Active Inference for Integrated State-Estimation, Control, and Learning.

Mohamed Baioumy  Paul Duckworth  Bruno Lacerda  Nick Hawes

Abstract—This work presents an approach for control, state-estimation and learning model (hyper)parameters for robotic manipulators. It is based on the active inference framework, prominent in computational neuroscience as a theory of the brain, where behaviour arises from minimizing variational free-energy. The robotic manipulator shows adaptive and robust behaviour compared to state-of-the-art methods. Additionally, we show the exact relationship to classic methods such as PID control. Finally, we show that by learning a temporal parameter and model variances, our approach can deal with unmodelled dynamics, damps oscillations, and is robust against disturbances and poor initial parameters. The approach is validated on the ‘Franka Emika Panda’ 7 DoF manipulator.

Supplementary Material

Code: https://github.com/MoBaioumy/active_inference_panda_paper
Video: https://youtu.be/38cEu-TkAXc

I. INTRODUCTION

Since it is infeasible to model all time-varying dynamics and disturbances a priori, it is crucial that intelligent robotics systems are able to adapt to the presence of unmodeled dynamics and model uncertainties. This is necessary for many applications such as aerial vehicles encountering unpredicted wind dynamics [1], manipulators handling objects of unknown weight [2] and autonomous vehicles on slippery road surfaces [3]. There exists a verity of methods to deal with such problems. For example, machine learning methods for either learning inverse dynamics or tuning conventional controllers. These require large amounts of training data and several iterations for learning [4], [5]. On the other hand, approaches from adaptive control such as Model Reference Adaptive Control suffer from scalability issues with the number of degrees of freedom [6], [7].

Humans are skilled in dealing with these problems and recent work in robotics has taken inspiration from ‘active inference’ [8], a neuroscientific theory of the brain and behaviour. Active inference provides a framework for understanding decision-making of biological agents. Under the active inference framework, optimal behavior arises from minimising variational free-energy: a measure of the fit between an internal model and (past) sensory observations. Additionally, agents act to fulfill prior beliefs about preferred future observations [9]. This framework has been employed to explain and simulate a wide range of behaviors including abstract rule learning [10], planning and navigation [11].

In [7], the authors use the active inference framework for joint space control of a robotic manipulator. The active inference controller (AIC) outperforms the state-of-the-art Model Reference Adaptive Control (MRAC) in tasks requiring adaptive behaviour. The active inference approach of [7] avoids the scalability issues of MRAC by requiring only a fixed number of parameters. However, this approach has a few limitations including sensitivity to its initial parameters. By slightly changing one of the model parameters or variances, the system suffers from severe oscillations and never settle at its target state.

The contributions of this paper are: 1) We present a novel formulation of active inference which includes the introduction of a temporal parameter $\tau$ that affects the controller performance. 2) We show that under this framework, we are able to derive approaches to automatically update the model hyperparameters including the variances as well as the introduced parameter $\tau$. This ensures the controllers is tuned properly during run-time.

Our approach provides a number of benefits: learning happens during execution-time, it has a fixed number of parameters to be specified regardless of the DoF (number of degrees of freedom), it does not require an accurate model of the dynamics, and it is robust to poor initial settings.

Additionally, we highlight theoretical results that show, if our introduced parameter $\tau$ approaches zero ($\tau^{-1} \to \infty$), the approach converts to a classic PID controller. When $\tau$ approaches $\infty$ ($\tau^{-1} \to 0$), the approach converts to a pure estimator. As a running example throughout the paper, the mass-spring-damper [12] system is used. However, we validate the approach on a ‘Franka Emika Panda’ 7 DoF
manipulator in the results section. Our approach is tested against previous work to carry different payloads and with different hyperparameters.

II. RELATED WORK

There exists a variety of methods to allow robotic systems to show adaptive and robust behaviour. Some work considers conventional controllers coupled with artificial intelligence methods. For example in [13], [14] PID controllers are tuned with fuzzy logic. Other approaches rely on estimating accurate inverse dynamic models using machine learning methods [4]. These approaches require large amount of training data and several iterations for learning [5], are hard to generalize [15], [16], and require expert definition (for instance, for the best topology of a neural network [17]).

Existing adaptive control approaches include self-tuning adaptive control, which represent the robot as a linear discrete-time model and estimate the unknown parameters online, substituting them in the control law [18]. Another method is Model Reference Adaptive Control (MRAC) [6], which finds a control signal to be applied to the robot actuators. This signal will force the system to behave as specified by a chosen reference model; however, these approaches suffer from scalability issues. For instance, [7] reports that in order to use the MRAC on a real 7 DoF robot previously tuned in simulation, a severe re-tuning of 63 parameters had to be performed.

