Goal-Predictive Robotic Teleoperation from Noisy Sensors

Christopher Schultz[1], Sanket Gaurav[1], Mathew Monfort[2], Lingfei Zhang[1], and Brian D. Ziebart[1]

***Abstract*— Robotic teleoperation from a human operator's pose demonstrations provides an intuitive and effective means of control that has been made feasible by improvements in sensor technologies in recent years. However, the imprecision of low-cost depth cameras and the difficulty of calibrating a frame of reference for the operator introduce inefficiencies in this process when performing tasks that require interactions with objects in the robot's workspace. We develop a goal-predictive teleoperation system that aids in "de-noising" the controls of the operator to be more goal-directed. Our approach uses inverse optimal control to predict the intended object of interaction from the current motion trajectory in real time and then adapts the degree of autonomy between the operator's demonstrations and autonomous completion of the predicted task. We evaluate our approach using the Microsoft Kinect depth camera as our input sensor to control a Rethink Robotics Baxter robot.**

I. INTRODUCTION

Effective robots are critically needed in a number of settings where human capabilities are limited. These include operation in settings that are unsafe for human presence [1], when lifting heavy objects that exceed the limits of human physical strength [2], and/or when precise movements are needed at fine scales that are beyond the precision of unaided human motor control [3]. Despite significant advances in artificial intelligence [4], [5], human teleoperators are still more adept at many of the robotic motion planning and manipulation tasks [6] encountered in these settings, which require versatility and high-level problem solving. Furthermore, programming by demonstration through teleoperation is an attractive modality for enabling end-users without computer programming expertise to create desired autonomous behavior [7]. Methods that improve the efficiency of teleoperation by sharing autonomy between human operator and autonomous controller [8], [9]—leveraging the advantages of each—are needed to further improve the efficiency of completing these tasks.

A natural input mechanism for humanoid robot teleoperation is for the end-user to simply demonstrate the desired robot movements to a passively observing sensor and have the movements imitated by the robot [10], [11], [12]. There are two key challenges to this form of unilateral teleoperation. First, the translation from human to robotic poses is often complicated due to differences in physical embodiment. Joint angle limitations of human operators and humanoid robots can differ significantly, for example. Second, apart from motion capture systems [13], [14], [15], which are very precise but require a calibrated environment and are relatively expensive, human pose estimates are often susceptible to sensor noise. Depth cameras, like the Microsoft Kinect, are relatively inexpensive, but can produce skeleton tracking errors that may make them unsuitable for fine-grained teleoperation tasks on their own. Despite being an inexpensive and intuitive setup, depth camera teleoperation has yet to be deployed in many industrial teleoperation applications [16], such as handling hazardous waste [17], due to these issues.

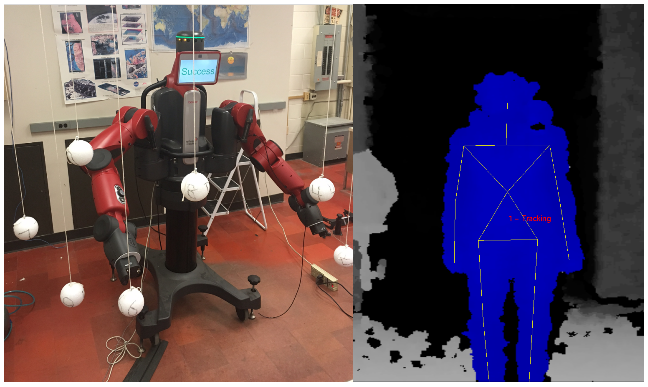

Fig. 1. The Baxter robot testing setup with a number of interaction objects suspended in the robot's workspace (left) and Kinect data of human teleoperator with tracked skeleton produced from the OpenNI system (right).

To address the issues with depth camera teleoperation, in this paper, we investigate **goal-predictive pose-based robotic teleoperation from noisy sensor data**: the task of using the knowledge of possible task completion goals for de-noising depth camera data to improve the efficiency of robotic teleoperation [8]. Our approach is composed of three main steps. First, we estimate a correspondence between the human operator's tracked skeleton and robotic joint positions. This is used to translate from human operator pose to robot pose. Second, we learn a model for goal prediction using inverse optimal control for linear-quadratic systems [18]. This provides a posterior probability estimate for each possible target or goal in the robot's work space given the partial trajectory. Third, we investigate control assistance policies based on the confidence of the goal prediction component. These increase the autonomy of the controller when predicted goal certainty is high. We evaluate the approach on teleoperated pointing tasks using the Baxter robot from Rethink Robotics [19] as our platform (Figure 1,

*The research in this paper was supported in part by NSF NRI Award No. 1227495.

[1]Christopher Schultz, Sanket Gaurav, Lingfei Zhang, and Brian D. Ziebart are with the Department of Computer Science, University of Illinois at Chicago, 851 S. Morgan St. (M/C 152) Chicago, IL 60607 cschul25@uic.edu, sgaura2@uic.edu, lzhang44@uic.edu, and bziebart@uic.edu

[2]Mathew Monfort is with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 32 Vassar St, Cambridge, MA 02139 mmonfort@mit.edu

left) and a Microsoft Kinect [20] as our input sensor (Figure 1, right).

