2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Probabilistic Inference of Human Capabilities from Passive Observations

Peter Tisnikar, Gerard Canal, and Matteo Leonetti

Abstract— **Modern robots need to adapt to diverse human partners with whom they collaborate. To this end, learning a representation of human capabilities enables the robot to personalize their behaviour to their collaborators across multiple tasks. We propose CApability Modeling from Observations (CAMO), a model-based estimation algorithm, in which human capabilities that parameterize a given model are inferred from observations of the human behaviour on known collaborative tasks. We apply the method to joint limit learning in order to predict future trajectories of a 7-DOF manipulator arm. Furthermore, we show that CAMO can be used as a sub-task assignment routine in a simulated human–robot collaboration scenario, allowing the robot to adapt its task allocation to perform tasks that the person is not able to do.**

I. Introduction

Roboticists have long aspired to create autonomous intelligent robots capable of working in unstructured environments shared with humans, such as homes, offices, or hospitals. These environments host diverse populations with people of varying capabilities. These characteristics result in different strategies people have when faced with solving a shared task with a robot partner. It is therefore necessary for robots to take into account the person with whom they are collaborating, and adapt to their characteristics.

Human-aware models have made their way into algorithms that ensure smooth handovers [1], [2], successful collaboration in a shared workspace [3], [4], as well as adaptive scheduling of tasks [5], [6], [7]. However, most of the capability representations either lack task generalization, as they focus on individual goals, or are representative only for the "typical" human, as they are calculated from aggregated data of able-bodied individuals.

We propose a task-independent model-based method to adapt to the characteristics of a specific human collaborator, as the robot carries out several tasks with them. We assume the existence of a model of the human collaborator, parameterized by their capabilities. The model can be used to compute collaborative plans when instantiated with a vector of parameters characterizing a particular person. We introduce CApability Modeling from Observations (CAMO), an algorithm to continually update a distribution over the parameter vector, so that the robot can estimate the person's characteristics from the observed trajectories and refine the estimate over tasks. CAMO uses Approximate Bayesian Computation (ABC) to infer what parameters likely explain the observed trajectory for an individual goal, and the set of solutions is converted to a distribution in the parameter space, which is used to calculate a joint probability of a parameter vector describing all trajectories observed so far. Through these continual updates, the robot adjusts its estimate of the model parameters, that is, the person's capability, to the particular collaborator over time, and can generalize the personalization over unseen tasks.

In this paper, we use CAMO to personalize assistance in a collaborative manipulation task, which is the most prominent form of human–robot collaboration in manufacturing [8]. Within such tasks, we choose to model joint limits, because they influence both trajectories and workspace, and can be affected by a number of conditions that interfere with mobility, including diabetes mellitus [9], radiotherapy [10], and haemophilia [11].

We show that CAMO can learn a representation of the partner's capabilities simply by observing their actions, without a calibration process, and irrespective of the observation space of the trajectories. Lastly, we show that CAMO can be used to learn personalized sub-task allocation in a collaborative block stacking scenario with a person with limited shoulder mobility.

II. Related Work

Human–Robot collaboration is a well-studied problem setting that models many complex interactions between robots and people. In this section, we present three aspects that are closely related to our work: goal inference, task allocation in human–robot teams, and parameter estimation from observations.

A. Goal Inference

Many proposed methods allow the robot to adapt to the human collaborator by inferring the goal of the collaborator through observing their actions in the form of partial trajectories [4], [3], [12], [13], or infer human intent through affordances [14] or eye gaze [15]. Unlike our method, which focuses on the collaborator's capabilities, the aforementioned approaches focus only on inferring the human's goal to avoid collisions with the user or to streamline the collaboration, whilst assuming that both the human and the robot can perform all actions in the collaboration.

This work was supported by UK Research and Innovation (grant number EP/S023356/1), in the UKRI Centre for Doctoral Training in Safe and Trusted Artificial Intelligence (www.safeandtrustedai.org). G. Canal was supported by the RAEng and the Office of the Chief Science Adviser for National Security under the UK IC Postdoctoral Research Fellowship programme.