The framework of active inference has the potential to facilitate building intelligent, robust and adaptive robotic systems. However there are only a handful of attempts to use active inference in robotics and control. In [19], a PR2 robot, simulated in ROS, is controlled by open-loop Active Inference; however, the computational complexity made an online implementation unfeasible. In [20], the authors use free-energy minimization for adaptive body perception. In [21] an implementation of active inference is presented with real hardware on a 3 DoF humanoid robot. It was capable of performing reaching behaviors with both arms and active head object tracking in the visual field with noisy observation. However, the control was performed using velocity commands rather than torque commands. This assumes the reliability of the low-level controllers to achieve the requested velocities.

The work about active inference described above mostly included simple tasks and wasn’t compared to state-of-the-art methods. In [7], a method for joint space control (using torque commands rather than velocity commands) of robotic manipulators is presented that outperforms the state-of-the-art MRAC in adaptability. Therefore, we use the work in [7] as comparison for our results.

III. ACTIVE INFERENCE FRAMEWORK

This section introduces active inference as a general framework and derives the key equations for free-energy ($F$). The free-energy term, $F$, is used in later sections to achieve state-estimation, control and hyperparameter learning.

A. Variational free energy

Active Inference considers an agent in a dynamic environment that receives an observation $o$ about a state $s$. Given a model of the agent’s world, Bayes’ rule can be used to infer the state as

$$p(s|o) = \frac{p(s)p(o|s)}{p(o)} = \frac{p(s)p(o|s)}{\int p(s)p(o|s)ds}$$

(1)

Performing such a calculation is computationally expensive especially if $p(o|s)$ is a non-standard distribution. The normalization term $p(o)$ involves calculating an integral making calculations of all but trivial examples infeasible. Instead, the agent can approximate the posterior distribution $p(s|o)$ with a ‘variational distribution’ over states $Q(s)$ which we can define to have a more simple form (such as a Gaussian). The goal is then to minimize the difference between the two distributions. The mismatch between the two distribution can be computed using the KL-divergence [22]:

$$KL(Q(s)||p(s|o)) = \int Q(s) \ln \frac{Q(s)}{p(s, o)} ds + \ln p(o)$$

$$= F + \ln p(o).$$

(2)

The quantity $F$ is referred to as the (variational) free-energy and minimizing $F$ minimizes the KL-divergence. $F$ is also often referred to as the evidence lower bound (ELBO) in the Machine Learning community. If we choose $Q(s)$ to be a Gaussian distribution with mean $\mu$, and utilize the Laplace approximation [23], the free-energy expression simplifies to:

$$F \approx -\ln p(\mu, o).$$

(3)

Now the expression for variational free-energy is solely dependent on one parameter, $\mu$, which is referred to as the ‘belief state’ or simply ‘belief’. The objective is to find the $\mu$ which minimizes $F$. This results in the agent finding the best estimate of its state. For a robotic manipulator the set of observations ($o$) and beliefs ($\mu$) are vectors with length depending on the number of degrees of freedom.

B. Generalized motions

The state of the robot is given by its joint position, velocities $\hat{\mu} = [\mu, \mu']$. In the Active Inference literature the state given by position and higher derivatives is also known as generalised coordinates of motions [24]. We also consider observations given by joint encoders, $\bar{o} = [o, o']$, which are the main source of information about the state.

C. Observation model and state transition model

Taking generalized motions into account, the joint probability from Equation (3) can be written as:

$$p(\bar{o}, \hat{\mu}) = p(\bar{o}|\hat{\mu}) p(\hat{\mu}) = p(o|\mu)p(o'|\mu')p(\mu|\mu)p(\mu'|\mu'),$$

(4)

1Minimizing the ELBO and thus the KL divergence is common in variational inference, a method for approximating probability densities [22]. Active inference on the otherhand is a framework that utilizes variational inference to explain behaviour of biological agents.
where \( p(o|\mu) \) is the probability of receiving an observation \( o \) while in (belief) state \( \mu \), and \( p(\mu'|\mu) \) is the state transition model (also referred to as the dynamic model or the generative model). The state transition model predicts the state evolution given the current state. These distributions are assumed Gaussian according to:

\[
\begin{align*}
  p(o|\mu) &= N(g(\mu), \Sigma_o), \quad p(o'|\mu') = N(g'(\mu'), \Sigma_{o'}), \\
  p(\mu'|\mu) &= N(f(\mu), \Sigma_f), \quad p(\mu''|\mu') = N(f'(\mu'), \Sigma_{f'}),
\end{align*}
\]

