2019 International Conference on Robotics and Automation (ICRA)
Palais des congres de Montreal, Montreal, Canada, May 20-24, 2019

Generalized Controllers in POMDP Decision-Making

Kyle Hollins Wray and Shlomo Zilberstein

***Abstract*— We present a general policy formulation for partially observable Markov decision processes (POMDPs) called *controller family policies* that may be used as a framework to facilitate the design of new policy forms. We prove how modern approximate policy forms: point-based, finite state controller (FSC), and belief compression, are instances of this family of generalized controller policies. Our analysis provides a deeper understanding of the POMDP model and suggests novel ways to design POMDP solutions that can combine the benefits of different state-of-the-art methods. We illustrate this capability by creating a new customized POMDP policy form called the *belief-integrated FSC (BI-FSC)* tailored to overcome the shortcomings of a state-of-the-art algorithm that uses non-linear programming (NLP). Specifically, experiments show that for NLP the BI-FSC offers improved performance over a vanilla FSC-based policy form on benchmark domains. Furthermore, we demonstrate the BI-FSC's execution on a real robot navigating in a maze environment. Results confirm the value of using the controller family policy as a framework to design customized policies in POMDP robotic solutions.**

I. Introduction

The partially observable Markov decision process (POMDP) is one of the most general single agent decision-making models [1]. Over the past two decades, POMDP solution formulations and approximate algorithms have enjoyed rapid growth and increased interest, specifically point-based methods [2], [3], finite state controller policies [4], [5], and compression techniques [6], [7]. With the improved tractability these solutions afford, POMDPs are increasingly used in real world deployed robotic applications ranging from aircraft collision avoidance systems [8] to self-driving cars [9], [10], [11]. However, POMDPs are PSPACE-complete, requiring improved techniques to be developed to facilitate more widespread use. Towards this goal, we provide a novel formulation of POMDP value and policy that unifies the state-of-the-art approaches, gives new insights, and provides a solid foundation on which to build the next generation of solutions that use these improved policy representations.

The current generation of POMDP solutions individually arose from disparate ideas about how to explore small sets of reachable beliefs, define node-based abstractions, or find reduced models representing a similar problem. Each method resulted in a distinct form of policy and value function representation, on top of which an algorithm was constructed. While each approach is successful in its own right, it is not clear how they are related to one another. This knowledge gap manifests itself by the lack of a unified view on the POMDP value equations, policies, and resulting algorithms, limiting the potential of new approaches and leaving many questions unanswered. For example, can we integrate point-based and FSC approaches? Is there a general method to introduce compression into point-based or FSC policy forms? What methods can automatically select the abstracted states (nodes)? In general, what is the relation between expanding more belief points, adding FSC nodes, or exploring the number of bases in a compressed model? We might even ask what the mathematical reason is for why value iteration and policy iteration exist? Finally, is there an underlying principled framework to design POMDP solutions? This paper aims at taking steps towards answering these questions.

This work was supported by NSF grants IIS-1724101 and IIS-1524797.
University of Massachusetts, Amherst, MA 01002, USA. Emails: {wray,shlomo}@cs.umass.edu

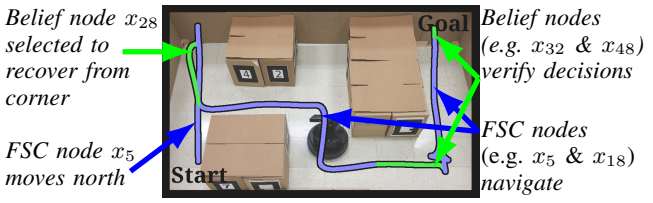

Fig. 1. An example controller family policy called a BI-FSC on a robot. It enables policies that combine belief points and FSC nodes.

We present the *controller family* of policy forms as a general formulation of policy and value. It consists of a set of nodes, an action selector function, a node selector function, and a function approximator with a specific form of value function acting as a constraint. These placeholder nodes and functions can be assigned to specific values or left unconstrained. We show how various policy forms emerge by constraining these elements.

As a motivational analogy, consider exponential family distributions which generalize popular distributions such as binomial, exponential, Dirichlet, and Gaussian. The exponential family presents a probability density function with placeholder functions (e.g., sufficient statistic function, natural parameter, and partition function). Constraining these placeholder functions in particular ways produces the specific popular distributions. In the same manner, we define controller family policies with a value function using placeholder functions. Constraining these placeholder functions in particular ways produces specific popular approximate policy forms: point-based, FSC, value-directed compression (VDC), and exponential-family principle components analysis compression (E-PCA). This paper defines the controller family and rigorously proves it encapsulates these policy forms.

Our contributions are: (1) a formal statement of the controller family (Section III); (2) a detailed theoretical analysis mapping the three widely-used policy forms to the controller family (Section IV); and (3) a novel belief-integrated FSC-based policy form with experiments (Section V).

II. Background

A partially observable Markov decision process (POMDP) is defined by the tuple $\langle S, A, \Omega, T, O, R \rangle$ [1]. S is a set of n states. A is a set of m actions. Ω is a set of z observations. $T: S \times A \times S \to [0,1]$ is a state transition that maps state s and action a to the probability of successor s' with $T(s,a,s') = Pr(s'|s,a)$. $O: A \times S \times \Omega \to [0,1]$ is an observation function that maps action a and successor s' to the probability of observation ω with $O(a,s',\omega) = Pr(\omega|a,s')$. $R: S \times A \to \mathbb{R}$ is a reward function that maps state s and action a to a reward $R(s,a)$.

The agent does not observe the true state of the system. Instead it maintains a *belief* $b \in \triangle^n$ over the true state. (Note: $\triangle^n$ is the standard $(n-1)$-simplex.) For belief b, performing a and observing ω yields successor belief b' at s':

$$b'(s') = Pr(\omega|b,a)^{-1} O(a,s',\omega) \sum_{s \in S} T(s,a,s') b(s). \quad (1)$$

It is convenient to refer to sets of beliefs $B \subseteq \triangle^n$, such as the set of *reachable beliefs* from an initial belief b^0 following belief updates (Equation 1) denoted $\mathcal{R}(b^0)$. A *policy* π can be described in two standard ways: a direct mapping of beliefs to actions—using the value function for belief points—or through an abstracted controller—using a stochastic finite state controller (FSC). In both policy forms, we compute a *value function* V^π that is the expected reward given fixed policy π, discount factor $\gamma \in [0,1]$, and horizon h. The objective is to find the policy π^* with maximal expected value V^*.

