Check for updates

Article

The International Journal of Robotics Research
1–18
© The Author(s) 2020
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/0278364920921935
journals.sagepub.com/home/ijr
SAGE

Data-driven Koopman operators for model-based shared control of human–machine systems

Alexander Broad[1,2,3], Ian Abraham[4], Todd Murphey[4] and Brenna Argall[2,3,4]

Abstract

We present a data-driven shared control algorithm that can be used to improve a human operator's control of complex dynamic machines and achieve tasks that would otherwise be challenging, or impossible, for the user on their own. Our method assumes no a priori knowledge of the system dynamics. Instead, both the dynamics and information about the user's interaction are learned from observation through the use of a Koopman operator. Using the learned model, we define an optimization problem to compute the autonomous partner's control policy. Finally, we dynamically allocate control authority to each partner based on a comparison of the user input and the autonomously generated control. We refer to this idea as model-based shared control (MbSC). We evaluate the efficacy of our approach with two human subjects studies consisting of 32 total participants (16 subjects in each study). The first study imposes a linear constraint on the modeling and autonomous policy generation algorithms. The second study explores the more general, nonlinear variant. Overall, we find that MbSC significantly improves task and control metrics when compared with a natural learning, or user only, control paradigm. Our experiments suggest that models learned via the Koopman operator generalize across users, indicating that it is not necessary to collect data from each individual user before providing assistance with MbSC. We also demonstrate the data efficiency of MbSC and, consequently, its usefulness in online learning paradigms. Finally, we find that the nonlinear variant has a greater impact on a user's ability to successfully achieve a defined task than the linear variant.

Keywords

Machine learning, shared control, human–robot interaction

1. Introduction

Robot autonomy offers great promise as a tool by which we can enhance, or restore, the natural abilities of a human partner. For example, in the fields of assistive and rehabilitative medicine, devices such as exoskeletons and powered wheelchairs can be used to assist a human who has severely diminished motor capabilities. However, many assistive devices can be difficult to control. This can be due to the inherent complexity of the system, the required fidelity in the control signal, or the physical limitations of the human partner. We can, therefore, further improve the efficacy of these devices by offloading challenging aspects of the control problem to an autonomous partner. In doing so, the human operator is freed to focus their mental and physical capacities on important high-level tasks such as path planning and interaction with the environment. This idea forms the basis of *shared control* (SC; see Figure 1), a paradigm that aims to produce joint human–machine systems that are more capable than either the human or machine on their own.

A primary challenge that researchers and engineers face when developing SC paradigms for generic human–machine systems is a lack of a priori knowledge of the human and robot partners. This issue is compounded by the fact that, in the real world, many users may operate the same mechanical device. It is therefore necessary to consider solutions that generalize to a variety of potential human and machine partners. In this work, we propose a data-driven methodology that learns all relevant

[1]Boston Dynamics, Waltham, MA, USA
[2]Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, USA
[3]Department of Mechanical Engineering, Northwestern University, Evanston, IL, USA
[4]Shirley Ryan AbilityLab, Chicago, IL, USA

Corresponding author:
Alexander Broad, Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208, USA.
Email: alex.broad@u.northwestern.edu

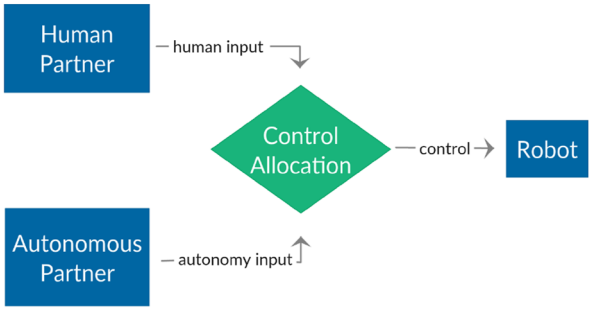

Fig. 1. Pictorial representation of a SC paradigm. Both the human and autonomy are capable of controlling the mechanical system, and a dynamic control allocation algorithm selects which agent is in control at any given moment.

information about how a given human and machine pair interact directly from observation. We then integrate the learned model of the joint system into a single SC paradigm. We refer to this idea as *model-based shared control* (MbSC).

In this work, we learn a model of the joint human–machine system through an approximation to the Koopman operator (Koopman, 1931), though any machine learning approach could be used. However, the Koopman operator is chosen specifically for this work as it has previously proven useful in human-in-the-loop systems (Broad et al., 2017) and can be computed efficiently (Williams et al., 2015b). This model is trained on observation data collected during demonstration of the human and machine interacting and therefore describes both the human's input to the system, and the robot's response to the human input and system state. We can then integrate the portion of the learned model that specifically describes the system and control dynamics of the mechanical device into an optimal control algorithm to produce autonomous policies. Finally, the input provided by the human and autonomous partners are integrated via a geometric signal filter to provide real-time, dynamic SC of unknown systems.

We validate our thesis that modeling the joint human–machine system is sufficient for the purpose of automating assistance with two human subjects studies consisting of 32 total participants. The first study imposes a linear constraint on the modeling and control algorithms, while the second study relaxes these constraints to evaluate the more general, nonlinear case. The linear variant of our proposed algorithm is used to validate the efficacy of our SC paradigm and was first presented in Broad et al. (2017). The nonlinear variant extends these results to a wider class of human–machine systems. The results of the two studies demonstrate that the nonlinear variant has a greater impact on overall task performance than the linear methods. We also find that our modeling technique is generalizable across users with results that suggest that individualizing the model offline, based on a user's own data, does not affect the ability to learn a useful representation of the dynamical system. Finally, we evaluate the efficacy of our SC paradigm in an online learning scenario, demonstrating the sample efficiency of the MbSC paradigm.

We provide background and related work in Section 2. We then define MbSC in Section 3. In Section 4 we describe the human subjects study we perform and detail the results in Section 5. We describe important takeaways in Section 6 and conclude in Section 7.

2. Background and related work

This section presents background and related work in the SC literature for human–machine systems. We also identify alternative methods of autonomous policy generation for SC, and provide a detailed background on the Koopman operator (Koopman, 1931) with a particular focus on its use in learning system dynamics.

2.1. SC

In this work, we explore the question of how automation can be used to adjust to, and account for, the specific capabilities of a human partner. In particular, we aim to develop a methodology that allows us to *dynamically adjust* the amount of control authority given to the robot and human partners (Hoeniger, 1998; Hoffman and Breazeal, 2004). If done intelligently, and with appropriate knowledge of the individual capabilities of each team member, we can improve the overall efficiency, stability and safety of the joint system (Lasota et al., 2017). Approaches to SC range from pre-defined, discretely adjustable methods (Kortenkamp et al., 2000) to probabilistic models (Javdani et al., 2015) to policy blending (Dragan and Srinivasa, 2013). In addition to blending in the original control signal space, SC has been researched through haptic control (Nudehi et al., 2005) and compliant control (Kim et al., 1992).

In this work, we allocate control using a filter (Tzorakoleftherakis and Murphey, 2015) described more thoroughly in Section 3.3. Our control allocation strategy is similar in practice to *virtual fixtures* and *virtual guides*, techniques that are common in the haptics literature (Forsyth and MacLean, 2005; Griffiths and Gillespie, 2005). In particular, virtual fixtures and guides are techniques by which autonomously generated forces are *added to the control of a system* to limit movement into undesriable areas and/or influence motion towards an optimal strategy (Abbink et al., 2012). These ideas have been explored most commonly in association with robotic telemanipulation (Abbott et al., 2007), including applications such as robotic surgery (Marayong and Okamura, 2004) and robot-assisted therapy (Noohi et al., 2016). A key difference between these approaches and our own is that our control allocation method does not incorporate additional information from the autonomous partner into the control loop. Instead, the autonomous partner simply rejects input

from the operator that does not meet the proposed criteria. Our approach therefore requires no a priori information about (or ability to sense) the environment, and no information about the system dynamics. In contrast, virtual fixtures/guides require information about (or the ability to detect) hard constraints in the environment, and knowledge of the system dynamics. This information is then used to compute forces, the virtual fixtures, that counteract user-generated forces that are defined as dangerous. The approach in this article does not have similar a priori information requirements, suggesting our approach can more easily be incorporated into novel human–machine systems. An important benefit of the methods proposed in the virtual fixtures/guide literature is that the techniques often provide an explicit guarantee of safety for the joint human–machine system. Our approach can be extended to provide the same guarantees by incorporating information about (or the ability to sense) the environment and using control barrier functions to implement safety requirements (Broad et al., 2018).

The effects of SC have been explored in numerous fields in which the addition of a robot partner could benefit a human operator. For example, in assistive and rehabilitation robotics, researchers have explored the effects of SC on teleoperation of smart wheelchairs (Erdogan and Argall, 2017; Trieu et al., 2008) and robotic manipulators (Kim et al., 2006). Similarly, researchers have explored SC as it applies to the teleoperation of larger mobile robots and human–machine systems, such as cars (de Winter and Dodou, 2011) and aircraft (Matni and Oishi, 2008). When dealing with systems of this size, safety is often a primary concern.

The above works are conceptually similar to our own as they use automation to facilitate control of a robot by a human partner. However, in this work, we do not augment the user's control based on an explicit model of the user. Instead, we use observations of the user demonstrations to build a *model of the joint human–robot system*. The effect of the human partner on the SC system is implicitly encoded in the model learned from their interactions.

2.2. Model-based reinforcement learning

MbSC is a paradigm that generalizes SC to generic human–machine partners (Broad et al., 2017). That is, MbSC assumes no a priori knowledge of either partner and instead uses data-driven techniques to learn models of the human and/or robot partner(s) from observation. In addition to providing a quantitative understanding of each partner, these models can be used to generate autonomous control policies by integrating the learned system and control dynamics into an optimal control framework.