II. BACKGROUND AND RELATED WORK

A. Robotic Teleoperation

Robotic teleoperation refers to robotic control with human participation [17]. There are generally two types of teleoperation systems: unilateral and bilateral control. In unilateral control, humans provide control inputs to a robot through a master control interface, such as a joystick, which then provides output to a robot slave controller. The slave controller performs the robotic system manipulations based off of the output from the master control interface. Bilateral control is unilateral control with an additional feedback loop in which a robotic slave controller provides feedback to the master control interface. This additional feedback allows the master control interface to provide feedback to human teleoperators. An example of bilateral control is a joystick that provides mechanical resistance to a human operator when a robot makes contact with objects or obstacles in its environment [17].

A number of recent teleoperation systems enable the operator to demonstrate desired control through pose data collected from depth cameras [21]. These types of teleoperation interfaces are unilateral and have been primarily developed to faithfully reproduce the operator's behaviors [22], [12]. Some use gesture recognition as a command signaling mechanism to improve teleoperation [23]. Our focus differs in that we infer intentions rather than recognizing pre-programmed directives from depth camera data.

Unfortunately, teleoperation using depth cameras can be difficult for human operators due to noisy sensor or tracking output, poor translation from human input to robotic output, and latency between the human input and robot output [17], [24]. Nevertheless, these interfaces are appealing from a cost and flexibility standpoint. A significant research direction has investigated assistive teleoperation [8], in which control is achieved through shared autonomy. One example is teleoperation for free-form tasks using mouse cursor input, in which the task is inferred and used to optimize low-level robotic motions [25]. A recent approach in the brain-computer interface teleoperation domain [9] has attempted to address these type of teleoperation issues in the BCI domain by infusing an correcting assist action, $\mathbf{A_{assist}}$, to a teleoperation control action, $\mathbf{A_{tel}}$. The exact $\mathbf{A_{assist}}$ value and the degree in which it is added to $\mathbf{A_{tel}}$ is based off of the predicted intention of the teleoperation action. This assisting action addition was shown in [9] to improve task completion metrics in robot teleoperation applications.

B. Inverse Optimal Control

Inverse optimal control (also known as inverse reinforcement learning) [26], [27], [28] seeks a reward or cost function that rationalizes demonstrated control sequences [28]. Though ill-posed in its simplest formulation [27], extensions that seek to provide predictive guarantees create a well-defined machine learning task [29]. Maximum entropy inverse optimal control [30], for example, seeks to provide robust predictions of the control policy under the logarithmic loss, while learning parameters that define a cost function for inverse reinforcement learning purposes.

In this work, we leverage extensions of maximum entropy inverse optimal control to linear-quadratic control (LQR) settings [18], [31], [32]. In these settings, it is assumed that the dynamics of a system being investigated can be represented by a continuous linear state-space representation,

$$\mathbf{s}_{t+1} = \mathbf{A}\mathbf{s}_t + \mathbf{B}\mathbf{a}_t + \epsilon_t, \tag{1}$$

where $\mathbf{s_t}$ denotes the state of the system at time t, $\mathbf{a_t}$ denotes the action at time t, ϵ_t denotes some zero mean Gaussian noise, and $\mathbf{A}$ and $\mathbf{B}$ define the system dynamics.

A state-action cost function,

$$cost(\mathbf{s}_t, \mathbf{a}_t) = \begin{bmatrix}\mathbf{a}_t \\ \mathbf{s}_t\end{bmatrix}^T \mathbf{M} \begin{bmatrix}\mathbf{a}_t \\ \mathbf{s}_t\end{bmatrix}, t < T, \tag{2}$$

and a final state cost penalizing the final state, $\mathbf{s_T}$, from deviating from the desired target, $\mathbf{s_G}$,

$$cost(\mathbf{s_T}) = (\mathbf{s_T} - \mathbf{s_G})^T \mathbf{M_f} (\mathbf{s_T} - \mathbf{s_G}), \tag{3}$$

are learned by updating the $\mathbf{M}$ and $\mathbf{M_f}$ coefficient matrices through demonstrated behaviors using the principle of maximum causal entropy [31]. Specifically, this is done by solving the constrained optimization problem maximizing the causal entropy [18],

$$H(\mathbf{a}||\mathbf{s}) \triangleq \mathbb{E}_{\hat{\pi}}\left[-\sum_{t=1}^{T} \log \hat{\pi}(\mathbf{a}||\mathbf{s})\right],$$

such that the predictive policy distribution, $\hat{\pi}(\mathbf{a}||\mathbf{s}) = \pi(\mathbf{a}_1|\mathbf{s}_1)\pi(\mathbf{a}_2|\mathbf{s}_2)\cdots\pi(\mathbf{a}_T|\mathbf{s}_T)$, matches the demonstrated quadratic state properties, $\tilde{\pi}$, in feature expectation through the following optimization constraints,

$$\mathbb{E}_{\hat{\pi}}\left[\sum_{t=1}^{T-1}\begin{bmatrix}\mathbf{a}_t \\ \mathbf{s}_t\end{bmatrix}\begin{bmatrix}\mathbf{a}_t \\ \mathbf{s}_t\end{bmatrix}^T\right] = \mathbb{E}_{\tilde{\pi}}\left[\sum_{t=1}^{T-1}\begin{bmatrix}\mathbf{a}_t \\ \mathbf{s}_t\end{bmatrix}\begin{bmatrix}\mathbf{a}_t \\ \mathbf{s}_t\end{bmatrix}^T\right], \text{ and}$$
$$\mathbb{E}_{\hat{\pi}}\big[(\mathbf{s}_T - \mathbf{s}_G)(\mathbf{s}_T - \mathbf{s}_G)^T\big] = \mathbb{E}_{\tilde{\pi}}\big[(\mathbf{s}_T - \mathbf{s}_G)(\mathbf{s}_T - \mathbf{s}_G)^T\big].$$

