A Framework for Efficient Teleoperation via Online Adaptation

Xuning Yang, Koushil Sreenath, and Nathan Michael

***Abstract*— We propose a task-independent adaptive teleoperation methodology that seeks to improve operator performance and efficiency by concurrently modeling user intent and adapting the set of available actions according to the predicted intent. User input selects a robot motion from a finite set of dynamically feasible and safe motions, represented as a motion primitive library. User intent is modeled as a probabilistic distribution with respect to future actions that represents the likelihood of action selection given recent user input, which can be formulated independent of task, environment, or user. As the intent model becomes increasingly confident, the action set is adapted in order to reduce the error between the intended and actual performance. Experimental evaluation of teleoperating a quadrotor for nonaggressive, single-intent maneuvers such as following a racetrack and conducting a free-hand helix motion shows improved performance, validating that the approach provides efficient adaptation towards achieving the user intent.**

I. Introduction

For high dimensional nonlinear robotic systems, the choice of using teleoperation has been the preferred method of control and planning for navigating robots in unfamiliar scenarios. With teleoperation, human intuition can be leveraged to improve task performance, as humans can alleviate computational demands of perception and semantic inference required in order to generate real-time dynamically feasible trajectories for an autonomous robot. We are interested in improving the operator efficiency in robot control by actively aiding the operator by methods of feedback, without requiring or assuming access to prior knowledge of the task, environment, or the user. Specifically, efficiency of the operator is improved by minimizing the entropy of the control inputs, thereby minimizing operator energy and achieving higher performance in the form of smoother trajectories. The proposed framework concurrently estimates user intent online and adaptively updates the action set available to the operator. This framework establishes equivalence between the action space and trajectories via the use of motion primitives, and formulate the operator as an optimal controller. As such, we remove the dependency of the user model on the environment and achieve inference of operator intent using recent operator action selection.

Humans are imperfect controllers that are prone to physiological stress, fatigue, inexperience, as well as random errors [1]. In addition, efficient control of a robotic system derives from experience interacting with the robot, such that the operator 'internalizes' the robot's dynamics [1]. The operator learns the dynamics of the vehicle by exploring the available actions and observing feedback of the change in the robot state, either visually or haptically. Effectively, the user is implicitly learning the mapping from the external input to the robot's dynamics and mastery is achieved when user is able to invert the dynamics to issue commands. The learning time for any user is correlated with the complexity of the system. In high dimensional systems such as aerial robots, this is especially difficult due to the coupling in dynamics.

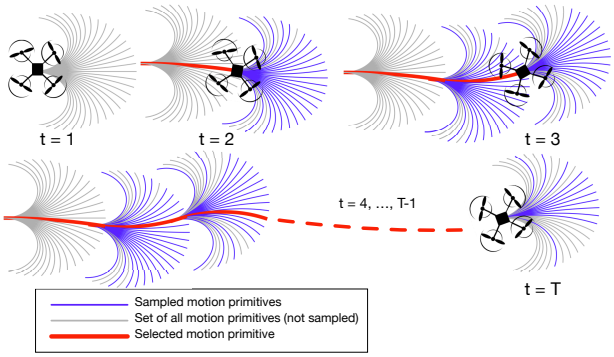


Fig. 1: The user begins each trial with access to an uniformly dense motion primitive library (MPL). As time progresses, a belief distribution is constructed about the underlying dense MPL. At each input time t, a subset of the motion primitives is sampled with respect to the belief distribution. The operator input corresponds to a motion primitive from the subsampled set via a selector function.

Prior work in introducing automation to assist operators in controlling robotic systems has primarily focused on leveraging information about the environment in order to augment the user's input with an optimal action to reach the predicted goal of the operator [2]. A direct method is to leverage *shared autonomy* in combining an autonomous assistive input with the user's input, based on varying levels of desired human intervention. The dominant methodology in shared autonomy is arbitration of the user's policy with a predicted policy [2]. In these frameworks, the robot typically infers the desired goal, trajectory, or task that the user wishes to achieve. An arbitration function blends the input from the prediction and the user input, with the blended input passed to the robot. These methods are suited for episodic tasks with a terminating goal, where a finite set of global goals are evaluated probabilistically, for example, using Maximum Entropy [3, 4]. Similarly, classification methods to identify tasks have also been employed. Gao et al. [5] utilize a recursive Bayesian filter to infer the appropriate scenario, such as doorway crossing and pole inspection, in order to generate the appropriate robot input to be arbitrated with the user input. The user model and the arbitration function employed in these formulations are generally trained or defined