P. Tisnikar, G. Canal, and M. Leonetti are with the Department of Informatics, Faculty of Natural, Mathematical, and Engineering Sciences, King's College London, WC2R 2LS London, the United Kingdom. {name.surname}@kcl.ac.uk

Other collaboration approaches use inverse reinforcement learning, or inverse optimal control to learn cost functions that allow the robot to predict how a person is going to behave in a collaborative setting [16], [17]. These methods learn the cost function over features such as distance between the links of the two collaborators to avoid collision, trajectory smoothness, and the distance from the rest pose. These features are not learned in a model-based context, and they are not necessarily informative about the person's true capability, but are specific to the particular task at hand.

Vahrenkamp et al. [1] study adaptation in the context of human–robot handovers, and the authors develop a planner that optimises the handover location of a tool based on the person's ability to reach it, and how well they can manipulate the tool at the handover location. This approach is extended in other works [2], [6], [18], [19], where authors develop human-aware motion planners that are able to optimise the interaction between human and robot based on different metrics indicating ergonomic posture, social rules, or preferences. However, these methods all average over the training data to produce "average" cost functions, or movements, and do not differentiate between individuals' capabilities.

The work that is closest to our study is that of Bestick et al. [20], in which the authors learn personalized ergonomic cost functions using active robot queries. We, on the other hand, adopt a purely observational approach, in which the robot does not have the power to induce human demonstrations by its own actions.

B. Task Allocation in Human–Robot Teams

Several approaches have been developed in which allocation of tasks in a collaborative setting depends on the properties of the human collaborator. Nikolaidis et al. [21] identify different collaboration strategies by collecting human–human demonstrations and performing clustering on the trajectories to recover reward functions, which the robot then uses to learn an effective collaboration policy for each human type. However, the method requires access to data of human–human teams, and it produces a discrete set of policies that the robot can use to adapt to new partners. Our method, in contrast, does not assume discrete types of collaborators, but learns continuous parameters that describe the behaviour of the collaborator, and does so for multiple tasks instead of just one.

Another similar approach is that of Görür et al. [22], where the adaptation to the collaboration is modeled as a policy selection problem. However, this approach does not model the capability of the person directly, and is task-specific as the policies do not generalize to unseen tasks.

Other task assignment approaches also take into account the capabilities of the human partner, and their preferences for the way in which they engage in the shared activity. Fiore et al. [23], equip the robot's planning system with reasoning capabilities about the human collaborator's preferences, ability to reach the object, and beliefs in simple tasks such as clearing a table. Lamon et al. [5] develop a set of indices such as agent dexterity, effort, and task complexity, which together act as action costs to assign optimally the tasks between a human and a robot in an industrial setting. Their approach, however, does not individualize these costs between different people. Munzer et al. [24] approach human–robot collaboration as a pure learning problem, where they represent the task through relational MDPs and learn what tasks the robot should do through repeated interactions, learning human preferences for the robot's contribution. Izquierdo-Badiola et. al. [25] frame collaboration with diverse people as a problem of learning appropriate action costs in a shared plan. The robot learns costs that ensure plan success and alignment with preferences of the human collaborator, using a simulator with a human model that links action outcomes to latent cognitive variables such as knowledge, strength, or focus. However, their approach works over predefined types of human collaborators and does not learn the collaborator type from scratch.

The difference in our approach compared to the methods above is that we do not require the knowledge of human capabilities a priori, nor do we learn them in a single task. Instead, we aim to learn representations of capabilities that translate across different tasks and are a property of the collaborator instead of being a property of the task itself.

C. Parameter Estimation

In this paper, we use approaches from Approximate Bayesian Computation (ABC) to find parameters that best describe the observed trajectory. ABC is a method from Bayesian statistics which directly computes the posterior of an unknown likelihood function by replacing it with a simulator. For a more detailed review, we refer the reader to the review of Beaumont [26]. A commonly used approach is rejection sampling, which has found its use in human–robot interaction scenarios [27], [28]. However, these implementations infer parameters that explain the goal (i.e., the person's reaching goal, or a pedestrian's trajectory), whereas we implement ABC to estimate a property of the human collaborator, which can be used to plan robot assistive actions. Sun et al. [29] estimate human capabilities through reinforcement learning. Their approach is model-free and, therefore, also task-dependent.

