2021 American Control Conference (ACC)
New Orleans, USA, May 25-28, 2021

A physics-guided reinforcement learning framework for an autonomous manufacturing system with expensive data

Md Ferdous Alam, *Student Member*, Max Shtein, Kira Barton, *Member*, David J. Hoelzle, *Member*

***Abstract*— Making intelligent decisions is the biggest challenge in building an autonomous manufacturing system that can build artifacts with desired properties without human intervention. Although reinforcement learning (RL) has favorable characteristics for such a task, the sample efficiency of RL is poor, which makes it difficult to implement on a manufacturing system due to the expense of producing parts to collect reward and action data. This paper focuses on building a framework for implementation of RL on manufacturing systems with expensive data and presents the framework for autonomous manufacturing of Phononic Crystals (PnCs), a type of acoustic metamaterial. Leveraging knowledge from physics-guided models and temporal abstraction ideas, we detail a framework that reduces the task of finding optimal manufacturing parameters from thousands of manufacturing samples to the order of 50 samples. The method is applied in simulation to a stochastic model of PnC production. Critically, we show that by using a long temporal abstraction horizon and order of 50 sample budget, the RL algorithm finds the optimal region greater than 95% of the time.**

I. Introduction

Modern manufacturing systems can benefit by applying data driven machine learning (ML) methods [1]. The design of many manufacturing processes is still a very *ad hoc* process in which engineers build extension design of experiments type independent parameter selection protocols to key in on parameter sets that optimize a given production objective [2]. Furthermore, the selected parameter set is often highly dependent on a multitude of uncontrollable factors, such as material feedstock characteristics and environmental conditions, requiring the process engineer to invest in expensive feedstock qualification studies and environmental control chamber, or perform new experiments to re-optimize the parameter set every time driving factor is uncontrolled. We along with others [3]–[5] believe that the next revolution in industrial applications will be the integration of cognitive elements that will autonomously optimize a process, and be continually applied to constantly update parameter sets.

Final manuscript submitted on March 22, 2021. This work was supported in part by NSF Award CMMI-1727894

Md Ferdous Alam is with the Department of Mechanical and Aerospace Engineering at the Ohio State University, Columbus, OH, 43210 USA (email: alam.92@osu.edu).

Max Shtein is with the Department of Materials Science and Engineering at University of Michigan, Ann Arbor, Michigan 48109 USA (email: mshtein@umich.edu).

Kira Barton is with the Department of Mechanical Engineering at University of Michigan, Ann Arbor, Michigan 48109 USA (email: bartonkl@umich.edu).

David J. Hoelzle is with the Department of Mechanical and Aerospace Engineering at the Ohio State University, Columbus, OH, 43210 USA (phone: +1 (614) 688-2942; email: hoelzle.1@osu.edu).

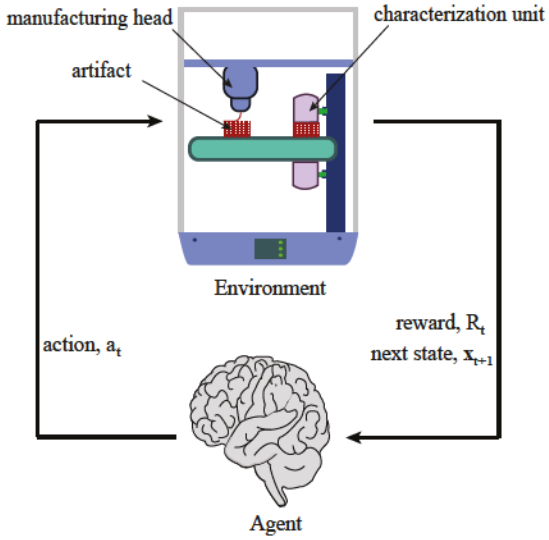


Fig. 1. Reinforcement learning framework for the proposed autonomous additive manufacturing system

Although recent literature suggest ML applications in a wide range of manufacturing systems, most works detail offline ML that is not capable of making real time online decisions. A potential idea that can solve the issue of online sequential decision making while dealing with uncertainty is the implementation of reinforcement learning (RL) [6] that builds upon the Markov decision process (MDP). In case of RL, a learning agent interacts with the environment and tries to find the optimal policy by observing the reward or the feedback it gets from the environment (see Fig. 1). Primary applications of reinforcement learning include robotics [7], [8], mastering complex games [9], operations research [10] and other relevant fields. Some studies have focused on using MDP or constrained MDP to solve the problem of sequential optimization for better build quality [11]. Most of these ideas are applied to systems in which data is cheap to procure or to produce, and oftentimes require on the order of to million data pairs. One of the fundamental challenges of extending RL to a manufacturing system is that, in general, the production of a part in a manufacturing line is expensive. For example, metal parts produced *via* directed energy deposition require hours of machine time and thousands of dollars in powdered metal feedstock [2]. However, most of the RL literature does not consider the practical difficulties of applying data intensive algorithms to the manufacturing problem.

Recently the authors proposed that RL can be used in

a complex task like the manufacture of metamaterials that attain desired wave propagation properties without manual design [12]. Sampling efficiency was quantified, but no methods to reduce sample number were proposed. In general reinforcement learning suffers from the need of large amount of samples to make a good decision. Another issue is the potential danger of implementing a non-optimal or a potentially unsafe policy while interacting with a physical system. We wish to learn the correct inputs for the manufacturing system that can produce artifacts with desired output properties with as few interactions with the real system as possible. One key challenge in doing so is that the goal is unknown for this learning task and we have to keep interacting with the system, incurring expense, to achieve convergence.