(5)

where the functions \( g(\mu) \) and \( g'(\mu') \) represent a mapping between observations and states. For many applications in robotics the state is directly observable. For instance, in the context of a robotic manipulator the state consists of the positions and velocities of all joints and the manipulator is provided with position and velocity encoders. Thus we can assume: \( g(\mu) = \mu \) and \( g'(\mu') = \mu' \). The functions \( f(\mu) \) and \( f'(\mu') \) represent the evolution of the belief state over time. This encodes the agent’s preference over future states (in this case the preferred future state is the target state, \( \mu_d \)). In our case \( f(\mu) = (\mu_d - \mu)\tau^{-1} \) and \( f'(\mu') = \tau^{-1} \mu' \), where \( \mu_d \) is the desired state and \( \tau \) a time scale (explained further in Section IV-D). In [7], \( \tau \) is set to one by default but we will show the limitations of that choice.

Now that all the terms have been defined, \( F \) can be expanded to:

\[
F = \frac{1}{2} \left( \varepsilon_o^\top \Sigma_{o}^{-1} \varepsilon_o + \varepsilon_{o'}^\top \Sigma_{o'}^{-1} \varepsilon_{o'} + \varepsilon_{\mu}^\top \Sigma_{\mu}^{-1} \varepsilon_{\mu} + \varepsilon_{\mu'}^\top \Sigma_{\mu'}^{-1} \varepsilon_{\mu'} + \ln |\Sigma_o| + \ln |\Sigma_{o'}| + \ln |\Sigma_\mu| + \ln |\Sigma_{\mu'}| \right) + C,
\]

(6)

where \( \varepsilon_o = o - \mu \) and \( \varepsilon_{o'} = o' - \mu' \). \( C \) refers to constant terms.

Equation (6) differs from the work presented in [7] in the third line (terms with \( \ln |\Sigma| \), which are not included). The importance of this difference is highlighted in Section V-A.

### IV. State-estimation and Control

We now introduce how to perform state-estimation and control by minimizing \( F \). We show how the estimation step uses the observations to refine its (state) estimate and then biases that estimate towards the goal state. The control step then steers the system from its observed state \( o \) to its estimated state \( \mu \). This can be considered an estimator coupled with a moving target PID. In Section IV-E we show that if \( \tau^{-1} \to \infty \), the approach converts to a classic PID controller. Additionally, if \( \tau^{-1} \to 0 \), the approach converts to a pure estimator.

#### A. State estimation by minimizing free-energy

Estimating the state of our system is achieved by finding a value \( \mu \) that minimizes \( F \). If we are able to compute the gradient of \( F \) with respect to the \( \mu \), gradient descent is a simple way to accomplish that:

\[
\hat{\mu} = D\mu - \kappa_\mu \frac{\partial F}{\partial \mu}.
\]

(7)

Where \( \kappa_\mu \) is a tuning parameter and \( D \) is a temporal derivative operator. Using Equation (7) the agent takes one-step in the gradient descent at every time-step. In this case the equation expands to:

\[
\begin{align*}
\mu' &= \mu + \kappa_\mu \Sigma^{-1} \varepsilon_o - \tau^{-1} \kappa_\mu \Sigma^{-1} \varepsilon_{\mu'} \\
\mu'' &= \mu'' + \kappa_\mu \Sigma^{-1} \varepsilon_{\mu'} - \kappa_\mu \Sigma^{-1} \varepsilon_{\mu''} - \tau^{-1} \kappa_\mu \Sigma^{-1} \varepsilon_{\mu'} \quad (8)
\end{align*}
\]

The first equation states that belief is refined using the term \( \kappa_\mu \Sigma^{-1} \varepsilon_o \) which moves our new estimate towards the value just observed. Additionally, the term \( \tau^{-1} \kappa_\mu \Sigma^{-1} \varepsilon_{\mu'} \) shifts the belief towards the target \( \mu_d \) \( (\mu' = \mu' - (\mu_d - \mu)\tau^{-1}) \). Essentially, this ‘biases’ the estimate towards preferred future states (the preferred future observation is this case is the target state \( \mu_d \)). The degree to which the system is biased (rather than refined using the observation) depends on the values \( \tau^{-1} \) and \( \Sigma^{-1} \).