The authors are affiliated with the Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15213, USA. {xuning, koushils, nmichael}@cmu.edu

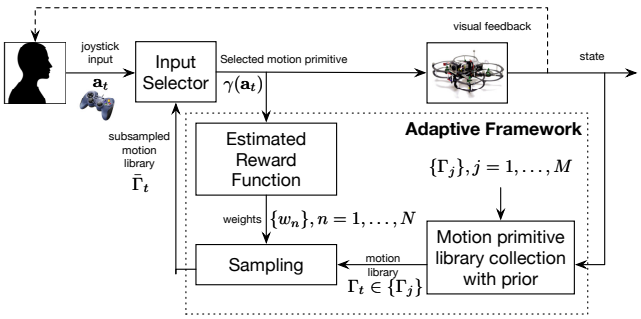


Fig. 2: System diagram of the proposed adaptive framework.

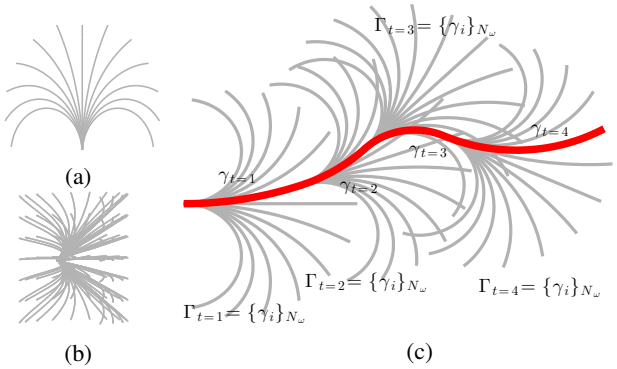


Fig. 3: (a) A motion primitive library generated for a 2D unicycle model with varying angular velocity ω. (b) A 3D motion library generated using (1). (c) A trajectory formulated for four time steps. The selected motion primitives at each time step (in red) form a single trajectory.

a priori, and require stringent assumptions with respect to operator proficiency, environmental restrictions, and system limitations [6, 7]. Instead of linear arbitration, Demeester et al. [8] presented a Bayesian approach in combining inputs that eliminates the explicit definition of a particular arbitration function, as commonly employed by previous works. Combined with a pre-trained user model, inference over the viable trajectories in a known configuration space modifies the control of the user using a Markov model [9]. For applications such as remote access mobile robot operation in unknown environments, lack of required prior information on explicit goal definition and user models renders these methods of assistance inadequate.

The outline of the paper is as follows. A novel generalization of action specification for high dimensional input spaces is introduced in Sect. II-A, delineated using motion primitive methods for ground and aerial systems. An operator model to be constructed online without explicit prior knowledge of the user is specified in Sect. II-B. A constructive adaptation scheme based on action selection is presented in Sect. III. Finally, the results of a micro air vehicle moving along a racetrack and in a helix motion controlled by a human operator with and without the adaptation framework are presented in Sect. V.

II. Operator Model

A. Action representation

Traditional control methodologies of mobile robot teleoperation compute the desired motion reference with respect to a velocity reference provided by an external input device (e.g. joystick, steering wheel). An alternate strategy is to represent the action space as a set of pre-computed *motion primitives*, called a *motion primitive library* (MPL) [10]. Instead of employing the external inputs as velocity references, the external input becomes a selector function that corresponds to a specific motion primitive at each instance in time. Motion primitives can be computed in advance and designed to ensure dynamic feasibility and vehicle stability assuming nominal operating conditions.

We define an action as $\mathbf{a} = \{a_1, \ldots, a_q\}$ for q input dimensions. A corresponding motion primitive generated using an action is denoted by $\gamma(\mathbf{a})$. An MPL, $\Gamma = \{\gamma_i(\mathbf{a}_i)\}$, $i = 1, \ldots, N$, is generated via an action set of size N, $\{\mathbf{a}_i\}, i = 1, \ldots, N$. We further define the set of MPL to be a *motion primitive library collection*, denoted by $\{\Gamma_j\}, j = 1, \ldots, M$.