For an infinite horizon, a policy $\pi: \triangle^n \to A$ maps beliefs to actions. The value function $V^\pi: \triangle^n \to \mathbb{R}$ maps beliefs to their expected reward. The Bellman optimality equation is for belief b:

$$V(b) = \max_{a \in A} \Big[R(b,a) + \gamma \sum_{\omega \in \Omega} Pr(\omega|b,a) V(b'_{a\omega}) \Big] \quad (2)$$

with $R(b,a) = \sum_s b(s) R(s,a)$ and $b'_{a\omega}$ following the belief update equation. The policy is stored by associating each α-vector with the maximizing action denoted $a_\alpha \in A$. The optimal policy is computed by applying Equation 2 to all uncountably infinite belief points until convergence.

For a finite horizon, Sondik [12] showed the value function V^π is piecewise linear and convex. Formally, a set of *α-vectors* $\Gamma = \{\alpha_1, \ldots, \alpha_r\}$ with $\alpha_i = [\alpha_i(s_1), \ldots, \alpha_i(s_n)]^T$ define the value of b:

$$V(b) = \max_{\alpha \in \Gamma} \sum_{s \in S} b(s) \alpha(s). \quad (3)$$

We approximate infinite horizon solutions over $\triangle^n$ with a finite horizon over $\mathcal{R}(b^0)$ starting at a given initial belief point b^0. By applying Equations 1 and 3 to Equation 2 we obtain:

$$V(b) = \max_{a \in A} \Big[R(b,a) + \gamma \sum_{\omega \in \Omega} \max_{\alpha' \in \Gamma} \sum_{s \in S} b(s) \sum_{s' \in S} T(s,a,s') O(a,s',\omega) \alpha'(s') \Big]. \quad (4)$$

III. Controller Family Policies

The **controller family** are policies defined by the form $\pi = \langle X, \psi, \eta, \sigma \rangle$. X is a set of r controller nodes, acting as an internal memory for the agent and referring to compact relevant aspects (e.g., belief points, state features, or compressed beliefs). $\psi: X \times \triangle^n \times A \to [0,1]$ denotes the stochastic selection of action a at node x and belief b with $\psi(x,b,a) = Pr(a|x,b)$. $\eta: X \times \triangle^n \times A \times \Omega \times X \to [0,1]$ denotes the stochastic selection of successor node x' given at node x and belief b, action a was performed yielding observation ω with $\eta(x,b,a,\omega,x') = Pr(x'|x,b,a,\omega)$. Lastly, $\sigma: X \times \triangle^n \to \mathbb{R}$ denotes a function approximator of V. Commonly, $\sigma(x,b) = \sum_s b(s) V(x,s)$, with node x's α-vector $V(x,s)$, is used to approximate infinite horizon with the finite horizon α-vectors. Generally, we will see that approximate algorithms assume $V(x,b) = \sigma(x,b)$ to compute their values.

The **value** of a controller family policy depends on the controller node x and the belief b:

$$V(x,b) = \sum_{a \in A} \psi(x,b,a) \Big[R(b,a) + \gamma \sum_{\omega \in \Omega} Pr(\omega|b,a) \sum_{x' \in X} \eta(x,b,a,\omega,x') \sigma(x', b'_{a\omega}) \Big] \quad (5)$$

with $R(b,a) = \sum_s b(s) R(s,a)$ and $b'_{a\omega}$ following the belief update equation. We often assume initial $x^0 \in X$ and $b^0 \in \triangle^n$. Also, it is convenient to define $Q(x,b,a)$ by the equation in $[\cdot]$. In the most general form, the **objective** is to compute values for *all components*, including the computation of V, to maximize the function approximator $\sigma(x,b)$ for all x and b. Given an initial node and belief, the objective is: $\max_{X,\psi,\eta,\sigma,V} \sigma(x^0, b^0)$ subject to the definition of $V(x,b)$ in Equation 5 and any extra constraints on $\langle X, \psi, \eta, \sigma \rangle$.

We define a controller family policy with a function approximator. Finite stochastic controllers (FSCs) and function approximators have been explored in various forms in-depth before. Critically, this paper presents this *unified formulation* and a *novel perspective* on the formal core of POMDP algorithm *policy and value representations*. We are not encompassing any specific algorithmic nuances or details. Our work establishes important new links among the various POMDP *policy and value representations* that are *used* by different algorithms. Early work informally described similar concepts, but lacked any formal results as modern algorithms were not yet developed [13] [14]. Other work compares the models themselves rather than the POMDP's policy and value forms [15] [16]. Surveys have derived algorithmic commonalities among strictly point-based approaches [17]. The remaining literature discussed in the paper compares algorithms, namely in terms of performance, whereas we find the common threads underlying their design.

At a high-level, we observe that X is a free set and ψ, η, and σ are free functions. If we condition them on the value function and policy, in an appropriate manner, we can enforce a particular structure to focus the resulting policy form. A major contribution of this paper are the formal proofs that select state-of-the-art approximate policy forms are actually specific instances of a broad family of policies.

IV. Theoretical Analysis

We now prove various policy forms are members of the controller family. Each time we: (1) define its normal policy form and value, (2) represent its policy as a controller family policy, and (3) prove the resulting value equation is identical.

A. Optimal Policy Formulations

Importantly, Equation 5 does not enable an "improved optimal equation" beyond what is achievable with Equation 2. The optimal solution can always be expressed by the original Bellman optimality equation, but the controller family equation does encapsulate it. Interestingly, we can construct policies that are not representable by a simple mapping of belief to action; however, no such policy can ever obtain values higher than that of the optimal formulation. This added flexibility is instead exploited in approximations. These simple but necessary facts are proven in Proposition 1.

