tasks being performed. The lower problem is solved by the task-learner agent and selects a task-parameter, $\phi_i(\theta)$, that minimizes the task loss function, $\mathcal{L}^{\text{task}}(\theta, \phi)$, for the ith task's training data, $\mathcal{D}_i^{\text{train}}$, with respect to a fixed meta parameter θ.

This task's approach to *meta-learning for control* integrates the bilevel optimization framework in equation (2) with the BOA-based generalized regulator shown in Fig. 4. In particular, we take the plant's BOA *and* associatd control gains, $\mathbf{K}$, as the model's *task parameters*, ϕ. An important fact that is not always well appreciated outside of the controls community is that controller synthesis is a recursive process in which the designer has to search for the best way of configuring the synthesis problem. The best known example of this is seen in the design of linear quadratic controllers where the designer selects a pair of weighting matrices that trade off the control effort expended against the controlled system's regulation error. The choice of these weighting matrices has a profound impact on the robustness of the controlled system [10] and so weight selection is a critical part of the design process. This project takes these weighting matrices as the meta parameters in the bilevel framework in equation (2). Our proposed approach, therefore, has the meta-learner determine the best way of trading off control effort and regulation in a manner that is *fair* to all tasks. The task-learner then uses that configuration of the synthesis problem to select the optimal control gains for the current task.

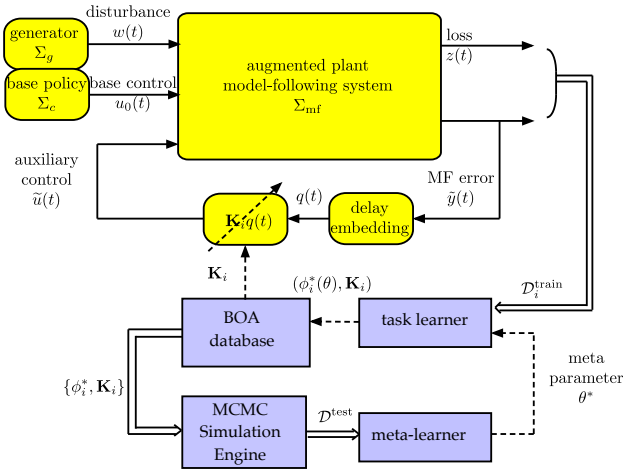

Figure 8: The BOA-based meta-learning framework applies meta-learning's bilevel optimization framework (blue blocks) to the generalized regulator (yellow blocks) whose augmented plant is formed from the BOA's MF-system.

We can then merge the BOA-based generalized regulator and the bilevel meta-learning frame-

work in equation (2) as shown Fig 8. In particular, let us assume the controlled system is currently performing task i. This task's training data, $\mathcal{D}_i^{\text{train}}$, consists of the augmented plant's outputs, z and $\tilde{y}$, and the task learner uses $\mathcal{D}_i^{\text{train}}$ to adaptively learn an optimal BOA and then select control gains, $\mathbf{K}_i$, that minimize the regulation error for this task. As mentioned above, that regulation problem is configured by the meta parameters, θ^*, trading off control effort against regulation performance. The meta-learner selects these meta parameters with respect to a test data set, $\mathcal{D}^{\text{test}}$, that is generated by Markov-Chain Monte Carlo (MCMC) simulations drawing from the BOA database of models and control gains that were obtained by the task learner in previous tasks. The MCMC sim-engine will generate simulation runs for all tasks that the system previously performed.

The main deliverable from this task will be software realizing a *distributed* implementation of the meta-learning framework in Fig. 8. The proposed work will build an MCMC simulation engine that uses the BOA database from research task 2.3.1 to generate testing data for the meta learner. This testing data must include representatives from all tasks that the CPS system has performed. We propose using MCMC methods to efficiently and fairly sample the BOA database for BOAs representative of all tasks the system has previously performed. The MF-models retrieved by the MCMC sim-engine will be used to automate the generation of computational simulation models of the MF-system with randomly generated auxiliary control inputs along those directions with the maximum degree of controllability. The resulting simulation outputs will then be added to the testing data set, $\mathcal{D}_{\text{test}}$. The main deliverable from this task will be the software components generating the data sets used by the meta-learner.