A parameterized action formulation for ground and air vehicles is chosen based on *forward-arc motion primitives* that propagate the dynamics of a unicycle model with a constant linear, angular, and vertical velocity for a specified amount of time, T [11]. The motion primitives are given by the solutions to the unicycle model:

$$\mathbf{x}_{t+T} = \mathbf{x}_t + \begin{bmatrix} \frac{v_{xt}}{\omega_t}(\sin(\omega_t T + \theta_t) - \sin(\theta_t)) \\ \frac{v_{xt}}{\omega_t}(\cos(\theta_t) - \cos(\omega_t T + \theta_t)) \\ v_{zt} T \\ \omega_t T \end{bmatrix}, \quad (1)$$

where $\mathbf{x}_t = [x_t, y_t, z_t, \theta_t]^{\mathrm{T}}$ represents the pose of the vehicle at time t, and v_{xt}, v_{zt}, and ω_t are the linear and angular velocities of the vehicle at time t in the body frame, respectively. Hence, the action space of the vehicle is given by uniformly dense sets: $\mathcal{V}_x = \{v_{xi}\}, i = 1, \ldots, N_{v_x}$, $\Omega = \{\omega_j\}, j = 1, \ldots, N_\omega$, $\mathcal{V}_z = \{v_{zk}\}, k = 1, \ldots, N_{v_z}$. A forward-arc motion primitive at each time t is represented by $\gamma_t = \{\mathbf{a}_t, T\}$, with $\mathbf{a}_t = \{v_{xt}, v_{zt}, \omega_t\}$. For ground vehicles, the heading of the vehicle is fixed to the yaw of the vehicle. Although aerial platforms such as quadrotors can independently control heading from yaw, we maintain the use of a unicycle model by ensuring that the heading is equivalent to the yaw of the vehicle, as humans naturally optimize for curved trajectories in robot control [12].

To ensure continuity in the selection of motion primitives such that the trajectory to which the vehicle follows is smooth, we generate an MPL collection based on a set of finely discretized initial states in the higher derivatives. The MPL in the collection to which the initial condition is sufficiently close is chosen to be the motion library at each time t. An example of the forward-arc MPL is shown in Figs. 3a-3b for ground and air vehicles, respectively. Figure 3c depicts the MPL selected at each time t with the initial condition matching that of the current robot state. The selected motion primitive at each time t forms a smooth trajectory.

B. Operator behavior representation

We model the underlying intent of the operator as an optimal controller [13]. For the operating scenarios studied in this paper, it is assumed that the operator is a rational, single

intent agent with smooth transitions in movements. Specifically, the operator inherently optimizes a reward function, but the choices of actions at each time step poorly reflect this function. Since the operator is assumed to achieve single intent tasks in this scenario, the reward function is assumed to not depend on time. At each input time t, the operator issues action $\mathbf{a}$ which is in some neighborhood of the optimal action $\mathbf{a}^\star$:

$$\mathbf{a}^\star = \underset{\mathbf{a}}{\operatorname{argmax}}\, R_t(\gamma(\mathbf{a})) \approx \underset{\mathbf{a}}{\operatorname{argmax}} \sum_i^P \alpha_t^i \phi_t^i(\gamma(\mathbf{a})), \quad (2)$$

where ϕ^i's are basis functions that describe intrinsic natural human or robot behavior that operators may seek to optimize, and α^i's are their corresponding weights for P basis functions. We assume the reward function can be described with linear basis terms, however other representations can also be used. We wish to infer the underlying reward function $\hat{R}_t = \sum_i^P \hat{\alpha}^i \phi_t^i$ from a series of noisy user inputs $\{\mathbf{a}_{t-m}, \mathbf{a}_{t-m+1}, \ldots, \mathbf{a}_{t-1}\}$.

Using this model, the inference over the user's behavior becomes the solution to the problem:

$$\begin{aligned} \hat{\gamma}_{t+1} &= \underset{\gamma_{t+1}(\mathbf{a}_{t+1})}{\operatorname{argmax}}\, R_t(\gamma_{t-m:t}(\mathbf{a}_{t-m:t}), \gamma_{t+1}(\mathbf{a}_{t+1})) \\ &= \underset{\gamma_{t+1}}{\operatorname{argmax}} \sum_i^P \alpha^i \phi_t^i(\gamma_{t-m:t}, \gamma_{t+1}) \end{aligned} \quad (3)$$

where $\gamma_{t-m:t} = \{\gamma_{t-m}, \gamma_{t-m+1}, \ldots, \gamma_t\}$ represents a trajectory from the past m motion primitives, and $\gamma_{t+1} \in \Gamma_{t+1}$. As user inputs are directly mapped to motion primitives, actions $\mathbf{a}_t$ and motion primitives γ_t are equivalent for some fixed duration T. This formulation removes the dependence of trajectories on the continuous input space, thus allowing inference to be made over motion primitives.