This optimization allows the state-conditioned probabilistic policy, $\hat{\pi}$, to be formed using the following recursively defined equations,

$$\hat{\pi}(\mathbf{a}_t|\mathbf{s}_t) = e^{Q(\mathbf{s}_t,\mathbf{a}_t)-V(\mathbf{s}_t)}, \tag{4}$$
$$Q(\mathbf{s}_t, \mathbf{a}_t) = \mathbb{E}_{\tau(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)}[V(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)] + cost(\mathbf{s}_t, \mathbf{a}_t), \tag{5}$$

$$V(\mathbf{s}_t) = \begin{cases}\underset{\mathbf{a_t}}{\text{softmax}} Q(\mathbf{s}_t, \mathbf{a}_t), & t < T \\ (\mathbf{s}_t - \mathbf{s}_G)^T \mathbf{M_f} (\mathbf{s}_t - \mathbf{s}_G), & t = T,\end{cases} \tag{6}$$

where the policy distribution is penalized for deviating from the desired goal location, $\mathbf{s}_G$, at time T and the softmax function is a smoothed interpolation of the maximum function, $\underset{\mathbf{x}}{\text{softmax}} f(x) = \log \int_x e^{f(x)} dx$.

After training is complete, Equations (4) through (6) allow us to estimate the probability of each possible target being the desired goal of an observed partial trajectory [18].

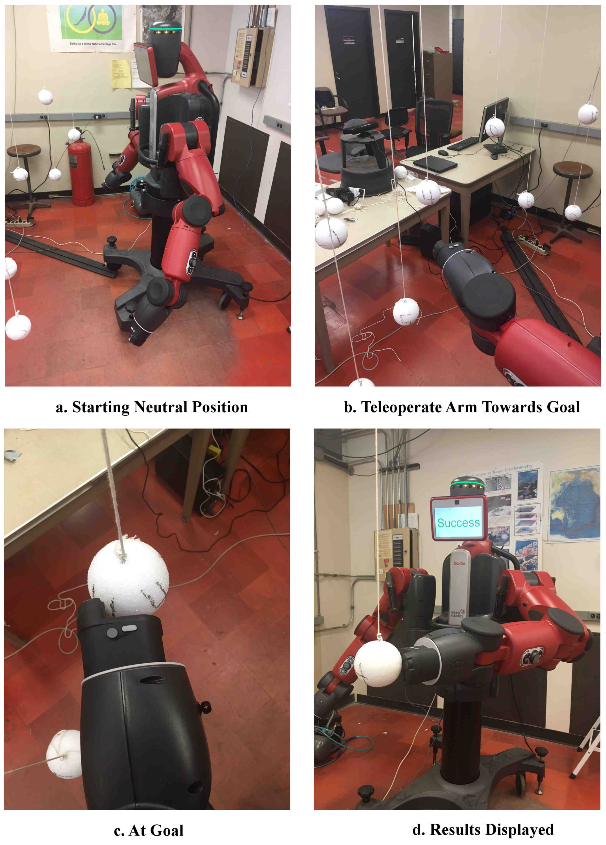

Fig. 2. The steps of a task in our testing sequence.

III. APPROACH

Our approach for real-time goal-predictive robotic teleoperation is based on the combination of three main components, which we describe in this section: (1) learning a correspondence between human operator pose and robot pose; (2) predicting the intended goal of the operator given a partial trajectory; and (3) control assistance when goal prediction confidence is high.

A. Learning Human-Robot Pose Correspondence

We use a Microsoft Kinect to obtain depth point cloud data of a human teleoperator and use the OpenNI framework[1] on the Kinect data to overlay a digital skeleton model on the human teleoperator's captured depth camera data. The OpenNI Skeleton model has 105 datapoints, 15 skeleton points each having x,y,z translation and x,y,z,w rotational data. These 105 datapoints are used as features to build a correspondence to a Rethink Robotics Baxter Research Robot's[2] two arm joint positions. The Baxter Robot arms are 7-degrees of freedom arms.

To build a correspondence between the arm joints of the Baxter arms to the OpenNI Skeleton, we collect data from the Baxter robot as one demonstrator moves the robot arms in "zero-gravity" mode. The arm movements start from a neutral arm position (Figure 2a) to a final position. While one robot arm is moved by a demonstrator, another demonstrator mimics the robot's arm motions with his/her arms and tries to stay synchronized with the robot arm movements. Both the robot arm joints and the corresponding OpenNI Skeleton data are recorded together. We vary the demonstrators to ensure a generalized correlation between human teleoperators and arm joints.

Given the amount of data we collect, we use a simple linear regression model with no regularization [33] to build the correspondence. We withhold 30% of the our collected data to act as a validation set for our linear correspondence model. The final fit correspondence model trained from ten demonstrators achieves an average variance between a predicted joint to the actual value in the validation set of 0.04 rad^2. We employ this correspondence model in our experiments.