The research issue to be addressed in this software development task concerns the fact that IoT-enabled CPS will have thousands of interacting subsystems in the cyber and physical fabrics. This means that it will be impractical to field a software platform whose task-learner and meta-learner are realized in a centralized manner. The raw information used by task learner will be generated across edge devices. The BOAs and controllers trained by our meta-learning framework will be geographically dispersed throughout the IoT system's cyber and physical fabrics. This means that the task and meta learners will need to be implemented in a distributed manner.

This problem is relevant to *federated machine learning* [42]. Federated learning is concerned with fusing edge data into a centralized model that usually takes the form of a feedforward neural network. There has already been extensive work focused on federated learning's statistical issues [97] and privacy issues [87]. This research task will pivot to focus on a federated learning paradigm that fuses edge data to train models that are distributed across the IoT manufacturing system's cyber and physical fabrics. Based on our prior work with distributed optimization [90], networked control [27], and consensus [50], we will explore algorithms that make use of *social learning* protocols to train distributed BOA models. We are aware that there has been a great deal of prior work using distribution schemes based on consensus protocols [70], but we also know these methods can lead to models that are unfairly biased by the data from edge devices with the greatest out-degree connectivity [50]. One way that has recently emerged to address this issue is through social learning [29] protocols (rather than consensus protocols [71]). Recent work has begun to formally explore the convergence properties of social learning [88]. This task's research will explore the extent to which social-learning algorithms can be used as a basis for developing distributed software implementations of the meta-learner framework in Fig. 8. This task will first build a centralized meta-learning software that can be quickly used by section 3's evaluation tasks. This task's main deliverable will be a distributed implementation of the meta-learning software based on social learning algorithms.

2.3.4 Research Task 4 - Robust Stabilization of DRL Policies: While Deep Reinforcement Learning (DRL) has provided impressive demonstrations controlling millipede robots over changing

terrain [67], it is widely known that these policies are not robustly stable [37]. When the plant models are linear, then one can establish the stability of traditional reinforcement learning policies [81] using adaptive dynamic programming (ADP) [51]. For nonlinear discrete-time plants one can use the methods in [21]. But neither of these works can be extended to deep neural networks in a way that provides a mature theory for the robust stability of DRL-derived control policies.

In our opinion BOA-based modeling provides a way to obtain theoretical guarantees on the robust stability of DRL control policies. The main reason for this belief is that we already know that the plant is an *approximately simulation* [19] of the moment matching (MM) model. An approximate simulation relation means that we can bound the model-following error, $\tilde{y}(t)$, with respect to bounded perturbations of the disturbances. An approximate simulation relation exists when we embed the MM-model in a hierarchical control system [18] with an appropriately chosen control interface. In particular, the BOA-based control architecture shown in Fig. 4 is that hierarchical control system proposed in [18] and this should provide formal guarantees on the robust stability of the DRL-derived base policy. These formal guarantees will be in the form of bounds on the model following error that can be obtained from simulation functions computed using SoS methods, similar to what was done in [44]. This task's main deliverable will be formal analysis providing guarantees on how BOA-based control can robustly stabilize DRL-derived policies.

2.4 CPS Research Focus: This project's challenge problem is to *learn how to control complex CPS that perform multiple tasks in a changing and uncertain world.* The project's *focus CPS attributes* lie in our target CPS application, the IoT-enabled manufacturing system in Fig. 1. This system has a physical fabric formed by the flow of materials across the factory floor and a cyber fabric formed by the digital communication network controlling the flow of information used to manage flows in the physical fabric. An important CPS attribute of this system is coupled dynamics in both fabrics since congestion in one fabric can trigger congestion in the other fabric and vice versa.

Learning how to control complex CPS such as the IoT manufacturing system will address a number of core CPS research areas in *control*, *real-time learning for control*, *IoT*, and *networking*. The project will develop new real time methods for learning how to control complex IoT systems. These methods will control the IoT system's cyber and physical fabrics in a coordinated real-time manner that is impossible to do using current reinforcement learning methods. The applications targeted by this project will be in the area of IoT-enabled manufacturing, a critical class of CPS system whose optimal management is projected to have a great impact on national economic activity [1, 58]. This suggests that the products delivered by this research may play a significant role in expanding the role that CPS control methods play in IoT-enabled applications.

