When and How to Help: An Iterative Probabilistic Model for Learning Assistance by Demonstration

Harold Soh and Yiannis Demiris

Abstract

Crafting a proper assistance policy is a difficult endeavour but essential for the development of robotic assistants. Indeed, assistance is a complex issue that depends not only on the task-at-hand, but also on the state of the user, environment and competing objectives. As a way forward, this paper proposes learning the *task of assistance* through observation; an approach we term Learning Assistance by Demonstration (LAD). Our methodology is a subclass of Learning-by-Demonstration (LbD), yet directly addresses difficult issues associated with proper assistance such as *when* and *how* to appropriately assist. To learn assistive policies, we develop a probabilistic model that explicitly captures these elements and provide efficient, online, training methods. Experimental results on smart mobility assistance — using both simulation and a real-world smart wheelchair platform — demonstrate the effectiveness of our approach; the LAD model quickly learns when to assist (achieving an AUC score of 0.95 after only one demonstration) and improves with additional examples. Results show that this translates into better task-performance; our LAD-enabled smart wheelchair improved participant driving performance (measured in lap seconds) by 20.6s (a speedup of 137%), after a single teacher demonstration.

I. Introduction

Over the years, the robotics research community has propelled the development of the intelligent robot assistant, making important scientific and technological advances that have resulted in (among many other examples) smart wheelchairs that predict user intent [1] and robotic laparoscopic camera assistants that help doctors perform complex surgery [2]. That said, robotic assistants have yet to achieve widespread use and remain largely confined to laboratory and other controlled environments.

One reason for this lack of expansion into real-world settings is that *proper* assistance (by human or robot) is challenging. By its nature, assistance is contextual, dependant not only on the current task, but also on the state of the user, the environment and the assistant's capabilities. Furthermore, competing goals may demand different degrees of assistance. In educational settings for example, the primary goal is long-term user development rather than short-term task completion [3]. Although it may be possible to derive assistive policies from "first principles", it requires an inordinate amount of prior knowledge to be integrated into a robotic system.

In this paper, we adopt a novel perspective on the problem, approaching it from a Learning-by-Demonstration (LbD) [4], [5] standpoint. As compared to developing policies by hand, LbD methods derive policies from teacher

H. Soh and Y. Demiris are with the Personal Robotics Lab, Intelligent Systems and Networks Group, Dept. of Electrical and Electronic Engineering, Imperial College London, SW7 2BT London, United Kingdom. haroldsoh at ic.ac.uk and y.demiris at ic.ac.uk

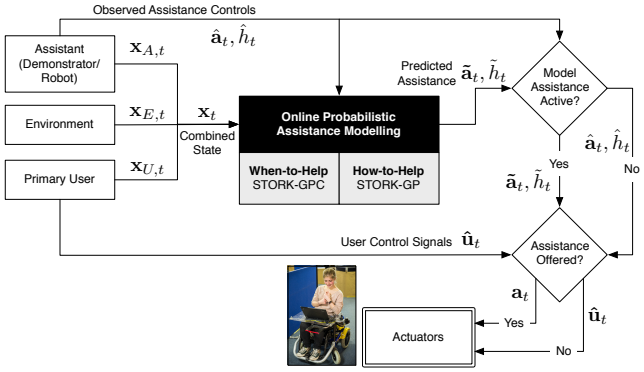


Fig. 1. Learning-Assistance-by-Demonstration (LAD) System Overview. Our model learns both *when* and *how* to assist iteratively from an assistant (demonstrator) helping a user accomplish a task. In this work, we apply our method to the problem of assisted smart mobility and present results using the ARTY smart wheelchair platform.

demonstrations. In a similar fashion, we aim to derive assistive policies by *observing an assistant*; an approach we term as Learning Assistance by Demonstration (LAD).

Framing assistive learning in this way allows us to retain many of the benefits associated with LbD; for complex tasks, demonstration is often more intuitive than hand-coding specific behaviours, allowing non-roboticists to participate in policy development [4].

In addition, LAD augments LbD by focussing on the assistive element. Current LbD systems are *task-centred* in that they focus on deriving a policy for completing the demonstrated task with or without a human-in-the-loop. For example, to teach a smart wheelchair to navigate, we would provide it with driving demonstrations to derive a "how-to-drive" policy (e.g., [6]). When the smart wheelchair is assisting a user, the robot faces the difficult decisions of not only *how* best to assist but also *when* it is appropriate to intercede. One potential solution is to infer the user's intent from (noisy) observations [7], [8], using the learned policies and failure models. In contrast, LAD is a direct approach. Instead of deriving a policy for "how-to-drive", we extract a policy for "how-to-help-a-user-drive".

A natural follow-up question is how to learn assistive policies from demonstrations? As a solution, we contribute a probabilistic model and provide efficient training algorithms. Described in Section III, our model is trainable *online* during the demonstration process (which facilitates interactive teaching) and more importantly, places emphasis on capturing both when and how to appropriately assist.

To demonstrate the feasibility of the LAD approach and our associated model, we focus on the problem of smart assisted mobility. Section IV describes experiments performed using a simulated robot and Section V extends this discussion to real-world experiments on the ARTY smart wheelchair platform [9], [10] with human participants. Finally, Section VI concludes the paper by highlighting research questions arising from this work.