III. Adaptive Feedback Framework

The behavior recognition and prediction update for a user is computed based on a distribution over the set of all motion primitives, given the traveled trajectory and an estimate of the reward function. For generality, it is assumed that the prediction window can be selected to accommodate temporal basis functions if needed. The probability of a motion primitive being selected given the past m motion primitives is given by:

$$\begin{aligned} p(\gamma_{t+1}|\gamma_{t-m:t}, \hat{R}_t) &= \frac{p(\hat{R}_t|\gamma_{t-m:t}, \gamma_{t+1}) p(\gamma_{t+1}|\gamma_{t-m:t})}{p(\hat{R}_t|\gamma_{t-m:t})} \\ &= \eta\, p(\hat{R}_t|\gamma_{t-m:t}, \gamma_{t+1}) p(\gamma_{t+1}|\gamma_{t-m:t}) \end{aligned} \quad (4)$$

where $p(\hat{R}|\gamma_{t-m:t}, \gamma_{t+1})$ is a distribution over the estimated reward function of the user, as introduced in Sect. II-B, and η is a normalization weight.

A. Online Operator Intent Estimation

To infer the true underlying reward function from a past window of m motion primitives, we construct a belief distribution of the reward, $p(\hat{R}|\gamma_{t-m:t}, \gamma_{t+1})$ given the set of motion primitives, via an online function approximation to estimate the reward function $\hat{R}_t(\gamma_{t-m:t}, \gamma_{t+1}) = \sum_i^P \hat{\alpha}^i \phi_t^i(\gamma_{t-m:t}, \gamma_{t+1})$. This function is estimated using Locally Weighted Projection Regression (LWPR), a computationally efficient online method for local approximations of high dimensional nonlinear functions [14]. The incremental algorithm performs global function approximation by taking a weighted sum of the local regressors that influence the region. Note that this formulation of operator intent is task-independent unless the basis functions incorporate environment or task information. The choice of basis functions is addressed further in Sect. V-B.

The regression over the reward bases is defined with respect to a linear global reward function that is estimated using LWPR:

$$\begin{aligned} \hat{R}_t &= \sum_i^P \hat{\alpha}^i \phi_t^i = \frac{1}{\sum_j^Q d_j} \sum_j^Q d_j(\hat{y}_j) \\ &= \hat{\boldsymbol{\alpha}}^\top \Phi_t = \frac{1}{\sum_j^Q d_j} \sum_j^Q d_j(\boldsymbol{\beta}_j^\top \Phi_j) \end{aligned} \quad (5)$$

where $\hat{y}_j = \boldsymbol{\beta}_j^\top \Phi_j$ are the individual receptive fields used in LWPR, and d_j is the measure of locality of the j^{th} receptive field, out of a total of Q receptive fields.

B. Adaptive prior

The online estimation of the user behavior provides insight into the prediction of the user behavior based on hindsight, which can be constructively utilized to aid the choice of selection in the control of the robot. Using the update rule (4), the framework infers an empirical distribution over the uniform, dense set of all motion primitives. The model prior is iteratively adapted following (4). At each time step, the prior reflects the distribution of the likelihood of the user selecting the next motion primitive that maximizes their intent function based on the estimate of the reward function at the previous time step.

C. Constructive subsampling of motion primitives

We adaptively modify the subset of available motion primitives from an underlying, dense uniform discretization, such that the density of the subsampling reflects the reward function distribution, $p(\hat{R}|\gamma_{t-m:t}, \gamma_{t+1})$. Adaptation of the allowable set of motion primitives provides fine-grained control of the action to be taken by the robot and expels inputs misaligned with the user's underlying intent. By construction, selected motions from the subsampled set closely advance the user's underlying intention and circumvent misaligned motions to the user's interest. The key assumption here is that the human operator follows a *satisficing* property [15], i.e. *the user tends to converge and operate near a small set of actions that are within the region of interest*, unless the region of interest changes. Sampling from a dense, underlying set of motion primitives with respect to a belief distribution that reflects the operator's predicted intent allows the user with full control by providing finer precision control near the region of interest while eliminating noisy inputs.

Let the weight of the n^{th} motion primitive be $w_n = p(\hat{R}|\gamma_{t-m:t}, \gamma_{t+1}^n)$. Given a motion primitive library Γ of

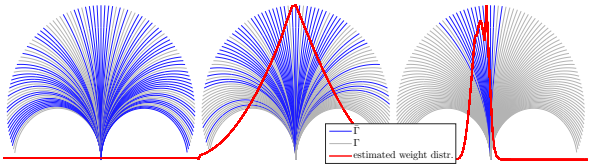


Fig. 4: Motion primitives and distribution over the motion primitives at selected times along a racetrack at $t = 0$, $t = 5$, and $t = 150$ respectively. The prediction becomes more peaked near the mean of the predicted motion primitive.