3.0 Evaluation and Experimentation Plan:

The fourth Industrial revolution (Industry 4.0) [45] refers to rapid change in the manufacturing sector driven by Internet-of-Things (IoT) technologies. The project's evaluation and experimentation plans revolve around a hardware testbed inspired by the IoT manufacturing system shown in Fig. 1. The physical fabric is formed from webcams and autonomous mobile robots that move along specified routes in the physical workspace. The cyber fabric is formed from a mesh ad hoc wireless 5 GHz network connected to the Internet and accessed through WiFi adaptors attached to each robot and webcam. This testbed may be seen as a simplified version of the IoT-manufacturing system in Fig. 1. This simplified testbed would have autonomous WiFi connected robots moving through the hallways of the engineering building.

The purpose of this testbed is not to duplicate how autonomous robots might be used in a real-life manufacturing 4.0 environment. The development of such a realistic industrial testbed is expensive and well beyond the scope of this project. The purpose of this testbed is to realistically capture the unpredictable variations in connectivity and message latency that are known to exist in

real-life mobile ad hoc networks. This real-life uncertainty cannot be accurately replicated through computer simulation models due to uncertainties in the indoor RF environment. The project's testbed, therefore, will provide a way to generate large data sets that are representative of the unpredictable task and environmental shifts we expect to see in real-world IoT-manufacturing systems. The proposed plan will use these data sets to experimentally evaluate how well BOA-based meta-learning performs relative to competing meta-versions of deep reinforcement learning. The plan will be carried out through three related tasks that 1) setup the testbed, 2) experimentally evaluate the performance of the BOA-based software in learning how to regulate traditional job shop scheduling and Internet rate control algorithms, and 3) experimentally evaluate how well BOA-based meta-learning can robustly stabilize DRL-based flow-control policies in the IoT system's cyber and physical fabrics.

Evaluation Task 3.1 - Testbed Setup:. This subtask will build the proposed testbed in the project's first year. The physical fabric will consist of five turtlebot robotic platforms equipped with laptop, 2d laser scanner, camera system, and WiFi adaptors supporting IEEE 802.11ac standards. We expect this to cost $20,000 and that cost has been included in the project's first year budget. The physical fabric will be built to emulate the movement of robots ferrying parts across a factory floor. The movement of these robots will be decided by a centralized planner using data gathered from edge devices over the wireless RF network. While the centralized plan schedules robot movements based on global information about customer orders, the implementation of that plan is left to the robots. This means that the robots may deviate from the schedule should local situations (i.e. congestion) require it. This configuration will capture shifts in workflow dynamics as a result of scheduling decisions moving between the centralized planner and edge devices.

The cyber fabric will be configured to host at least two different network configurations. One configuration will have all robots connect to a traditional WiFi access point (AP) from which they can access the job shop planner. The other configuration will support ad hoc networking based on the WiFi Direct standard. WiFi Direct [16] enables peer to peer communication without an access point. This is done by forming groups of users with a single user (device) acting as the group owner (GO). The GO supports some of the traditional AP functionality associated with assigning IP addresses and broadcasting. This is done to avoid flooding issues that plagued earlier mesh networking based on the 802.11s standard [91]. The use of multiple network configurations will allow us to study how shifts in network topology impact flow rates.

Evaluation Task 3.2 - Evaluating BOA-based Meta-Learning Performance:. This subtask has two objectives. It will first build the base controller for both the physical and cyber fabrics. At this point, we intend to implement flow controllers similar to the primal-dual controller used in our preliminary work (see Fig. 5). This will be done for both the cyber and physical fabrics. In the cyber fabric, flow control will be used to determine how much information each robot transmits across the network. In the physical fabric, flow control will be used to adjust the speed with which a given robot traverses the route assigned to it by the centralized planner. The second part of this subtask will then use the testbed to generate the data sets used to train a BOA for the flows in this combined cyber/physical fabric. The meta-learning software from the task 2.3.3 will be implemented and used to evaluate the testbed performance when there are shifts in the physical fabric (i.e. congestion events) and the cyber fabric (i.e. shifts in network configuration).