Proposition 1: The optimality policy form is a member of the controller family.

Proof: We must write Equations 2 and 4 using Equation 5. Let $X=\triangle^n$, $\psi(x,b,a)=[Q(x,b,a)\geq Q(x,b,a')\forall a']$, $\eta(x,b,a,\omega,x')=[x'=b'_{a\omega}]$, and $\sigma(x,b)=V(x,b)$ with Iverson bracket $[\cdot]$, $b'_{a\omega}$ resulting from the belief update equation, and $b\equiv x$ in all cases. This produces Equation 2. For finite horizon, let $X=\mathcal{R}(b^0)$ with ψ as above. Let $\eta(x,b,a,\omega,x')=[\sum_{s'} b'_{a\omega}(s')V(x',s')\geq \sum_{s'} b'_{a\omega}(s')V(x'',s')\forall x'']$ (i.e., an argmax) with α-vector $V(x,s)$ for belief/node x. Let $\sigma(x,b)=\sum_s b(s)V(x,s)$. This produces Equation 4. Also, we can write a policy $\pi:\triangle^n\rightarrow A$ as the controller family policy $\pi_c=\langle X,\psi,\eta,\sigma\rangle$. Let X, η, and σ be as above. Let $\psi(x,b,a)=[\pi(x)=a]$ since $x=b$. We can add arbitrary stochasticity to any beliefs $x=b$, both in action ψ and successor η, representing policies unobtainable otherwise. ∎

Insights (1) The *node selection* X is the policy's domain (e.g., beliefs). (2) We write ψ and η as maximizations by constraining them by parts of the equation. (3) Equation 3 can be interpreted as: $\eta\equiv\max_\alpha$ and $\sigma\equiv\sum_s b(s)\alpha(s)$.

B. Point-Based Policy Formulations

Point-based policies avoid the exponential growth of reachable belief points by exploring a subset $B\subseteq\mathcal{R}(b^0)$. Given any belief $b\in\triangle^n$, we can extract the policy's action $\pi(b)\mapsto a_\alpha$ using $\alpha=\text{argmax}_{\alpha'}\sum_s b(s)\alpha'(s)$. We denote a point-based policy by $\pi=\langle B,\Gamma\rangle$. The point-based update equation, given previous α-vectors Γ', is [2]:

$$\Gamma_{a\omega}=\{[V^{\alpha'}_{a\omega}(s_1),\ldots,V^{\alpha'}_{a\omega}(s_n)]^T,\forall\alpha'\in\Gamma'\},$$

$$\Gamma_b=\{R(\cdot,a)+\sum_{\omega\in\Omega}\operatorname*{argmax}_{\alpha'\in\Gamma_{a\omega}}\sum_{s\in S}b(s)\alpha'(s),\forall a\in A\},$$

$$\Gamma=\{\operatorname*{argmax}_{\alpha\in\Gamma_b}\sum_{s\in S}b(s)\alpha(s),\forall b\in B\} \quad (6)$$

with $V^{\alpha'}_{a\omega}(s)=\gamma\sum_{s'}O(a,s',\omega)T(s,a,s')\alpha'(s')$ and vector $R(\cdot,a)=[R(s_1,a),\ldots,R(s_n,a)]^T$. Each initial $\alpha(s)=\min_{s'}\min_{a'}R(s',a')/(1-\gamma)$ to ensure α-vectors weakly monotonically increase [18].

The original forms of point-based methods apply Equation 4 on a fixed grid over the belief simplex [18]. Point-based value iteration (PBVI) in Equation 6 originally operated over all beliefs, selecting them by solving linear programs to find "witness" (i.e., improvable in value) beliefs [19]. The tractable general incarnation of PBVI explores reachable beliefs and interleaves belief updates with belief expansion techniques [2]. Perseus does all belief expansion initially then intelligently orders the beliefs to do less updates overall, as an α-vector can improve many beliefs simultaneously [20]. HSVI2 [21] and SARSOP [22] both have a tighter interleaving of update and expansion, maintaining lower and upper bounds to test convergence and cleverly selecting action-observation pairs to tighten these bounds. Recent work suggests a modular approach to mix the algorithms' components, which can be competitive in some cases [17].

The formal mapping between a point-based policy and a controller family policy is in Proposition 2.

Proposition 2: Point-based policies are a member of the controller family.

Proof: We must write Equation 6 using Equation 5. Let $X=B$ with each $x\in X$ corresponding to a point-based belief $x\in B$, distinct from the current belief $b\in\mathcal{R}(b^0)$. Let $\sigma(x,b)=\sum_s b(s)V(x,s)=\sum_s x(s)V(x,s)$ always requiring $x=b\in B$. This is the critical assumption that makes point-based algorithms work: to compute the values the only beliefs that matter are in B. Since the objective only optimizes values over X, it removes the dependence on $\mathcal{R}(b^0)$. We rewrite Equation 5, noting $Pr(\omega|b,a)$ cancels with $b'_{a\omega}$, as:

$$V(x,s)=\sum_{a\in A}\psi(x,b,a)\Big[R(s,a)+\sum_{\omega\in\Omega}\sum_{x'\in X}\eta(x,b,a,\omega,x')\gamma\sum_{s'\in S}T(s,a,s')O(a,s',\omega)V(x',s')\Big]$$

with $\sum_s b(s)$ moved outside the summations such that $V(x,b)=\sum_s b(s)V(x,s)$. Simply reference x and x' here as α and α' to rename $\alpha(s)=V(x,s)$ and $\alpha'(s')=V(x',s')$. Recognize $V^{\alpha'}_{a\omega}(s)$ to obtain $\alpha(s)=$

$$\sum_{a\in A}\psi(x,b,a)\Big[R(s,a)+\sum_{\omega\in\Omega}\sum_{x'\in X}\eta(x,b,a,\omega,x')V^{\alpha'}_{a\omega}(s)\Big].$$