III. Problem Formulation

We consider an environment with two agents, an agent A and an observer O, who is learning about A. The task of agent A is modeled as an episodic goal-based contextual Markov Decision Process (MDP) $m_\theta(g) = \langle X, U, T(x,u,\theta), R(x,u \mid g,\theta)\rangle$ where X and U are sets of states and actions respectively, $T(x,u,\theta)$ is the transition function, which we assume to be deterministic, $R(x,u \mid g,\theta)$ is the goal and context-conditioned reward function, $g \in X$ is an absorbing goal state, and θ is a vector of latent parameters, or context, which influences the agent's policy. For any goal g, the agent A computes a deterministic policy $\pi(x \mid g,\theta)$. When acting in the environment from an initial state x_0, A produces state trajectories $\tau_{\theta,g} = \langle x_0, \ldots x_l\rangle$, such that $x_l = g$, which are observable by agent O. Agent A maximizes the

cumulative return $G_g = \sum_{i=1}^{H} R(x_i, u_i | g, \theta)$ over a horizon H. Furthermore, O knows the goal g of each observed trajectory.

We assume that the observer has a model $\hat{m}_{\hat{\theta}}(g)$ of the contextual MDP $m_\theta(g)$, which can be used, in conjunction with an estimate $\hat{\theta}$ of the parameter vector θ, to plan for a given goal. The aim of O is to compute the estimate $\hat{\theta}$ that allows it to predict A's trajectories as best as possible, that is, the solution of:

$$argmin_{\hat{\theta}} \sum_{g \in \mathcal{G}} L(\tau_{\theta,g}, \tau_{\hat{\theta},g}), \tag{1}$$

where $\mathcal{G}$ is the set of observed goals, and L is the distance between the trajectories:

$$L(\tau_{\theta,g}, \tau_{\hat{\theta},g}) = \|\tau_{\theta,g} - \tau_{\hat{\theta},g}\|_2 = \sqrt{\sum_{i=1}^{l} \left(\tau_{\theta,g}^{(i)} - \tau_{\hat{\theta},g}^{(i)}\right)^2}. \tag{2}$$

In general, the trajectories $\tau_{\hat{\theta},g}$ are not directly parameterized by $\hat{\theta}$, so the gradient of the trajectory with respect to the parameters cannot be computed. They are the result of planning on the model $\hat{m}_{\hat{\theta}}(g)$, where the planner is a black box. Furthermore, tasks and trajectories are generated continually during the lifetime of the agent, and are not all available at once.

In the next section, we introduce an iterative method based on ABC and probability density estimation, which refines a distribution over $\hat{\theta}$ as more trajectories become available.

IV. Method

In general, many parameter values may result in the same trajectories, that is, Equation 1 has multiple minima. Since this is a continual learning setting, we should not commit to a particular parameter vector at any point, since future tasks may prove incompatible with it. For this reason, we track the solution set across tasks. The set of solutions is continuous and may be formed by disjoint non-convex subsets. To be able to capture the possible representation complexity of the solution set, we use a non-parametric model. Lastly, evaluating a parameter vector implies planning on the model, which is computationally expensive and prohibits exhaustive search. For this reason, we set a budget b of maximum number of parameter vectors to consider at any given task. Algorithm 1 shows the steps of the method, also illustrated in Figure 1.