To achieve a minimal interaction learning algorithm, we employ the options framework [13], which is a form of temporal abstraction, from RL with physics-based guidance. We propose to leverage physics guided knowledge that maps the inputs to outputs of the workpiece fabricated in a manufacturing system. By doing so, RL transforms from a fully empirical method to semi-empirical method which still retains beneficial features of RL such as exploration to compensate for uncontrollable factors, whilst using physics knowledge to penalize much of the sub-optimal parameter space. Additionally, the options framework permits state transitions that jump past large swaths of sub-optimal parameter space (Fig. 2). Traditional RL transitions through states by applying primitive actions, unnecessarily requiring multiple primitive actions to transit out of a sub-optimal region of the state space. The options framework permits a multi-primitive horizon to quickly emerge from sub-optimal regions. We term this integration of physics-guided RL and the options framework "Physics guided Reinforcement learning enabled Autonomous Manufacturing" ($PRLAM$).

The main contributions of this paper are threefold: (1) Development of a complete framework for autonomous manufacturing system that can be generalized to many applications; (2) Demonstration of a detailed case study that shows the potential of this framework in a real-world manufacturing system; and (3) Identification of key challenges of building such a framework. The paper is organized as follows: section II provides the preliminary ideas that play crucial role in developing the rest of the paper, section III discusses problem formulation for an autonomous manufacturing system, section IV develops the learning framework with minimal number of sample fabrication, section V implements the proposed framework on a case study of acoustic metamaterial fabrication and section VI provides the corresponding results and analysis.

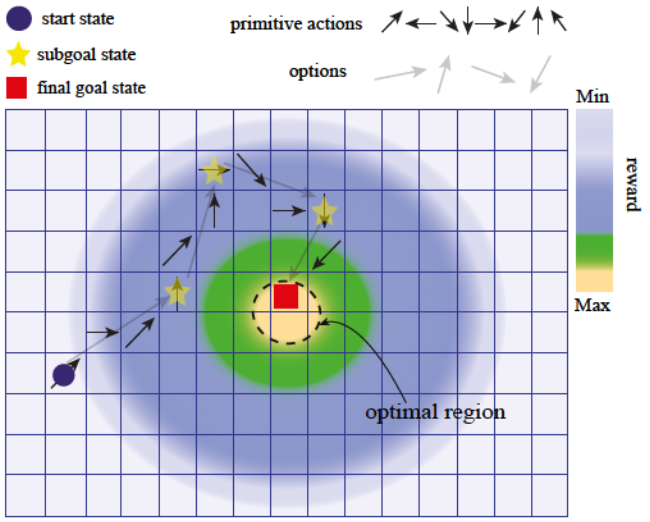


Fig. 2. Conceptual visualization of the problem and strategy

II. Preliminaries

The framework proposed in this paper is based on value function based RL with combination of temporal abstraction and transfer learning. In this section we provide relevant background on these topics.

A. Value based reinforcement learning

In principle RL deals with the problem of learning to control a dynamical system using a Markov decision process (MDP) [6].

Definition 1. (Markov decision process). The Markov decision process is defined as a tuple $\mathcal{M} = \langle \mathcal{X}, \mathcal{A}, \mathcal{P}, R, \gamma \rangle$, where $\mathcal{X}$ is a set of states $\mathbf{x} \in \mathcal{X}$ (i.e state-space), $\mathcal{A}$ is a set of actions $a \in \mathcal{A}$ (i.e. action-space), $\mathcal{P}$ is the transition probability that describes the dynamics of the system, $R : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ defines a reward function and $\gamma \in [0, 1]$ is a scalar discount factor.

Formally, the goal of an agent is to learn a policy $\pi(\mathbf{x}_t|a_t) : \mathcal{X} \to \mathcal{A}$ that maximizes the expected discounted future reward from state $\mathbf{x}_t$, in the case of the state-value function $V^\pi(\mathbf{x}_t)$, or when starting from a state-action tuple $(\mathbf{x}_t, \mathbf{a}_t)$, in the case of the state-action value function,

$$Q^\pi(\mathbf{x}_t, a_t) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R_t | \mathbf{x}_t, a_t \right]. \quad (1)$$

In value based RL, the value function works as an intermediate step to find the policy. Intuitively the idea is to recover a near-optimal policy from an accurate estimation of a state or a state-action value function. Various algorithms can be used to find value function estimation. For example, we can use the SARSA algorithm [6] to continually estimate Q^π for the behavior policy π and use a similar policy for learning. These types of algorithms are known as on-policy algorithms and the update rule is derived from the recursive definition of the action-value function with learning rate α,

$$Q(\mathbf{x}, a) \leftarrow Q(\mathbf{x}, a) + \alpha[R + \gamma Q(\mathbf{x}', a') - Q(\mathbf{x}, a)]. \quad (2)$$

B. Transfer learning

Transfer learning [14] often helps to accelerate the learning procedure in a target task by transferring previous knowledge from a source task. For example, simulation to

real-world transfer is an active research field that uses high-fidelity simulators for learning policies and then transfer those policies into real world [15].