#### B. Control by minimizing free-energy

Similar to state-estimation, to find the control action which minimizes \( F \), gradient descent is used:

\[
\hat{a} = -\kappa_a \frac{\partial F}{\partial a} = -\kappa_a \frac{\partial F}{\partial o} \frac{\partial o}{\partial a},
\]

(9)

where \( \kappa_a \) is a tuning parameter. The term \( \frac{\partial o}{\partial a} \) is assumed linear, and equal to the identity matrix (multiplied by a constant) similar to existing work [7], [21]. Actions are then computed as:

\[
\hat{a} = -\kappa_a (\Sigma^{-1} (o - \mu) + \Sigma^{-1} (o' - \mu')).
\]

(10)

This controller essentially steers the system from its observed state \( o \) to its estimated state \( \mu \).

Note how the current control law does not contain any information about the dynamical system, it is thus a reactive controller. The control law only requires \( o \) and \( \mu \) which is biased towards the future desired state \( \mu_d \). This controller thus operates in the presence of unmodelled dynamics similar to a PID controller.

#### C. Simultaneous state-estimation and control

The presented approach performs state estimation and control simultaneously. The estimation and control step are dependent. This is because the estimation step refines the belief using the observation and biases the belief \( \mu \) towards the target \( \mu_d \). The controller then steers the system from the observation \( o \) to the refined then biased estimated state \( \mu \). If \( \tau^{-1} \) and \( \Sigma^{-1} \) are large, the estimate \( \mu \) is biased more towards the target \( \mu_d \).

To illustrate this, consider the mass damper system. It’s given by the equation: \( \ddot{x} = a(t) - k_1 x - k_2 \dot{x} \), where \( x \) is the position of the mass, \( a(t) \) the control action, \( k_1 \) the spring constant (set to \( 1N/m \)), \( k_2 \) the damper coefficient (set to \( 0.1Ns/m \)) and the system has unit mass. It’s simulated with initial conditions \( x(0) = -0.5m, \dot{x}(0) = -1m/s \) and \( a(t) = 0N \). Equations (8) are used separately performing state estimation. To challenge the system, the initial beliefs...
Position (m)

−0.50
−0.25

Position (m)

0.00
0.25
0.50
1.00
1.25

Fig. 2: Separately performing state-estimation for different values of $\tau^{-1}$. Higher values of $\tau^{-1}$ give more bias towards the target.

Control behaviour for different values of $\tau^{-1}$

−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
1.25

Fig. 3: Control for different values of $\tau^{-1}$. Higher values of $\tau^{-1}$ provide more bias towards the target and thus more aggressive control causing overshoot oscillations.

are inaccurate ($\mu(0) = 0m$ and $\mu'(0) = -1.5m/s$). The system is simulated for different values of $\tau^{-1}$ and presented in Figure 2. Code for the simulations is available on the accompanying Github repository.

It’s clear that higher values of $\tau^{-1}$ give more bias towards the target. For $\tau^{-1} = 8$ (green line), the estimate is close to the target (black dashed line) and far away from the actual position (blue dashed line) as opposed to setting $\tau^{-1} = 0.1$ (red line), the estimate better follows the real trajectory (not perfectly since the observations are noisy and the trajectory is highly non-linear). If $\tau^{-1} \rightarrow 0$, the estimation step reduces to a pure estimator, which would follow the trajectory without any bias towards the target.

Enabling control steers the system to its target. The $\tau^{-1}$ in this case determines how aggressive the controller is. Since larger values move the estimate more towards the target, the difference $(o - \mu)$ is larger and thus the controller is more aggressive. An illustration for different values of $\tau^{-1}$ is shown in Figure 3.

D. Understanding temporal parameter $\tau$

Recall, the generative model specified by Equations 4 and 5 includes the function $f(\mu)$ which determines how the state evolves over time, $f(\mu) = (\mu_d - \mu)\tau^{-1}$.

How the state is specified to evolve over time is the derivative between the current state $\mu$ and target $\mu_d$. This can be evaluated as the $(\mu_d - \mu)$ divided by a time scale $\tau$. The smaller $\tau$, the larger the derivative. If $\tau$ approaches zero ($\tau^{-1} \rightarrow \infty$), the function $f(\mu)$ approaches $\infty$. As a result, the estimate is infinitely biased towards the target and $\mu \approx \mu_d$.

E. Relationship to a classic PID Controller

A classic PID controller defines an error term $e = (\mu_d - o)$. The control law is then designed as

$$a = P \cdot e + I \int e dt + D \frac{de}{dt},$$

where ‘P’, ‘I’ and ‘D’ are tuning parameters.