Evaluation Task 3.3 - Evaluating Robust Stabilization of DRL Policies:. This subtask has two objectives. We will first use well known DRL toolkits to develop control policies that optimize flow rates in the testbed's physical and cyber fabrics. This policy will be implemented in a centralized computer that gathers information from all edge devices, decides the best global policy and then broadcasts that policy to all edge devices. We will then study the robustness of this DRL policy to

perturbations in the physical and cyber fabric. Perturbations to the cyber-fabric will be introduced through a robot initiating a denial-of-service attack on the network. Perturbations to the physical fabric will be introduced through intermittent obstacles that create congestion along robot paths. The objective will be to experimentally evaluate how large of a disturbance is needed to cause a catastrophic breakdown in the DRL policy. We will use the robust stability results from Task 2.3.4 to predict stability margins for the DRL policy. These stability margins will be experimentally evaluated on the testbed. We will then experimentally assess the extent to which BOA-based controls enlarge the stability margin of the DRL-derived base control policy.

4.0 Project Management and Collaboration

Prof. M.D. Lemmon will direct the project. Dr. Lemmon is an expert in networked control systems. The PI will meet weekly with project students where students present their work and receive the PI's feedback. These meetings will facilitate regular review of the project schedule to keep track of progress on project deliverables. We are budgeting for two Ph.D. students to assist with the work.

5.0 Broader Impacts:

Community Outreach: The project will use ND's Research Experience for Teachers in Engineering program (EngRET@ND) for outreach to local high schools. This outreach would host a high school teacher for four weeks to work alongside graduate students. Later followup will assess the extent to which the teacher has integrated IoT concepts into the high school curriculum.

Broadening the Participation of Women in Computing: This project will broaden the participation of women in computing through an existing partnership with Saint Mary's College. Saint Mary's is a four-year women's liberal arts college that is one mile from Notre Dame. The percentage of women at Saint Mary's graduating with STEM degrees is twice the national average. One opportunity available to Saint Mary's students is the 4+1 dual-degree program that is offered in partnership with Notre Dame. As part of this program, students earn a degree of their choice from Saint Mary's in four years while earning an engineering degree from Notre Dame in the fifth year. The PI will apply for REU supplements to host Saint Mary's students. The PI will work with Saint Mary's program coordinator to identify students that are a good fit for this proposed project. This activity's success will be measured by the number of Saint Mary's students joining the PI's REU program and the ultimate placement of these students once they graduate.

Curriculum Development: Prof. Lemmon is the director of graduate studies for Notre Dame's department of electrical engineering. He is currently working with Notre Dame's wireless institute to stand up an in-person professional MS degree that would focus on CPS systems with an emphasis on IoT. The program is similar to others across the country (Vanderbilt). The novelty from other programs rests with our leadership in 5G wireless technologies for IoT applications. We expect to stand up the program in this project's second year.

5.0 Results from Prior NSF Sponsored Research:

Lemmon was PI of NSF project CNS-1239222 CPS: Synergy: Resilient Wireless Sensor-Actuator networks, $1,000,000, 9/2012–09/2016. *Intellectual Merits*: This project developed resilient wireless sensor-actuator networks through the coordinated management of the control system and the associated communication infrastructure. The approach rested on integrating innovations in event-triggered control and machine-to-machine communication. *Broader Impacts*: The project broadened its impact through Notre Dame's Wireless Institute, the Midwest Control and Game Theory workshop, the development of a multi-robot research lab, and the graduation of 6 Ph.D. students. Publications: 18 conference papers [24–26, 33, 35, 38–41, 46, 52, 53, 65, 82–84, 92, 95], and 6 journal papers [27, 34, 36, 47, 54, 96].

References

[1] Shadi Al-Sarawi, Mohammed Anbar, Rosni Abdullah, and Ahmad B Al Hawari. Internet of things market analysis forecasts, 2020–2030. In *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pages 449–453. IEEE, 2020.