C. Temporal abstraction

Previous studies [13], [16] have proposed that temporal abstraction in RL can increase the speed of learning by reasoning in terms of high-level abstract actions. These multi-time actions are often known as "options". Note that any MDP with a fixed set of options no longer stays an MDP, rather the MDP forms a semi-Markov decision process (SMDP) [13]. To deal with SMDP, a Q-learning update rule has been proposed and is known as SMDP-Q learning [13], [17].

$$Q(\mathbf{x}, o) \leftarrow Q(\mathbf{x}, o) + \alpha \left[r + \gamma^{\Delta} Q(\mathbf{x}', o') - Q(\mathbf{x}_t, o_t) \right] \quad (3)$$

$$r = R_t + \gamma R_{t+1} + \cdots + \gamma^{\Delta-1} R_{t+\Delta-1} \quad (4)$$

Here, o' is the next option that can be chosen at the next state $\mathbf{x}'$ and Δ is the number of time steps that the option takes to execute.

III. Problem Formulation

A. System definition

We consider a manufacturing system at timestep t with output y_t, inputs $\mathbf{x}_t$ and stochastic noise ξ_t. The noise-free system can be expressed as $\mathcal{F}(\cdot) : \mathbb{R}^N \rightarrow \mathbb{R}$, a mapping between the controllable inputs and the true or measured output.

$$y_t = \mathcal{F}(\mathbf{x}_t) + \xi_t \quad (5)$$

where $\mathbf{x}_t = [x_1^{(t)}, x_2^{(t)}, \ldots, x_N^{(t)}]^T$. In our case, the timestep is also a manufacturing observation index ($t \in \mathbb{Z}_+$). The stochastic variable ξ_t is a product of the uncontrollable inputs and unmodeled factors. This definition is subjected to some constraints due to the manufacturing limitation (i.e. bounded inputs, fabrication resolution, raw materials feedstock capacity etc.). We assume this system has a total of $N+1$ number of constraints, $c_i = \underline{\mathbf{x}} \preceq \mathbf{x} \preceq \bar{\mathbf{x}}, \forall i = 1, \ldots, N$ and manufacturing index $t \in [0, T]$, where $\underline{\mathbf{x}}$ is the minimum allowable limit of the input variables and $\bar{\mathbf{x}}$ is the maximum allowable limit of the input variables. Here, $\preceq$ is used to define the element-wise inequality: $\mathbf{x}_1 \preceq \mathbf{x}_2$ means that $[\mathbf{x}_1]_i \leq [\mathbf{x}_2]_i \ \forall i$. From here on we will label T as the manufacturing budget or simply budget. We formalize this learning problem as an MDP where the goal of the agent is to find the highest reward region or optimal region (unknown apriori) with minimal number of interactions T with the real manufacturing system (Fig. 2). Note that each real interaction produces an artifact. The learning problem is described as the following. We start with a vector of randomly chosen input parameters,

$$\mathbf{x}_0 \sim \mathcal{U}\left[\underline{\mathbf{x}}, \bar{\mathbf{x}}\right],$$

interact with the real manufacturing system and obtain some sort of feedback (reward, R). Then we make an intelligent transition through an action a to a new vector of input parameters. From now on we define this vector as the state to adopt RL terminologies. As the optimal region/goal state is unknown, we consider the continuing case of RL implementation rather than episodic case. Hence the dynamic evolution of states can be expressed as the markov chain in Fig. 3. As the goal state $\mathbf{x}_{\text{goal}}$ is unknown we wish to find $\mathbf{x}_T$ as close as possible to $\mathbf{x}_{\text{goal}}$.

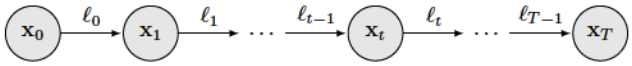


Fig. 3. Evolution of states

B. Reward function

Each time there is a transition from current state to the next state ($\mathbf{x} \xrightarrow{a} \mathbf{x}'$), the agent incurs a loss, ℓ_t. Here,

$$\ell_t = \mathcal{L}(y_{\text{desired}}, y_t)$$

where, $\mathcal{L}(\cdot, \cdot)$ is an appropriate similarity metric between desired output y_{desired} and current output y_t of a user defined characteristic of the artifact. This metric can be a simple squared error or a carefully designed complex relation, for example, a L_2 norm based similarity measure would be

$$\mathcal{L}(y_{\text{desired}}, y_t) = ||y_{\text{desired}} - y_t||_2.$$

Now, we can define the reward function from the loss as

$$R_t = B - \ell_t, \quad (6)$$

where we use a baseline value B to keep the reward positive.

IV. Framework

We propose a complete framework that we term $PRLAM$ to implement RL algorithms for online sequential decision making on data-expensive, stochastic autonomous manufacturing system. The framework is presented in Fig. 4 and the corresponding algorithm is described in Algorithm 1. It is assumed that we have access to a physics-guided external simulator $\mathcal{M}_{\text{sim}}$ of the real system $\mathcal{M}_{\text{real}}$ that can map the inputs $\mathbf{x}$ to output y of the corresponding sample fabricated from the manufacturing system. $\mathcal{M}_{\text{sim}}$ can be a first principle based model of the system with differential equations or a physics-based computational engine (FEA, CFD etc.). Let's consider the following representation of $\mathcal{M}_{\text{sim}}$ similar to [18] through a set of possibly nonlinear equations $\mathcal{G}(\cdot, \cdot)$ and inequalities $\mathcal{H}(\cdot, \cdot)$.

$$\mathcal{M}_{\text{sim}} : \begin{cases} \mathcal{G}(\hat{y}, \mathbf{x}) = 0 \\ \mathcal{H}(\hat{y}, \mathbf{x}) \leq 0 \end{cases}$$

$$\hat{y} = \mathcal{M}_{\text{sim}}(\mathbf{x})$$

where $\hat{y}$ is the estimation of the system output obtained from $\mathcal{M}_{\text{sim}}$.