B. Goal Prediction via Inverse Linear-Quadratic-Regulation

We define a goal as a location in x,y,z translational space that we may want the robot arm end-effector to reach. The end-effector is the endpoint of the robot arm, which can be calculated through the arm's geometry using forward kinematics [34]. The end-effector has both x,y,z translational and x,y,z,w rotational dimensions referenced from the associated robot's coordinate frame. We only consider translational dimensions for goal positions. End-effector space is used since the control task can be modeled as a Linear-Quadratic Regulation problem [18].

Our approach to goal prediction is to predict the intent of an end-effector trajectory through space in reference to a goal position. Following the approach outlined in [18], we assume the linear dynamics of (1), in which we define the state of the end-effector as:

$$\mathbf{s_t} = [x_t, y_t, z_t, \dot{x}_t, \dot{y}_t, \dot{z}_t, \ddot{x}_t, \ddot{y}_t, \ddot{z}_t, 1]^T, \tag{7}$$

and end-effector actions as:

$$\mathbf{a_t} = [\dot{x}_t, \dot{y}_t, \dot{z}_t]^T, \tag{8}$$

where $(\dot{x}_t, \dot{y}_t, \dot{z}_t)$ are velocities, $(\ddot{x}_t, \ddot{y}_t, \ddot{z}_t)$ are accelerations, and a constant of 1 is added to the state representation to incorporate linear features into the quadratic cost function in (2). Additionally, goal state i of the end-effector is represented using only the goal's translational position,

$$\mathbf{s_{G_i}} = [x_{G_i}, y_{G_i}, z_{G_i}, 0, 0, 0, 0, 0, 0, 0]^T. \tag{9}$$

The cost matrix coefficients $\mathbf{M}$ and $\mathbf{M_f}$ are learned from end-effector position data where a human demonstrator moves the robot arms in "zero-gravity" mode from a neutral start position to a goal position. We train these matrices by maximizing causal entropy [31] using gradient descent with an adaptive learning rate [18].

From these learned cost matrices, we infer the probabilities of different possible goal states that the operator may want the end-effector to reach given the observed partial trajectory of the end-effector in real time. We define these goal state probabilities as $P(\mathbf{s_{G_i}}|trajectory_{init \to t})$ and the probability

[1]https://github.com/ros-drivers/openni$_{l}aunch$
[2]http://sdk.rethinkrobotics.com/wiki/Main_Page

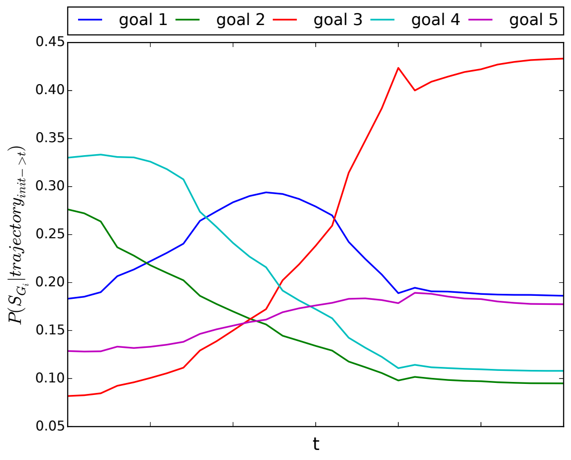

Fig. 3. The predicted goal probabilities of the trained inverse LQR model over time for a sequence of poses as the operator's left arm reaches towards goal 3.

of the most likely intended goal of the partial trajectory, I, as,

$$I = \max_i P(\mathbf{s_{G_i}}|\text{ trajectory}_{init\to t}). \quad (10)$$

The inverse LQR goal prediction method is a Bayesian inference method that benefits greatly from a prior distribution over the possible goals [18]. In this application, we use a distance prior similar to the one used in previous work [18],

$$P(\mathbf{s_{G_i}}|\mathbf{s_t}) \propto e^{-\beta dist(\mathbf{s_t},\mathbf{s_{G_i}})}, \quad (11)$$

where $dist(\mathbf{s_t},\mathbf{s_{G_i}})$ is a function that computes the Euclidean distance between the spatial coordinates of $\mathbf{s_t}$ and $\mathbf{s_{G_i}}$, and β is an adjustable coefficient that increases the importance of distance on the distribution. As $dist(\mathbf{s_t},\mathbf{s_{G_i}})$ decreases, $P(\mathbf{s_{G_i}}|\mathbf{s_t})$ increases effectively making closer targets more probable. Fig. 3 depicts the goal probability change over time when the operator attempts to reach goal 3.

C. Goal-Based Control Assistance

Using the goal predictions from the previous section, we adjust the level of autonomy between human and autonomous control. The specific approach we employ from [35], [9] is to consider an action taken by a robot at time t, $\mathbf{A_{r,t}}$, to be a linear combination of a teleoperation action, $\mathbf{A_{tel,t}}$ and an assisting action, $\mathbf{A_{assist,t}}$:

$$\mathbf{A_{r,t}} = \alpha\mathbf{A_{tel,t}} + (1-\alpha)\mathbf{A_{assist,t}} \quad (12)$$

where α is a mixing coefficient. When α is close to 1, $\mathbf{A_{r,t}}$ is mainly a function of the teleoperation translated action $\mathbf{A_{tel,t}}$. As alpha decreases towards 0, $\mathbf{A_{r,t}}$ becomes increasingly a function of an assist action $\mathbf{A_{assist,t}}$. By defining α as a function of predicted intent [18], a noisy teleoperation action can be corrected by mixing in an appropriate assisting action that reflects the true intention of the teleoperation action.