For the control law defined by active inference, our $(o - \mu)$ is similar to the error term. Additionally, as explained in the previous section, when $\tau^{-1} \rightarrow \infty$ then $\mu \approx \mu_d$. Now the control law of active inference can be rewritten in terms of the error term as:

$$\dot{a} = \kappa_o \Sigma_o^{-1} e + \kappa_o \Sigma_o^{-1} \frac{de}{dt}.$$

This means that if $\tau^{-1} \rightarrow \infty$, the active inference controller is equivalent to a PI Controller (PID with $D = 0$) with a ‘P’ gain of $\kappa_o \Sigma_o^{-1}$ and an ‘I’ gain of $\kappa_o \Sigma_o^{-1}$. If one considers the generalized motions (from section III-B) up to a third order rather than a second, the resulting control law would include a non-zero ‘D’ terms.

The relationship to a pure estimator is straightforward. As previously mentioned, if $\tau^{-1} \rightarrow 0$, the estimation step reduces to a pure estimator. Essentially, this indicates, the estimation step has zero bias towards the target. As Figure 2 has shown, for very small values of $\tau^{-1}$, the estimator follows the real position without bias.

V. LEARNING HYPERPARAMETERS AS ACTIVE INFEERENCE

We have shown that state estimation and control can be performed using gradient decent on the free-energy $F$. The same applies to the hyperparameters. Estimating $\tau$ and the model variances is done using gradient decent on $F$.

A. Learning model variances

As illustrated in Section IV-E, the model variances $\Sigma_o^{-1}$ and $\Sigma_o'$ can be considered as gains for the controller, similar to the ‘P’ and ‘I’ gains in a PID controller. Additionally, the values $\Sigma_o^{-1}$ and $\Sigma_o'$ affect how much the estimation step biases the controller towards the desired position ($\tau^{-1}$ also affects the bias towards the target).

We can update $\Sigma_o$ and $\Sigma_o'$ using gradient decent on $F$ as:

$$\dot{\Sigma}_o = -\kappa_o \frac{\partial F}{\partial \Sigma_o}, \quad \dot{\Sigma}_o' = -\kappa_o \frac{\partial F}{\partial \Sigma_o'}.$$  (11)

The presented update rules have several practical issues. First, in any high dimensional case, $\Sigma_o$ would be a matrix. Since in most equations presented so far, the inverse $\Sigma_o^{-1}$ is used, updating the covariance matrix using Equations 11 then inverting it would be computationally expensive. A work
around is to simply update the inverse covariance matrix, sometimes referred to as the precision matrix or information matrix, as done in [25].

\[
\Sigma_o^{-1} = -\kappa_\sigma \frac{\partial F}{\partial \Sigma^{-1}_o}, \quad \Sigma_{\mu}^{-1} = -\kappa_\mu \frac{\partial F}{\partial \Sigma^{-1}_{\mu}}. \tag{12}
\]

The second issue is that a covariance needs to be positive and a covariance matrix needs to be positive semi-definite. However, the update rules from Equations 12 may violate these conditions. One way to avoid this problem is by setting a positive lower bound on the variance (as suggested in [26]). In the case of the covariance matrix, all diagonal elements have a positive lower bound and all non-diagonal elements are set to zero. Other workarounds are suggested in the discussion.

We demonstrate this using \(\Sigma_{\mu}^{-1}\) fixed at 0.5 and \(\Sigma_\mu^{-1}\) will be varied. If \(\Sigma_\mu^{-1}\) is too high, the systems suffers from oscillations and overshoot. However, if \(\Sigma_\mu^{-1}\) and \(\Sigma_o^{-1}\) are updated during run-time, the controller shows improved behaviour. Results are shown in Figure 4.

The convergence of \(\Sigma_o^{-1}\) occurs when \(\frac{\partial F}{\partial \tau^{-1}} = 0\). Since the observations change over time and have a certain level of noise, \(\Sigma_o\) converges to the expected value of \(\mathbf{e}_\epsilon \mathbf{e}_\epsilon^T\). This does not necessarily happen upon reaching the target state.