During the first phase of $PRLAM$ framework we use $\mathcal{M}_{\text{sim}}$ as the source task for training the agent. While learning in the source task we implement any on-policy RL algorithm $\mathscr{A}_{\text{sim}}$ (i.e. SARSA etc.) to interact with $\mathcal{M}_{\text{sim}}$ using policy π and receive a reward $\hat{R}$ with each transition of

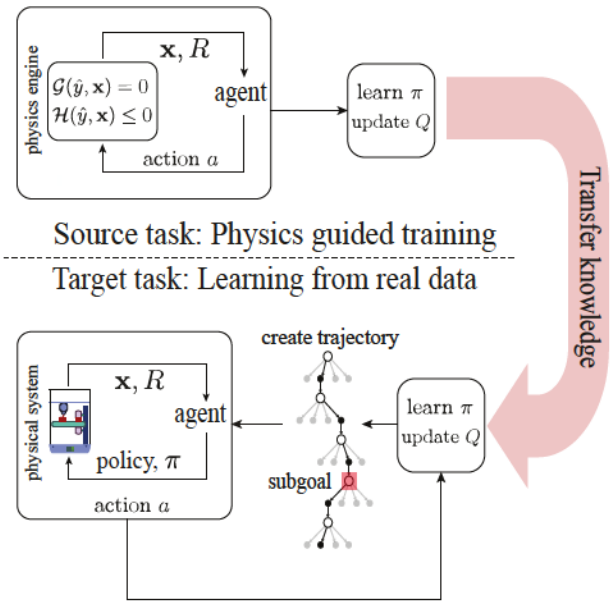


Fig. 4. *PRLAM* framework

states. Thus, the agent collects experiences and updates the action-value function Q^{π}. After a sufficiently large number of interactions, the agent will have an accurate estimation of Q^{π} and the near-optimal policy π^* can be retrieved from Q^{π}. Now we transfer this knowledge to $\mathcal{M}_{\text{real}}$. At the start of the second phase, using a Model Predictive Control (MPC) [19] like approach we roll out an imaginary trajectory $\tau^*_{\sim\pi^*}$ of finite horizon H using the learned policy π^*.

$$\tau^*_{\sim\pi^*} = (\hat{\mathbf{x}}_t, a_t, \hat{R}_t, \hat{\mathbf{x}}_{t+1}, a_{t+1}, \hat{R}_{t+1}, \ldots, \hat{\mathbf{x}}_{t+H}) \quad (7)$$

Next we identify a sub-goal state $\mathbf{x}_{\text{subgoal}}$ from this trajectory. There may be a variety of ways to identify $\mathbf{x}_{\text{subgoal}}$; for example, a simple approach is to choose the state with maximum visitation within this trajectory obtained from $\mathcal{M}_{\text{sim}}$. Note that we only perform an actual fabrication in these sub-goal states and achieve an actual reward R. Then we update the reward in (6) by replacing the Δ-th reward of the trajectory with real reward (line 14 in Algorithm 1). Next we update the learning rule in (3). We keep interacting with $\mathcal{M}_{\text{real}}$ and learning with real data as long as our budget T allows.

V. Case Study

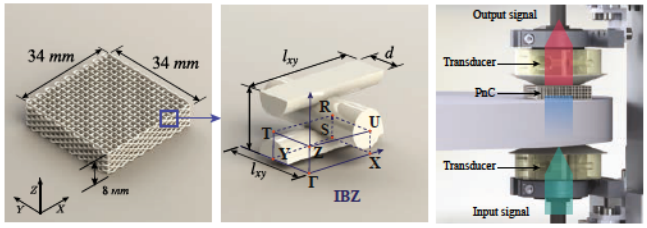
A. PnC materials

This study implements the *PRLAM* framework for the autonomous discovery of PnC materials using fused deposition (FDM)-based additive manufacturing (AM) system as a case study. The fabricated artifacts exhibit an acoustic spectrum with special passband properties corresponding to their geometric parameters [20], [21]. The spectrum can be characterized by piezoelectric transducers, as shown in Fig. 5. In this study we parameterize PnC design using filament diameter (d) and lattice constant (l_{xy}). Our goal

Algorithm 1 Physics-guided Reinforcement learning enabled Autonomous Manufacturing (*PRLAM*)