We use a joint angle position based arm controller for this work[3]. Because of this, we consider the actions in (12) to be in the arm joint angle position space. Specifically, $\mathbf{A_{tel,t}}$ is the joint values from the linear correspondence model we trained from a human teleoperator to robot arm joints. $\mathbf{A_{assist,t}}$ is calculated by finding the end-effector goal with the highest probability from our goal prediction model and applying inverse kinematics [34] to goal position with the current arm joint positions as seeds to inverse kinematic calculations. We then linearly mix $\mathbf{A_{tel,t}}$ and $\mathbf{A_{assist,t}}$ together per (12) to form $\mathbf{A_{r,t}}$ which is used by the joint position arm controller to manipulate the robot arms.

We consider three different ways to set the value of the mixing coefficient α. The first is a no assist approach where α is 1.0 for all I values. As α remains at 1.0 regardless of I, this method prevents any assist from being applied to a teleoperation control action. For this reason, we refer to this α setting method as the *no assist* method.

The second approach sets α using a sigmoid function of the highest probability goal, I, of the following form [9]:

$$\alpha = \frac{1}{1+e^{-a(1-I)+o}}, \quad (13)$$

where a and o are adjustable parameters. this method varies α as I varies so this method mixes in an assisting action with a teleoperation control action. We refer to this α setting method as the sigmoid assist.

The last approach is α as a step function of I of the following form [8]:

$$\alpha = \begin{cases} 1.0 \text{ if } I < I_{threshold} \\ c \text{ else,} \end{cases} \quad (14)$$

where $I_{threshold}$ and c are adjustable parameters. Like the sigmoid assist, this method mixes in an assisting action with a teleoperation control action based on $I_{threshold}$. We refer to this α setting method as the step assist. The three alpha variation are shown in Fig. 4

As discussed in [9], it is important to set an α_{min} where the value of α cannot drop below. If α decreases too much, $A_{assist,t}$ becomes dominant in (12) where if the guessed intent of a partial trajectory is wrong the human teleoperator will never be able to correct the arm because $A_{tel,t}$ will not be a large enough component in $A_{r,t}$. An α_{min} is set in the two α approaches above where α varies as a function of I.

IV. EXPERIMENTS

To test our approach, ten different end-effector goal locations are selected, five for each of the Baxter arms. The selected goals are different than any of the goals used to train the goal prediction and linear correspondence models. The goals are visually shown in the testing space around the robot with, approximately, four-inch diameter spheres numbered to reflect the goal index they are associated with. There is a Kinect camera directly in-front of the robot testing space setup where the human demonstrator's OpenNI

[3]http://sdk.rethinkrobotics.com/wiki/Arm_Control_Modes

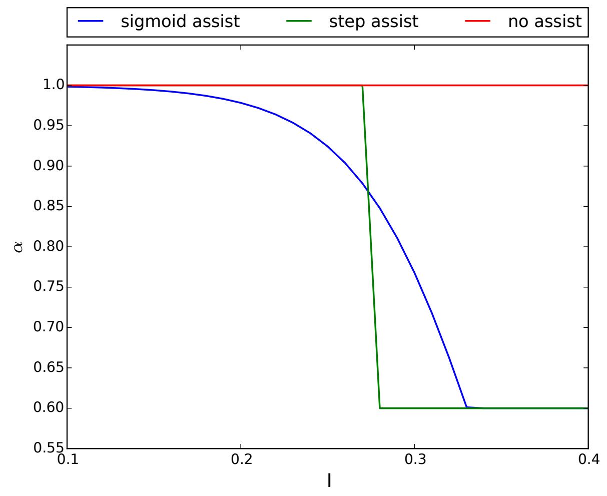

Fig. 4. Three α selection methods after final parameter adjustments.

skeleton data is captured. The Kinect camera is positioned so that a demonstrator can see both the Baxter robot and the goals.

For each demonstrator, all three of the α selection criteria are tested twice. The order in which the assist and no assist α selection methods are used is randomized between demonstrators. A demonstrator is never told what α selection method is being used at a given time.

For a given α selection test, all ten goals are tested in a random order. We refer to one goal test as a goal test sequence. A sequence always starts with the robot arms at neutral positions that are the same for every sequence. The demonstrator is told what the objective goal is prior to the start of any goal test sequence through a graphics display and given five seconds to prepare. After this countdown, the sequence starts.

At the start of a test sequence, the associated arm position controller of the selected goal is enabled. The input used by the arm controller is the output from (12). The objective of the sequence is to get the associated arm's end-effector within 0.13 meters (Euclidean distance) of the selected goal. The demonstrator must have the end-effector stay within that distance tolerance for 2.0 seconds to register the sequence as a successful sequence. The demonstrator has 15 seconds to complete this sequence, otherwise, a timeout is recorded for the sequence. When using an assist method, the α value is kept at 1.0 until the arm end-effector travels at least 0.6 meters. This is to allow a partial trajectory to be established. With no partial trajectory, the inverse LQR goal predictive method output will be based off of the distance away from the goal solely because we are using a distance prior and there is no trajectory at that point to update the posterior.