As Proposition 1, $\psi(x,b,a)=[Q(x,b,a)\geq Q(x,b,a')\forall a']$ and $\eta(x,b,a,\omega,x')=[\sum_s b(s)V^{\alpha'}_{a\omega}(s)=\sum_s b(s)V^{\alpha''}_{a\omega}(s)\forall\alpha'']$, both equivalent to argmax. This produces Equation 6. Optionally, we could use the more flexible probabilistic softmax function, such as:

$$\eta(x,b,a,\omega,x')=\frac{\exp\{\sum_s b(s)V^{\alpha'}_{a\omega}(s)/\tau\}}{\sum_{x''}\exp\{\sum_s b(s)V^{\alpha''}_{a\omega}(s)/\tau\}} \quad (7)$$

with softmax *temperature* $\tau\rightarrow 0^+$. Also, we can write a point-based policy $\pi_p=\langle B,\Gamma\rangle$ as a controller family policy $\pi_c=\langle X,\psi,\eta,\sigma\rangle$ as in Proposition 1. ∎

Insights (1) Node selection X is the set of explored beliefs. (2) *Node iteration*—iterating to improve the set of selected nodes—is what most point-based approaches rely on. (3) *Node (successor) selector* η and *action selector* ψ can be written with softmax, enabling easy derivatives [14].

C. Finite State Controller Policy Formulations

Finite state controller (FSC) methods instead describe a policy as an FSC that is executed from an initial belief [23]. An FSC policy is defined by $\pi = \langle X, \hat{\psi}, \hat{\eta} \rangle$. X is a set of nodes. $\hat{\psi}: X \times A \to [0,1]$ and $\hat{\eta}: X \times A \times \Omega \times X \to [0,1]$ ignore any dependence on an explicitly maintained belief and follow the FSC alone. Policy iteration (PI) is commonly used with FSC policy representations. PI alternates between policy evaluation and policy improvement steps, though techniques exist to perform them simultaneously [4]. Evaluating policy π requires solving a system of equations:

$$V(x,s) = \sum_{a \in A} \hat{\psi}(x,a) \Big[R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') \sum_{\omega \in \Omega} O(a,s',\omega) \sum_{x' \in X} \hat{\eta}(x,a,\omega,x') V(x',s') \Big] \quad (8)$$

with $V(x,b) = \sum_s b(s) V(x,s)$, $x^0 \in X$, and $b^0 \in \triangle^n$. This formula is derived from the recognition and use of a cross-product MDP formed from states and controller nodes.

The original PI defines a policy as bounded regions on the belief simplex (e.g., resulting from Γ) and converts it to an FSC for policy evaluation [12]. This proved intractably complex. Instead the policy itself can be represented as an FSC. Policy improvement performs the Bellman update in Equation 4, assigns these α-vectors as new potential nodes, then adds, merges, or prunes them [23]. To avoid exponential growth, bounded PI (BPI) explores single new node at each iteration, at the cost of getting stuck in local optima [24]. Point-based PI (PBPI) does Hansen's PI but uses the point-based update in Equation 6 [25]. Recent work casts both evaluation and improvement as one non-linear program (NLP) [4]. A dual formulation exists, but only for deterministic FSCs [26].

The straight-forward but necessary mapping from FSC to the controller family is in Proposition 3.

Proposition 3: FSC policies are a member of the controller family.

Proof: We must write Equation 8 using Equation 5. Let X be the same in both. Let $\psi(x,b,a) = \hat{\psi}(x,a)$ and $\eta(x,b,a,\omega,x') = \hat{\eta}(x,a,\omega,x')$ simply ignore the current maintained b. Let $\sigma_c(x,b) = \sum_s b(s) V(x,s)$. Rewrite Equation 5, apply the definition of $Pr(\omega|b,a)$, and recognize we can again pull $\sum_s b(s)$ outside the summations to obtain Equation 8 with $V(x,b) = \sum_s b(s) V(x,s)$. For deterministic FSCs, we also ensure $\hat{\psi}$ and $\hat{\eta}$ are 0 or 1 via parameters $y_x \in A$ and $z_{xa\omega} \in X$ such that we also have $\hat{\psi}(x,a) = [a = y_x]$ and $\hat{\eta}(x,a,\omega,x') = [x' = z_{xa\omega}]$. Also, we must write an FSC policy $\pi_f = \langle X_f, \psi_f, \eta_f \rangle$ as a controller family policy $\pi_c = \langle X_c, \psi_c, \eta_c, \sigma_c \rangle$, which can be done as above. ■

Insights (1) Node iteration to improve X occurs in BPI and PBPI. (2) Nodes are constrained (i.e., fixed-sized controller) in the NLP solution, but it does *simultaneous value and policy iteration* because V, ψ, and η are free variables. (3) Parameters can be added to a controller family's elements with a fixed structure encoding these parameters in the elements (e.g., y for ψ, and z for η, above).

D. Compression Policy Formulations

Compression techniques construct a mapping from the original POMDP to a smaller reduced model. There are two standard general approaches that we consider: value directed compression (VDC) [7] and exponential family principle components analysis (E-PCA) [6]. Both approaches attempt to find a function $f: \triangle^n \to \triangle^r$ that projects beliefs in n dimensional space to a lower $r \leq n$ dimensional space. In PCA and linear VDC, this takes the form of a matrix $F \in \mathbb{R}^{n \times r}$ that acts as a change of basis. In general, this function f is applied to a set of beliefs $B \subseteq \triangle^n$ to produce an approximate POMDP described by $\tilde{B}$, $\tilde{T}: \tilde{B} \times A \times \tilde{B} \to [0,1]$, and $\tilde{R}: \tilde{B} \times A \to \mathbb{R}$. We prove that all these forms are representable in the controller family.