II. Learning Assistance by Demonstration: Problem Statement

In this section, we formally introduce LAD and detail its elements and distinguishing features. To help guide this discussion, Fig. 1 illustrates an overview of our system.

Recall that we are principally interested in robots that learn how to assist humans accomplish tasks through observation of an assistant. For simplicity, let us consider a scenario where there is one assistant (U_A) helping a primary user (U_P) complete a primary task (T_P)[1]. In effect, U_A is engaging in an *assistive task* (T_A) and our robot's goal is to extract an assistive policy, i.e., a mapping from states $\mathbf{x}_t \in \mathcal{X}$ to assistive actions $\mathbf{a}_t \in \mathcal{A}$, denoted $\pi_A(\mathbf{x}_t) : \mathcal{X} \rightarrow \mathcal{A}$. This should be contrasted against the typical LbD approach which is to model the primary task and obtain a task policy $\pi_P(\mathbf{x}_t) : \mathcal{X} \rightarrow \mathcal{U}$ where $\mathcal{U}$ is the set of control actions.

To enable us to extract this policy, we have access to (partial and noisy) observations of the state $\mathbf{x}_t = (\mathbf{x}_{A,t}, \mathbf{x}_{U,t}, \mathbf{x}_{E,t})$ where $\mathbf{x}_{A,t}$, $\mathbf{x}_{U,t}$, $\mathbf{x}_{E,t}$ are the state of the assistant, the user and the environment. Note that $\mathbf{x}_{A,t}$ represents either the assistant or the robot, depending on whether the robot is observing or executing the policy. We also observe the actions performed by U and A, denoted as $\hat{\mathbf{u}}_t$ and $\hat{\mathbf{a}}_\mathbf{t}$ respectively. In addition, we assume a visible signal $\hat{h}_t$ which indicates when the assistance is given.

To make these concepts more concrete, let us consider a specific smart wheelchair training scenario where an occupational therapist U_A is teaching a young boy U_P with special needs to navigate his environment. In addition, she wishes to train a the smart wheelchair R to assist the child when she is not present. To do so, whenever U_P is unable to complete a particular sub-task or is having difficulty, U_A engages a "guidance-mode" where she takes control and guides U_P appropriately with actions $\hat{\mathbf{a}}_\mathbf{t}$. R is able to observe these actions as well as the states $\mathbf{x}_t$ when interventions occur, and aims to extract the policy $\pi_A(\mathbf{x}_t)$ used by U_A.

An important element of our approach is the recognition that assistance may not be needed (or desired) continuously but only at *key* points during the process. This is relevant in many real-world applications where constant intervention may interfere with task completion or competing objectives. Using our example, if U_A does not provide sufficient assistance, the child may become frustrated with the activity. On the other hand, if U_A assists too much, the child learns to rely on her assistance, which negatively impacts his long-term development (a phenomena known as "learned helplessness" [11]). As such, deciding when to assist is paramount. In the next section, we discuss our probabilistic assistive model which formally captures these elements.

III. An Online Probabilistic Model for Learning Assistance by Demonstration

At a high-level, our model is a representation of an assistant making two choices at each time-step; 1) whether she should help and if so, 2) what she should do.

Let us begin by assuming that the assistive actions are continuous random variables $\mathbf{a}_t \in \mathbb{R}^D$, which can represent motor commands to an actuator for example. In addition, we define a "null" or zero action $\mathbf{a}_0 \triangleq \mathbf{0}$ (the assistant is not helping in the task). Our Bayesian model considers that assistive actions $\mathbf{a}_t$ are generated from an "assistance

[1]Our method can be extended to scenarios involving multiple assistants and primary users.

component" or a "do-nothing component" (where the action probability is a Dirac delta centered at $\mathbf{a}_0$):

$$p(\mathbf{a}_t|\mathbf{x}_t) = p(h_t = 1|\mathbf{x}_t)\mathcal{N}(\mathbf{a}_t|f(\mathbf{x}_t), \mathbb{V}[f(\mathbf{x}_t)]) + (1 - p(h_t = 1|\mathbf{x}_t))\delta_0(\mathbf{a}_t). \quad (1)$$

A critical part of our model is the probability of assistance (when-to-help) represented by a discrete/binary random variable $h_t \in \{-1, 1\}$ conditioned on the current state $\mathbf{x}_t$. In this work, h_t and $\mathbf{a}_t$ are modelled using a Gaussian Process classifier (GPC) and regressor (GP) respectively:

$$h_t|\mathbf{x}_t \sim \mathcal{GPC} \quad (2)$$

$$f(\mathbf{x}_t) \sim \mathcal{GP} \quad (3)$$

Since standard GPs are computationally expensive (requiring $O(n^3)$ time for training and $O(n^2)$ space) and we typically prefer our robotic assistants to learn iteratively, we used the Spatio-Temporal Online Recursive Kernel Gaussian Process (STORK-GP) [12], a specialised sparse GP with memory to account for temporal dependencies. In the following subsections, we flesh out our model by giving specifics on how our model can be trained efficiently in real-time.