1: Source task: $\mathcal{M}_{\text{sim}} : \{\mathcal{G}(\hat{y}, \mathbf{x}) = 0, \mathcal{H}(\hat{y}, \mathbf{x}) \leq 0\}$
2: Target task: Physical Manufacturing system, $\mathcal{M}_{\text{real}}$
3: Learn in $\mathcal{M}_{\text{sim}}$ using $\mathscr{A}_{sim}$ with policy π
4: Identify optimal policy π^* from low level knowledge
5: Transfer low level knowledge (i.e. $\pi^*, Q^{\pi}(\mathbf{x}, a)$)
6: Input: trajectory horizon H, budget T, initialize $Q(\mathbf{x}, o)$
7: **for** $t \in T$ **do**
8: $\quad \hat{\mathbf{x}}_t \equiv \mathbf{x}$: current state in physical system
9: $\quad \tau^*_{\sim\pi^*} = (\hat{\mathbf{x}}_t, a_t, \hat{R}_t, \hat{\mathbf{x}}_{t+1}, a_{t+1}, \hat{R}_{t+1}, \ldots, \hat{\mathbf{x}}_{t+H})$
10: $\quad$ Identify subgoal state $\hat{\mathbf{x}}_{t+\Delta}, \quad \Delta \in \{0, 1, \ldots, H\}$
11: $\quad$ option, $o \leftarrow \{a_t, a_{t+1}, a_{t+2}, \ldots, a_{t+\Delta-1}\}$
12: $\quad \mathbf{x}' \leftarrow \hat{\mathbf{x}}_{t+\Delta}$ where $\hat{\mathbf{x}}_{t+\Delta} \xleftarrow{o_t} \hat{\mathbf{x}}_t$
13: $\quad$ Perform an actual fabrication at $\mathbf{x}'$ and obtain R
14: $\quad r = \hat{R}_t + \gamma \hat{R}_{t+1} + \cdots + \gamma^{\Delta-2} \hat{R}_{t+\Delta-2} + \gamma^{\Delta-1} R$
15: $\quad Q(\mathbf{x}, o) += \alpha \left[r + \gamma^{\Delta} Q(\mathbf{x}', o') - Q(\mathbf{x}, o) \right]$
16: $\quad \mathbf{x} \leftarrow \mathbf{x}'$

is to obtain desired spectrum y_{desired} through proper tuning of $\mathbf{x} = [l_{xy}\ d]^T$.



(a) (b) (c)

Fig. 5. **Unit cell description and PnC characterization:** (a) PnC material, (b) Unit cell with design parameters l_{xy} and d, (c) PnC Characterization using ultrasonic transducers

B. Physics guided simulation engine

We perform eigenfrequency analysis of PnCs to identify the passbands. These calculations are performed on a representative unit cell that is the building block of the PnC and repeats periodically along the three spatial directions (x, y, z) (Fig. 5). We develop a computational engine that uses the acoustics module package of COMSOL® to solve the elastic wave propagation problem by Finite Element Method (FEM) at discrete parameter values of $l_{xy} = [700, 705, \ldots, 1035]\mu m$, and $d = [300, 305, \ldots, 635]\mu m$, for a total of 4624 unique design parameter combinations. Note that in this study $\mathcal{G}(\hat{y}, \mathbf{x}), \mathcal{H}(\hat{y}, \mathbf{x})$ are embedded inside the FEM solver. We refer the interested readers to [22], [23] for additional details on PnC eigenfrequency analysis. We use the same spectra characterization and loss definition from [12]:

$$\ell_t = \mathcal{L}(y_{\text{desired}}, y_t) = \mathcal{L}_1 + a\mathcal{L}_2 + b\mathcal{L}_3.$$

Here, $\mathcal{L}_1$ compares the amplitude of y_{desired} and y_t, $\mathcal{L}_2$ is a normalized metric of the difference between peak amplitude frequency of y_{desired} and y_t, $\mathcal{L}_3$ is a normalized metric of

the difference between peak amplitude of y_{desired} and y_t and a, b are scaling constants. Finally we define the reward using (6) with base value B = 80.

C. Model of the environment

To computationally test the proposed framework we build $\mathcal{M}_{\text{sim}}$ and $\mathcal{M}_{\text{real}}$ of $\mathcal{F}(\mathbf{x})$ using non-parametric regression technique Gaussian Process (GP) [24] and denote them as $\mathcal{M}_{\text{sim},\mathcal{GP}}$ and $\mathcal{M}_{\text{real},\mathcal{GP}}$ respectively. A GP can be considered as a prior over plausible functions $\mathcal{F}$ and is characterized by a mean function $m(\mathbf{x})$ and a covariance function $k(\mathbf{x}, \mathbf{x}')$.

$$\mathcal{F} \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')) \tag{8}$$

Assuming noise $\xi \sim \mathcal{N}(0, \sigma_y^2)$ and using a fixed $m(\mathbf{x})$ and data $\{\mathbf{X}, \mathbf{y}\}$ the predictive distribution for a deterministic input $\mathbf{x}_i$ is calculated as

$$\begin{aligned}
\mathcal{F}_i &\sim \mathcal{N}(\mu_i, \Sigma_i) \\
\mu_i &= m(\mathbf{x}_i) + k(\mathbf{x}_i, \mathbf{X})(K + \sigma_y^2 I)^{-1}(\mathbf{y} - m(\mathbf{X})) \\
\Sigma_i &= k(\mathbf{x}_i, \mathbf{x}_i) - k(\mathbf{x}_i, \mathbf{X})(K + \sigma_y^2 I)k(\mathbf{X}, \mathbf{x}_i)
\end{aligned}$$

where $K = k(\mathbf{X}, \mathbf{X})$. We choose the squared-exponential covariance function [25] to build the models.