The metrics we evaluate each sequence on are the completion time and the end-effector distance traveled. The end-effector distance traveled is calculated by summing up the Euclidean distances between controller steps. If a timeout occurs for a sequence, we record the completion time as the

TABLE I
AVERAGE IMPROVEMENT OF COMPLETION TIME AND DISTANCE TRAVELED AND CORRESPONDING P-VALUE STATISTICS

	Δt [s][1]	p-value[2]	Δd [m][1]	p-value[2]
No Assist to Sigmoid	2.1	2.8e-6	0.47	2.2e-5
No Assist to Step	1.3	5.8e-5	0.29	4.9e-5
Step to Sigmoid	0.8	1.3e-2	0.18	5.7e-5

[1] Average improvement across all participants with both left and right arm results. Results shown are decrease improvements.
[2] The p-values computed using pairwise t-testing for each participant comparing participant's average results from each of the control strategies. Shown p-values are one-sided.

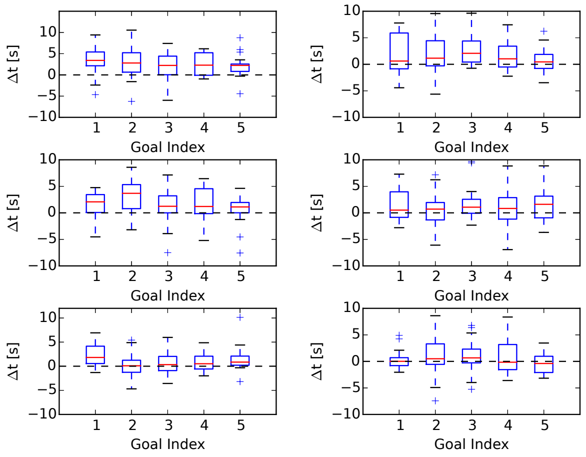

Fig. 5. Time improvements results for both arms. Plots in the first row show the improvement from using no assist to using the sigmoid assist, plots in the second row show the improvement from using no assist to using the step assist, and plots in the last show the improvement from using step assist to sigmoid assist. The left column is the left arm results and the right column is the right arm results.

timeout setting amount, which is 15 seconds.

Overall, we had 18 completed testing sets of data from human demonstrators. Each demonstrator contributed 60 datapoints, 6 α selection tests each having 10 goals.

Averaging the sequence results of all 18 demonstrators (Table I), we see an average decrease in the completion time of a sequence on the order of 1 to 2 seconds and a decrease of distance traveled by less than 0.5 meters when comparing the assist methods to the no assist method. These results might appear to be moderate improvements. However, considering the relatively easy task we used to test with, these results are of practical importance. We also note that the sigmoid assist method provides a larger improvement than the step assist method. This result is not surprising given that the sigmoid assist provides a continuous range of α values compared to the two α value outputs of the step assist method. The p-values of these results, calculated using a paired statistical test for different methods using paired teleoperation sequences to each goal, show that the improvements are statistically significant.

Averaging the sequence results on a per-goal basis (Fig. 5,

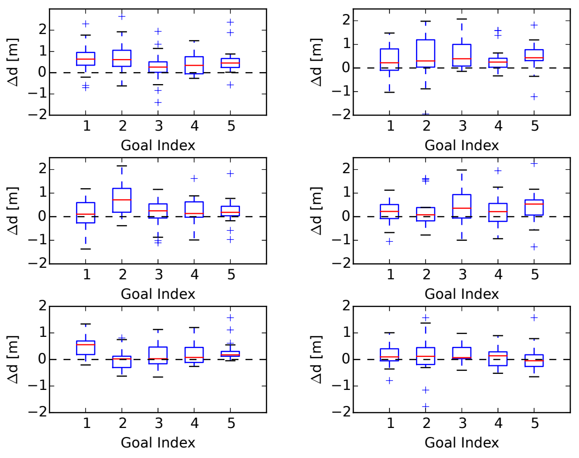

Fig. 6. Distance traveled results for both arms. Plots in the first row show the improvement from using no assist to using the sigmoid assist. Plots in the second row show the improvement from using no assist to using the step assist. Plots in the last show the improvement from using step assist to sigmoid assist. The left column contains the left arm results and the right column contains the right arm results.

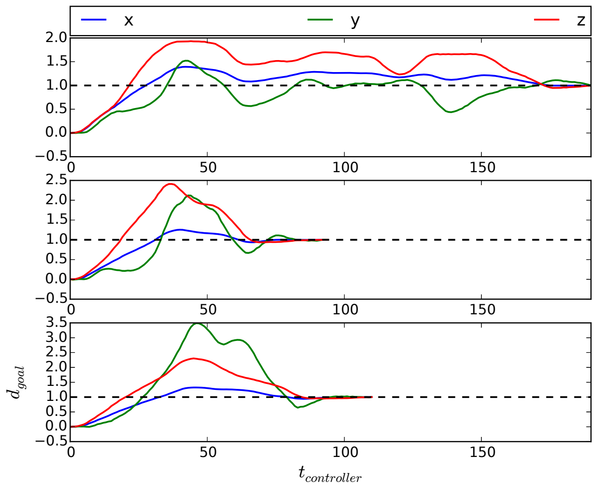

Fig. 7. Example of the improvement that the assisting methods provide to the arm end-effector trajectory. Data is from the same participant attempting to teleoperate a robot arm from the neutral position to the same goal position. From the top, the first plot is the no assist method end-effector trajectory result, the second plot is the sigmoid assist method, and the last plot is the step assist method. All three translational dimensions (x,y,z) of an end-effector are shown. The $t_{controller}$ axis represents time referenced by the arm controller time step since the start of a testing sequence. The d_{goal} axis represents the fraction of the distance the arm end-effector has traveled towards the goal.