size N, we sample K motion primitives using the weights $\{w_n\}, n = 1, \ldots, N$ with replacement such that we obtain a subsampled set

$$\bar{\Gamma} = \{\gamma^k\} \subseteq \Gamma, k = 1, \ldots, K$$

The choice of motion primitive is limited to this set. Using selector function (6), the motion primitive with the closest parameterization of the actual joystick value $\mathbf{a}_{\text{joy}}$ is selected.

$$\gamma_{\text{selected}} = \gamma\{\underset{\mathbf{a}}{\text{argmin}}\, \mathbf{a} - \mathbf{a}_{\text{joy}}\} \in \bar{\Gamma} \tag{6}$$

A visualization of this algorithm is provided in Fig. 4. The algorithm for constructing the subsampled set is provided in Algorithm 1.

Algorithm 1 Algorithm for updating construction of the subsampled set online

$\Gamma_{t+1} \leftarrow \{\gamma_i\};\ i = 1, \ldots, N;\ \Gamma_{t+1} \sim U(a, b)$
$\bar{\Gamma}_{t+1} \leftarrow \emptyset$
for $n = 1 : N$ **do**
    Calculate weights for each motion primitive $w_n = p(\hat{R}|\gamma_{t-m:t}, \gamma^n_{t+1})$
end for
for $k = 1 : K$ **do**
    Draw $\gamma_k \in \Gamma_t$ with probability w_k with replacement
    $\bar{\Gamma}_{t+1} \leftarrow \bar{\Gamma}_{t+1} + \gamma_n$
end for
return $\bar{\Gamma}_{t+1}$

IV. User Efficiency Evaluation

Joystick Steering Entropy, introduced by Nakayama et al. [16] is a direct, online method to efficiently evaluate the workload performance of an operator given continuous inputs, without asking the operator to deviate their focus. Steering entropy quantifies the smoothness of the operator's actions directly from past inputs such that only hindsight information is used. We utilize steering entropy to measure the efficiency of the operator for each trial. Given a continuous input $u \in \mathbb{R}$, the input error is derived from the difference between a second order Taylor approximation of the input at time t with the actual input provided by the user:

$$e_t = u_t - \hat{u}_t, \tag{7}$$

where

$$\begin{aligned}\hat{u}_t &= u_{t-1} + (u_{t-1} - u_{t-2}) \\ &\quad + \frac{1}{2}((u_{t-1} - u_{t-2}) - (u_{t-2} - u_{t-3})).\end{aligned} \tag{8}$$

A frequency distribution of the error is then constructed and divided into 9 bins [16]. The total steering entropy, Hp, for each run is given by

$$Hp = \sum_i -P_i \log_9 P_i. \tag{9}$$

A slight modification to the algorithm is made by padding the proportion of each bin by ϵ in order to avoid asymptotes:

$$P_i = \frac{n_i}{\sum_i n_i} + \epsilon,\ i = 1, \ldots, 9. \tag{10}$$

As efficiency increases, the steering entropy decreases accordingly. The proposed teleoperation methodology is evaluated based on steering entropy through simulation and experimental trials with and without adaptation.

TABLE I: LWPR parameters for estimating the reward function

D_{init}	7	w_{gen}	250
α_{init}	250	w_{cutoff}	0.5
meta	false	penalty	1.0

TABLE II: Motion primitive library parameters

$v_{x\,\text{max}}$	0.5 m/s	$v_{x\,\text{min}}$	-0.5 m/s	N_{v_x}	101
$v_{z\,\text{max}}$	0.5 m/s	$v_{z\,\text{min}}$	-0.5 m/s	N_{v_z}	31
ω_{max}	3 rad/s	ω_{min}	-3 rad/s	N_ω	101

V. Experiments

A. System design and implementation details

The proposed adaptive framework is validated with a quadrotor operating in a motion capture equipped flight arena. A Logitech F310 gamepad is used to control vehicle lateral motion, thrust, and angular velocity. Simulation results are obtained via a high-fidelity quadrotor simulator that captures the nonlinear dynamics of the vehicle. For the purposes of this study, it is assumed that the information provided by the visual feedback to the user is sufficient in order for the operator to control the robot.

B. Reward function formulation

For quadrotor vehicles, several cost bases that naturally optimize an operator's motion are defined and inverted for the purpose of constructing a reward function. In general, the defined bases are purely functions of past trajectories, or *hindsight bases*. However, if information regarding the environment or task is known a priori and is well defined, for example in the case of certain obstacle fields and goal locations, these can be additionally incorporated as bases and further aid in refining the optimization. Here, we present three hindsight cost bases (smoothness, orthogonality, and time), and an additional penalty in distance given a desired trajectory to follow (distance error).