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp(-\frac{1}{2l^2}(\mathbf{x} - \mathbf{x}')^T(\mathbf{x} - \mathbf{x}'))$$

We do not use evidence maximization, a mathematical

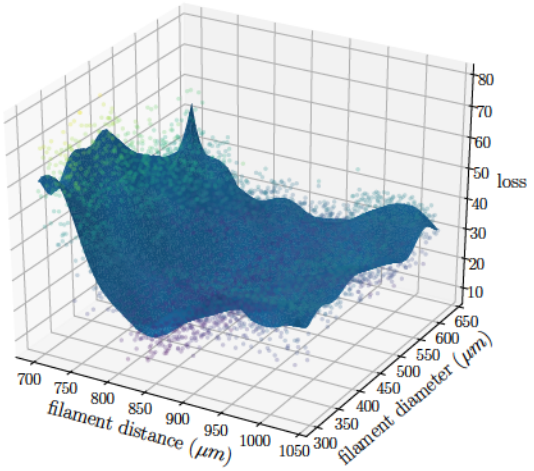


Fig. 6. $\mathcal{M}_{\text{sim},\mathcal{GP}}$ obtained from non-parametric GP regression

way to remove hyperparameter σ_f, l, σ_y from GP, as it overfits the original data and $\mathcal{M}_{\text{sim},\mathcal{GP}}$ will work as too accurate approximation of $\mathcal{M}_{\text{real},\mathcal{GP}}$. We manually choose $\sigma_f = 50.0, l = 52.0$. The GP model of $\mathcal{M}_{\text{sim}}$ is shown in Fig. 6 that is built using the noise variance parameter, $\sigma_y = 1.0$. On the other hand we use $\sigma_y = 10.0$ to build stochastic model $\mathcal{M}_{\text{real},\mathcal{GP}}$ which is analogous to a noisy physical manufacturing system, $\mathcal{M}_{\text{real}}$. Hence, $\mathcal{M}_{\text{sim},\mathcal{GP}}$ will be treated as the source task and we will build a stochastic model $\mathcal{M}_{\text{real},\mathcal{GP}}$ in each iteration as the target task that mimics the physical system. Note that we only consider

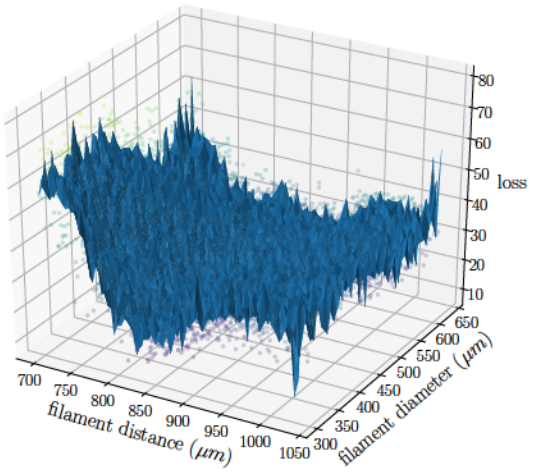


Fig. 7. An example $\mathcal{M}_{\text{real},\mathcal{GP}}$ that mimics the noisy experimental data in a physical system at timestep t.

unbiased noise in $\mathcal{M}_{\text{real},\mathcal{GP}}$ because the goal of this study is to show that temporal abstraction is suitable in a noisy environment. A detailed study of the effect of noise on the learning algorithm is out of the scope of this paper.

D. Implementation of PRLAM framework

According to Algorithm 1, initially we train our agent in $\mathcal{M}_{\text{sim},\mathcal{GP}}$ using on-policy algorithm average reward differential SARSA [6]. Here we start with an action space $|\mathcal{A}| = 8$ where the primitive actions are eight possible directions the agent can move. If the agent hits any constraint c_i it stays in the same place on the state space. As there is no identified goal state $\mathbf{x}_{\text{goal}}$, the agent interacts with the environment continually and keeps updating the Q-function. The agent is trained for 10 million timesteps using ϵ-greedy policy with slightly higher exploration factor ($\epsilon = 0.25$). It means that on average out of every 4 actions the agent tries one random action. We find out that on-policy algorithms tend to work better in the training phase. After the agent is sufficiently trained, we transfer the Q-function to the stochastic model $\mathcal{M}_{\text{real},\mathcal{GP}}$. At each current state, the agent rolls out a trajectory of fixed horizon length H based on training data. The most challenging part is to identify $\mathbf{x}_{\text{subgoal}}$ from the trajectory. For this study we choose the maximum visited state within each trajectory as $\mathbf{x}_{\text{subgoal}}$. This makes sense because the agent will visit a state more often than other states if that state has higher action-values than others. Once the agent identifies $\mathbf{x}_{\text{subgoal}}$ from the rolled out trajectory it moves to the sub-goal state through the execution of temporally extended action or option o (in fact it is a sequence of primitive actions). Once the agent observes a real reward, the transferred Q-value is also updated using Q-learning update rule. We repeat this process until we run out of budget T. To test the performance of the framework, we perform 50 such experiments and an experiment is considered a successful attempt if $\mathbf{x}_T$ can achieve a reward $R \geq 55.0$. We choose this reward because

when the actual spectrum is very close to the desired spectrum of the PnC material.