A. How-to-Help with STORK-GP Regression

Because regression with GPs is more straightforward than classification, we first introduce the "how-to-help" component which models/generates the assistive actions, $\mathbf{a}_t$. Given observations $\mathbf{x}_t \in \mathcal{X}$, a GP is a set of random variables whereby any finite subset has a joint Gaussian distribution [13]. It is specified by its mean function, $m(\mathbf{x}_t) = \mathbb{E}[f(\mathbf{x}_t)]$ and its covariance function, $k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))]$.

The STORK-GP is an online GP that uses sparse approximations [14], [15] used previously for learning-by-demonstration [16] and a *recursive kernel* with automatic relevance determination (ARD). In contrast to prevailing methods, the STORK-GP can be updated sequentially in real-time and models temporal dependencies. Because its internals are relatively involved and discussed elsewhere [17], [12], we focus on describing the three main aspects of the algorithm and provide references for readers wanting additional detail:

1) The Projected Process Approximation: For online learning, we revise our model as observations arrive using a Bayesian update that minimises the Kullback-Leibler divergence, $KL(\hat{p}_t||q)$ [14]. The GP in its "natural parameterisation" form is given by:

$$m_t(\mathbf{x}) = \boldsymbol{\alpha}_t^{\mathrm{T}}\mathbf{k}(\mathbf{x}) \quad (4)$$

$$k_t(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}') + \mathbf{k}(\mathbf{x})^{\mathrm{T}}\mathbf{C}_t\mathbf{k}(\mathbf{x}') \quad (5)$$

where $\boldsymbol{\alpha}$ and $\mathbf{C}$ are model parameterisations that are iteratively updated (See [14], [15]).

2) Maintaining Sparsity: Although equations (4)-(5) allow us to update the GP sequentially, $\boldsymbol{\alpha}$ and $\mathbf{C}$ grow with each processed datum. To control the model's growth, we limit the number of the datapoints retained, termed the

"basis vectors" (BVs), denoted $\mathbf{b}_i \in \mathcal{B}$. Each incoming point is scored for "novelty" using:

$$\gamma(\mathbf{x}_{t+1}) = k(\mathbf{x}_{t+1}, \mathbf{x}_{t+1}) - \mathbf{k}_{t+1}^{\mathrm{T}} \mathbf{K}^{-1} \mathbf{k}_{t+1} \tag{6}$$

where $\mathbf{k}_{t+1} = [k(\mathbf{b}_i, \mathbf{x}_{t+1})]$ and $\mathbf{K}^{-1} = [k(\mathbf{b}_i, \mathbf{b}_j)]$ with $\mathbf{b}_i, \mathbf{b}_j \in \mathcal{B}$. If $\gamma(\mathbf{x}_{t+1})$ is below some threshold, ϵ_γ (10^{-4} in our work), then an *approximate* update is performed, which incorporates observations but does not increase the number of BVs. To limit the maximum size or *capacity* of the BV set, it often becomes necessary to delete a BV. Again, we score each $\mathbf{b}_i \in \mathcal{B}$ and remove the lowest scoring BV using a reduced update.

3) Recursive Spatio-Temporal Kernel: The main ingredient of a GP is the covariance function $k(\mathbf{x}, \mathbf{x}')$. Because real-world environments are often partially observable in nature, the model has to make decisions and inferences based on a *history* of observations. To account for this, we used the recursive kernel:

$$\kappa_t^{\mathrm{ARD}}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2}(\mathbf{x}_t - \mathbf{x'}_t)^{\mathrm{T}} \mathbf{M} (\mathbf{x}_t - \mathbf{x'}_t)\right) \exp\left(\frac{\kappa_{t-1}(\mathbf{x}, \mathbf{x}') - 1}{\sigma_\rho^2}\right) \tag{7}$$

where $\mathbf{M}$ is a symmetric $d \times d$ matrix that controls the "importance" of the inputs [12]. It is conventional that $M = \mathrm{diag}(\boldsymbol{l})^{-2}$ where $\boldsymbol{l} = [l_i]_{i=1}^d$. Modifying the l_i's allows us to control the impact that the inputs have on the predictions (the kernel function's responsiveness to input dimension k is inversely related to l_k). Moreover, unlike the commonly used squared-exponential, this kernel accounts for temporal dependencies where the parameter σ_ρ^2 controls how much the past is "weighted".

Finally, to perform predictions with the STORK-GP, we compute the mean of the predicted distribution at an unknown test point $\mathbf{x}_t^*$:

$$\mu_* = \mathbf{k}_t(\mathbf{x}_t^*)^{\mathrm{T}} \boldsymbol{\alpha}_t \tag{8}$$

and variance:

$$\sigma_*^2 = k(\mathbf{x}_t^*, \mathbf{x}_t^*) + \mathbf{k}_t(\mathbf{x}_t^*)^{\mathrm{T}} \mathbf{C}_t \mathbf{k}_t(\mathbf{x}_t^*) \tag{9}$$

B. When-to-Help with STORK-GP Classification

Let us re-use the machinery developed in the previous section by modelling the "when-to-help" random variable h_t using another STORK-GP. Since h_t is a binary random variable, we adopt a classification perspective. In this work, we have assumed that h_t is visible during training, allowing us to apply supervised training directly. In particular, whenever assistance is offered by the demonstrator, we observe $h_t = 1$ and $h_t = -1$ otherwise.