Smoothness is the magnitude of change in the input:

$$\phi_{\text{smoothness}}(\gamma_{t+1}, \gamma_{t-m:t}) = \sum_{j=t-m+1}^{t+1} \|\mathbf{a}_j - \mathbf{a}_{j-1}\|_1 \tag{11}$$

Orthogonality penalizes drastic deviations in heading from the previous trajectory and defined as a ratio of three points:

$$\phi_{\text{orthogonality}}(\gamma_{t+1}, \gamma_{t-m:t}) = \frac{\|p_t - p_{t-m}\| - \|p_{t+1} - p_t\|}{\|p_{t+1} - p_{t-m}\|} - 1 \tag{12}$$

where $p_k = \gamma_k(\mathbf{a}, t = T)$.

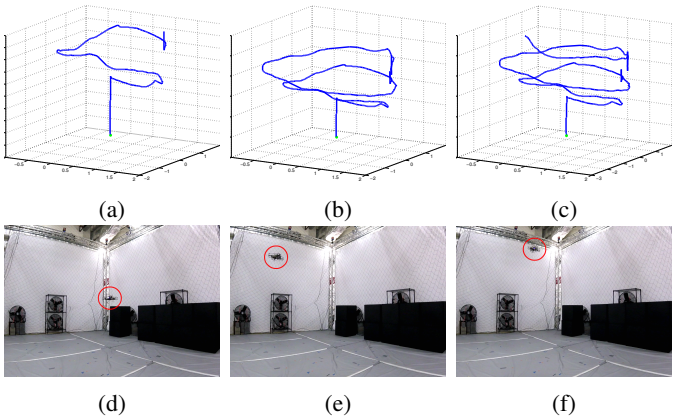


Fig. 5: Quadrotor in action (a)-(c) in the simulation environment and (d)-(f) in the flight arena for the helix flight.

Flight time cost is defined as the inverse of the linear body velocity to permit inference over the desired speed of flight:

$$\phi_{\text{time}} = \frac{1}{v_x} \tag{13}$$

Distance Error given a desired trajectory is the error between the motion primitive and the trajectory and calculated by approximating the area or volume for 2D and 3D trajectories respectively [17]. Given two paths defined by points $\{p_1, ..., p_n\}$ and $\{q_1, ..., q_n\}$ in $\mathbb{R}^2$ or $\mathbb{R}^3$, the distance error is computed as:

$$\phi_{\text{distance}} = \sum_{i=2}^{n} \frac{1}{2} \left(\|p_i - p_{i-1}\|_2 + \|q_i - q_{i-1}\|_2 \right) \|p_i - q_i\|_2 \tag{14}$$

C. Parameters and definitions

The choice of LWPR parameters used to approximate the cost function is provided in Table I. The LWPR parameters not listed here use default values provided by the LWPR toolbox [14].

The motion primitive library was generated using the discretization and range values provided in Table II. The velocity bounds are chosen in order to limit the aggressiveness of the motions for this particular study. The motion primitive libraries are generated with a fixed time $T = 1$ s. The number of underlying motion primitives is randomly chosen to be a large number in order to finely discretize the inputs. The experimental results in Sect. V-E suggest that the the values used in this study suffice to validate the proposed framework.

D. Experimental scenarios

We verify the proposed framework with two scenarios: a generic racetrack with size 30 m×10 m at a height of 1 m, and a continuous, single heading helix motion. Both scenarios are tested in simulation, and the helix scenario is further tested with a quadrotor vehicle to supplement the simulation results subject to the full complexity of the quadrotor (Fig. 5). For simplicity, we only perform inference over the heading of the robot, such that the motion primitives form a library with varying yaw velocity ω only.

Operators are asked to follow a pre-defined racetrack to the best of their ability. As the trajectory that the user is trying