[2] L.B. Almeida. A learning rule for asynchronous perceptrons with feedback in a combinatorial environment. In *IEEE International Conference on Neural Networks*, pages 609–618, 1987.

[3] Alessandro Astolfi. Model reduction by moment matching for linear and nonlinear systems. *IEEE Transactions on Automatic Control*, 55(10):2321–2336, 2010.

[4] Matthew Brand. Fast low-rank modifications of the thin singular value decomposition. *Linear algebra and its applications*, 415(1):20–30, 2006.

[5] S.L. Brunton, B.W. Brunton, J.L. Proctor, and J.N. Kutz. Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PLoS ONE*, 11(2-e0150171), 2016.

[6] Vladimir Cherkassky, Xuhui Shao, Filip M Mulier, and Vladimir N Vapnik. Model complexity control for regression using vc generalization bounds. *IEEE transactions on Neural Networks*, 10(5):1075–1089, 1999.

[7] Charles C Conley. *Isolated invariant sets and the Morse index*. American Mathematical Soc., 1978.

[8] Alexander Davydov, Anton V Proskurnikov, and Francesco Bullo. Non-euclidean contractivity of recurrent neural networks. *arXiv preprint arXiv:2110.08298*, 2021.

[9] Jared Di Carlo, Patrick M Wensing, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. In *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 1–9. IEEE, 2018.

[10] John Doyle. Guaranteed margins for lqg regulators. *IEEE Transactions on Automatic Control*, 23(4):756–757, 1978.

[11] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[12] Robert W Easton. *Geometric methods for discrete dynamical systems*. Oxford University Press on Demand, 1998.

[13] Harald Edquist, Peter Goodridge, and Jonathan Haskel. The internet of things and economic growth in a panel of countries. *Economics of Innovation and New Technology*, 30(3):262–283, 2021.

[14] Nicholas Fearn. Germany exceeds us in iot investment. *Control Engineering*, 63(8):25–26, 2016.

[15] Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot visual imitation learning via meta-learning. In *Conference on robot learning*, pages 357–368. PMLR, 2017.

[16] Colin Funai, Cristiano Tapparello, and Wendi Heinzelman. Enabling multi-hop ad hoc networks through wifi direct multi-group networking. In *2017 International Conference on Computing, Networking and Communications (ICNC)*, pages 491–497. IEEE, 2017.

[17] Robert Ghrist. Barcodes: the persistent topology of data. *Bulletin of the American Mathematical Society*, 45(1):61–75, 2008.

[18] Antoine Girard and George J Pappas. Hierarchical control system design using approximate simulation. *Automatica*, 45(2):566–571, 2009.

[19] Antoine Girard and George J Pappas. Approximate bisimulation: A bridge between computer science and control theory. *European Journal of Control*, 17(5-6):568–578, 2011.

[20] Xiaolong Guo, Song Han, X Sharon Hu, Xun Jiao, Yier Jin, Fanxin Kong, and Michael Lemmon. Towards scalable, secure, and smart mission-critical iot systems: review and vision. In *Proceedings of the 2021 International Conference on Embedded Software*, pages 1–10, 2021.

[21] Pingan He and Sarangapani Jagannathan. Reinforcement learning neural-network-based controller for nonlinear discrete-time systems with input constraints. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 37(2):425–436, 2007.

[22] J.K. Hodgins and M.H. Raibert. Adjusting step length for rough terrain locomotion. *IEEE Transactions on Robotics Automation*, 7(3):289–298, 1991.

[23] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*, 2020.

[24] B. Hu and M.D. Lemmon. Distributed switched supervisory control to achieve amost sure safety for a class of interconnected networked systems. In *IEEE International Conference on Control and Automation (ICCA)2*, Taichung, Taiwan, 2014.

[25] B. Hu and M.D. Lemmon. Distributed switching control to achieve resiience to deep fades in leader-follower non-holonomic systems. In *Conference on High Confidence Networked Systems (HiCONS)*, Berlin, Germany, 2014.

[26] Bin Hu and M.D. Lemmon. Using channel state feedback to achieve resilience to deep facesin wireless networked control systems. In *Proceedings of Conference on High Confidence Networked Systems*, Philadelphia, PA, 2013.