### B. Learning the temporal parameter

Figure 3 showed the importance of choosing appropriate values for \(\tau^{-1}\): if the value is too high, the controller suffers from overshoot and oscillations. On the other hand, a low value results in a slow response. Ideally, the value for \(\tau\) would be high in the start but decrease as the system reaches the target. This would essentially be the value that minimizes \(F\) and can be found using gradient descent on \(F\) as:

\[
\dot{\tau}^{-1} = -\kappa_\tau \frac{\partial F}{\partial \tau^{-1}}
\]

\[
\frac{\partial F}{\partial \tau^{-1}} = -2\Sigma_\mu' \mathbf{e}_\mu (\mathbf{d} - \mu) + 2\Sigma_\mu' \mathbf{e}_\mu' \mathbf{\mu}' = 0.
\]

Note how the inverse \(\tau^{-1}\) is updated rather than \(\tau\) directly. Similar to the variances, all previous equations contain \(\tau^{-1}\) and since inverting a matrix is computationally expensive, the inverse is directly updated. Additionally, \(\tau^{-1}\) requires the definition of a lowerbound. The optimization can results in \(\tau^{-1}\) approaching zero which means the controller converts to a pure estimator. In this work, \(\tau^{-1}\) is set to have a minimum value of 0.5 on all diagonal elements and zero elsewhere.

Using Equation 13 the oscillations can be damped as well as improving settling time as shown in Figure 5. Note how updating \(\tau^{-1}\) only is satisfactory to eliminate the oscillations (no update of \(\Sigma_o^{-1}\) or \(\Sigma_\mu^{-1}\) was used).

The convergence of \(\tau^{-1}\) happens when

\[
\frac{\partial F}{\partial \tau^{-1}} = -2\Sigma_\mu \mathbf{e}_\mu (\mathbf{d} - \mu) + 2\Sigma_\mu \mathbf{e}_\mu' \mathbf{\mu}' = 0.
\]

This occurs when \(\mathbf{d} = \mu\) and \(\mathbf{\mu}' = 0\) which corresponds to the controller settling at its target position. The updates for \(\tau\) will thus retune the controller appropriately until the the target is reached. This gives a preference for updating \(\tau\) rather than updating \(\Sigma_o^{-1}\) or \(\Sigma_\mu^{-1}\) in most cases, since at the desired state \(\tau\) is not updated anymore, unlike \(\Sigma_o^{-1}\) and \(\Sigma_\mu^{-1}\).

### VI. Results on a robotic manipulator

This section evaluates the presented approach and uses the active inference controller (AIC) from [7] as a benchmark since the authors have shown their work outperforms state-of-the-art MRAC. We show that our approach outperforms the AIC from [7] for carrying different payloads, different initial parameters for the variances and different values of \(\tau\). A summary of the results is reported in this section; however, full results are posted along with the video demonstrations.

The AIC from [7] achieves adaptive control without explicit model dynamics of the system. However, it is sensitive to the initialization of its parameters. By slightly changing \(\Sigma_\mu^{-1}\) for instance, the system suffers from severe oscillations and never settles at its target state. Our approach overcomes this problem by updating the variances during run-time.
Consider the task of reaching a target $\mu_d$ starting from $\mu(0)$

$$\mu_d = [-1, 0.5, 0, -1.2, 0, 1.6, 0, 0]$$
$$\mu(0) = [0, 0, 0, 0, 0, 0, 0]$$

where each element in these vectors corresponds to one of the 7 joints from the Panda manipulator (Figure 1). If the AIC from [7] is tuned properly ($\Sigma_\mu^{-1} = 1.5I$, $\Sigma_\eta^{-1} = 0.5I$, $\Sigma_\mu' = 0.1I$ and $\Sigma_\eta' = 0.5I$), this results in satisfactory behaviour. In this case, $I$ refers to the 7x7 identity matrix. However, if we vary $\Sigma_\mu^{-1}$ to other values ($0.3I$ and $0.5I$), the performance gets considerably worse. In our approach, we update $\tau$, $\Sigma_\eta$ and $\Sigma_\eta'$ online to retune the controller online. We ran the experiment of moving from $\mu(0)$ to $\mu_d$ for several values of $\Sigma_\mu^{-1}$ and recorded the Mean Absolute Error (MAE) for all joints in Table I.

The Mean Absolute Error (MAE) is defined as:

$$MAE = \frac{1}{n_t} \sum_{j=1}^{n_t} |\mu_d - \mu|$$

When the AIC is properly tuned $\Sigma_\mu^{-1} = 0.1I$, the two cases have the same MAE. However, when $\Sigma_\mu^{-1} = 0.3I$, the controller suffers from severe oscillations and never settle around its target (visualized in Figure 6). The MAE increase to more the triple its value while in the case of tuning the hyperparameters, the MAE actually decreases. This is due to the fact that increasing $\Sigma_\mu^{-1}$ makes the controller more aggressive and when tuned, it does not oscillate and also has a slightly faster response.

In a similar fashion, results for changing the value of $\tau^{-1}$ are recorded in Table II. Again, the MAE is much lower when tuning the hyperparameters.