Each approximate belief $\tilde{b} \in \triangle^r$ is projected from some original belief $b \in \triangle^n$ by $\tilde{b}^T = b^T F$ for linear and $\tilde{b} = f(b)$ for non-linear. For the linear case, a belief b can be reconstructed as $\hat{b} \approx b$ with $\hat{b} = (F^\dagger)^T \tilde{b}$; orthonormal columns yields $\hat{b} = F\tilde{b}$. For the non-linear case, a belief b can be reconstructed with a function inverse $\hat{b} = f^{-1}(\tilde{b})$. In general, we consider these beliefs to be non-negative and renormalized, as both VDC and E-PCA papers discuss. If they are not, then the resulting compressed model would not be a valid (PO)MDP, as beliefs and state transitions must be probabilities over $\triangle^n$. It does not necessarily affect the change of basis, and without it algorithms can run into issues. In the cases when it does still work, it falls outside the proper definition of a (PO)MDP.

Value Directed Compression VDC compresses the beliefs with either a lossless or lossy f creating a smaller POMDP on which any solver can be applied [7]. To focus our analysis, we consider the *linear* VDC using F as it is the favored form to extend [27] [28] [29]. Similar logic applies for general approaches using non-linear f. The original model forms lossless compression F by *Krylov iteration*. Let $R^a \in \mathbb{R}^n$ be defined as $R^a_i = R(s_i, a)$. Let $T^{a\omega} \in \mathbb{R}^{n \times n}$ be defined as $T^{a\omega}_{ij} = Pr(s_j, \omega | s_i, a) = T(s_i, a, s_j) O(a, s_j, \omega)$. Krylov iteration methods start with $F_{1j} = R^a_j$. Then, for each a and ω, they iteratively assign $F_{\cdot i+1} = T^{a\omega} F_{\cdot i}$ if it is linearly independent for all $F_{\cdot 1}$, ..., $F_{\cdot i-1}$ [7]. Lossy variants can compute F by linear programming or truncated Krylov iteration. Truncated Krylov iteration selects up to a fixed $r \leq n$ and removes the largest error column in F at each step. Other lossy variants employ non-negative matrix factorization (NMF) over a given fixed set of beliefs, either by orthogonal NMF to ensure $FF^\dagger \approx I$ [27] or by locality preserving NMF to ensure Lipschitz continuity is preserved in the compressed model [28].

The VDC compressed problem requires that $R^a = F\tilde{R}^a$ and $T^{a\omega} F = F\tilde{T}^{a\omega}$. Since the columns of F are linearly independent by construction, we take the pseudoinverse to derive the compressed model:

$$\tilde{R}^a = F^\dagger R^a \qquad \text{and} \qquad \tilde{T}^{a\omega} = F^\dagger T^{a\omega} F. \quad (9)$$

The policy $\tilde{\pi}: \triangle^r \to A$ is be evaluated with $\tilde{V}^{\tilde{\pi}}: \triangle^r \to \mathbb{R}$ by:

$$\tilde{V}^{\tilde{\pi}}(\tilde{b}) = \tilde{b}^T \tilde{R}^{\tilde{\pi}(\tilde{b})} + \gamma \sum_{\omega \in \Omega} \tilde{V}^{\tilde{\pi}}(\tilde{b}^T \tilde{T}^{\tilde{\pi}(\tilde{b})\omega}). \quad (10)$$

Lemma 1: The linear VDC policy is a member of the controller family.

Proof: We must write Equation 10 as Equation 5. Let $X=\hat{B}\subseteq\triangle^n$ be all possible reconstructed beliefs, with each $x=\hat{b}$. For example, if starting at $x=b^0$, $X=\hat{\mathcal{R}}(b^0)$ are all possible reconstructed beliefs following $T^{a\omega}$ and F. Let $\psi(x,b,a)=[\tilde{\pi}(\tilde{b})=a]$, since VDC uses a policy evaluation form. Let $\eta(x,b,a,\omega,x')=[\hat{b}^T T^{a\omega}=x']$. Let $\sigma(x,b)=V(x,\hat{b})/\|\hat{b}\|_1=\hat{b}^T\hat{\alpha}/\|\hat{b}\|_1$, with the α-vector dot product, and $\hat{\alpha}=F\tilde{\alpha}$. By the definition of $\sigma(x,b)$, the only beliefs ever visited are the reconstructed $\hat{b}$ corresponding to x. Apply to Equation 5:

$$V(x,\hat{b})=R(\hat{b},\tilde{\pi}(\tilde{b}))+\gamma\sum_{\omega\in\Omega}Pr(\omega|\hat{b},\tilde{\pi}(\tilde{b}))\\ V(x',\hat{b}^T T^{\tilde{\pi}(\tilde{b})\omega})/\|\hat{b}^T T^{\tilde{\pi}(\tilde{b})\omega}\|_1$$

with $x=\hat{b}$, $\tilde{b}^T=\hat{b}^T F$, and $x'=\hat{b}^T T^{\tilde{\pi}(\tilde{b})\omega}$ from η's definition. It is simple to show that $\|\hat{b}^T T^{\tilde{\pi}(\tilde{b})\omega}\|_1=Pr(\omega|\hat{b},\tilde{\pi}(\tilde{b}))$. Apply this fact with the definitions of R (as vectors) and V:

$$V(x,\hat{b})=\hat{b}^T R^{\tilde{\pi}(\tilde{b})}+\gamma\sum_{\omega\in\Omega}\hat{b}^T T^{\tilde{\pi}(\tilde{b})\omega}F\tilde{\alpha}.$$

Apply definition of $\hat{b}$, and then recognize both $\tilde{R}$ and $\tilde{T}$:

$$V(x,\hat{b})=\tilde{b}^T F^{\dagger}R^{\tilde{\pi}(\tilde{b})}+\gamma\sum_{\omega\in\Omega}\tilde{b}^T F^{\dagger}T^{\tilde{\pi}(\tilde{b})\omega}F\tilde{\alpha}\\ =\tilde{b}^T\tilde{R}^{\tilde{\pi}(\tilde{b})}+\gamma\sum_{\omega\in\Omega}\tilde{b}^T\tilde{T}^{\tilde{\pi}(\tilde{b})\omega}\tilde{\alpha}.$$

With $\tilde{b}$ we know x and $\hat{b}$, so we can rename $V(x,\hat{b})=\tilde{V}(\tilde{b})$. Recognize $\tilde{V}(\tilde{b})=\tilde{b}^T\tilde{\alpha}$ to obtain Equation 10. Also, we can rewrite any policy $\tilde{\pi}:\triangle^r\to A$ as the controller family policy $\pi=\langle X,\psi,\eta,\sigma\rangle$ by assigning the elements be as above. ■