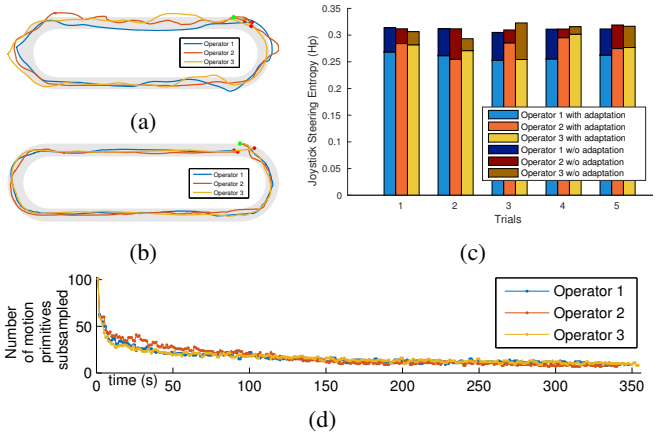


Fig. 6: Trajectory taken by the quadrotor around the racetrack (a) without the adaptation framework and (b) with adaptation, with start and end marked in green and red, respectively. (c) Joystick steering entropy for three operators over five trials. Lower entropy is observed when using adaptation. (d) Number of motion primitives sampled at each time instance for one lap around the racetrack consistently decreased across all trials and converges to approximately 15 motion primitives, downsampled from 101.

to follow is known, this information can be incorporated into the distance error basis as discussed in Sect. V-B. For the helix motion, the operators are asked to perform a freehand helix motion without visual guidance. As no such trajectory is specified, predictions are based purely on the hindsight bases.

Each scenario is tested with 10 trials (5 with adaptation and 5 without). For consistency, we perform the same experiment with local researchers, not including the authors. All operators do not have any prior experience in teleoperating a quadrotor using the joystick setup and the proposed methodology.

E. Discussion of Results

For consistency, teleoperation of a quadrotor in the racetrack scenario is repeated with multiple operators. We first observe smoother flight performance with the adaptation framework for all operators as shown in Figs. 6a and 6b. The robustness of the method with respect to the choice of parameters provided in Table II is validated with a consistent decrease in the number of subsampled motion primitives across multiple users, as shown in Fig. 6d. This indicates that the number of motion primitives for each action (N_{v_x}, N_{v_z} and N_ω) only needs to be sufficiently dense such that the operator finds the discretization fine enough to represent a continuous input. User efficiency with and without adaptation is compared using joystick steering entropy on the control

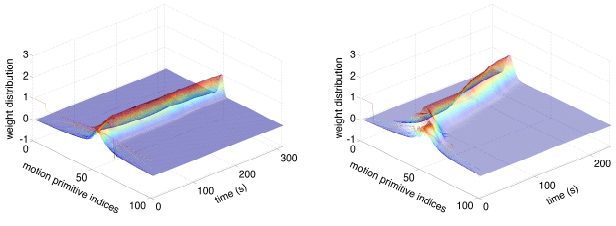
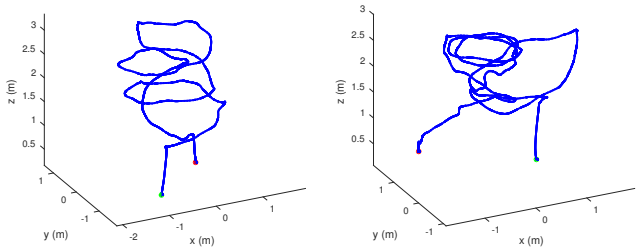


Fig. 7: Belief distribution over the set of motion primitives over time for one trial of (a) the racetrack scenario and (b) helix motion in simulation, with mean converging to the directional intent.



(a) (b)

Fig. 8: Experimental position data obtained for the helix scenario for a single operator (a) with adaptation and (b) without adaptation.

of yaw velocity. For all of the users, we observe that the steering entropy with adaptation to be signficantly less than without adaptation, as shown in Fig. 6c. An average decrease of 13% in entropy is observed across all trials.

The results of the convergence of the weight distribution with respect to the motion primitives are shown in Fig. 7 for a trial of the racetrack as well as the helix scenario. Here, we observe that the belief distribution heavily veers towards the motion primitive with a high yaw velocity in the helix case (as expected). Similarly, we observe a convergence to a more centered distribution in the case of the racetrack. These outcomes suggest that the distribution over the motion primitives converge in finite time. The convergence of the distribution validates the correct prediction of the user's intent independent of prior information, as in the case of the helix, the predictions of the weights are determined purely using hindsight bases. Hence, addition of environmental information is constructive to the prediction of the user's reward function, but not required. We further assess the performance of the adaptive framework with experimental results in addition to the high-fidelity simulation analysis, and yield similar improvement in performance as shown in Fig. 8.

VI. Conclusion and Future Work

We present a task independent framework to improve efficiency of teleoperation via online estimation of the user intent. In contrast to previous work, our approach in user intent modeling does not require prior knowledge of the environment, task, or user characteristics. However, prior information regarding task can be leveraged if available. In addition, the proposed framework eliminates the need to train user models offline, thus allowing the framework to operate only on hindsight information obtained online. We validate that the method yields increased efficiency by measuring the steering entropy and increased performance in achieving smoother flights and convergence to a sufficient motion primitive set within finite time via repeated trials.