A. Constructing a Representative Set

At each new task g_n, we perform Approximate Bayesian Computation to obtain a set of points representative of the solution set. We sample b points $\hat{\theta}_i$ from a prior distribution $P(\hat{\theta})$ and generate the corresponding trajectories for the current goal $\tau_{\hat{\theta}_i,g_n}$. The prior distribution encodes any previous knowledge on the value of the parameters, and can be uniform. Out of the sampled points, we retain all those for which:

$$L(\tau_{\theta,g_n}, \tau_{\hat{\theta}_i,g_n}) < \varepsilon,$$

for an $\varepsilon > 0$, and L as defined in Equation 2. The outcome of the procedure RejectionSample is the set $Q_n = \{\hat{\theta} \mid L(\tau_{\theta,g_n}, \tau_{\hat{\theta}_i,g_n}) < \varepsilon.\}$ returned at line 2. Because the trajectories may have different lengths, we compute the loss by performing Dynamic Time Warping (DTW) [30] on the pair of trajectories.

Algorithm 1: CAMO

Data: $C = \langle \tau_{\theta,g_n} \rangle_{n=1}^{N}$, $max = 0$, $P_{All} = \emptyset$, $Q_{All} = \emptyset$
Result: $\hat{\theta}^*$

1 **for** $\tau_{\theta,g_n} \in C$ **do**
2 Q_n = RejectionSample(τ_{θ,g_n});
3 ▷ Rejection sample using the observation;
4 $P_n = \frac{1}{|Q_n|} \sum K_H(Q_n)$;
5 ▷ Fit Gaussian KDE;
6 $P_{All} = P_{All} \cup P_n$;
7 $Q_{All} = Q_{All} \cup Q_n$
8 **for** $\hat{\theta}_i \in Q_{all}$ **do**
9 $P(\hat{\theta}_i) = \prod_{i=1}^{|P_{All}|} P_i(\hat{\theta}_i)$;
10 ▷ Compute joint probability across trajectories;
11 **if** $P(\hat{\theta}) > max$ **then**
12 $max \leftarrow P(\hat{\theta}_i)$;
13 $\hat{\theta}^* \leftarrow \hat{\theta}_i$;
14 **end**
15 **end**
16 **end**
17 **return** $\hat{\theta}^*$

B. Estimating Parameter Density

The selected parameter vectors, that is, those that for g_n induce a trajectory close to the observed one, are used to define a distribution over the parameter space $P(\theta_{g_n})$ which expresses the probability density function of a particular parameter to be able to generate the observed trajectory for g_n. We approximate the parameter density using Kernel Density Estimation (KDE), a non-parametric density estimation technique that computes the likelihood of a given sample based on the number of samples in its local neighbourhood. We employ a multivariate KDE with a Gaussian kernel to approximate the sample density in parameter space of an individual solution set (line 4 in Algorithm 1). We define the probability density function of a particular vector to have generated the observed trajectory as:

$$P(\theta_{g_n}) = \frac{1}{|Q_n|} \sum_{\hat{\theta} \in Q_n} K_H(\theta_{g_n} - \hat{\theta}), \tag{3}$$

where K_H is the Gaussian kernel function:

$$K_H(x) := \det(H)^{-\frac{1}{2}} \cdot 2\pi e^{-\frac{1}{2}(H^{-\frac{1}{2}}x)^T (H^{-\frac{1}{2}}x)}, \tag{4}$$

with diagonal matrix $H = \mathrm{diag}(h)$ whose diagonal entry is the kernel width h, which is a hyperparameter of the density estimation method.