Exponential Family Principle Component Analysis E-PCA computes a non-linear compression of the reachable beliefs using E-PCA [6] and applies *fitted value iteration* [30] on the resultant reduced belief MDP. E-PCA is used over PCA to better represent the fact that beliefs are probabilities. This generalized form requires a *link function* $\ell:\triangle^n\to\triangle^n$. The exponential link function is given by $\ell(F\tilde{b})=\exp\{F\tilde{b}\}$ with $F\in\mathbb{R}^{n\times r}$. It enables us to recover a belief $b\approx\hat{b}$ with $\hat{b}=\exp\{F\tilde{b}\}$. Thus, the link function acts an inverse $f^{-1}(\tilde{b})=\hat{b}=\ell(F\tilde{b})$. In general, the link function determines the type of exponential family random variable. A loss function is then defined by minimizing the generalized Bregman divergence between b and $\hat{b}$. For the choice of an exponential link function, we have a Poisson belief error model, and equates to minimizing unnormalized KL divergence. The solutions to F and $\tilde{B}$ are convex optimization problems solvable using the partial derivatives of this loss function. The process for $\tilde{B}$ is also used to project belief b to the compressed belief space $\tilde{b}$, defining $f(b)=\tilde{b}$. Fitted value iteration (FVI) is used in the E-PCA compressed belief MDP with a averager function approximator. E-PCA policy $\tilde{\pi}:\tilde{B}\to A$ is determined from value $\tilde{V}:\tilde{B}\to\mathbb{R}$:

$$\tilde{V}(\tilde{b})=\max_{a\in A}\tilde{R}(\tilde{b},a)+\gamma\sum_{\tilde{b}'\in\tilde{B}}\tilde{T}(\tilde{b},a,\tilde{b}')\tilde{V}(\tilde{b}') \qquad (11)$$

with k-nearest neighbors computing the nearest $k>0$ belief neighbors such that $w:\tilde{B}\times\triangle^r\to[0,1]$ and $w(\tilde{b},\tilde{b}')=(1/k)$ if $\tilde{b}$ is one of k closest beliefs from $\tilde{B}$ for $\tilde{b}'$, with $w(\tilde{b},\tilde{b}')=0$ otherwise. The reward and state transition are defined, with $\hat{b}=f^{-1}(\tilde{b})$ and $\tilde{b}'_{a\omega}=f(\hat{b}'_{a\omega})$, as:

$$\tilde{R}(\tilde{b},a)=\sum_{s\in S}\hat{b}(s)R(s,a) \qquad (12)$$

$$\text{and}\quad \tilde{T}(\tilde{b},a,\tilde{b}')=\sum_{\omega\in\Omega}Pr(\omega|\hat{b},a)w(\tilde{b}',\tilde{b}'_{a\omega}).$$

Lemma 2: The E-PCA policy is a member of the controller family.

Proof: We must write Equation 11 as Equation 5. Let $X=\tilde{B}$ be all compressed beliefs considered, with $\hat{b}=f^{-1}(\tilde{b})$. Let $\psi(x,b,a)=[Q(\hat{b},a)\geq Q(\hat{b},a')\forall a']$, since E-PCA's FVI uses an optimality form. Let $\eta(x,b,a,\omega,x')=w(\tilde{b}',\tilde{b}'_{a\omega})$, noting that from $x=\tilde{b}$ we compute $\hat{b}$, then $\hat{b}'_{a\omega}$, and finally $\tilde{b}'_{a\omega}$. Let $\sigma(x,b)=V(x,\hat{b})$ with $\hat{b}$ computed from $x=\tilde{b}$. By the definition of $\sigma(x,b)$, the only beliefs ever visited are the reconstructed $\hat{b}$ corresponding to x. Apply this to Equation 5 to yield:

$$V(x,\hat{b})=\max_{a\in A}\sum_{s\in S}\hat{b}(s)R(s,a)\\ +\gamma\sum_{x'\in X}\sum_{\omega\in\Omega}Pr(\omega|\hat{b},a)w(\tilde{b}',\tilde{b}'_{a\omega})V(x',\hat{b}')$$

with the definition of R and reordering of summations. Rename $V(x,\hat{b})=\tilde{V}(\tilde{b})$ since $\tilde{b}$ uniquely determines x and $\hat{b}$. Recognize $\tilde{R}$ and $\tilde{T}$ to obtain Equation 11. Also, we can rewrite any policy $\tilde{\pi}:\triangle^r\to A$ as the controller family policy $\pi=\langle X,\psi,\eta,\sigma\rangle$ by assigning the elements as above. ■

Compression Policy Formulations Proposition 4 formally combines the above lemmas that show linear and non-linear compression forms are generalized and remarks on other subsumed methods.

Proposition 4: Compression policies are members of the controller family.

Proof: By Lemmas 1 and 2, we find that the controller family generalizes compression equations and policy representations. Since VDC generalizes state aggregation [31], model minimization [32], and linear predictive state representations (PSR) [33], so too does the controller family. Additionally, since the proof is independent of ℓ, controller family generalizes any choice of ℓ for E-PCA. ■

Insights (1) Node selection *is the compression technique itself*, since X is the compressed or reconstructed beliefs. (2) Function approximators σ make value constant over the belief; instead, the value's belief (i.e., b in $V(x,b)$) is replaced with node's reconstructed belief. (3) Node selection are thus assigned to ensure the node's belief properly updates.

Domain		BI-FSC-Based Policy			FSC-Based Policy	
Name	$\|N\|$	Time	ADR	$\|B\|$	Time	ADR
Aloha-10	10	109.2	**523.0**	50	**15.0**	163.0
Grid-4x3	5	**16.4**	**0.83**	10	18.8	0.58
Hallway2	7	155.4	0.25	10	**67.2**	0.27
Tiger	3	**13.8**	**11.7**	6	20.7	-20.0

TABLE I
RESULTS USING NEW POLICY FORM WITH SAME NLP ALGORITHM AND SAME NUMBER OF FSC NODES ($|N|$). METRICS: TIME (SECONDS, 10 TRIALS) AND AVERAGE DISCOUNTED REWARD (ADR, 100 TRIALS). BOLD INDICATES STATISTICALLY SIGNIFICANT IMPROVEMENT.