Future work will include fast adaptation of both single-intent and multi-intent scenarios, which is contingent on analysis of prediction error in user intent. To this end, we seek to further develop robust feedback mechanisms in online verification of user intent prediction. In addition, the presented framework is extensible for multi-input, multi-agent systems, notably systems with well-defined motion primitives, such as exoskeleton control and formation control of multiple vehicles.

References

[1] C. C. MacAdam, "Understanding and Modeling the Human Driver," *Vehicle System Dynamics*, vol. 40:1-3, no. January 2014, pp. 101–134, 2003.

[2] A. D. Dragan and S. S. Srinivasa, "A policy-blending formalism for shared control," *Int. J. Robot. Res.*, vol. 32, no. 7, pp. 790–805, 2013.

[3] K. Muelling, A. Venkatraman, J.-S. Valois, J. E. Downey, J. Weiss, S. Javdani, M. Hebert, A. B. Schwartz, J. L. Collinger, and J. A. Bagnell, "Autonomy Infused Teleoperation with Application to BCI Manipulation," in *Proc. of Robot.: Sci. and Syst.*, Ann Arbor, MI, Jan. 2015.

[4] S. Javdani, S. S. Srinivasa, and J. A. Bagnell, "Shared Autonomy via Hindsight Optimization," in *Proc. of Robot.: Sci. and Syst.*, Rome, Italy, July 2015.

[5] M. Gao, J. Oberl, T. Schamm, and J. Z. Marius, "Contextual Task-Aware Shared Autonomy for Assistive Mobile Robot Teleoperation," in *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, Chicago, IL, Sep. 2014, pp. 3311–3318.

[6] A. Goil, M. Derry, and B. D. Argall, "Using Machine Learning to Blend Human and Robot Controls for Assisted Wheelchair Navigation," in *IEEE Int. Conf. Rehabil. Robot*, Seattle, WA, Jun. 2013.

[7] S. Jain and B. Argall, "An Approach for Online User Customization of Shared Autonomy for Intelligent Assistive Devices," in *Proc. of the Workshop on Human-Robot Interfaces for Enhanced Physical Interactions IEEE Int. Conf. on Robot. and Autom.*, Stockholm, Sweden, May 2016.

[8] E. Demeester, A. Hüntemann, E. V. Poorten, and J. D. Schutter, "ML, MAP and Greedy POMDP Shared Control: Comparison of Wheelchair Navigation Assistance for Switch Interfaces," in *Proc. of the Int. Sym. on Robotics*, Taipei, Taiwan, Aug. 2012, pp. 1106–1111.

[9] E. Demeester, A. Hüntemann, D. Vanhooydonck, G. Vanacker, H. V. Brussel, and M. Nuttin, "User-Adapted Plan Recognition and User-Adapted Shared Control: A Bayesian Approach to Semi-Autonomous Wheelchair Driving," in *Autonomous Robots*, 2008, pp. 193–211.

[10] A. Kelly and B. Nagy, "Reactive Nonholonomic Trajectory Generation via Parametric Optimal Control," *Int. J. Robot. Res.*, vol. 22, 2003.

[11] M. Pivtoraiko, I. A. Nesnas, and A. Kelly, "Autonomous robot navigation using advanced motion primitives," in *Proc. of the IEEE Aerospace Conf.*, Big Sky, USA, 2009, pp. 1–7.

[12] N. Delson and H. West, "Robot Programming by Human Demonstration: The Use of Human Inconsistency in Improving 3D Robot Trajectories," in *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, Sep. 1994, pp. 1248–1255.

[13] J. W. Crandall and M. A. Goodrich, "Characterizing Efficiency of Human Robot Interaction: A Case Study of Shared-Control Teleoperation," in *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, Lausanne, Switzerland, Oct. 2002, pp. 1–6.

[14] S. Vijayakumar, A. D'Souza, and S. Schaal, "Incremental Online Learning in High Dimensions," *Neural Computation*, vol. 17, no. 12, pp. 2602–2634, 2005.

[15] J. March and H. Simon, *Organizations*. New York: Wiley, 1958.

[16] O. Nakayama, T. Futami, and T. Nakamura, "SAE Technical Development of a Steering Entropy Method for Evaluating Driver Workload," Tech. Rep. 724, 1999.

[17] A. G. Bachrach, "Trajectory Bundle Estimation For Perception-Driven Planning," Ph.D. dissertation, Dept. Elect. Eng. and Comp. Sci., Massachusetts Institute of Technology, Cambridge, MA, 2013.