MbSC is therefore highly related to model-based reinforcement learning (MbRL), a paradigm that explicitly learns a model of the system dynamics in addition to learning an effective control policy. MbSC extends MbRL to systems that integrate control from various sources. Early work in MbRL includes Barto et al. (1995) and Kaelbling et al. (1996). More recently, researchers have considered integrating learned system models with optimal control algorithms to produce control trajectories in a more data-efficient manner (Mitrovic et al., 2010). These algorithms compute control through an online optimization process, instead of through further data collection (Barto et al., 1995). There are, of course, many viable model learning techniques that can be used to describe the system and control dynamics. For example, neural networks (Williams et al., 2017), Gaussian processes (Nguyen-Tuong et al., 2009), and Gaussian mixture models (Khansari-Zadeh and Billard, 2011) have all shown great promise in this area. Often the best choice of modeling algorithm is related specifically to the application domain. For example, Gaussian processes perform well in low-data regimes, but scale poorly with the size of the dataset where neural networks fit naturally. In this work, we explore a modeling technique that easily integrates with derivative-based optimal control algorithms. A survey of learning for control can be found in Schaal and Atkeson (2010).

From a motivational standpoint, related work also includes methods that model not only the dynamics of a robotic system, but combined human–machine systems from data. For example, researchers have explored learning control policies from user demonstrations, thereby incorporating both system dynamics and the user's desires (Argall et al., 2009; Celemin et al., 2019). Building on these ideas, researchers have proposed learning SC policies directly from demonstration data using deep reinforcement learning (Reddy et al., 2018). To improve the human partner's intuition for the interaction paradigm, researchers have also proposed learning latent spaces to allow users to control complex robots with low-dimensional input devices (Losey et al., 2019). Relatedly, people have also proposed techniques for modeling both the dynamics of a system, and a policy for deciding when a human or autonomous partner should be in control. One such method is to learn local approximations to the system's dynamics and only provide autonomous assistance when the system is nearby a state it has previously observed (Peternel et al., 2016). Our approach utilizes a linear representation of the nonlinear human–robot dynamics which avoids the use of local models in exchange for a higher capacity linear model which globally represents the complex system. This is also distinct from the virtual fixtures/guide literature where system models are known a priori, and frequently nonlinear.

From a methodological standpoint, the most closely related research is recent work that computes control trajectories by integrating learned dynamics models with model predictive control (MPC) algorithms (Drews et al., 2017; Williams et al., 2017). These algorithms are defined by an iterative, receding horizon optimization process instead of using an infinite horizon. Similar to our own work, these researchers first collect observations from live demonstrations of the mechanical device to learn a model of the system dynamics. They then integrate the model with a MPC algorithm to develop control policies. Beyond

methodological differences (e.g., choice of machine learning and optimal control algorithms), the key theoretical distinction between these works and our own is our focus on SC of joint human–machine systems, instead of developing fully autonomous systems. In particular, we learn a model of the joint system that is integrated into a SC system to improve a human operator's control of a dynamic system. We therefore consider the influence of the human operator both during the data-collection process and at run-time in the control of the dynamic system.

In this work, we learn a model of the system and control dynamics through an approximation to the Koopman operator (Koopman, 1931). As the Koopman operator is a relatively new concept in robot learning for control, we now provide additional information on its description in the following section.

2.3. The Koopman operator

The Koopman operator is an infinite-dimensional linear operator that can capture all information about the evolution of nonlinear dynamical systems. This is possible because the operator describes a linear mapping between sequential *functions of states* instead of the state itself. In particular, the Koopman operator acts on an infinite-dimensional Hilbert space representation of the state. To define the Koopman operator, let us consider a discrete time dynamic system $(\mathcal{X}, t, F)$:

$$x_{t+1} = F(x_t) \tag{1}$$

where $\mathcal{X} \subseteq \mathbb{R}^N$ is the state space, $t \in \mathbb{R}$ is time and $F : \mathcal{X} \to \mathcal{X}$ is the state evolution operator. We also define ϕ, a nominally infinite-dimensional observation function

$$y_t = \phi(x_t) \tag{2}$$

where $\phi : \mathcal{X} \to \mathbb{C}$ defines the transformation from the original state space into the Hilbert space representation that the Koopman operator acts on. The Koopman operator $\mathcal{K}$ is defined as the composition of ϕ with F, such that

$$\mathcal{K}\phi = \phi \circ F \tag{3}$$

By acting on the Hilbert state representation, the *linear* Koopman operator is able to capture the complex, nonlinear dynamics described by the state evolution operator.

While the Koopman operator is nominally infinite-dimensional, recent work has demonstrated the ability to approximate a finite-dimensional representation using data-driven techniques (Budišić et al., 2012; Rowley et al., 2009). In the limit of collected observation data, the approximation to the Koopman becomes exact (Williams et al., 2015a). These data-driven methods have renewed an interest in using the Koopman operator in applied engineering fields. In contemporary work, the Koopman operator has been used successfully to learn the dynamics of numerous challenging systems. This includes demonstrations that show the Koopman operator can differentiate between cyclic and non-cyclic stochastic signals in stock market data (Hua et al., 2016) and that it can detect specific signals in neural data that signify non-rapid eye movement (NREM) sleep (Brunton et al., 2016). More recently these systems have included physical robotics systems (Abraham and Murphey, 2019; Bruder et al., 2019).

3. MbSC

Our primary goal is to develop a SC methodology that improves the skill of human–machine systems without relying on a priori knowledge of the relationship between the human and the machine. To define our MbSC algorithm we now describe the (1) model learning process, (2) method for computing the policy of the autonomous agent (*autonomy input* in Figure 1), and (3) control allocation method (the *green box* in Figure 1). A pictorial depiction of our MbSC paradigm can be found in Figure 2. Our learning-based approach develops a model of the joint human–machine system solely from observation, and this model can be used by the policy generation method to develop autonomous control trajectories. The control allocation method then describes how we integrate the input provided by the human partner and the autonomous agent into a single command that is sent to the dynamic system.

3.1. Model learning via the Koopman operator

When designing assistive SC systems, it is important to consider both the human and autonomous partners. To ensure that our paradigm is valid for generic human–machine systems, we learn both the *system dynamics* and information about the *user interaction* directly from data. In this work, we develop a model of the joint human–machine through an approximation to the Koopman operator, which can be computed offline or online (discussed further in Section 5.3). The model learning process is depicted in Figure 2(a). As mentioned previously, there are of course a variety of other machine learning algorithms and representations one could choose to learn the system dynamics. In this work, we use the Koopman operator, which is particularly well suited to MbSC of human–machine systems for two main reasons. First, it is possible to approximate the Koopman operator in low-data regimes (see Section 5.3), which allows us to quickly expand the set of human–machine systems we can control under the general MbSC paradigm. Second, there are a variety of highly efficient learning algorithms (Klus et al., 2015; Rowley et al., 2009; Williams et al., 2015a) that make the Koopman operator well suited to an online learning paradigm, an important feature in SC where it is unlikely that we have a priori knowledge of the joint human–machine system.

We use Extended Dynamic Mode Decomposition (EDMD) to approximate the Koopman operator (Williams et al., 2015a). EDMD belongs to a class of data-driven techniques known as Dynamic Mode Decomposition

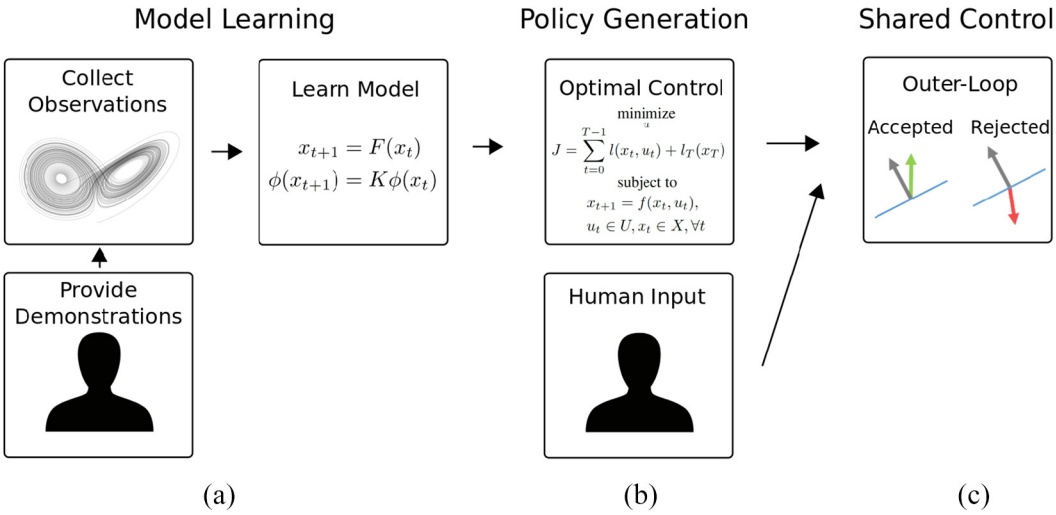

Fig. 2. Pictorial depiction of the our MbSC paradigm. (a) Collect observations from user interaction and learn a model of the joint human–machine system through an approximation to the Koopman operator. This can be computed offline or online. (b) Compute control policy of autonomous agent by solving optimal control problem using the learned model. (c) Allocate control to integrate autonomy (gray) and user input (green/red).