We also define a success rate to investigate the performance of over these experiments for different values of horizon length and budget ,

$$\frac{\text{number of experiments where has}}{\text{Total number of experiments performed}} \quad (9)$$

VI. Results

We only allow a maximum of interactions with and the framework is able to maintain high rewards within this extremely small number of interactions. The mean reward obtained from experiments is shown in Fig. 8. In this case, starting from a lower reward the agent is able to achieve and maintain . Initially uncertainty is high due to random initialization and as the budget increases higher rewards are achieved. Eventually the agent is able to find the optimal region with high rewards. In

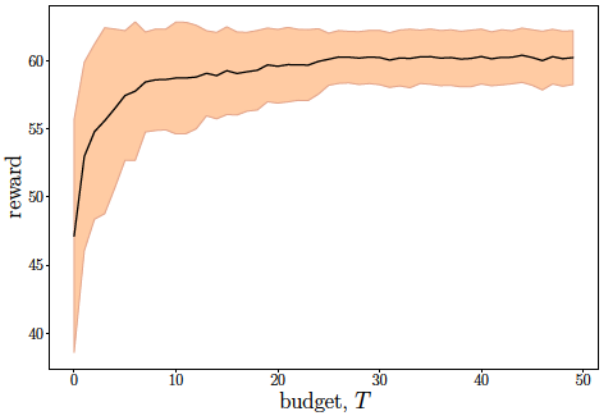


Fig. 8. Final state reward for different budgets, rewards are shown within one standard deviation limit for experiments with random initialization.

Fig. 9 we show the final state of these experiments in a heat map of a sample stochastic environment for . We observe that most form cluster around high reward region while very few of them fail to converge. The effects of horizon length and the budget T (interactions with) on the success rate is summarized in table I. We use while varying . Also we keep fixed horizon length while varying . The results match the intuitive sense, should increase as we increase the budget T and also when we increase the trajectory horizon length . As the interaction with increases the agent gets better at exploiting the domain knowledge. On the other hand when we increase the horizon length this allow the agent to plan for longer period of times. Note that longer horizon length may harm the performance where noise in is very high while very short horizon length may also harm the performance for small budget. We achieve a maximum of success rate at and .

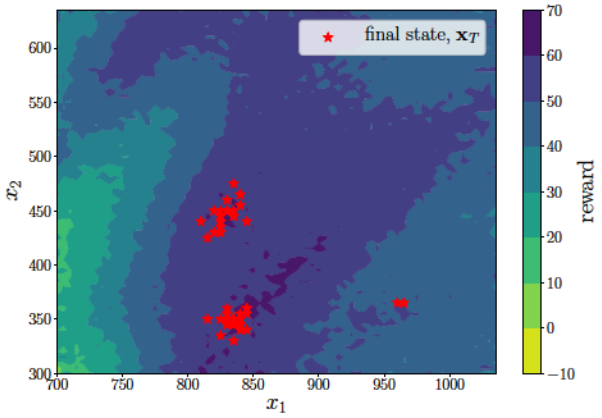


Fig. 9. obtained from experiments shown on a sample stochastic . Most form clusters around high reward region except very few.

The effect of temporal abstraction can be observed from Fig. 10 which is shown for a representative experiment with only interactions with . The agent is able to execute temporally extended actions and jump multiple states to reach . Also note that the agent makes longer jumps from low reward region and and smaller jumps from high reward region. To verify how is selected based on visit counts we present Fig. 11. This figure showcases a single trajectory obtained from in a single timestep of an experiment with . We verify that only the maximum visited state is selected as the subgoal state .

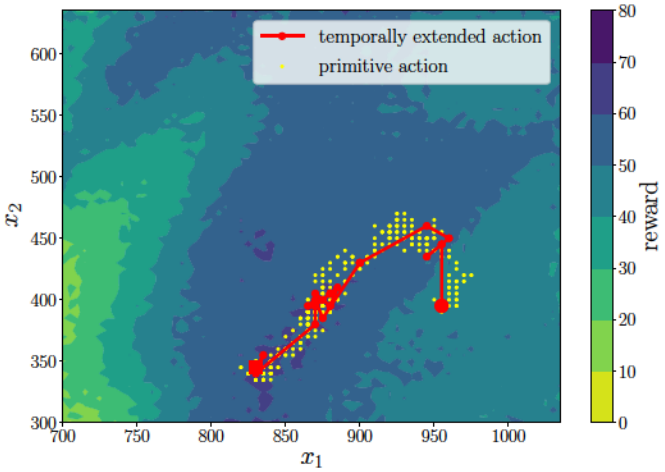


Fig. 10. Temporal abstraction in a single experiment plotted on an example stochastic model. Temporally extended actions can execute multiple primitive actions as a sequence to reach the sub-goal state.

VII. Conclusion

We propose a novel framework for autonomous manufacturing system and implements on a computational case study of PnC manufacturing with desired properties. These early simulation results demonstrate promise for experimental application and translation to other autonomous manufacturing systems. There are a multitude important follow-up

TABLE I

SUCCESS RATE FOR PARAMETER VALUES

Parameter name	Parameter value	Success rate, S_r
Horizon, H	10	0.74
	25	0.88
	50	0.92
Budget, T	10	0.80
	25	0.92
	50	0.96

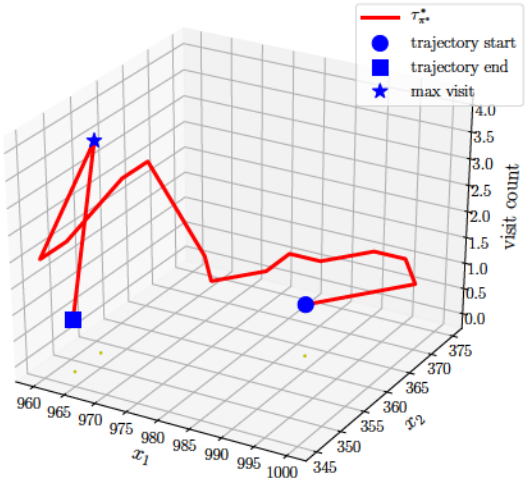


Fig. 11. Subgoal state ($\mathbf{x}_{\text{subgoal}}$) selection criteria based on visit counts to a state from trajectory τ^* using learned policy π^*

studies that are prompted by this first work. For instance, the selection of a sub-goal was largely unexplored in this studying, opting to select the sub-goal most visited during the policy development on the physics emulator; one can envision more sophisticated algorithms which incorporate parameter region statistics for more robustness or other selection criteria. Additionally, many manufacturing problems have a high dimensional input parameter space; although we believe physics integration is necessary to make the higher dimensional optimization possible, the dimensional limits on this method are yet to be explored.