Fig. 6) we see improvements in both task completion time and distance traveled across all 10 goals when comparing results of the assisting methods and no assist method as well. For a majority of the goals, the sigmoid assist method out performs the step assist.

The improvements that the assist methods provide can also be seen in the actual arm end-effector trajectories of the test sequences. For example, in Fig. 7 we see that one demonstrator was able to teleoperate the arm in a much more efficient path to the goal with the assist methods. When no assist was applied, the end-effector trajectory appears more sporadic, taking longer to stabilized near the goal coordinates. Although this is one specific example of improvement, it provides insight into how the assisting actions improve the efficiency of teleoperation.

A possible concern with our results is that the observed improvements could have been achieved using the distance prior, defined in Eq. (11), solely for the likelihood of a robot arm goal state instead of using our inverse LQR model output $P(S_G|trajectory_{init \to t})$ which incorporates learned robotic arm system dynamics. To show we believe this is not the case, in Fig. 8, we see differences between $P(S_G|S_t)$ and $P(S_G|trajectory_{init \to t})$ for the probability of the selected goal of a trial across all of the data gathered from the validation experiment trials. Although $P(S_G|S_t)$ and $P(S_G|trajectory_{init \to t})$ are often very similar in value, there are times when these values differ from 0.1 to 0.7, which shows that the learned system dynamics provide additional information in predicting goal likelihood over using distance measures alone.

V. CONCLUSIONS

In this work, we have shown that inverse optimal control approaches, specifically inverse LQR, can be used to extract the intent of teleoperation control actions in real time. This intent can be used to improve teleopation task completion efficiency by applying an assist action to a teleoperation control action. Specifically, we have shown these results in a depth camera teleoperation setting.

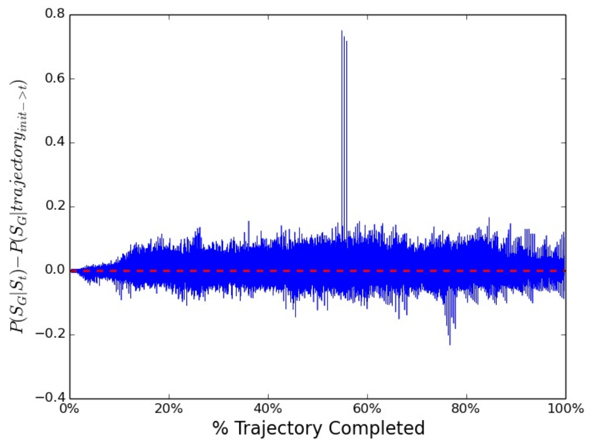

Fig. 8. The difference between the selected goal's probability under the distance prior and the inverse LQR model as a function of the percentage of the trajectory completed for all data gathered in our validation experiments.

There are many interesting extensions to this work to consider in the future. First, we believe the improvements shown in this work will increase as the difficulty of the teleoperation task increases. Therefore, we plan to extend this work to more complicated teleoperation tasks, such as grasping objects and more difficult arm navigation tasks.

Second, it is possible to predict the best action to get a robot arm end-effector into a goal state from the inverse LQR model. Instead of using goal coordinates and inverse kinematics for $A_{assist,t}$, we would like to use these predicted actions as $A_{assist,t}$ instead.

Lastly, incorporating obstacle avoidance with the assist action would be very beneficial. In [36], it was shown that the inverse LQR prediction method can incorporate waypoint states $s_{W,i}$ into this inverse optimal control formulation with an additional cost term in the cost function shown above. Using this waypoint formulation, we believe it is possible through arm demonstrations and including obstacle avoidance using waypoints during training that a cost function can be learned that provides next actions that avoid obstacles in a testing environment. With the completion of the second extension just mentioned above, this next action could be used as $A_{assist,t}$.

ACKNOWLEDGMENT

We thank David Labak who helped out with writing code and gathering data early in this work.

REFERENCES

[1] K. Nagatani, S. Kiribayashi, Y. Okada, K. Otake, K. Yoshida, S. Tadokoro, T. Nishimura, T. Yoshida, E. Koyanagi, M. Fukushima, *et al.*, "Emergency response to the nuclear accident at the fukushima daiichi nuclear power plants using mobile rescue robots," *Journal of Field Robotics*, vol. 30, no. 1, pp. 44–63, 2013.

[2] K. Harada, S. Kajita, H. Saito, M. Morisawa, F. Kanehiro, K. Fujiwara, K. Kaneko, and H. Hirukawa, "A humanoid robot carrying a heavy object," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. IEEE, 2005, pp. 1712–1717.

[3] T. S. Lendvay, B. Hannaford, and R. M. Satava, "Future of robotic surgery," *The Cancer Journal*, vol. 19, no. 2, pp. 109–119, 2013.

[4] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *Journal of Machine Learning Research*, vol. 17, no. 39, pp. 1–40, 2016.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[6] J.-C. Latombe, *Robot motion planning*. Springer Science & Business Media, 2012, vol. 124.

[7] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[8] A. D. Dragan and S. S. Srinivasa, *Formalizing assistive teleoperation*. MIT Press, July, 2012.

[9] K. Muelling, A. Venkatraman, J.-S. Valois, J. 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," *arXiv preprint arXiv:1503.05451*, 2015.