(DMD) (Rowley et al., 2009; Schmid, 2010; Tu et al., 2014). These algorithms use snapshots of observation data to approximate the Koopman modes that describe the dynamics of the observed quantities. We now provide a mathematical treatment of the EDMD algorithm. We start by redefining the observation function ϕ from Equation (2) as a vector valued set of basis functions chosen to compute a finite approximation to the Hilbert space representation. We can then define the following approximation to the Koopman operator

$$\phi(x_{t+1}) = \mathcal{K}^T \phi(x_t) + r(x_t) \tag{4}$$

where $r(x_t)$ is a residual term that represents the error in the model. The Koopman operator is therefore the solution to the optimization problem that minimizes this residual error term

$$\begin{aligned} J &= \frac{1}{2}\sum_{t=1}^{T} |r(x_t)|^2 \\ &= \frac{1}{2}\sum_{t=1}^{T} |\phi(x_{t+1}) - \phi(x_t)K)|^2 \end{aligned} \tag{5}$$

where T is the time horizon of the optimization procedure, and $|\cdot|$ is the absolute value. The solution to the least-squares problem presented in Equation (5) is

$$K = G^{\dagger} A$$

where $\dagger$ denotes the Moore–Penrose pseudo-inverse and

$$G = \frac{1}{T}\sum_{t=1}^{T} \phi(x_t)^T \phi(x_t)$$

$$A = \frac{1}{T}\sum_{t=1}^{T} \phi(x_t)^T \phi(x_{t+1})$$

3.1.1. Basis. In this work, we require that the finite basis ϕ *includes both the state and control variables* (Proctor et al., 2016). This ensures that the Koopman operator models both the natural dynamics of the mechanical system and the control dynamics as provided by the user demonstration. In this work, we empirically select a fixed set of basis functions to ensure that all models (across the different users in our validation study) are learned using the same basis. Here we choose ϕ such that

$$\begin{aligned} \phi = [&1, x_1, x_2, x_3, x_4, x_5, x_6, u_1, u_2, u_1 * x_1, u_1 * x_2, u_1 * x_3, \\ &u_1 * x_4, u_1 * x_5, u_1 * x_6, u_2 * x_1, u_2 * x_2, u_2 * x_3, \\ &u_2 * x_4, u_2 * x_5, u_2 * x_6, u_1 * \cos(x_3), u_1 * \sin(x_3), \\ &u_2 * \cos(x_3), u_2 * \sin(x_3)] \end{aligned} \tag{6}$$

These 25 basis functions were chosen to combine information about the geometry of the task (e.g., the trigonometric functions capture specific nonlinearities present in the system dynamics, see Section 4.1) with information related to how the user responds to system state. For this reason, we include terms that mix state information with control information. To evaluate the accuracy of the learned approximation to the Koopman operator we compute the H-step prediction accuracy (see Figure 3).

There are, of course, a variety of methods that one could use to select an appropriate basis for a given dynamical system. This step is particularly important as selecting a poor basis will quickly degrade the validity of the learned model

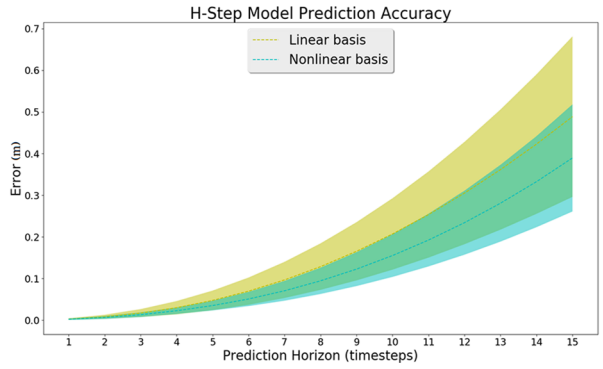

Fig. 3. H-step prediction accuracy of Koopman operator models based on linear and nonlinear bases. Error is computed as a combination of the Euclidean distance between the predicted (x, y) values and the ground truth (x, y).

(Berrueta et al., 2018). One such method is to integrate known information about the system dynamics into the chosen basis functions, such as the relationship between the heading of the lander and the motion generated by the main thruster. This approach works well when the system dynamics are easy to understand, however it can prove challenging when the dynamics are more complex. For this reason, one could also choose the set of basis functions through purely data-driven techniques. Sparsity promoting DMD (Jovanović et al., 2014) is one such algorithm. SP-DMD takes a large initial set of randomly generated basis functions and imposes an ℓ_1 penalty during the learning process to algorithmically decides which basis functions are the most relevant to the observable dynamics (Tibshirani, 1996). An example of this purely data-driven approach being applied to human–machine systems can be found in Broad et al. (2019).

3.2. Autonomous policy generation

To generate an autonomous control policy, we can integrate the portion of the learned model that relates to the system and control dynamics into a MPC algorithm. In particular, we use Koopman operator model-based control (Abraham et al., 2017; Broad et al., 2017), which we detail now in full. To compute the optimal control sequence, u, we must solve the following MPC problem

$$\begin{aligned} \underset{u}{\text{minimize}} \quad & J = \sum_{t=0}^{T-1} l(x_t, u_t) + l_T(x_T) \\ \text{subject to} \quad & x_{t+1} = f(x_t, u_t), \\ & u_t \in U, x_t \in X, \forall t \end{aligned} \tag{7}$$

where $f(x_t, u_t)$ is the system dynamics, l and l_T are the running and terminal cost, and U and X are the set of valid control and state values, respectively.

In this work, we define

$$l(x_t, u_t) = \frac{1}{2}(x_t - x_d)Q_t(x_t - x_d) + \frac{1}{2}u_t R_t u_t$$

where $Q_t = Diag[6.0, 10.0, 20.0, 2.0, 2.0, 3.0]$

x_t is the current state and x_d is the desired goal state. In addition,

$$l_T(x_T) = \frac{1}{2}(x_t - x_d)Q_T(x_t - x_d)$$

where $Q_T = Diag[3.0, 3.0, 5.0, 1.0, 1.0, 1.0]$

These values were chosen empirically based on results observed from the system operating fully autonomously.

To integrate our learned system model, we re-write the system dynamics as such:

$$\phi(x_{t+1}) = f_{\mathcal{K}}(x_t, u_t) \tag{8}$$

where $f_{\mathcal{K}} = \mathcal{K}^T\phi(x_t, u_t)$ is the learned system dynamics parameterized by a Koopman operator $\mathcal{K}$. This equation demonstrates the fact that the Koopman operator does not map directly from state to state, but rather operates on functions of state. We can then evaluate the evolved state by recovering the portion of the basis that represents the system's state

$$x_{t+1} = \phi(x_{t+t})_{1:N} \tag{9}$$

where values $1 : N$ are the state variables, as per our definition in Equation (6), and N is the dimension of the state space. The policy generation process is depicted in Figure 2(b).

3.2.1. Nonlinear MPC algorithm. We solve Equation (7) with sequential action control (SAC) (Ansari and Murphey, 2016). SAC is a real-time, model-based nonlinear optimal control algorithm that is designed to iteratively find a single value, a time to act, and a duration that maximally improves performance. Other viable nonlinear optimal control algorithms include iLQR (Li and Todorov, 2004) and DDP (Mayne, 1966; Tassa et al., 2014). SAC is particularly well suited for our SC algorithm because it searches for single, short burst actions, which aligns well with our control allocation algorithm (described in detail in Section 3.3). In addition, SAC can compute closed-loop trajectories very quickly (1 kHz), an important feature for interactive human–machine systems such as that presented in this work.

3.2.2. Integrating the Koopman model and SAC. SAC is a gradient-based optimization technique and it is therefore necessary to compute derivatives of a system during the optimization process. The linearization of the discrete time system is defined by the following equation

$$x_{t+1} = Ax_t + Bu_t$$

By selecting a differentiable ϕ, one can compute A and B

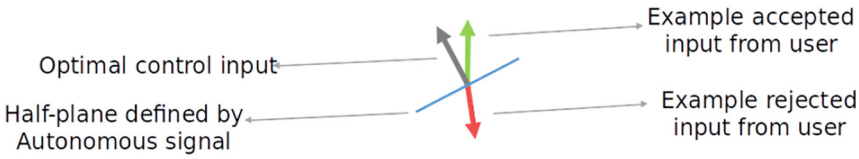

Fig. 4. Maxwell's Demon Algorithm (MDA).

Algorithm 1 Maxwell's Demon Algorithm (MDA)

if $\langle u_h, u_a \rangle \geqslant 0$ **then**
  $u = u_h$;
else
  $u = 0$;
end if

$$A = \mathcal{K}^T_{1:N} \frac{\partial \phi}{\partial x}$$
$$B = \mathcal{K}^T_{N:N+P} \frac{\partial \phi}{\partial u} \tag{10}$$

where N is again the dimension of the state space, and P is the dimension of the control space.

3.3. Control allocation method

To close the loop on our SC paradigm, we define a control allocation method that uses the solution from the optimal control algorithm to provide outer-loop stabilization. We use a geometric signal filter that is capable of dynamically shifting which partner is in control at any given instant based on optimality criteria. This technique is known as Maxwell's Demon Algorithm (MDA) (Tzorakoleftherakis and Murphey, 2015). Our specific implementation of MDA is detailed in Algorithm 1 where u_h is the control input from the human operator, u_a is the control produced by the autonomy, and u is applied to the dynamic system. We also provide a pictorial representation of the algorithm in Figure 4.

This control allocation method restricts the user's input to the system to be in the same half-plane as the optimal control solution, and places no other limitations on the human–machine interaction. If the user's input is in the opposite half-plane, no input is provided to the system. This control allocation method is lenient to the human partner, as notably, *the autonomous agent does not add any information into the system* and instead only blocks particularly bad input from the user. Therefore, *any signal sent to the system originates from the human partner*. We use this filter because we are motivated by assistive robotics, in which prior research has shown that there is no consensus across users on desired assistance level (Erdogan and Argall, 2017). By allowing the user a high level of control freedom, the system encourages input from the human operator and restricts how much authority is granted to the autonomous partner. This method is depicted in Figure 2(c).

We use MDA in this work primarily because it has been experimentally validated in prior studies on human–machine systems for assistive robotics (Fitzsimons et al., 2016). Notably, this method does not guarantee optimal (or even "good") performance, as a human operator could theoretically always provide input orthogonal to the autonomous solution, resulting in no control ever being applied to the system. However, this technique does allocate a large amount of control authority to the human-in-the-loop, a desirable feature in our motivating application domains. There are also alternative methods that can be used for similar purposes, including extensions to MDA that incorporate additional information from the autonomous partner, which can be used to improve performance or safety (Broad et al., 2018). A review paper of alternative control allocation techniques can be found in Losey et al. (2018).

4. Human subjects study

Here, we detail the experimental setup that we use to study three main aspects of the described system.

- First, our aim is to evaluate the efficacy of MbSC as it relates to task success and control skill. Concurrently, we aim to evaluate the generalizability of the learned system models with respect to a wide range of human operators.
- Second, we aim to evaluate the efficacy of MbSC under an online learning paradigm: specifically, the sample efficiency of the Koopman operator representation.
- Finally, we aim to evaluate the impact of nonlinear modeling and policy generation techniques through a comparison with a second human subjects study that enforces linear constraints on our MbSC algorithm.