For the last experiment, the robot is supposed to carry varying payloads. We test three different masses: $m = 1kg$, $m = 2kg$ and $m = 3kg$ (max payload for the Panda arm). The MAE for these cases is recorded in the Table III. Again, our approach outperforms the approach from previous work even when it is properly tuned.

The controller presented does not require a dynamic model and performs robustly on both the mass-spring-damper and an industrial manipulator. Additionally, the approach does not need any offline training. These benefits also apply to approach presented in [7]; however, when initial parameters are slightly altered, the performance of the presented approach is clearly superior. The presented approach damp oscillations robustly and updating $\tau$ ensure converges to the target. Additionally, the presented approach performs better when carrying different payloads.

![Fig. 6: Results comparing the active inference controller with and without updating hyperparameters ($\tau$, $\Sigma_\eta$ and $\Sigma_\eta'$). This graph corresponds to the second column of Table I](image)

Note the difference in the scale of the y axes!
VII. DISCUSSION AND FUTURE WORK

In section III, the generative model was selected to have the form $p(\hat{o}, \hat{\mu})$ which does not explicitly include any notion of an action $a$. Thus to choose the action that minimizes free-energy, the chain rule was utilized (Equation 9). Alternatively, the actions could be explicitly added in the generative model $p(\hat{o}, \hat{\mu}, a)$. Additionally, the presented models could efficiently solved as factor graphs [27], [28].

To improve the estimation, several modifications are possible: a prior factor could be introduced $p(\mu_{\tau} | \mu_{\tau-1})$ or a sliding window could be considered based on the last $H$ steps. Additionally, the current method only returns the control actions for the next timestep and thus planning ahead is not possible. Solving this can be achieved by a forward sliding window (receding horizon) similar to model predictive control [29].

For updating the variances, a lower bound was set since the value has to be strictly positive (or positive semi-definite for a matrix). An alternative would be to put the variable through a mapping to a strictly positive function. For instance, $\Sigma^{-1} = \exp j_\sigma$ and the update rules would choose $j_\sigma$ to minimize $F$.

Finally, using the Laplace approximation allowed us to only optimize the mean of the variational distribution $Q(s)$. The covariance was not computed or used. Future work should look into utilizing the covariances and using full variational inference rather than the Laplace approximation.

VIII. CONCLUSIONS

In this paper a method for state-estimation, control and learning model (hyper)parameters is introduced based on minimizing free-energy. Online estimation of relevant quantities can be achieved with one step of gradient descent on the free-energy for each iteration of the controller. We showed that when a temporal parameter $\tau$ approaches zero, the approach converts to a PID controller and if $\tau$ approaches $\infty$, it converts to a pure estimator. We then demonstrated the effectiveness of the framework for a 7 DOF robotic arm and showed adaptability and robustness outperforming previous work by a large margin. Our approach provides a number of benefits: it doesn’t require training data (or trials to learn), has a fixed number of parameters to be specified regardless of DoF, it does not require an accurate model of the dynamics (see Section IV-B), and it damps oscillations while being robust against poor initial settings (see Section VI).

ACKNOWLEDGMENT

Authors thank MatiasMattamala and Mees Vanderbroeck for helpful comments and feedback.

REFERENCES

[1] Jesús Enrique Sierra and Matilde Santos. Wind and payload disturbance rejection control based on adaptive neural estimators: application on quadrotors. Complexity, 2019.

[2] Bin Wei. Adaptive control design and stability analysis of robotic manipulators. In Actuators, volume 7, page 89. Multidisciplinary Digital Publishing Institute, 2018.

[3] Bustanil Arifin, Bhakti Yudo Suprapto, Sri Antini Dwi Prasetyowati, and Zainuddin Nawawi. The lateral control of autonomous vehicles: A review. In 2019 International Conference on Electrical Engineering and Computer Science (ICECOS), pages 277–282. IEEE, 2019.

[4] Duy Nguyen-Tuong, Jan R Peters, and Matthias Seeger. Local gaussian process regression for real time online model learning. In Advances in neural information processing systems, pages 1193–1200, 2009.

[5] Fernando Díaz Ledezma and Sami Haddadin. First-order-principles-based constructive network topologies: An application to robot inverse dynamics. In 2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids), pages 438–445. IEEE, 2017.

[6] Dan Zhang and Bin Wei. A review on model reference adaptive control of robotic manipulators. Annual Reviews in Control, 43:188–198, 2017.