To perform classification, we have used the probit model where the predictive distribution is given by:

$$p(h_t | \mathbf{x}_t, \boldsymbol{\alpha}_t, \mathbf{C}_t) = \Phi\left(\frac{h_t \mu_*}{\sigma_*^2}\right) \tag{10}$$

where $\Phi(z) = \int_{-\infty}^{z} \mathcal{N}(z|0,1)dz$ is a cumulative density function of a standard normal distribution and μ_* and σ_*^2 are given by (8) and (9) respectively[2].

[2]More details on classification using the sparse online GP can be found in [14], [15].

C. From Training to Prediction and Control

By using STORK-GPs, our LAD model can be trained online as the demonstration is being conducted. For example, in our experiments, the when-to-help classifier is continuously trained throughout the demonstration with both positive and negative observed samples $(\mathbf{x}_t, \hat{h}_t)$. The how-to-help regressor is only trained when assistance is offered ($\hat{h}_t = 1$) with the observed sample pairs $(\mathbf{x}_t, \hat{\mathbf{a}}_t)$.

Once both models are trained, we can apply the models directly in an assistive policy $\pi_A(\mathbf{x}_t)$. In this work, we applied a simple threshold W_h and assistance ($\tilde{h}_t = 1$) was offered whenever the probability of assistance $p(h_t|\mathbf{x}_t) \geq W_h$, where $p(h_t|\mathbf{x}_t)$ is obtained using (10). The threshold W_h is a user-defined parameter that controls the minimum level of confidence before offering assistance. The assistive control signals, $\tilde{\mathbf{a}}_t$, are taken as the mean prediction of the STORK-GP regressor given by (8).

IV. Simulation Experiments

To illustrate how LAD and our probabilistic model can be applied, let us focus on the real-world problem of assisted mobility for individuals with special needs. An estimated 61-91% of wheelchair users (1.4 to 2.1 million people in the US) could gain from an assistive smart wheelchair [18]. Research in this area has produced a multitude of obstacle avoidance algorithms and real-world platforms that help wheelchair users avoid collisions [19], [20], [21], [22], [7], [23]. For a more complete review of smart wheelchairs, we refer readers to [24].

As a prelude to our real-world trials (described in Section V), we conducted a simulation study designed to test if our model was capable of capturing when and how to help a wheelchair user after a minimal set of demonstrations. Our "first-cut" scenario assumes that the user lacks a specific fine motor control skill which inhibits his ability to make *sharp* right turns. This loss is permanent and thus, assistance should be offered to prevent user frustration. Since the user is otherwise capable of controlling the joystick, assistance should withheld during *soft* right turns (less than 60°), left turns and forward/backward movements to encourage the development of wheelchair driving skills. The efficacy of our approach was measured using the model accuracy (using the human demonstrator signals as the "gold standard") and the time required to complete a lap around a driving course.

A. Simulation Experiment Setup

Based on the real-world environment shown in Fig. 2, we created a simulated world using the Stage simulator [25]. Our test subject was an autonomous robot driver (RD) developed using the Robot Operating System (ROS) [26] navigation stack. For this test, RD's maximum forward and rotational velocities were set at 0.7m/s and 0.5 rad/s and local obstacle avoidance was performed using Dynamic Window Approach (DWA) [27]. We have made our model implementation and experimental setup freely available for download[3].

We tasked RD to drive from the start point (S), through R_1 and R_2 and back to the finish point (F). Note that RD always made right turns at both R_1 and R_2. Under normal operation, RD was able to complete a lap in an average

[3]Available on the author's website at `http://www.haroldsoh.com` and the Personal Robotics website `http://www3.imperial.ac.uk/PersonalRobotics`.

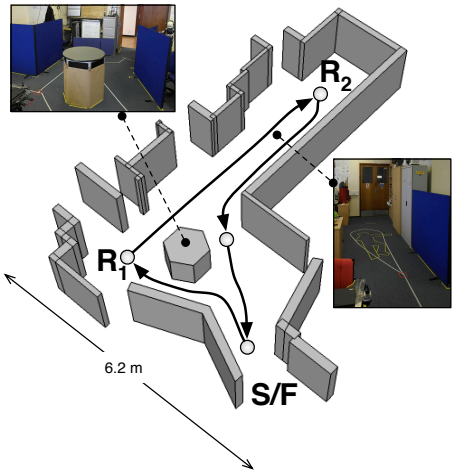


Fig. 2. Driving course used in our experiments. Participants (and our simulated robot) were tasked to drive from S to F, passing through and making right turns at R_1 and R_2.

of 34.59s (sd: 0.356s). We induced the control limitation by scaling right turns to a maximum of 0.15 rad/s. This simple constraint had a drastic impact on RD's lap performance, almost doubling lap times to 59.11s (sd: 0.95s).

B. Assistance Demonstration

A human demonstrator[4] was tasked to provide assistance in the form of a control takeover (in the spirit of "hand-over-hand" control used by occupational therapists) during the right turns at R_1 and R_2. Demonstration was performed using a wireless joystick controller which provided an observable takeover signal, $\hat{h}_t$ (1 whenever assistance was provided and -1 otherwise) and the assistive command velocities, $\hat{\mathbf{a}}_t = (\hat{a}_{x,t}, \hat{a}_{\theta,t})$. During takeover, the assistive control signals replace those sent by RD (an alternative approach would be to augment RD's control velocities).