4.1. Experimental environment

The proposed SC framework is evaluated using a simulated lunar lander (see Figure 5). We use a simulated lunar lander (rocket) as our experimental environment for a number of reasons. This environment is challenging for a novice user, but performance can be improved (and sometimes mastered) given enough time and experience. Similar to a real rocket, one of the main control challenges is the stability of the system. As the rocket rotates along its yaw axis, firing the main thruster can produce nonintuitive dynamics for a

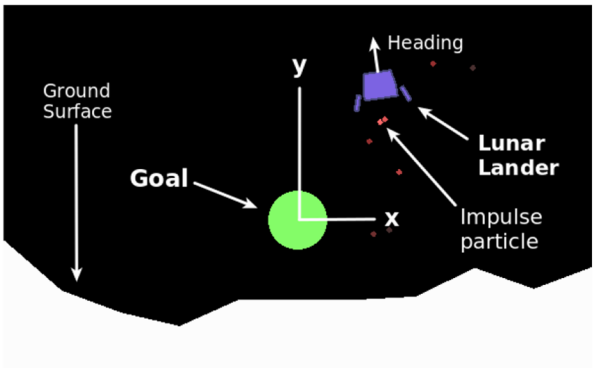

Fig. 5. Simulated lunar lander system. The green circle is the goal location. The red dots represent an engine firing.

Fig. 6. Control interface.

novice. Furthermore, once the rocket has begun to rotate, momentum can easily overwhelm a user who is unfamiliar with such systems. Therefore, it is often imperative—particularly for non-expert users—to maintain a high degree of stability at all times in order to successfully complete the task. In addition to the control challenges, we choose this environment because the simulator abstracts the system dynamics through calls to the Box2D physics engine; therefore, we do not have an exact model and thus have an *explicit need to learn one*.

4.2. System description

The dynamic system is a modified version of an open-source environment implemented in the Box2D physics engine and released by OpenAI (Brockman et al., 2016). Our modifications (1) allow for continuous-valued multi-dimensional user control via a joystick, and (2) incorporate the codebase into the open-source ROS framework. We have made our code available online at https://github.com/asbroad/model_based_shared_control.

The lunar lander is defined by a six-dimensional state space made up of the position (x, y), heading (θ), and their rates of change $(\dot{x}, \dot{y}, \dot{\theta})$. The control input to the system is a continuous two-dimensional vector (u_1, u_2), which represents the throttle of the main and rotational thrusters. The main engine can only apply positive force. The left engine fires when the second input is negative, while the right engine fires when the second input is positive. The main engine applies an impulse that acts on the center of mass of the lunar lander, while the left and right engines apply impulses that act on either side of the rocket. We remind the reader that our goal is to learn both the system dynamics and user interaction. For this reason, we must collect data both on the system state and also the control input. Together, this defines an eight-dimensional system:

$$\mathcal{X} = [x, y, \theta, \dot{x}, \dot{y}, \dot{\theta}, u_1, u_2]$$

where the first six terms define the lunar lander state and u_1, u_2 are the main and rotational thruster values, through which the user interacts with the system.

4.3. Trial description

The task in this environment requires the user to navigate the lander from its initial location to the goal location (represented by the green circle in Figure 5) and to arrive with a heading nearly perpendicular to the ground plane and with linear and rotational velocities near zero. A trial is considered complete either (1) when the center of an upright lunar lander is fully contained within the goal circle (i.e., the Euclidean distance between the center of the lander and the center of the goal is less than 0.9 m) and the linear and angular velocities are near zero (i.e., the linear velocities must be less than 1.0 m/s and the angular velocity must be less than 0.3 rad/s), or (2) when the lander moves outside the bounded environment (i.e., when the lander moves off the screen to the left or right) or crashes into the ground.

In each trial, the lunar lander is initialized to the same x, y position (10.0 m, 13.3 m), to which we added a small amount of Gaussian noise ($\mu = 0.2$ m). In additionally, a random force is applied at the start of each trial (uniform($-1,000$ N,1, 000 N)). The goal location (10.0 m, 6.0 m) is constant throughout all trials and is displayed to the user as a green circle (see Figure 5).

The operator uses a PS3 controller to interact with the system (Figure 6). The joystick controlled by the participant's dominant hand fires the main thruster, and the opposing joystick fires the side thrusters. As the user moves through the environment, we keep track of the full state space at each timestep (10 Hz). We provide a video of the system, task, and user interaction under SC as part of the supplementary material.

4.4. Analysis I: efficacy and generalizability of MbSC

4.4.1. Control conditions. To study the efficacy and generalizability of our SC system and the generalizability of the learned system dynamics, we compare four distinct control conditions.

- In the first condition, the user is in full control of the lander and is not assisted by the autonomy in any way; we call this approach *User Only* control. As each user undergoes repeated trials with the same goal, this can also be considered a natural learning paradigm.

In the remaining three conditions an autonomous agent provides outer-loop stabilization on the user's input as described in Section 3. The main distinction between these three control conditions is the source of the data used to compute the model of the joint system.

- In the second condition, the model is defined by a Koopman operator learned on data captured from earlier observations of the current user; we call this approach *Individual Koopman*.
- In the third condition, the model is defined by a Koopman operator learned on data captured from observations of three novice participants prior to the experiment (who were not included in our analysis); we call this approach *General Koopman*.
- In the fourth condition, the model is defined by a Koopman operator learned on data captured from observations of an expert user (the first author of the article, who has significant practice controlling the simulated system); we call this approach *Expert Koopman*.

We analyze the viability of MbSC by comparing the *User Only* condition with each of the SC conditions. We analyze the generalizability of the learned models by comparing the results from the *Individual Koopman, General Koopman*, and *Expert Koopman* conditions. All of the data we analyze to evaluate the ideas presented in this section comes from the nonlinear MbSC study.

4.4.2. Protocol and participants. Each experiment begins with a training period for the user to become accustomed to the dynamics of the system and the interface. This training period continues until the user is able to successfully complete the task three times in a row or 15 minutes elapses. During the next phase of the experiment, we collect data from 10 user-controlled trials, which we then use to develop a model. Finally, each user performs the task under the four conditions detailed above (10 trials each). The order in which the control paradigms are presented to the user is randomized and counter-balanced to reduce the effect of experience. The study consisted of 16 participants (11 female, 5 male). All subjects gave their informed consent and the experiment was approved by Northwestern University's Institutional Review Board.

4.5. Analysis II: online MbSC

To study the efficacy of our MbSC algorithm in an online learning paradigm, we collect data from a fifth experimental condition, which we call *Online Koopman*.

4.5.1. Control condition.

- The main difference between the *Online Koopman* paradigm and the three previously described SC conditions is that the model of the joint human–machine system is learned online in real-time. In all other control conditions, all models were trained offline from observations gathered during a data collection phase. In the online paradigm, the model is updated continuously starting with the first collected set of observations.

In addition to the lack of a separate data collection phase, the online learning paradigm is distinct from the other SC conditions because of the data that we use to learn the model. In the SC conditions that use a model learned offline, we use all of the observations collected from the user demonstrations to learn the model. In the online learning paradigm we only update the model when the user input is admitted by the MDA controller. We choose this learning paradigm because it fits well conceptually with our long-term goal of using the outer-loop algorithm to provide stability and safety constraints on the SC system. It is important to note that at the beginning of the online learning paradigm, the MDA controller relies on randomly initialized control and system dynamics models. For this reason, the control let through to the system will be very noisy during the first few moments of the experiment, making the system difficult to control successfully for any human-in-the-loop. For this reason, it is important that the system dynamics and control models can be learned quickly, something we evaluate in Section 5.3. As soon as the learning process produces a model of any kind, the policy is computed using MPC techniques.

4.5.2. Protocol and participants. The online learning paradigm consists of 15 trials per user to allow us to evaluate possible learning curves. The model is updated at the same rate as the simulator (10 Hz) and is initialized naively (i.e., all values are sampled from a uniform distribution $[0, 1)$). This paradigm is presented as the final experimental condition to all subjects. The subjects are the same 16 participants as in Section 4.4. All of the data we analyze to evaluate the ideas presented in this section comes from the nonlinear MbSC study.

4.6. Analysis III: comparison of linear and nonlinear MbSC

To study the impact of nonlinear modeling and policy generation techniques on our MbSC paradigm, we compare results from the above study with a second study (consisting of a separate group of 16 participants) that enforces linear constraints on these parts of the system.

4.6.1. Control conditions. The same four control conditions from Analysis I are evaluated. The differences lie in

(1) the choice of basis function used to approximate the Koopman operator and (2) the choice of optimal control algorithm used to generate the autonomous policy. In this study, we use a linear basis, instead of a nonlinear basis, to approximate the Koopman operator, which consists of the first nine terms in Equation (6). We furthermore use a linear quadratic regulator (LQR), instead of a nonlinear MPC algorithm (SAC).

4.6.2. Protocol and participants. The same experimental protocol described in Section 4.4 was used, allowing us to perform a direct comparison between the two studies. Unlike the prior sections, the data we analyze to evaluate the ideas presented in this section comes from both the linear and nonlinear MbSC studies. The data from the linear MbSC study was previously analyzed in Broad et al. (2017) and was collected from a different set of 16 subjects, resulting in 32 total participants.

4.7. Statistical analysis

We analyze the results of the human subjects studies using statistical tests to compare the performance of participants along a set of pertinent metrics under the control conditions described in Section 4.4. Our analysis consists of one-way analysis of variance (ANOVA) tests conducted to evaluate the effect of the SC paradigm on each of the dependent variables in the study. These tests allow us to statistically analyze the effect of each condition while controlling for overinflated type I errors that are common with repeated *t*-tests. Each test is computed at a significance value of 0.05. When the omnibus *F*-test produced significant results, we conducted post-hoc pair-wise Student's *t*-tests using Holm–Bonferroni adjusted alpha values (Wright, 1992). The post-hoc *t*-tests allowed us to further evaluate the cause of the significance demonstrated by the ANOVA by comparing each pair of control paradigms separately. Similar to the ANOVA test, the Holm–Bonferroni correction was used to reduce the likelihood of type I errors in the post-hoc *t*-tests.