[27] Bin Hu and Michael D Lemmon. Distributed switching control to achieve almost sure safety for leader-follower vehicular networked systems. *IEEE Transactions on Automatic Control*, 60(12):3195–3209, 2015.

[28] Alberto Isidori and Christopher I Byrnes. Steady-state behaviors in nonlinear systems with an application to robust disturbance rejection. *Annual Reviews in Control*, 32(1):1–16, 2008.

[29] Ali Jadbabaie, Pooya Molavi, Alvaro Sandroni, and Alireza Tahbaz-Salehi. Non-bayesian social learning. *Games and Economic Behavior*, 76(1):210–225, 2012.

[30] Yi Jiang, Jialu Fan, Tianyou Chai, and Frank L Lewis. Dual-rate operational optimal control for flotation industrial process with unknown operational model. *IEEE Transactions on Industrial Electronics*, 66(6):4587–4599, 2018.

[31] Cheng Jin, David X Wei, and Steven H Low. Fast tcp: motivation, architecture, algorithms, performance. In *IEEE INFOCOM 2004*, volume 4, pages 2490–2501. IEEE, 2004.

[32] Mason Kamb, Eurika Kaiser, Steven L Brunton, and J Nathan Kutz. Time-delay observables for koopman: Theory and applications. *SIAM Journal on Applied Dynamical Systems*, 19(2):886–917, 2020.

[33] A. Karimoddini and H. Lin. Hierarchcial hybrid symbolic control for robot motion planning. In *Proc. of 10th IEEE International Conference on Control and Automation*, P.R. China, 2013.

[34] A. Karimoddini, H. Lin, B.M. Chen, and T.H. Lee. A bumpless hybrid supervisory control algorithm for the formation of unmanned helicopters. *Mechatronics*, 23(6):677–688, 2013.

[35] A. Karimoddini, H. Lin, B.M. Chen, and T.H. Lee. A smooth hybrid symbolic control for a team of uavs over a partitioned space. In *Proceedings of the American Control Conference*, Washington D.C., 2013.

[36] A. Karimoddini, H. Lin, X. Dong, F. Lin, G. Cai, B.M. Chen, and T.H. Lee. Hybrid modeling and control of the regulation layer of unmanned helicoter. to appear in International Journal of Control, 2013.

[37] Pramod P Khargonekar and Munther A Dahleh. Advancing systems and control research in the era of ml and ai. *Annual Reviews in Control*, 45:1–4, 2018.

[38] M. Khoshnevisan and J. Nicholas Laneman. Intermittent multi-access communication: achievable rates. In *IEEE International Symposium on Information Theory*, Istanbul, Turkey, 2013.

[39] M. Khoshnevisan and J.N. Laneman. Achievable rates for intermittent communication. In *Proc. IEEE Int. Symp. Information Theory*, Boston, 2012.

[40] M. Khoshnevisan and J.N. Laneman. Intermittent communication and partial divergence. In *Proc. Allerton Conf. Communications, Control, and Computing*, Monticello, IL, 2012.

[41] M. Khoshnevisan and J.N. Laneman. Upper bonds on the capacity of binary intermittent communication. In *Workshop on Information Theory and Applications*. Univ. of California, San Diego, 2013.

[42] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[43] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[44] Vince Kurtz, Patrick M Wensing, Michael D Lemmon, and Hai Lin. Approximate simulation for template-based whole-body control. *arXiv preprint arXiv:2006.09921*, 2020.

[45] Heiner Lasi, Peter Fettke, Hans-Georg Kemper, Thomas Feld, and Michael Hoffmann. Industry 4.0. *Business & information systems engineering*, 6(4):239–242, 2014.

[46] M.D. Lemmon. Towards a passivity framework for power control and reponse time management in cloud computing. In *7th International Workshop on Feedback Computing*, San Jose, CA, Sept. 17 2012.

[47] M.D. Lemmon. Achieving ecological resilience through regime shift management. *Foundations and Trends in Systems and Control*, 7(4), 2020.

[48] M.D. Lemmon, P.M. Wensing, H. Lin, and V. Kurtz. Learning to control robot hopping over uneven terrain. In *American Control Conference*, October 2021.