C. Model Setup and Parameters

In this experiment, the (partially-observable) state of the system was modelled as $\mathbf{x}_t = (\mathbf{x}_U, \mathbf{x}_A, \mathbf{x}_E)$ where the state of the robot was represented by its current velocities $\mathbf{x}_A = (v_{x,t}, v_{\theta,t})$ and the environmental state was captured by forward laser scan readings (separated into $M = 15$ segments), $\mathbf{x}_E = \mathbf{s}_t = (s_{1,t}, s_{2,t}, \ldots, s_{M,t})$. Since we did not employ "user-specific" sensors, we used the user's desired translational and rotational velocities as a proxy for intent (an internal state), i.e., $\hat{\mathbf{u}}_t = \mathbf{x}_U = (u_{x,t}, u_{\theta,t})$. It is important to note that, in the limited control state, the model is only able to observe the scaled right turns (instead of the full commanded rotational velocity of 0.5 rad/s). More state complexity can be accommodated by our model but we found this representation to be sufficient for the task. The parameters for our probabilistic assistive model were tuned experimentally and are shown in Table I[5].

[4]The demonstrator was the lead author in the experiments presented.

[5]We are currently experimenting with real-time parameter tuning using likelihood maximization and will report results in future work.

TABLE I

ASSISTIVE MODEL PARAMETERS

Parameters	Value
Capacity (C)	400
Recursion Depth (τ)	4
Temporal scale (σ_ρ)	1.1
Lengthscale (l_U)	0.1
Lengthscale (l_A)	1.0
Lengthscale (l_E)	2.0
Signal Variance (σ_f^2)	0.1
Noise (σ_n^2)	0.1 (GPC) / 0.01 (GP)
Assistance Threshold (W_h)	0.5

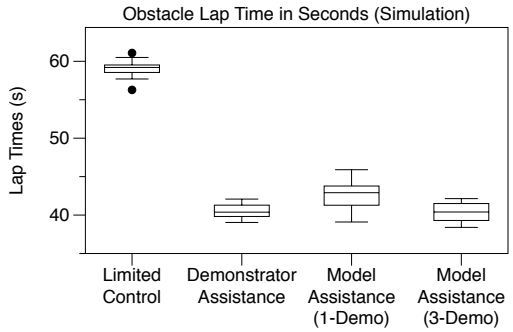


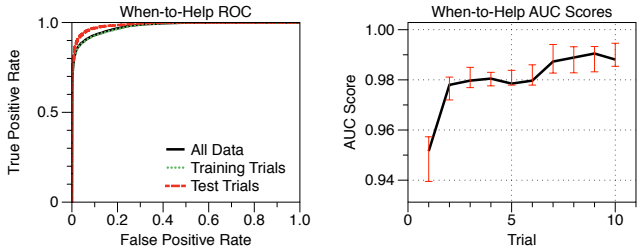
Fig. 3. Lap times in seconds. Without assistance, the control limited robot (RD) completes a lap in 59.1s. With human demonstrator assistance, RD is able to complete a lap in approximately 40.5s (a speedup improvement of 146%). With a *single* demonstration, our learned model is able to improve lap times to 42.6s (139% speedup) and with *three* demonstrations, our learned model achieves a performance statistically similar to a human demonstrator (average lap time: 40.4s).

D. Model Accuracy

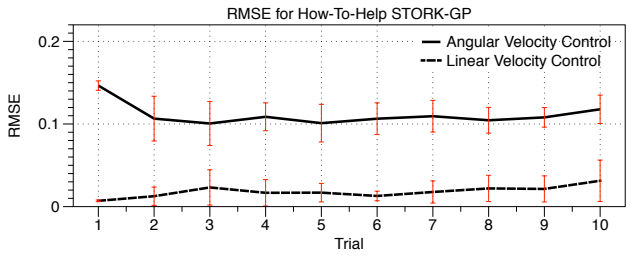
To better analyse the learning capability of our model, we used ten supervisor demonstrations as a single dataset to train and test our model. The model learned in an online fashion, i.e., trained/tested at each time step during nine laps. The final tenth lap was used for testing. Fig. 4 summarises the performance of both the classifier and regressor averaged over fifteen runs (with the ten supervisor demonstrations randomly permuted).

The ROC curves obtained by varying the assistance threshold, W_h are shown in Fig. 4a. Our trained model achieves a high true positive to false positive ratio, indicative of strong classifier performance. This is confirmed by the median and upper/lower quantile AUC scores over the nine training trials and the tenth testing lap (Fig. 4b). We observed that our model achieves a high AUC (≈ 0.95) after only a single demonstration. The median AUC scores are 0.9905 for the training portion and 0.9881 for the final test trial.