In addition to reporting the results of the statistical tests, we also use box-and-whisker diagrams to display specific metrics. In these plots, the box represents the *interquartile range* (IQR), which refers to the data that lies between the first and third quartiles. This area contains 50% of the data. The line inside the box represents the median value and the whiskers above and below the box are the minimum and maximum values inside 1.5 times the IQR. The small circles are outliers. The plots also depict the results of the reported statistical tests. That is, if a post-hoc *t*-test finds statistically significant differences between two conditions, we depict these results on the box-and-whisker diagrams using asterisks to represent the significance level ($*p<0.05$, $**p<0.01$, $***p<0.005$).

We note that this analysis is used for *all reported results*. Therefore, if we present the results of a *t*-test, it signifies that we have previously run an ANOVA and found a statistically significant difference. The reader can also assume that any unreported post-hoc *t*-tests failed to reject the null hypothesis.

5. Results

We now present the results of the desired analyses described in Sections 4.4, 4.5, and 4.6. Our analyses support the premise that MbSC is a valid and effective data-driven method for improving a human operator's control of an a priori unknown dynamic system. We also find the learned system models are generalizable across a population of users. Finally, we find that these models can be learned online in a fast, data-efficient manner.

5.1. Efficacy of MbSC

To evaluate the efficacy of our MbSC algorithm, we compute the average success rate under each control paradigm and examine the distribution of executed trajectories. Our analysis compares the User Only control condition with each of the SC conditions (Individual Koopman, General Koopman, and Expert Koopman). All of the data we analyze to evaluate the ideas presented in this section comes from the nonlinear MbSC study.

5.1.1. Task success and user skill. A trial is considered a success when the user is able to meet the conditions defined in Section 4.3. We can interpret the success rate of a user, or SC system, on a set of trials as a measure of skill. The greater the skill, the higher the success rate. By comparing the average success rate under the User Only control paradigm with the average success rate under the SC paradigms, we can analyze the impact of the assistance provided by the autonomous partner.

The average number of successful trials produced in each control condition are displayed in Figure 7. An ANOVA shows that the choice of control paradigm has a significant effect on the success rate ($F(3,59)=4.58$, $p<0.01$). Post-hoc *t*-tests find that users under the SC conditions show statistically significant improvements in the success rate when compared with their performance under the User Only control condition ($p<0.01$, for all cases). No other pairings are found to be statistically distinct. This result demonstrates that the assistance provided by the autonomous agent significantly improves the skill of the joint human–machine system, thereby validating the efficacy of MbSC. This result is inline with related work that aims to provide similar assistance, such as can be found in the virtual fixtures/guides literature (Abbink et al., 2012; Forsyth and MacLean, 2005; Griffiths and Gillespie, 2005). Importantly, however, unlike these prior methods, MbSC does not require a priori knowledge of the system dynamics or the human operator.

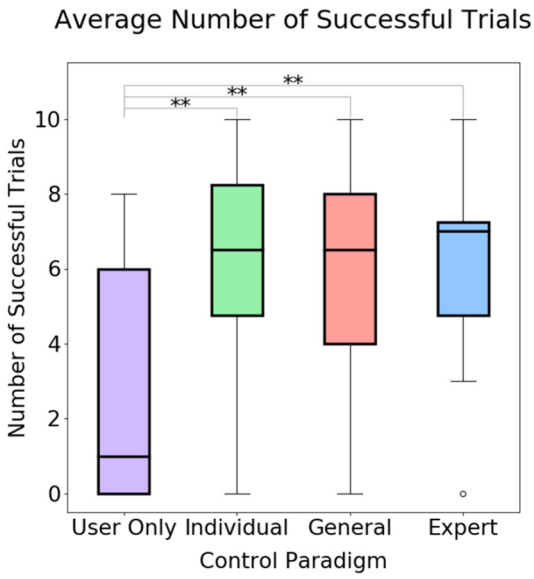

Fig. 7. Number of successful trials under each control condition.

The fact that there are no observed differences in task performance between the individual, general, and expert cases suggests that the source of the data used to learn the model may not be important in developing helpful autonomous assistance in the SC of dynamic systems (discussed further in Section 5.2). An alternative interpretation of this data could be that the discrepancy of skill demonstrated by the participants in the individual, general, and expert cases was not large enough to produce any potential difference in performance. This, however, is not likely, as the expert demonstrator (the lead author) was able to easily achieve the desired goal state during every demonstration (i.e., 10 out of 10 trials). In contrast, the average subject who provided data in the individual and general cases performed similarly to how participants performed under the User Only cases in Figure 7 (i.e., about 1 in 10 successful demonstrations).

5.1.2. Distribution of trajectories: qualitative. We further analyze the different control conditions through a comparison of the distribution of trajectories we observe in each condition. Unlike the success metric, this analysis is not based on task performance, and is instead performed to evaluate the control skill exhibited by either the human operator alone or the joint human–machine system. Figure 8 depicts trajectory plots that represent the most frequently occupied sections of the state space. The plots are generated using data separated based on the control condition (columns) and whether the user was able to complete the task on a given trial (rows).

The first distinction we draw is between the User Only control condition and the three SC conditions. In particular, the distribution of trajectories in the User Only condition depicts both larger excursions away from the target and lower levels of similarity between individual executions. When we focus specifically on which parts of the state space users spend the most time in (as represented by the intensity of the plots), we see two main clusters of high intensity (around the start and goal locations) in the SC conditions, whereas we see a wider spread of high-intensity

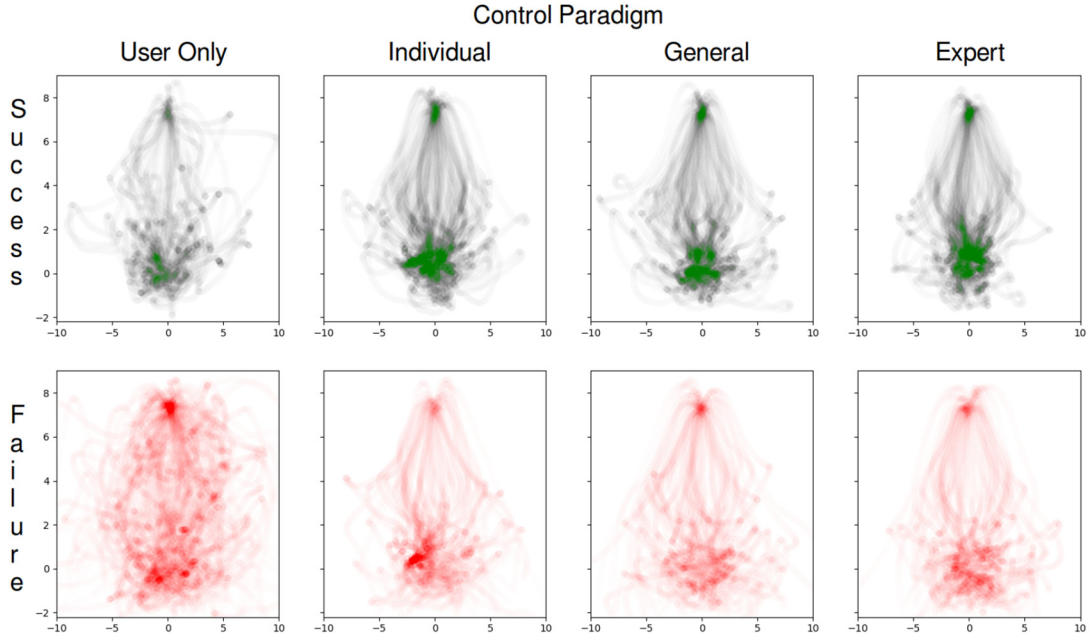

Fig. 8. Trajectory plots that visualize the most frequently visited parts of the state space. The data is broken down by control condition (columns) and whether the trial was successful (rows). The plots are generated by overlaying each trajectory with a low opacity and the intensity of the plots therefore represents more frequently visited portions of the state space.

values in the User Only control condition. This suggests more purposeful motions under the SC conditions.

The second distinction we draw focuses on a comparison between the successful and unsuccessful trials. Specifically, we note that trajectory plots computed from the failed trials under the SC conditions demonstrate similar properties (e.g., the extent of the excursions away from the target, as well as two main clusters of intensity) to the trajectory plots computed from successful trials under the SC conditions. This suggests that users may have been closer to succeeding in these tasks than the binary success metric gives them credit for. By comparison, the trajectory plot computed from the failed trials under the User Only control condition depicts a significantly different distribution of trajectories with less structure. Specifically, we observe numerous clusters of intensity that represent time spent far away from the start and goal locations. This suggests that users were particularly struggling to control the system in these cases.

5.1.3. Distribution of trajectories: quantitative. These observations are supported by an evaluation of the ergodicity (Mathew and Mezić, 2011) of the distributions of trajectories described previously. We find users under the SC paradigm are able to produce trajectories that are more ergodic with respect to the goal location than users under User Only control, which means that they spend a significantly larger proportion of their time navigating near the goal location under SC.

To perform this comparison, we compute the ergodicity of each trajectory with respect to a probability distribution defined by a Gaussian centered at the goal location (which represents highly desirable states). This metric can be calculated as the weighted Euclidean distance between the Fourier coefficients of the spatial distribution and the trajectory (Miller et al., 2016).

Similar to our qualitative analysis of the trajectory plots in Figure 8, we first compare ergodicity between the different control conditions by analyzing *all* the trajectories observed under each condition. An ANOVA showed that the effect of the SC paradigm on trajectory ergodicity is significant ($F(3, 640) = 12.97,\ p < 0.00001$). Post-hoc t-tests find statistically significant differences between the performance of the users in the User Only control condition and users in the SC conditions based on the individual, general, and expert datasets ($p < 0.0005,\ p < 0.001,\ p < 0.0005$, respectively). No other pairings demonstrate statistically distinct results. We interpret this result as additional evidence that MbSC improves the skill of the human partner in controlling the dynamic system.

We further analyze the ergodicity results by separating the trajectories based on whether they come from an unsuccessful or successful trial. An ANOVA computed over all control conditions showed that the effect of the SC paradigm on trajectory ergodicity is significant for both unsuccessful ($F(3, 310) = 6.60,\ p < 0.0005$) and successful ($F(3, 325) = 7.20,\ p < 0.0005$) trials. Post-hoc t-tests find statistically significant differences between the performance of the users in the User Only control condition and users in the SC conditions ($p < 0.005$ in all unsuccessful cases, $p < 0.05$ in all successful cases). No other pairings reject the null hypothesis. These results suggest that the SC paradigm is helpful in improving the user's skills even when they provide input that is ultimately unsuccessful in achieving the task. Furthermore, our SC paradigm is helpful, even when users are performing at their best. Thus, for both failed and successful trials, users exhibit a greater amount of control skill than when there is no assistance.