[49] Michael D Lemmon, Kevin X He, and Ivan Markovsky. Supervisory hybrid systems. *IEEE Control Systems Magazine*, 19(4):42–55, 1999.

[50] Michael D Lemmon and Yashan Sun. Cohesive swarming under consensus. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 5091–5096. IEEE, 2006.

[51] Frank L Lewis and Draguna Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE circuits and systems magazine*, 9(3):32–50, 2009.

[52] Lichun Li, Bin Hu, and M.D. Lemmon. Resilient event triggered systems with limited communication. In *IEEE Conference on Decision and Control*, Hawaii, December, 2012.

[53] Lichun Li and M.D. Lemmon. Polynomial approximations of optimal event triggers for state estimation problems using sostools. In *Proceedings of the American Control Conference*, Washington D.C., 2013.

[54] Lichun Li, Xiaofeng Wang, and M.D. Lemmon. Efficiently attentive event-triggered control systems with limited bandwidth. *IEEE Transactions on Automatic Control*, 2017 (to appear).

[55] Tong Lin and Hongbin Zha. Riemannian manifold learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):796–809, 2008.

[56] Winfried Lohmiller and Jean-Jacques E Slotine. On contraction analysis for non-linear systems. *Automatica*, 34(6):683–696, 1998.

[57] Steven H Low and David E Lapsley. Optimization flow control. i. basic algorithm and convergence. *IEEE/ACM Transactions on networking*, 7(6):861–874, 1999.

[58] Denise Lund, Carrie MacGillivray, Vernon Turner, and Mario Morales. Worldwide and re-

gional internet of things (iot) 2014–2020 forecast: A virtuous circle of proven value and demand. *International Data Corporation (IDC), Tech. Rep*, 1(1):9, 2014.

[59] Nguyen Cong Luong, Dinh Thai Hoang, Shimin Gong, Dusit Niyato, Ping Wang, Ying-Chang Liang, and Dong In Kim. Applications of deep reinforcement learning in communications and networking: A survey. *IEEE Communications Surveys & Tutorials*, 21(4):3133–3174, 2019.

[60] Krithika Manohar, J Nathan Kutz, and Steven L Brunton. Optimal sensor and actuator selection using balanced model reduction. *IEEE Transactions on Automatic Control*, 67(4):2108–2115, 2021.

[61] Gary Marcus. Deep learning: A critical appraisal. *arXiv preprint arXiv:1801.00631*, 2018.

[62] Iven MY Mareels, Brian DO Anderson, Robert R Bitmead, Marc Bodson, and Shankar S Sastry. Revisiting the mit rule for adaptive control. In *Adaptive Systems in Control and Signal Processing 1986*, pages 161–166. Elsevier, 1987.

[63] Vikas Kumar Mishra, Ivan Markovsky, and Ben Grossmann. Data-driven tests for controllability. *IEEE Control Systems Letters*, 5(2):517–522, 2020.

[64] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, and Georg Ostrovski. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[65] E. MolavianJazi and J.N. Laneman. A random coding approach to gaussian multiple access channels with finite blocklength. In *Proc. Allerton Conf. Communications, Control, and Computing*, Monticello, IL, 2012.

[66] A Morse. Structure and design of linear model following systems. *IEEE Transactions on Automatic Control*, 18(4):346–354, 1973.

[67] Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018.

[68] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7559–7566. IEEE, 2018.

[69] Kumpati S Narendra and Anuradha M Annaswamy. *Stable adaptive systems*. Courier Corporation, 2012.

[70] Angelia Nedic, Asuman Ozdaglar, and Pablo A Parrilo. Constrained consensus and optimization in multi-agent networks. *IEEE Transactions on Automatic Control*, 55(4):922–938, 2010.

[71] Reza Olfati-Saber, J Alex Fax, and Richard M Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[72] Yash Vardhan Pant, Houssam Abbas, Kartik Mohta, Truong X Nghiem, Joseph Devietti, and Rahul Mangharam. Co-design of anytime computation and robust control. In *2015 IEEE Real-Time Systems Symposium*, pages 43–52. IEEE, 2015.