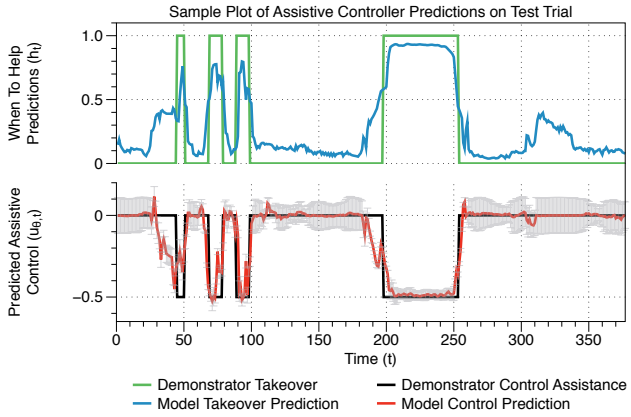
Turning our attention to assistive control accuracy, we see from Figs. 4c and 4d that the STORK-GP is able to generate assistive actions that are very similar to those generated by the demonstrator. The rotational command



(a) When-to-Help ROC (b) When-to-Help AUC



(c) How-to-Help RMSE



(d) Model Predictions during a Sample Test Lap

Fig. 4. Performance measures summarising LAD model accuracy. Our model achieves high AUC scores (for when-to-help classification) and low RMSE (for the how-to-help regressor), which get progressively better as more examples are given. See text for additional details.

velocities have higher RMSE as compared to the linear controls; a possible explanation is that the help was principally rotational (and thus more varied) in nature. Interestingly, Fig. 4d shows that the model is able to "anticipate" when assistance should be given — we see a rise in the model's predicted $p(h_t)$ and a decrease in the variance of $p(\mathbf{a}_t)$ well before the assistance is actually performed.

E. Lap Performance

Fig. 3 illustrates the lap times attained by RD when assisted by a human demonstrator and by our learned assistive model (over 10 independent laps). After only a single demonstration, our model is able to improve RD's lap time to 42.6s; a speedup of 139% over the control limited scenario and only an average of 2.6s behind that of the human

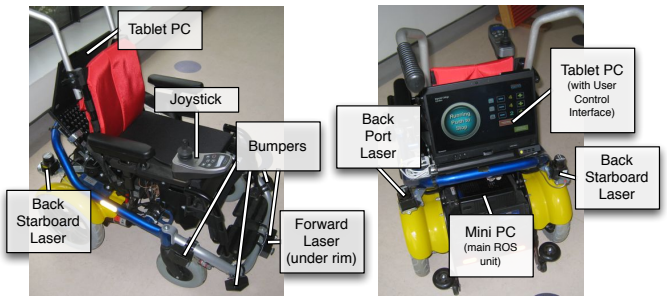


Fig. 5. The Assistive Robotic Transport for Youngsters (ARTY) Smart Wheelchair [9], [10].

assistance (40.5s). After three demonstrations, the model's assistive performance is statistically similar to that of the human demonstrator, reducing lap times to an average of 40.4s (Kolmogorov-Smirnov test with $p = 0.4370$ and test statistic $k_s = 0.3125$).

V. Real-World Experiments with the ARTY Smart Wheelchair

Based on the positive simulation results, we conducted a real-world experiment using the ARTY smart wheelchair (Fig. 5) [9], [10]. ARTY is a smart paediatric wheelchair developed at the Personal Robotics Laboratory at Imperial College London that has been successfully tested with end-users at a children's hospital.

More specifics about ARTY can be found in [9], but we note here that sensory information was obtained using three Hokuyo URG-04LX laser scanners and an inertial measurement unit connected to a base computational unit running ROS. Our assistive probabilistic model was developed in MATLAB and run as a ROS node on a Tablet PC connected to the base unit.

To prevent damage to the wheelchair and potential injury to participants, simple safeguarding using a DWA variant [9] was employed to prevent hard collisions with obstacles. This setup also enabled us to test how our assistive model can be used in conjunction with current obstacle avoidance methods.

A. Experimental Setup

Our real-world trials were set up similar to the simulation experiment; the test environment was a lab office space (Fig. 2) and we invited participants to drive the wheelchair along the specified track (from S to F) a total of four times. Before beginning the trial, the participants were allowed to familiarise themselves with the wheelchair by driving around the experimental route until they felt comfortable with its operation.

During the first lap, full control was conferred, allowing us to obtain the baseline performance of an average user. The control limitation on right turns was applied during the latter three laps; human demonstrator assistance was offered during the third lap (during which the model was simultaneously trained) and LAD model assistance

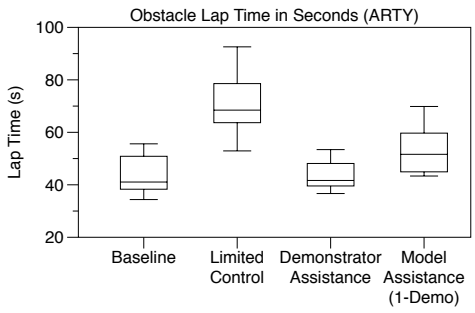


Fig. 6. Lap times in seconds for all twelve participants categorised by driving mode. The learned model improved participant lap performance by 20.6s (speedup of 137%) after only a single demonstration.

was offered during the final lap[6]. After each lap, the participants were given a survey to complete indicating how much they agreed with eleven statements[7] on a 5-point Likert scale:

Survey Questions:

1) I found navigating the obstacle course with the wheelchair easy.
2) I found navigating the obstacle course frustrating.
3) I performed well on the task.
4) The task was difficult at times.
5) I found the driving assistance to be helpful.
6) The driving assistance interfered with my driving in a negative way.
7) The driving assistance enabled me to complete the task faster.
8) I found the driving assistance to be timely.
9) The driving assistance negatively impacted my driving ability.
10) I would have completed the task easily even without the driving assistance.
11) I liked having driving assistance for this task.