5.2. Generalizability of SC paradigm

We continue the evaluation of our human subjects study with an analysis of the generalizability of the learned system models and our MbSC algorithm. All of the data we analyze to evaluate the ideas presented in this section comes from the nonlinear MbSC study. As reported in Section 5.1, we find no statistical evidence that the source of the data used to train the model impacts the efficacy of the SC paradigm. This test was again conducted using an ANOVA, which can be used to evaluate differences between groups by comparing the mean and variance computed from the data collected during the experimental trials. When we compare the success rate of users in each SC condition, we find no statistically significant difference. However, we do find a significant difference between the user's performance under each SC condition and the User Only condition. The same result holds when we compare each control condition along the ergodic metric described in Section 5.1 and visualized by trajectory plots in Figure 8. Taken together, these results suggest that the efficacy of the assistance provided by the autonomous agent is *independent of the source of the data used to learn a model of the joint system*. That is, models trained on data collected from an individual user generalize to a larger population of human partners.

To further analyze the generalizability of the MbSC paradigm, we evaluate the participants' interactions with the outer-loop autonomous control. We are interested in whether users agree more often with the autonomy when control signals are produced based on models learned from their personal demonstration data. To evaluate this idea, we look at the percentage of user inputs that are let through as control to the dynamic system based on our control allocation method (MDA). The average agreement metric is broken down by control condition and presented in Figure 9.

An ANOVA shows that the effect of the source of the model data on the average agreement is not significant in either the main thruster ($F(2, 44) = 0.87,\ p = 0.43$) nor the side thruster ($F(2, 44) = 0.38,\ p = 0.69$). These results show a uniformity in the response to system state across users and suggest that the system is able to adapt to the person, instead of requiring a personalized notion of the user and system.

We interpret this finding as further evidence of the generalizability of our MbSC paradigm. In particular, we find that it is

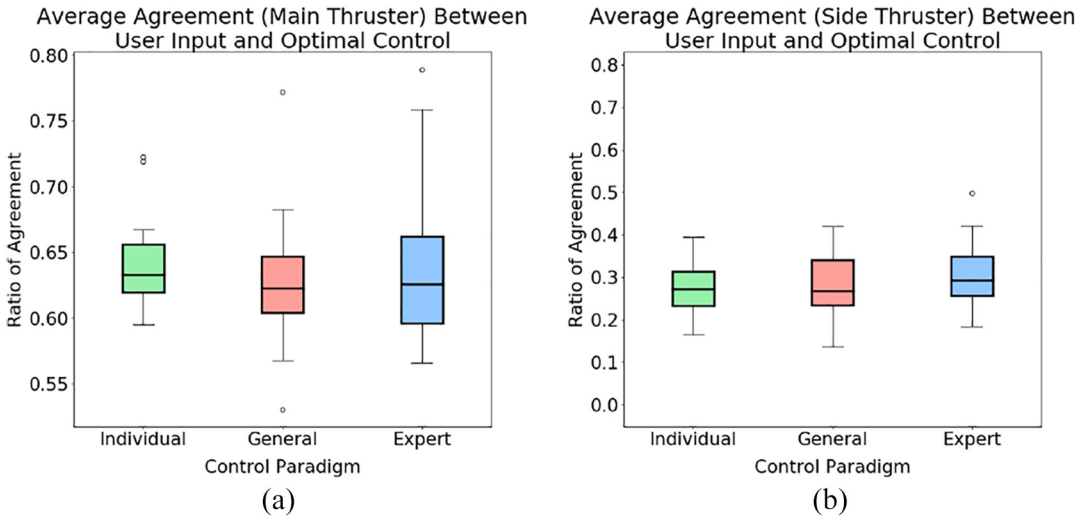

Fig. 9. Average agreement between user and optimal control algorithm as defined by the MDA (Equation (1)) along the (a) main and (b) side thrusters.

not necessary to incorporate demonstration data from individual users when developing MbSC. This result replicates findings from our analysis of data collected under a SC paradigm that enforced a linear constraint on the model learning and policy generation techniques (Broad et al., 2017).

5.3. Online learning SC

We next evaluate our MbSC algorithm in an online learning paradigm. Our evaluation considers the sample complexity of our model-based learning algorithm through a comparison of the *impact each SC paradigm has on the skill of the joint system over time*. All of the data we analyze to evaluate the ideas presented in this section comes from the nonlinear MbSC study. Our statistical analysis is a comparison of the percent of participants who succeed under each paradigm *by trial number*, shown in Figure 10. We remind the reader that users participate in 15 trials of the Online Koopman condition while they participate in 10 trials of the 4 other experimental conditions. For comparison, we only plot the first 10 trials of the Online Koopman data, though we note that the improved success rate is sustained over the final 5 trials. From this plot, we can see that users in the Online Koopman SC condition start off performing poorly, but by around trial 7 start performing on a par with the other SC conditions.

Here, we also note the number of trials used to train the model of the system and control dynamics in each condition. In the individual and expert conditions, data is collected from 10 trials to train the model. In the general condition, data is collected from three different users who each control the system for 10 trials each, which means the model is trained from a total of 30 trials. Finally, as discussed above, in the online condition, the model is learned continuously over the course of 15 trials.

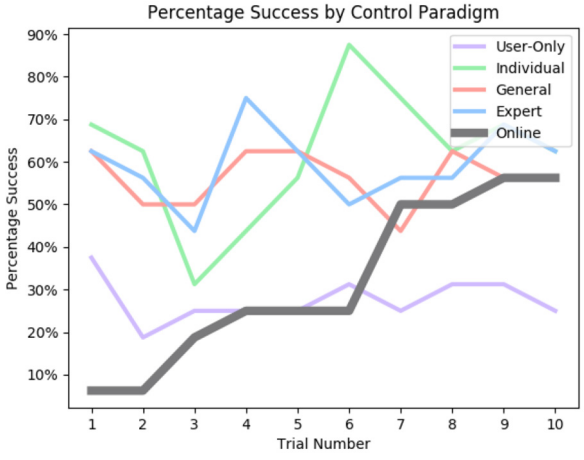

Fig. 10. Average percentage success by trial for the first 10 trials by control condition. Users under all SC conditions using models learned offline (individual, general, expert) outperform the User Only control condition across all trials. Users under the SC condition using models learned online (online) start off performing poorly, but quickly begin to outperform the User Only control condition and, in the end, achieve the same level of success as those under the offline SC conditions.

To provide quantitative evidence of this visual trend, we perform the same types of statistical analyses as in previous sections, but now include data from the *Online Koopman* as a fifth experimental condition. For ease of discussion we refer to the *Individual, General*, and *Expert Koopman* MbSC conditions as the offline learning conditions, and the *Online Koopman* MbSC as the online learning condition. As users provide more data in the Online Koopman condition than in all other conditions, we perform two sets of analyses. First, we compare the data from the first 10

trials from the Online Koopman condition to all other control conditions. We then re-perform the same tests, but use the final 10 trials from the Online Koopman condition. By comparing these results, we can evaluate the efficacy of the online learning paradigm, and also analyze the effect of the amount of data used during the learning process.

5.3.1. Statistical analysis of the first 10 trials. An ANOVA finds a statistically significant difference between the various control conditions along the primary success metric ($F(4, 74) = 5.35,\ p < 0.001$). Post-hoc t-tests find that all offline learning conditions significantly outperform the Online Koopman and User Only control conditions ($p < 0.05$ for all cases). We do not find the same statistically significant difference between the User Only and Online Koopman conditions. These results suggest that users under the online learning paradigm initially perform on a par with how they perform under the user only control paradigm, but worse than under the offline control conditions. This analysis is consistent with our expectations since, in the online condition, the model of the joint system is initialized randomly and therefore does a poor job of assisting the user. However, it is also important that this online SC does not degrade performance in comparison with the User Only paradigm, suggesting that there is little downside to employing the online learning paradigm during learning.

5.3.2. Statistical analysis of the final 10 trials. As a point of comparison, we now re-run the same statistical tests using the final 10 trials from the Online Koopman condition. An ANOVA finds a statistically significant difference between the various control conditions along the primary success metric ($F(4, 74) = 3.55,\ p < 0.05$; see Figure 11). Post-hoc t-tests find that all SC conditions (using models learned offline and online) significantly outperform the User Only control paradigm ($p < 0.01$ for all conditions). This result is different from our analysis of the first 10 trials and suggests that the learned model improves significantly with more data and now is on a par with the models learned in the offline conditions. No other pairings show statistically significant differences.

The visual trend present in Figure 10 and the statistical analysis demonstrated in Figure 11 suggest that the Koopman operator is able to quickly learn an actionable representation of the joint human–machine system. These results also demonstrate the efficacy of our MbSC algorithm in an online learning scenario and in limited data regimes. Here we note that follow-up studies are required to tease apart the impact of the model learning process and the user's experience controlling the dynamic system when comparing the offline paradigms to the online paradigm. Notably, in the offline learning paradigm, users undergo 10 trials of training at the start of the experiment. In contrast, and as stated in Section 4.5.2, all users operate the system under the online learning paradigm as the final condition.

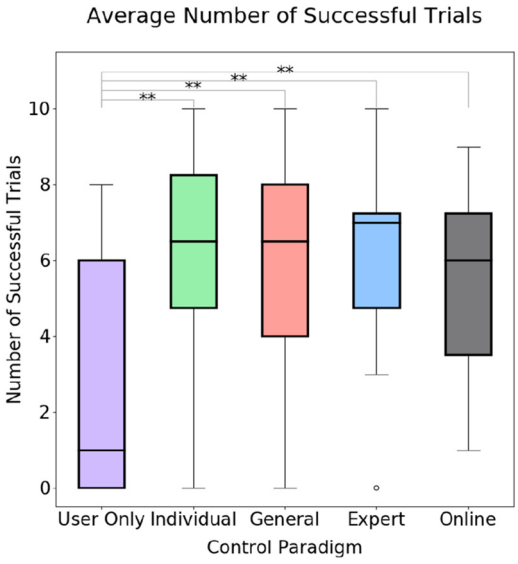

Fig. 11. Number of successful trials under each control condition (including an online learning paradigm). We find statistically significant differences between the User Only condition and each SC condition ($p < 0.01$).