V. EVALUATION

Now we demonstrate the efficacy of the controller family as a tool for describing novel policy and value representations. Importantly, the main contribution of this paper remains the formulation of controller family and theoretical analysis, rather than this new example of a controller family policy. Recall, the controller family is *not an algorithm*; it is a representation of policy and value. This is included to provide evidence of its usefulness and guidance for how to create new forms of policy.

Intuitively, we define a belief-integrated FSC as a member of the controller family with some FSC nodes with some belief point nodes. FSC nodes follow the free stochastic successor node selector (e.g., NLP). Beliefs nodes follow an argmax (e.g., PBVI). The belief points used, however, are reconstructed from compressed beliefs (e.g., E-PCA). The resulting value equation is amenable to node iteration via PBVI's expand step followed by E-PCA, and policy iteration via an NLP. Formally, a **belief-integrated FSC (BI-FSC)** is a controller family policy $\pi = \langle X, \psi, \eta, \sigma \rangle$ that has its nodes X represent both FSC nodes *and* compressed belief-points. Let $X = N \cup \tilde{B}$ with N a set of FSC nodes and $\tilde{B} \subseteq \triangle^n$. Let $\lambda \in [0,1]$ be a weight between both approaches. Let $\eta(x,b,a,\omega,x')$ be either: $(1-\lambda)\bar{\eta}(x,b,a,\omega,x')$ if $x,x' \in N$; $\lambda\,\mathrm{softmax}$ over $\tilde{B}$ if $x \in N$ and $x' \in \tilde{B}$; or $\bar{\eta}(x,b,a,\omega,x')$ if $x \in \tilde{B}$ and $x' \in N$. We can use a Taylor series approximation of $\mathrm{softmax}$ (e.g., the 0-th order is $1/|\tilde{B}|$). Let $\sigma(x,b)$ be either: $\sum_s b(s)V(x,s)$ if $x \in N$; or $\sum_s \hat{b}'_{a\omega}(s)V(x,s)$ if $x \in \tilde{B}$ and $\hat{b} = f^{-1}(x)$.

Understanding the BI-FSC Policy Form What does this novel policy form look like in practice? Figure 2 shows results from real robot POMDP navigation. The POMDP has S as a 7×5 grid, A as eight directions and stop, and Ω as the "bump" sensor. We solve the BI-FSC with *nova* [34] using SNOPT [35] on the NEOS Server [36]. The traversed path includes intentional "feeling" for walls with successful localization. Interestingly, this policy uses the interspersed belief nodes' decisions to help guide the stochastic FSC nodes' decisions to the goal (blue and green in Figure 2).

How does this new policy form improve performance over just a pure FSC policy form? Table I shows results of a pure FSC-based policy versus a BI-FSC-based policy ($\lambda = 0.5$) on standard benchmark domains. The NLP algorithm is used by *both* policy forms to *isolate a direct comparison* of different policy forms, instead of the algorithms used on top of a chosen policy form. Experiments were run with SNOPT on the NEOS Server. BI-FSC policies are suited for domains that have many local optima (e.g., Aloha-10 and Tiger) in which solvers like SNOPT easily get stuck. BI-FSCs obviate the issue by infusing important beliefs to help find the global optima. Thus, BI-FSCs are able to greatly outperform FSC-based policies in ADR (especially Aloha-10 and Tiger) and can even improve time.

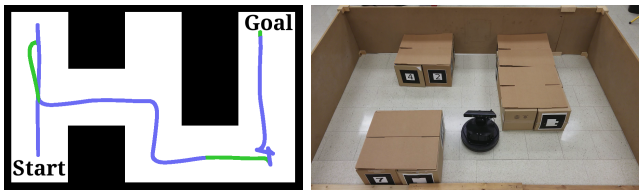

Fig. 2. Robot experiment. Path: FSC (blue), belief (green) actions.

VI. CONCLUSION

We now revisit the questions from the introduction. How can we *simultaneously* integrate belief point-based and FSC-based techniques? Nodes can be defined to be beliefs *and* FSC nodes, with the successor node selector to be a free variable or argmax. Is there a way to introduce compression into policy forms? Simply select compressed or reconstructed beliefs for some or all nodes; other elements (e.g., successor node selectors) use these reconstructed beliefs. Can we automatically select some of the nodes (i.e., *node iteration*)? Simply use any prior approach appropriate to the type of node—belief nodes use belief exploration (e.g., HSVI2), FSC nodes use node addition/pruning (e.g., BPI), or compressed nodes use compression (e.g., E-PCA). In general, how are iteration techniques related? Each is an operator on a different type of node as described above.

Lastly, what is the mathematical reason for the existence of value and policy iteration? From the perspective offered by the controller family, it is determined by the action selector ψ. When ψ is constrained to a fixed function, we have value iteration. This assignment prevents policy iteration because the policy is determined by the other free elements; the values V determine the mapping from belief to action. Conversely, when ψ is at all unconstrained, V and ψ are both free and can be computed separately (e.g., policy iteration's evaluation/improvement steps) or together (e.g., NLP).

In conclusion, is there an underlying principled framework to design POMDP policies? This paper defines the controller family as an answer to this question. We show they generalize the policy and value representations used by state-of-the-art solutions. To validate its effectiveness, we construct a novel policy formulation that infuses beliefs into an FSC. We demonstrate this improved policy form's execution on a real robot acting in the world, and show it overcomes some well-known issues with a vanilla FSC. Finally, we will provide our source code with the goal of building new controller family policies to improve POMDP solutions under this unified formal language with the greater research community.

REFERENCES

[1] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, no. 1, pp. 99–134, 1998.