REFERENCES

[1] T. Wuest, D. Weimer, C. Irgens, and K.-D. Thoben, "Machine learning in manufacturing: advantages, challenges, and applications," *Production & Manufacturing Research*, vol. 4, no. 1, pp. 23–45, 2016.

[2] D. J. Corbin, A. R. Nassar, E. W. Reutzel, A. M. Beese, and N. A. Kistler, "Effect of directed energy deposition processing parameters on laser deposited inconel® 718: External morphology," *Journal of Laser Applications*, vol. 29, no. 2, p. 022001, 2017.

[3] P. Nikolaev, D. Hooper, F. Webber, R. Rao, K. Decker, M. Krein, J. Poleski, R. Barto, and B. Maruyama, "Autonomy in materials research: a case study in carbon nanotube growth," *npj Computational Materials*, vol. 2, p. 16031, Oct. 2016.

[4] K. G. Reyes and B. Maruyama, "The machine learning revolution in materials?" *MRS Bulletin*, vol. 44, no. 7, pp. 530–537, 2019.

[5] Daehn, Glenn, J. Allison, E. Bilitz, D. Bourne, J. Cao, K. Clarke, J. DeLoach Jr., E. Herderick, J. Lewandowski, T. Schmitz, H. Sizek, and A. E. Tekkaya, "Metamorphic Manufacturing: Shaping the Future of On-Demand Components," The Minerals, Metals & Materials Society, Tech. Rep., Mar. 2019.

[6] R. S. Sutton, A. G. Barto, *et al.*, *Introduction to reinforcement learning*. MIT press Cambridge, 1998, vol. 135.

[7] A. Y. Ng, A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger, and E. Liang, "Autonomous inverted helicopter flight via reinforcement learning," in *Experimental robotics IX*. Springer, 2006, pp. 363–372.

[8] S. Ha, P. Xu, Z. Tan, S. Levine, and J. Tan, "Learning to walk in the real world with minimal human effort," *arXiv preprint arXiv:2002.08550*, 2020.

[9] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, "Mastering the game of go without human knowledge," *nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[10] S. Mahadevan and G. Theocharous, "Optimizing production manufacturing using reinforcement learning." in *FLAIRS Conference*, vol. 372, 1998, p. 377.

[11] B. Yao and H. Yang, "Constrained markov decision process modeling for sequential optimization of additive manufacturing build quality," *IEEE Access*, vol. 6, pp. 54 786–54 794, 2018.

[12] M. F. Alam, M. Shtein, K. Barton, and D. J. Hoelzle, "Autonomous manufacturing using machine learning: A computational case study with a limited manufacturing budget," in *International Manufacturing Science and Engineering Conference*, vol. 84263. American Society of Mechanical Engineers, 2020, p. V002T07A009.

[13] R. S. Sutton, D. Precup, and S. Singh, "Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning," *Artificial intelligence*, vol. 112, no. 1-2, pp. 181–211, 1999.

[14] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey." *Journal of Machine Learning Research*, vol. 10, no. 7, 2009.

[15] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," *arXiv preprint arXiv:1804.10332*, 2018.

[16] D. Precup, "Temporal abstraction in reinforcement learning." *Ph. D. thesis, University of Massachusetts*, 2000.

[17] S. J. Bradtke and M. O. Duff, "Reinforcement learning methods for continuous-time markov decision problems," *Advances in neural information processing systems*, pp. 393–400, 1995.

[18] A. Karpatne, W. Watkins, J. Read, and V. Kumar, "Physics-guided neural networks (pgnn): An application in lake temperature modeling," *arXiv preprint arXiv:1710.11431*, 2017.

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

[20] M. S. Kushwaha, P. Halevi, L. Dobrzynski, and B. Djafari-Rouhani, "Acoustic band structure of periodic elastic composites," *Physical review letters*, vol. 71, no. 13, p. 2022, 1993.

[21] A. Kruisová, M. Ševčík, H. Seiner, P. Sedlák, B. Román-Manso, P. Miranzo, M. Belmonte, and M. Landa, "Ultrasonic bandgaps in 3d-printed periodic ceramic microlattices," *Ultrasonics*, vol. 82, pp. 91–100, 2018.

[22] W. Setyawan and S. Curtarolo, "High-throughput electronic band structure calculations: Challenges and tools," *Computational materials science*, vol. 49, no. 2, pp. 299–312, 2010.

[23] F. Warmuth and C. Körner, "Phononic band gaps in 2d quadratic and 3d cubic cellular structures," *Materials*, vol. 8, no. 12, pp. 8327–8337, 2015.

[24] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on Machine Learning*. Springer, 2003, pp. 63–71.

[25] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.