For this reason, we do not account for user experience in this condition and therefore highlight the data efficiency of the model learning process instead of the overall task performance in this condition. The main takeaway from this portion of the analysis is therefore that an actionable Koopman operator can be learned *quickly*, from significantly less data than alternative approaches commonly found in the literature, such as neural networks (Nagabandi et al., 2018).

5.4. Linear and nonlinear MbSC

The final piece of analysis we perform in this work is related to the impact that nonlinear modeling and policy generation techniques have on our MbSC paradigm. For this analysis we compare the User Only control condition with the three offline SC conditions. Unlike the prior sections, the data we analyze to evaluate the ideas presented in this section comes from both the linear and nonlinear MbSC studies.

The average success rate of users under each control paradigm for both studies is presented in Figure 12. In the linear study (Broad et al., 2017) we observe a trend (see Figure 12(a)) that suggests users perform better under the SC paradigm, but we do not find statistically significant evidence of this observation. In contrast, we find that MbSC using nonlinear modeling and policy generation techniques does statistically improve the success rate when compared with a User Only control paradigm.

One potential explanation for the difference we find in the results of the two studies is that the nonlinear basis produces more accurate models of the system dynamics than the linear

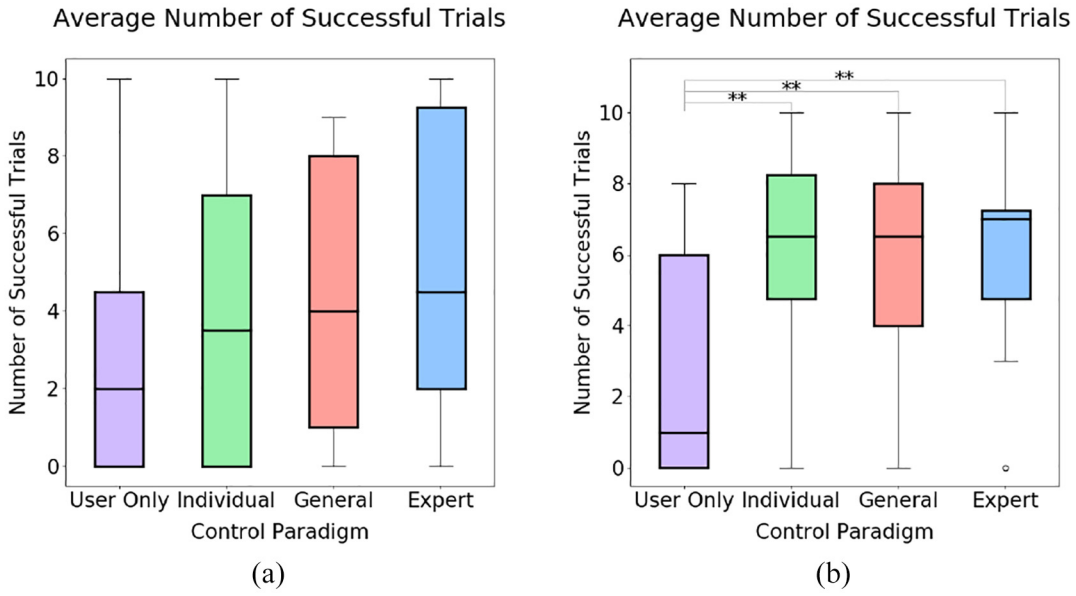

Fig. 12. A comparison of the average success rate under (a) linear and (b) nonlinear MbSC to user only control.

basis. To explore this explanation, we evaluate the predictive capabilities of a Koopman operator learned with a linear basis to one learned with a nonlinear basis. This analysis is performed by comparing the predicted system states with ground truth data. We evaluate the error (mean and variance) as a function of prediction horizon (also known as the H-step error). Figure 3 depicts the raw error (in meters) of Koopman operators trained using linear and nonlinear bases.

Our analysis of the predictive capabilities of the Koopman operator models demonstrates that each is highly accurate. The nonlinear model does slightly outperform the linear model as the prediction horizon grows; however, we find that both models are able to produce single-step predictions with error on the scale of 10^{-3} meters. As a reminder to the reader, the state space is bounded with $X \in (-10, 10)$, $Y \in (0, 16)$. This analysis suggests that the choice of basis function does not cause the observed difference in average success rate between the two studies. Instead, the important design decision may be the choice of MPC algorithm. In the linear study we use an infinite horizon LQR to produce autonomous control policies, whereas in the nonlinear study we use a receding-horizon MPC to produce autonomous control. Our interpretation of these results is that the receding horizon nature of MPC is better suited to the visual planning approach that human operators use when solving the lunar lander task.

6. Discussion

In this section, we highlight a number of main takeaways that stem from our analysis. To begin, the results of our human subjects studies demonstrate that our MbSC paradigm is able to (1) successfully learn a model of the joint human–machine system from observation data, and (2) use the learned system model to generate autonomous policies that can help assist a human partner achieve a desired goal. We evaluate the predictive capabilities of the learned system models through a comparison with ground truth trajectory data (see Figure 3) and evaluate the impact of the assistive SC system through a comparison of performance (success rate, see Figure 7) with a User Only (or natural learning) control paradigm. All analyses support the idea that MbSC can help improve the control skill of a human operator both when they are able to achieve a task on their own and when they are not.

Additional evaluations demonstrate that the learned system and control dynamics generalize across users, and suggests that, unlike in other human–machine interaction paradigms (Javdani et al., 2015; Macindoe et al., 2012; Sadigh et al., 2016), personalization is not required for defining SC paradigms of generic human–machine systems. Specifically, we find that the demonstration data used to learn the system and control models does not need to come from an optimal, or expert, controller, and can instead come from *any* human operator. Therefore, at a base level, the controller does not need to be personalized to each individual user as the learned model captures all necessary information. This idea is important for application in real-world scenarios where personalization of control paradigms can be time-consuming, costly, and challenging to appropriately define, often due to the variety in preferences described by human operators (Erdogan and Argall, 2017; Gopinath et al., 2016).

We also demonstrate that our approach can be used in an online learning setting. Importantly, we find that the model is able to learn very quickly, from limited amounts of data. In the Online Koopman condition, each trial took an average of 18 seconds, and therefore provided 180 data

points. From our analysis in Section 5.3.2, we find that we are able to learn an effective model of the joint system after only 5 trials (or about 900 data points). Our model learning technique is also well suited for an online learning paradigm as it is not computationally intensive and can easily run at 50 Hz on a Core i7 laptop with 8 GB of RAM. In addition, we find that, even during the learning process, the application of the online MbSC algorithm does not significantly degrade the performance of the human operator.

Finally, we also evaluate the impact that nonlinear modeling and policy generation techniques have on our MbSC algorithm (Broad et al., 2017). In particular, we replace the nonlinear modeling and policy generation techniques with linear counterparts and compare how they impact the ability of a human operator to achieve a desired task. This requires using a nonlinear basis when computing the approximation to the Koopman operator and using nonlinear MPC (SAC) to generate the autonomous policy. We find that the nonlinear MbSC paradigm produces a joint human–machine system that is significantly better along the primary performance metric (task success) than users under a user only control paradigm. The same result is not found from the data collected under a SC paradigm that enforced linear constraints (see Figure 12).

7. Conclusion

In this work, we have introduced MbSC. A particularly important aspect of this work is that *we do not rely on a priori knowledge, or a high-fidelity model, of the system dynamics*. Instead, we learn the system dynamics *and* information about the user interaction with the system directly from data. We learn this model through an approximation to the Koopman operator, an infinite-dimensional linear operator that can exactly model nonlinear dynamics. By learning the joint system dynamics through user interaction, the robot's understanding of the human is implicit to the system definition.

Results from two human subjects studies (consisting of 32 total participants) demonstrate that incorporating the learned models into our SC framework statistically improves the performance of the operator along a number of pertinent metrics. Furthermore, an analysis of trajectory ergodicity demonstrates that our SC framework is able encourage the human–machine system to spend a significantly greater percentage of time in desirable states. We also find that the learned system models are able to be used in SC systems that generalize across a population of users. Finally, we find that, using this approach, models can be efficiently learned online. In conclusion, we believe that our approach is an effective step towards SC of human–machine systems with unknown dynamics. This framework is sufficiently general that it could be applied to any robotic system with a human in the loop. In addition, we have made our code available online at https://github.com/asbroad/model_based_shared_control, and include a video depicting a user's control of the dynamic system and the impact of MbSC in the supplementary material.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This material is based upon work supported by the National Science Foundation (grant number CNS 1329891). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the aforementioned institutions.

ORCID iD

Alexander Broad https://orcid.org/0000-0001-9230-7891
Todd Murphey https://orcid.org/0000-0003-2262-8176

References

Abbink DA, Mulder M and Boer ER (2012) Haptic shared control: smoothly shifting control authority? *Cognition, Technology and Work* 14(1): 19–28.

Abbott JJ, Marayong P and Okamura AM (2007) Haptic virtual fixtures for robot-assisted manipulation. In: *International Symposium on Robotics Research*. New York: Springer, pp. 49–64.

Abraham I, De La Torre G and Murphey TD (2017) Model-based control using Koopman operators. In: *Robotics: Science and Systems*.

Abraham I and Murphey TD (2019) Active learning of dynamics for data-driven control using Koopman operators. *Transactions on Robotics* 35(5): 1071–1083.

Ansari AR and Murphey TD (2016) Sequential action control: Closed-form optimal control for nonlinear and nonsmooth systems. *Transactions on Robotics* 32(5): 1196–1214.

Argall BD, Chernova S, Veloso M and Browning B (2009) A survey of robot learning from demonstration. *Robotics and Autonomous Systems* 57(5): 469–483.