[2] J. Pineau, G. Gordon, and S. Thrun, "Anytime point-based approximations for large POMDPs," *Journal of Artificial Intelligence Research*, vol. 27, pp. 335–380, 2006.

[3] K. H. Wray and S. Zilberstein, "Approximating reachable belief points in POMDPs," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017, pp. 117–122.

[4] C. Amato, D. S. Bernstein, and S. Zilberstein, "Optimizing fixed-size stochastic controllers for POMDPs and decentralized POMDPs," *Autonomous Agents and Multi-Agent Systems*.

[5] K. H. Wray, A. Kumar, and S. Zilberstein, "Integrated cooperation and competition in multi-agent decision-making," in *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018, pp. 4751–4758.

[6] N. Roy, G. Gordon, and S. Thrun, "Finding approximate POMDP solutions through belief compression," *Journal of Artificial Intelligence Research*, vol. 23, pp. 1–40, 2005.

[7] P. Poupart, "Exploiting structure to efficiently solve large scale partially observable Markov decision processes," Ph.D. dissertation, University of Toronto, 2005.

[8] M. J. Kochenderfer, *Decision Making Under Uncertainty: Theory and Application*. MIT Press, 2015.

[9] K. H. Wray, S. J. Witwicki, and S. Zilberstein, "Online decision-making for scalable autonomous systems," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 2017, pp. 4768–4774.

[10] K. H. Wray, L. Pineda, and S. Zilberstein, "Hierarchical approach to transfer of control in semi-autonomous systems," in *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, 2016, pp. 517–523.

[11] K. H. Wray and S. Zilberstein, "Multi-objective POMDPs with lexicographic reward preferences," in *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, 2015, pp. 1719–1725.

[12] E. J. Sondik, "The optimal control of partially observable Markov processes over the infinite horizon: Discounted costs," *Operations Research*, vol. 26, no. 2, pp. 282–304, 1978.

[13] K. P. Murphy, "A survey of POMDP solution techniques," University of California Berkeley, Tech. Rep., 2000.

[14] D. Braziunas, "POMDP solution methods," University of Toronto, Tech. Rep., 2003.

[15] C. Boutilier, T. Dean, and S. Hanks, "Decision-theoretic planning: Structural assumptions and computational leverage," *Journal of Artificial Intelligence Research*, vol. 11, pp. 1–94, 1999.

[16] B. Bonet and H. Geffner, "Planning and control in artificial intelligence: A unifying perspective," *Applied Intelligence*, vol. 14, no. 3, pp. 237–252, 2001.

[17] G. Shani, J. Pineau, and R. Kaplow, "A survey of point-based POMDP solvers," *Autonomous Agents and Multi-Agent Systems*, pp. 1–51, 2013.

[18] W. S. Lovejoy, "Computationally feasible bounds for partially observed Markov decision processes," *Operations Research*, vol. 39, no. 1, pp. 162–175, 1991.

[19] N. L. Zhang and W. Zhang, "Speeding up the convergence of value iteration in partially observable Markov decision processes," *Journal of Artificial Intelligence Research*, vol. 14, pp. 29–51, 2001.

[20] M. Spaan and N. Vlassis, "Perseus: Randomized point-based value iteration for POMDPs," *Journal of Artificial Intelligence Research*, vol. 24, pp. 195–220, 2005.

[21] T. Smith and R. Simmons, "Point-based POMDP algorithms: Improved analysis and implementation," in *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*, 2005, pp. 542–549.

[22] H. Kurniawati, D. Hsu, and W. S. Lee, "SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces." in *Robotics: Science and systems*, 2008.

[23] E. A. Hansen, "Solving POMDPs by searching in policy space," in *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, 1998, pp. 211–219.

[24] P. Poupart and C. Boutilier, "Bounded finite state controllers," in *Proceedings of Advances in Neural Information Processing Systems 16*, 2004, pp. 823–830.

[25] S. Ji, R. Parr, H. Li, X. Liao, and L. Carin, "Point-based policy iteration," in *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, 2007, pp. 1243–1249.

[26] A. Kumar and S. Zilberstein, "History-based controller design and optimization for partially observable MDPs," in *Proceedings of the 25th International Conference on Automated Planning and Scheduling*, 2015, pp. 156–164.

[27] X. Li, W. K. Cheung, J. Liu, and Z. Wu, "A novel orthogonal NMF-based belief compression for POMDPs," in *Proceedings of the 24th International Conference on Machine Learning*, 2007, pp. 537–544.

[28] G. Theocharous and S. Mahadevan, "Compressing POMDPs using locality preserving non-negative matrix factorization," in *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, 2010, pp. 1147–1152.

[29] Z. Wang, P. A. Crook, W. Tang, and O. Lemon, "On the linear belief compression of POMDPs: A re-examination of current methods," *arXiv preprint arXiv:1508.00986*, 2015.

[30] G. J. Gordon, "Stable function approximation in dynamic programming," in *Proceedings of the 12th International Conference on Machine Learning*, 1995, pp. 261–268.

[31] C. Boutilier and D. Poole, "Computing optimal policies for partially observable decision processes using compact representations," in *Proceedings of 13th National Conference on Artificial Intelligence*, 1996, pp. 1168–1175.

[32] R. Givan, T. Dean, and M. Greig, "Equivalence notions and model minimization in Markov decision processes," *Artificial Intelligence*, vol. 147, no. 1, pp. 163–223, 2003.

[33] M. L. Littman, R. S. Sutton, and S. Singh, "Predictive representations of state," in *Proceedings of Advances in Neural Information Processing Systems 14*, 2002, pp. 1555–1561.

[34] K. H. Wray and S. Zilberstein, "A parallel point-based POMDP algorithm leveraging GPUs," in *Proceedings of the 2015 AAAI Fall Symposium on Sequential Decision Making for Intelligent Agents*, 2015, pp. 95–96.

[35] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Review*, vol. 47, no. 1, pp. 99–131, 2005.

[36] J. Czyzyk, M. P. Mesnier, and J. J. Moré, "The NEOS Server," *IEEE Computational Science and Engineering*, vol. 5, no. 3, pp. 68–75, July 1998.