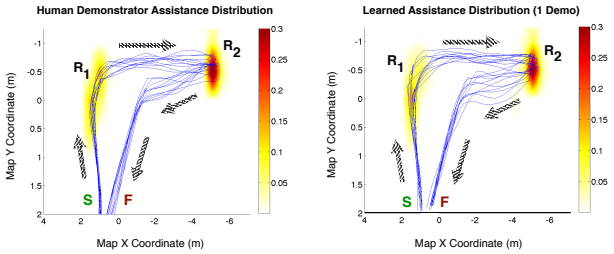
B. Empirical Results: Lap Performance

Twelve able-bodied participants (five female) ages 21-38 (mean: 27, sd: 4.91) participated in our experiment. Fig. 7 shows the paths taken by our subjects and a smoothed density plot of when assistance was given. Both distributions are similar, indicating that assistance was offered by our learned model under the correct situations. Note that the model only assisted at R_1 and R_2 and did *not* assist during the soft right turns (as shown by the driven paths on the return lap from R_2 to F).

Fig. 6 shows the lap-times achieved by our participants under the four aforementioned conditions. Under the baseline condition (where right turns were normal), participants were able to complete a lap in ≈ 48.5s (sd: 20.0s).

[6]Although the lap ordering may introduce a bias towards lower lap times in the assistance trial, it was only possible to offer the assistance after the model was suitably trained. Future work would look into introducing more laps with and without assistance to correct for possible bias.

[7]For the first two laps where assistance was not given, surveys consisting of only questions 1-4 were given.



(a) Demonstrator (b) Model (1-Demo)

Fig. 7. Driven paths and normalised smoothed density plots illustrating *when* assistance was offered by the human demonstrator and LAD model (best viewed in colour). The modelled density is more spread out with a greater variation in the paths accounting for the higher lap completion times. That said, both distributions are similar illustrating the learned model is able to capture when to appropriately assist; assistance was *not* offered between R_2 and F despite the user making "soft" right turns, as indicated by the paths.

When right turns were constrained, lap times increased by 62% to 76.7s (sd: 24.8s). Although the lap times during LAD model assistance was higher than the demonstrator performance (mean: 44.55s, sd: 8s), it is worth noting that the model was only trained using a single demonstration and still improved participants lap times by an average of 20.6s to 56.1s (a speedup of 137% over the limited control setting). This result is similar to that achieved in simulation and we posit that additional demonstrations would improve assistance capability.

C. Survey Results

Based on eleven surveys completed[8], the participants found navigating the course more difficult and more frustrating (Q1 and Q2) during limited control as compared to the other conditions (as was expected). However, the responses during (H)uman and LAD (M)odel assistance were similar to the baseline condition. A close examination suggests that the participants may have been able distinguish between the two conditions; 100% of the participants agreed or strongly agreed that human assistance was helpful in completing the task (Q7) versus 73% for the model assistance (the remaining 27% were neutral to the statement). That said, a Wilcoxon rank test did not find these differences to be statistically significant. In general, a majority of the participants liked the assistance (Q11 - H:63%, M:63%), finding the assistance to be timely (Q8 - H:63%, M:73%) and helpful in completing the task (Q5 - H:80%, M:73%).

VI. Conclusions

This work has proposed and discussed Learning Assistance by Demonstration (LAD); a LbD approach to the complex problem of deriving assistance policies. In addition, we have contributed a novel probabilistic model which captures both when and how to assist, and efficient methods for online training. Both simulation and real-world experiments demonstrate the efficacy of our approach; our smart wheelchair is able to learn rapidly to provide *contextual* assistance where needed, speeding up lap times by $137\% - 146\%$ (after only one to three demonstrations).

[8]One participant did not manage to complete the entire survey and as such, her survey was left out of the analysis. However, her completed portions did not reveal responses significantly different from other participants.

We believe LAD to be a rich and promising area for future exploration. From an experimental perspective, it is important to validate the model on more complex scenarios with the intended target population. In addition, we highlight three interesting research questions particular to this approach:

1) How can we train the model efficiently when the assistive signal is latent?: In this work, we have assumed that h_t is visible. In certain scenarios, h_t is hidden or latent, complicating the training process.

2) How can the model adapt to improvements on the part of the assisted user?: A potential solution may be to combine ideas from human-robot cross-training for collaborative robots [28] with our approach.

3) Can we extract more than control from the learnt assistive policies?: In this work, we have illustrated how LAD can be applied to derive assistive control policies. However, the learnt policy can also be used for studying how humans assist under various circumstances, which may supplement recent studies [29] and lead to interesting findings applicable to the general design of assistive robots.

ACKNOWLEDGMENT

The authors thank members of the Personal Robotics Lab, ICL, for their help with the experiments. This work was partially funded by the EU FP7 ALIZ-E project (248116).

REFERENCES

[1] T. Carlson and Y. Demiris, "Collaborative Control for a Robotic Wheelchair: Evaluation of Performance, Attention, and Workload," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 42, no. 3, pp. 876 –888, June 2012.