[73] F.J. Pineda. Generalization of back-propagation to recurrent neural networks. *Physical Review Letters*, 59(19):2229–2232, 1987.

[74] Joshua L Proctor, Steven L Brunton, and J Nathan Kutz. Dynamic mode decomposition with control. *SIAM Journal on Applied Dynamical Systems*, 15(1):142–161, 2016.

[75] Marc H Raibert. *Legged robots that balance*. MIT press, 1986.

[76] Adrian Redder, Arunselvan Ramaswamy, and Daniel E Quevedo. Deep reinforcement learning for scheduling in large-scale networked control systems. *IFAC-PapersOnLine*, 52(20):333–338, 2019.

[77] D.E. Rumelhart, J.L. McClelland, and the PDP Research Group. *Parallel Distributed Processing, explorations in the microstructure of cognition*, volume 1. MIT Press, 1986.

[78] Giordano Scarciotti and Alessandro Astolfi. Data-driven model reduction by moment matching for linear and nonlinear systems. *Automatica*, 79:340–351, 2017.

[79] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *Departmental Papers (CIS)*, page 107, 2000.

[80] D.D. Siljak. *Large-scale Systems Stability & Structure*. North-Holland, 1978.

[81] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[82] T. Tamba and M.D. Lemmon. Using first passage times to manage eco-system regime shifts. In *IEEE Conference on Decision and Control*, Florence, Italy, 2013.

[83] T. Tamba and M.D. Lemmon. Forecasting the resilience of networked dynamical systems under environmental perturbation. Symposium on High Confidence Networked Control Systems Work-in-Progress session (HiCONS), 2014.

[84] T.A Tamba and M.D. Lemmon. The distance-to-bifurcation problem in non-negative dynamical systems with kinetic realizations. IEEE International Conference on Control and Automation (ICCA), 2014.

[85] Le Thanh Tan and Rose Qingyang Hu. Mobility-aware edge caching and computing in vehicle networks: A deep reinforcement learning. *IEEE Transactions on Vehicular Technology*, 67(11):10190–10203, 2018.

[86] Sebastian Thrun and Lorien Pratt. *Learning to learn*. Springer Science & Business Media, 2012.

[87] Stacey Truex, Nathalie Baracaldo, Ali Anwar, Thomas Steinke, Heiko Ludwig, Rui Zhang, and Yi Zhou. A hybrid approach to privacy-preserving federated learning. In *Proceedings of the 12th ACM workshop on artificial intelligence and security*, pages 1–11, 2019.

[88] Cesar A Uribe, Alexander Olshevsky, and Angelia Nedich. Non-asymptotic concentration rates in cooperative learning part i: Variational non-bayesian social learning. *IEEE Transactions on Control of Network Systems*, 2022.

[89] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

[90] Pu Wan and Michael D Lemmon. Event-triggered distributed optimization in sensor networks. In *2009 International Conference on Information Processing in Sensor Networks*, pages 49–60. IEEE, 2009.

[91] Xudong Wang and Azman O Lim. Ieee 802.11 s wireless mesh networks: Framework and challenges. *Ad Hoc Networks*, 6(6):970–984, 2008.

[92] Z. Wang, M. Xia, and M.D. Lemmon. Voltage stability of weak power distribution networks with inverter connected sources. In *Proceedings of the American Control Conference*, Washington D.C., 2013.

[93] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[94] Ronald J Williams and Jing Peng. An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural computation*, 2(4):490–501, 1990.

[95] B. Wu, H. Lin, and M.D. Lemmon. Stability analysis for wireless networked control system in unslotted ieee 802.15.4 protocol. In *IEEE International Conference on Control and Automation (ICCA)2*, Taichung, Taiwan, 2014.

[96] Bo Wu, Michael D Lemmon, and Hai Lin. Formal methods for stability analysis of networked control systems with ieee 802.15. 4 protocol. *IEEE Transactions on Control Systems Technology*,

2017.

[97] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[98] Kemin Zhou and John Comstock Doyle. *Essentials of robust control*, volume 104. Prentice hall Upper Saddle River, NJ, 1998.