[7] Corrado Pezzato, Riccardo MG Ferrari, and Carlos Hernandez. A novel adaptive controller for robot manipulators based on active inference. IEEE Robotics and Automation Letters, 2020.

[8] Karl Friston, Thomas FitzGerald, Francesco Rigoli, Philipp Schwartenbeck, and Giovanni Pezzulo. Active inference: a process theory. Neural computation, 29(1):1–49, 2017.

[9] Karl Friston. Friston k. the free-energy principle: a rough guide to the brain? trends cogn sci 13: 293-301. Trends in cognitive sciences, 13:293–301, 07 2009.

[10] Giovanni Pezzulo, Francesco Rigoli, and Karl J Friston. Hierarchical active inference: A theory of motivated control. Trends in cognitive sciences, 22(4):294–306, 2018.

[11] Raphael Kaplan and Karl J. Friston. Planning and navigation as active inference. Biological cybernetics, 112(4):323–343, 2018.

[12] Marie Dillon Dahleh and William T Thomson. Theory of vibration with applications. Prentice-Hall Inc, 1998.

[13] Bao Yu, Xu Bingfeng, and Pan Guanlin. The study of excitation controller based on genetic algorithms and fuzzy pid. In 2010 International Conference on Computer, Mechatronics, Control and Electronic Engineering, volume 2, pages 330–332. IEEE, 2010.

[14] Hamed Khodadadi and Hamid Ghaediri. Self-tuning pid controller design using fuzzy logic for half car active suspension system. International Journal of Dynamics and Control, 6(1):224–232, 2018.

[15] Lorenzo Jamone, Bruno Dasas, and Jos Santos-Victor. Incremental learning of context-dependent dynamic internal models for robot control. In 2014 IEEE international symposium on intelligent control (ISIC), pages 1336–1341. IEEE, 2014.

[16] Daniel Kappler, Franziska Meier, Nathan Ratliff, and Stefan Schaal. A new data source for inverse dynamics learning. In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 4723–4730. IEEE, 2017.

[17] Matteo Matteucci. Eleam: Evolutionary learning of rich neural network topologies. Technical report, Carnegie-Mellon University, 2006.

[18] RG Walters and MM Bayoumi. Application of a self-tuning pole-placement regulator to an industrial manipulator. In 1982 21st IEEE Conference on Decision and Control, pages 323–329. IEEE, 1982.

[19] Léon Pio-Lopez, Ange Nizard, Karl Friston, and Giovanni Pezzulo. Active inference and robot control: a case study. Journal of The Royal Society Interface, 13(122):20160616, 2016.

[20] Pablo Lanillos and Gordon Cheng. Adaptive robot body learning and estimation through predictive coding. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 4083–4090. IEEE, 2018.

[21] Guillermo Oliver, Pablo Lanillos, and Gordon Cheng. Active inference body perception and action for humanoid robots. arXiv preprint arXiv:1906.03022, 2019.

[22] Charles W Fox and Stephen J Roberts. A tutorial on variational bayesian inference. Artificial intelligence review, 38(2):85–95, 2012.

[23] Karl Friston, Jérémie Mattout, Nelson Trujillo-Barreto, John Ashburner, and Will Penny. Variational free energy and the lapiace approximation. Neuroimage, 34(1):220–234, 2007.

[24] Christopher L Buckley, Chang Sub Kim, Simon McGregor, and Anil K Seth. The free energy principle for action and perception: A mathematical review. Journal of Mathematical Psychology, 81:55–79, 2017.

[25] Mohamed Baiouny, Matias Mattamala, Paul Duckworth, Bruno Lac erda, and Nick Hawes. Adaptive manipulator control using active inference with precision learning. In UKRAS20 Conference: "Robots into the real world" Proceedings, Lincoln, United Kingdom, May 2020.

[26] Rafał Bogacz. A tutorial on the free-energy framework for modelling perception and learning. Journal of mathematical psychology, 76:198–211, 2017.
[27] Hans-Andrea Loeliger, Justin Dauwels, Junli Hu, Sascha Korl, Li Ping, and Frank R Kschischang. The factor graph approach to model-based signal processing. *Proceedings of the IEEE*, 95(6):1295–1322, 2007.

[28] Mees Vanderbroeck, Mohamed Baioumy, Daan van der Lans, Rens de Rooij, and Tiis van der Werf. Active inference for robot control: A factor graph approach. *Student Undergraduate Research E-journal?*, 5:1–5, 2019.

[29] Basil Kouvaritakis and Mark Cannon. Model predictive control. *Switzerland: Springer International Publishing*, 2016.