[2] C. Nezhat, O. Lavie, M. Lemyre, E. Unal, C. H. Nezhat, and F. Nezhat, "Robot-assisted laparoscopic surgery in gynecology: scientific dream or reality?" *Fertility and Sterility*, vol. 91, no. 6, pp. 2620 – 2622, 2009.

[3] J. Earwaker, *The tutorial relationship*, ser. Taking Issue: Debates in Guidance and Conselling in Learning. Routledge, 1998, ch. 6.

[4] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, May 2009.

[5] R. Dillmann, "Teaching and learning of robot tasks via observation of human performance," *Robotics and Autonomous Systems*, vol. 47, pp. 109 – 116, 2004.

[6] H. Chow and Y. Xu, "Learning human navigational skill for smart wheelchair in a static cluttered route," *Industrial Electronics, IEEE Transactions on*, vol. 53, no. 4, pp. 1350–1361, June.

[7] T. Carlson and Y. Demiris, "Human-wheelchair collaboration through prediction of intention and adaptive assistance," *IEEE International Conference on Robotics and Automation*, pp. 3926–3931, 2008.

[8] Y. Demiris, "Prediction of intent in robotics and multi-agent systems," vol. 8, no. 3, pp. 151–158, 2007.

[9] H. Soh and Y. Demiris, "Towards early mobility independence: An intelligent paediatric wheelchair with case studies," in *IROS 2012 Workshop on Progress, Challenges and Future Perspectives in Navigation and Manipulation Assistance for Robotic Wheelchairs*, 2012.

[10] ——, "Involving young children in the development of a safe, smart paediatric wheelchair," *ACM/IEEE HRI-2011 Pioneers Workshop*, 2011.

[11] M. E. Seligman, "Learned helplessness," *Annual Review of Medicine*, vol. 23, no. 5, p. 407, 1972.

[12] H. Soh, Y. Su, and Y. Demiris, "Online spatio-temporal Gaussian process experts with application to tactile classification," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, Oct., pp. 4489–4496.

[13] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[14] L. Csató, "Gaussian processes: iterative sparse approximations," Ph.D. dissertation, Aston University, 2002.

[15] L. Csató and M. Opper, "Sparse on-line gaussian processes," *Neural Computation*, vol. 14, no. 3, pp. 641–668, March 2002.

[16] D. Grollman and O. Jenkins, "Sparse incremental learning for interactive robot control policy estimation," in *IEEE International Conference on Robotics and Automation*, may 2008, pp. 3315 –3320.

[17] H. Soh and Y. Demiris, "Iterative temporal learning and prediction with the sparse online echo state Gaussian process," in *Neural Networks, The 2012 International Joint Conference on*, 2012, pp. 1–8.

[18] R. C. Simpson, E. F. LoPresti, and R. A. Cooper, "How many people would benefit from a smart wheelchair?" *Journal Of Rehabilitation Research And Development*, vol. 45, no. 1, pp. 53–71, 2008.

[19] H. A. Yanco, "A Robotic Wheelchair System: Indoor Navigation and User Interface," *LNCS Assistive Technology and Artificial Intelligence Applications in Robotics User Interfaces and Natural Language Processing*, vol. 1458, pp. 256–268, 1998.

[20] T. Gomi and A. Griffith, "Developing Intelligent Wheelchairs for the Handicapped," in *Assistive Technology and Artificial Intelligence*, ser. Applications in Robotics, User Interfaces and Natural Language Processing, vol. 1458, 1998, pp. 150 – 178.

[21] D. P. Miller and M. G. Slack, "Design and testing of a low-cost robotic wheelchair prototype," *Autonomous Robots*, vol. 2, no. 1, pp. 77–88, 1995.

[22] S. P. Levine, D. A. Bell, L. A. Jaros, R. C. Simpson, Y. Koren, and J. Borenstein, "The NavChair Assistive Wheelchair Navigation System." *IEEE Transactions on Rehabilitation Engineering*, vol. 7, no. 4, pp. 443–451, 1999.

[23] J. K. Tsotsos, G. Verghese, S. Dickinson, M. Jenkin, A. Jepson, E. Milios, F. Nuflo, S. Stevenson, M. Black, D. Metaxas, S. Culhane, Y. Ye, and R. Mann, "PLAYBOT: A visually-guided robot for physically disabled children," *Image & Vision Computing, Special Issue on Vision for the Disabled*, vol. 16, no. 4, pp. 275–292, 1998.

[24] R. C. Simpson, "Smart wheelchairs: A literature review." *J. Rehab. Research And Dev.*, vol. 42, no. 4, pp. 423–436, 2005.

[25] B. P. Gerkey, R. T. Vaughan, and A. Howard, "The player/stage project: Tools for multi-robot and distributed sensor systems," in *In Proceedings of the 11th International Conference on Advanced Robotics*, 2003, pp. 317–323.

[26] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, vol. 3, no. 3.2, 2009.

[27] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[28] S. Nikolaidis and J. Shah, "Human-robot cross-training: computational formulation, modeling and evaluation of a human team training strategy," in *Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction*, ser. HRI '13, 2013, pp. 33–40.

[29] A. D. Dragan and S. S. Srinivasa, "Assistive teleoperation for manipulation tasks," in *Proceedings of the 7th annual ACM/IEEE international conference on Human-Robot Interaction*, 2012, pp. 123–124.