Barto AG, Bradtke SJ and Singh SP (1995) Learning to act using real-time dynamic programming. *Artificial Intelligence* 72(1-2): 81–138.

Berrueta TA, Pervan A, Fitzsimons K and Murphey TD (2018) Dynamical system segmentation for information measures in motion. *Robotics and Automation Letters* 4(1): 169–176.

Broad A, Murphey T and Argall B (2017) Learning models for shared control of human–machine systems with unknown dynamics. In: *Robotics: Science and Systems*.

Broad A, Murphey T and Argall B (2018) Operation and imitation under safety-aware shared control. In: *Workshop on the Algorithmic Foundations of Robotics*.

Broad A, Murphey T and Argall B (2019) Highly parallelized data-driven MPC for minimal intervention shared control. In: *Robotics: Science and Systems*.

Brockman G, Cheung V, Pettersson L, et al. (2016) OpenAI Gym. *arXiv Preprint* abs/1606.01540.

Bruder D, Gillespie B, Remy CD and Vasudevan R (2019) Modeling and control of soft robots using the Koopman operator and model predictive control. In: *Robotics: Science and Systems*.

Brunton BW, Johnson LA, Ojemann JG and Kutz JN (2016) Extracting spatial-temporal coherent patterns in large-scale neural recordings using dynamic mode decomposition. *Journal of Neuroscience Methods* 258: 1–15.

Celemin C, Ruiz-del Solar J and Kober J (2019) A fast hybrid reinforcement learning framework with human corrective feedback. *Autonomous Robots* 43(5): 1173–1186.

de Winter JC and Dodou D (2011) Preparing drivers for dangerous situations: A critical reflection on continuous shared control. In: *International Conference on Systems, Man, and Cybernetics*, pp. 1050–1056.

Dragan A and Srinivasa S (2013) A policy blending formalism for shared control. *The International Journal of Robotics Research* 32(7): 790–805.

Drews P, Williams G, Goldfain B, Theodorou EA and Rehg JM (2017) Aggressive deep driving: Combining convolutional neural networks and model predictive control. In: *Conference on Robot Learning*, Vol. 78, pp. 133–142.

Erdogan A and Argall B (2017) The effect of robotic wheelchair control paradigm and interface on user performance, effort and preference: An experimental assessment. *Robotics and Autonomous Systems* 94: 282–297.

Fitzsimons K, Tzorakoleftherakis E and Murphey T (2016) Optimal human-in-the-loop interfaces based on Maxwell's demon. In: *American Control Conference*, pp. 4397–4402.

Forsyth BA and MacLean KE (2005) Predictive haptic guidance: Intelligent user assistance for the control of dynamic tasks. *Transactions on Visualization and Computer Graphics* 12(1): 103–113.

Gopinath D, Jain S and Argall BD (2016) Human-in-the-loop optimization of shared autonomy in assistive robotics. *Robotics and Automation Letters* 2(1): 247–254.

Griffiths PG and Gillespie RB (2005) Sharing control between humans and automation using haptic interface: Primary and secondary task performance benefits. *Human Factors* 47(3): 574–590.

Hoeniger T (1998) Dynamically shared control in human–robot teams through physical interactions. In: *International Conference on Intelligent Robots and Systems*, pp. 744–749.

Hoffman G and Breazeal C (2004) Collaboration in human–robot teams. In: *Intelligent Systems Technical Conference*, pp. 6434–6452.

Hua JC, Roy S, McCauley JL and Gunaratne GH (2016) Using dynamic mode decomposition to extract cyclic behavior in the stock market. *Physica A: Statistical Mechanics and its Applications* 448: 172–180.

Javdani S, Srinivasa S and Bagnell JA (2015) Shared autonomy via hindsight optimization. In: *Robotics: Science and Systems*, Rome, Italy.

Jovanović MR, Schmid PJ and Nichols JW (2014) Sparsity-promoting dynamic mode decomposition. *Physics of Fluids* 26(2): 024103.

Kaelbling LP, Littman ML and Moore AW (1996) Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* 4: 237–285.

Khansari-Zadeh SM and Billard A (2011) Learning stable nonlinear dynamical systems with Gaussian mixture models. *IEEE Transactions on Robotics* 27(5): 943–957.

Kim HK, Biggs J, Schloerb W, et al. (2006) Continuous shared control for stabilizing reaching and grasping with brain–machine interfaces. *IEEE Transactions on Biomedical Engineering* 53(6): 1164–1173.

Kim WS, Hannaford B and Fejczy A (1992) Force-reflection and shared compliant control in operating telemanipulators with time delay. *Transactions on Robotics and Automation* 8(2): 176–185.

Klus S, Koltai P and Schütte C (2015) On the numerical approximation of the Perron–Frobenius and Koopman operator. *arXiv Preprint* 1512.05997.

Koopman BO (1931) Hamiltonian systems and transformation in Hilbert space. *Proceedings of the National Academy of Sciences* 17(5): 315–318.

Kortenkamp D, Keirn-Schreckenghost D and Bonasso RP (2000) Adjustable control autonomy for manned space flight. In: *Aerospace Conference*, Vol. 7. IEEE, pp. 629–640.

Lasota PA, Fong T, Shah JA, et al. (2017) A survey of methods for safe human–robot interaction. *Foundations and Trends in Robotics* 5(4): 261–349.

Li W and Todorov E (2004) Iterative linear quadratic regulator design for nonlinear biological movement systems. In: *International Conference on Informatics in Control, Automation and Robotics*, Vol. 1, pp. 222–229.

Losey DP, McDonald CG, Battaglia E and O'Malley MK (2018) A review of intent detection, arbitration, and communication aspects of shared control for physical human–robot interaction. *Applied Mechanics Reviews* 70(1): 010804.

Losey DP, Srinivasan K, Mandlekar A, Garg A and Sadigh D (2019) Controlling assistive robots with learned latent actions. *arXiv Preprint* 1909.09674.

Macindoe O, Kaelbling LP and Lozano-Pérez T (2012) POMCoP: Belief space planning for sidekicks in cooperative games. In: *Artificial Intelligence and Interactive Digital Entertainment Conference.*

Marayong P and Okamura AM (2004) Speed–accuracy characteristics of human–machine cooperative manipulation using virtual fixtures with variable admittance. *Human Factors* 46(3): 518–532.

Mathew G and Mezić I (2011) Metrics for ergodicity and design of ergodic dynamics for multi-agent systems. *Physica D: Nonlinear Phenomena* 240(4): 432–442.

Matni N and Oishi M (2008) Reachability-based abstraction for an aircraft landing under shared control. In: *American Control Conference*, pp. 2278–2284.

Mayne D (1966) A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems. *International Journal of Control* 3(1): 85–95.

Miller LM, Silverman Y, MacIver MA and Murphey TD (2016) Ergodic exploration of distributed information. *IEEE Transactions on Robotics* 32(1): 36–52.

Mitrovic D, Klanke S and Vijayakumar S (2010) Adaptive optimal feedback control with learned internal dynamics models. In: *From Motor Learning to Interaction Learning in Robots*. Berlin: Springer, pp. 65–84.

Nagabandi A, Kahn G, Fearing RS and Levine S (2018) Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In: *International Conference on Robotics and Automation*, IEEE, pp. 7559–7566.

Nguyen-Tuong D, Peters JR and Seeger M (2009) Local Gaussian process regression for real time online model learning. In: *Advances in Neural Information Processing Systems*, pp. 1193–1200.

Noohi E, Žefran M and Patton JL (2016) A model for human–human collaborative object manipulation and its application to human–robot interaction. *Transactions on Robotics* 32(4): 880–896.

Nudehi SS, Mukherjee R and Ghodoussi M (2005) A shared-control approach to haptic interface design for minimally invasive telesurgical training. *Transactions on Control Systems Technology* 13(4): 588–592.

Peternel L, Oztop E and Babič J (2016) A shared control method for online human-in-the-loop robot learning based on locally weighted regression. In: *International Conference on Intelligent Robots and Systems*, pp. 3900–3906.

Proctor JL, Brunton SL and Kutz JN (2016) Generalizing koopman theory to allow for inputs and control. arXiv Preprint abs/1602.07647.

Reddy S, Levine S and Dragan A (2018) Shared autonomy via deep reinforcement learning. In: *Robotics: Science and Systems*.

Rowley CW, Mezić I, Bagheri S, Schlatter P and Henningson DS (2009) Spectral analysis of nonlinear flows. *Journal of Fluid Mechanics* 641: 115–127.

Sadigh D, Sastry SS, Seshia SA and Dragan A (2016) Information gathering actions over human internal state. In: *International Conference on Intelligent Robots and Systems*. IEEE, pp. 66–73.

Schaal S and Atkeson CG (2010) Learning control in robotics. *IEEE Robotics and Automation Magazine* 17(2): 20–29.

Schmid PJ (2010) Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics* 656: 5–28.

Tassa Y, Mansard N and Todorov E (2014) Control-limited differential dynamic programming. In: *International Conference on Robotics and Automation*. IEEE, pp. 1168–1175.

Tibshirani R (1996) Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society. Series B (Methodological)* 58: 267–288.

Trieu HT, Nguyen HT and Willey K (2008) Shared control strategies for obstacle avoidance tasks in an intelligent wheelchair. In: *International Conference on Engineering in Medicine and Biology Society*, pp. 4254–4257.

Tu JH, Rowley CW, Luchtenburg DM, Brunton SL and Kutz JN (2014) On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics* 1: 391–421.

Tzorakoleftherakis E and Murphey T (2015) Controllers as filters: Noise-driven swing-up control based on Maxwell's demon. In: *Proceedings of the Conference on Decision and Control*, pp. 4368–4374.

Williams G, Wagener N, Goldfain B, et al. (2017) Information theoretic MPC for model-based reinforcement learning. In: *International Conference on Robotics and Automation*.

Williams MO, Kevrekidis IG and Rowley CW (2015a) A data-driven approximation of the Koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science* 25(6): 1307–1346.

Williams MO, Rowley CW and Kevrekidis Y (2015b) A kernel-based approach to data-driven Koopman spectral analysis. *Journal of Computational Dynamics* 2(2): 247–265.

Wright SP (1992) Adjusted *p*-values for simultaneous inference. *Biometrics* 48(4): 1005–1013.