[10] W. Song, X. Guo, F. Jiang, S. Yang, G. Jiang, and Y. Shi, "Teleoperation humanoid robot control system based on kinect sensor," in *Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2012 4th International Conference on*, vol. 2. IEEE, 2012, pp. 264–267.

[11] G. Du, P. Zhang, J. Mai, and Z. Li, "Markerless kinect-based hand tracking for robot teleoperation," *International Journal of Advanced Robotic Systems*, vol. 9, 2012.

[12] G. Du and P. Zhang, "Markerless human–robot interface for dual robot manipulators using kinect sensor," *Robotics and Computer-Integrated Manufacturing*, vol. 30, no. 2, pp. 150–159, 2014.

[13] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," *Computer vision and image understanding*, vol. 104, no. 2, pp. 90–126, 2006.

[14] A. Shingade and A. Ghotkar, "Animation of 3d human model using markerless motion capture applied to sports," *arXiv preprint arXiv:1402.2363*, 2014.

[15] T. Shiratori, H. S. Park, L. Sigal, Y. Sheikh, and J. K. Hodgins, "Motion capture from body-mounted cameras," in *ACM Transactions on Graphics (TOG)*, vol. 30, no. 4. ACM, 2011, p. 31.

[16] M. R. Andersen, T. Jensen, P. Lisouski, A. K. Mortensen, M. K. Hansen, T. Gregersen, and P. Ahrendt, "Kinect depth sensor evaluation for computer vision applications," *Electrical and Computer Engineering Technical Report ECE-TR-6*, 2012.

[17] J. Vertut, *Teleoperation and robotics: applications and technology*. Springer Science & Business Media, 2013, vol. 3.

[18] M. Monfort, A. Liu, and B. D. Ziebart, "Intent prediction and trajectory forecasting via predictive inverse linear-quadratic regulation," in *Proceedings of The Twenty-Ninth AAAI Conference on Artificial Intelligence*, vol. 5, June 2015, pp. 3672–3678.

[19] Z. Ju, C. Yang, and H. Ma, "Kinematics modeling and experimental verification of baxter robot," in *Control Conference (CCC), 2014 33rd Chinese*. IEEE, 2014, pp. 8518–8523.

[20] Z. Zhang, "Microsoft kinect sensor and its effect," *IEEE multimedia*, vol. 19, no. 2, pp. 4–10, 2012.

[21] J. Shotton, R. Girshick, A. Fitzgibbon, T. Sharp, M. Cook, M. Finocchio, R. Moore, P. Kohli, A. Criminisi, A. Kipman, *et al.*, "Efficient human pose estimation from single depth images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2821–2840, 2013.

[22] H. Reddivari, C. Yang, Z. Ju, P. Liang, Z. Li, and B. Xu, "Teleoperation control of baxter robot using body motion tracking," in *Multisensor Fusion and Information Integration for Intelligent Systems (MFI), 2014 International Conference on*. IEEE, 2014, pp. 1–6.

[23] C. Hu, M. Q. Meng, P. X. Liu, and X. Wang, "Visual gesture recognition for human-machine interface of robot teleoperation," in *Intelligent Robots and Systems,(IROS 2003). Proceedings. IEEE/RSJ International Conference on*, vol. 2. IEEE, 2003, pp. 1560–1565.

[24] J. Y. Chen, E. C. Haas, and M. J. Barnes, "Human performance issues and user interface design for teleoperated robots," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 6, pp. 1231–1245, 2007.

[25] K. Hauser, "Recognition, prediction, and planning for assisted teleoperation of freeform tasks," *Autonomous Robots*, vol. 35, no. 4, pp. 241–254, 2013.

[26] R. Kalman, "When is a linear control system optimal?" *Trans. ASME, J. Basic Engrg.*, vol. 86, pp. 51–60, 1964.

[27] A. Y. Ng, S. J. Russell, *et al.*, "Algorithms for inverse reinforcement learning." in *Icml*, 2000, pp. 663–670.

[28] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proc. International Conference on Machine Learning*, 2004, pp. 1–8.

[29] N. Ratliff, J. A. Bagnell, and M. Zinkevich, "Maximum margin planning," in *Proc. International Conference on Machine Learning*, 2006, pp. 729–736.

[30] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Human behavior modeling with maximum entropy inverse optimal control." in *Association for the Advancement of Artificial Intelligence Spring Symposium: Human Behavior Modeling*, 2009.

[31] B. D. Ziebart, J. A. Bagnell, and A. K. Dey, "The principle of maximum causal entropy for estimating interacting processes," *IEEE Transactions on Information Theory*, vol. 59, no. 4, pp. 1966–1980, 2013.

[32] S. Levine and V. Koltun, "Continuous inverse optimal control with locally optimal examples," in *International Conference on Machine Learning*, 2012.

[33] M. H. Kutner, C. Nachtsheim, and J. Neter, *Applied linear regression models*. McGraw-Hill/Irwin, 2004.

[34] M. W. Spong and M. Vidyasagar, *Robot dynamics and control*. John Wiley & Sons, 2008.

[35] A. D. Dragan and S. S. Srinivasa, "A policy-blending formalism for shared control," *The International Journal of Robotics Research*, vol. 32, no. 7, pp. 790–805, 2013.

[36] A. Byravan, M. Montfort, B. Ziebart, B. Boots, and D. Fox, "Layered hybrid inverse optimal control for learning robot manipulation from demonstration," in *NIPS workshop on autonomous learning robots*. Citeseer, 2014.