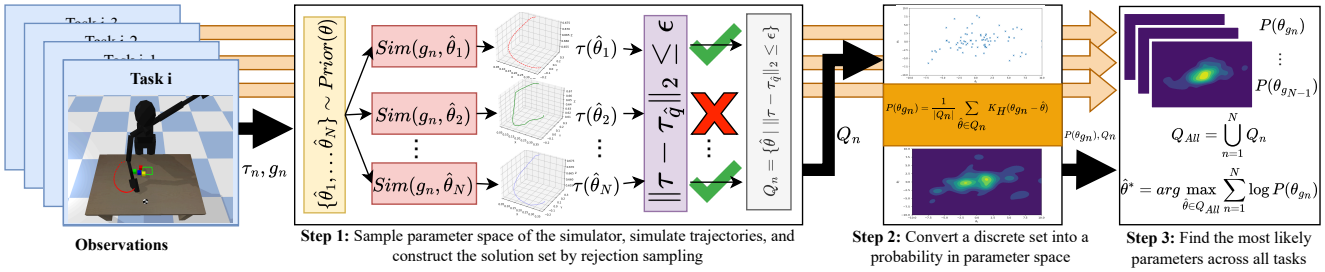

Fig. 1. Overview of CAMO. For every new goal, rejection sampling is performed over the parameter space of the human model, and samples are accepted if they induce a trajectory that is similar to the observation. Then a distribution is defined over the selected samples. The current best estimate is computed by finding the joint probability of a sample across all observed goals.

C. Obtaining Current Best Estimate

After having observed N tasks, we have computed a distribution $P(\theta_{g_n})$ over the parameters for each task (line 4) . As we want to obtain the parameter vector that best explains the tasks we observed so far, we compute the joint distribution over individual tasks:

$$P(\hat{\theta}) = \prod_{n=1}^{N} P(\theta_{g_n}). \tag{5}$$

We compute $\hat{\theta}^*$ from $Q_{all} = \cup_{n=1}^{N} Q_n$, which is the set of all collected parameter vectors (line 7). To obtain the best estimate $\hat{\theta}^*$, we maximize Equation 5 (lines 8 to 14 in Algorithm 1):

$$\hat{\theta}^* = \arg\max_{\hat{\theta}} P(\hat{\theta}) = \arg\max_{\hat{\theta}} \sum_{n=1}^{N} \log P(\theta_{g_n}). \tag{6}$$

This solution is the parameter vector that most likely explains all trajectories observed so far. As a consequence, it approximates the minimiser of Equation 1.

V. Experiments

As discussed in the Introduction, as a representative instance of this problem we consider a collaborative manipulation setting, in which a robot and a person share a common goal. The robot knows the goal that the person is supposed to reach, and intends to learn about their joint limits to be able to predict the person's trajectories across future tasks. The vector of parameters is $\theta = [q_{ll_1}, q_{ul_1}, .., q_{ll_n}, q_{ul_n}]^T$, where q_{ll_i} and q_{ul_i} represent the upper and lower limit for the i-th joint respectively. We consider the following two scenarios: a redundant manipulator with joint limits reaching random goals in 3D space, and a human model with joint range of motion deficiencies in a collaborative block stacking task.[1]

A. Redundant Manipulator

We first study our method in a simulated environment in which the observed agent is a 7-DOF manipulator moving in 3D task space, implemented using the Robotics Toolbox [31].

We sample random configurations within the joint limits of the manipulator and pass them as goals to the controller, generating motion trajectories. We fix the start state for each goal; only the goal state varies across tasks. We collect the trajectories to use them as observations for CAMO.

We draw 2000 parameter vector samples from a uniform prior, bounded between the minimum and maximum joint limits of the manipulator.

We intend to assess experimentally the following properties of our proposed method: how accurately can the observer predict the observed trajectories? How fast does the accuracy improve with the number of observed tasks? In addition to these general properties, we also considered the type of observation specifically pertaining joint limits. We considered state trajectories both in joint and Cartesian space, to assess the level of observation required for this method to be viable in the manipulation context.

In our simulated environments, both joint configurations and end-effector pose are available at all times. However, estimating all joint angles for an observed person from vision is much more difficult. Therefore we ask the question: if an accurate observation of the joint trajectory was available, what impact would it have on the accuracy of the trajectory prediction? In particular, we intend to assess whether sufficient accuracy can be achieved only through end-effector pose observations.

To the best of our knowledge (cf. Section II), there is no other model-based parameter estimation algorithm that learns parameters of the collaborator's model across multiple tasks. As a baseline, we assume that perfect tracking of joint values is available on the manipulator, and that we can obtain minimum and maximum joint values elicited at every task. Using this information, we can determine the maximum and minimum observed joint values across all tasks, and use it as a reasonable estimation of the joint limits of the manipulator. We compare this naïve approach to two variants of our method, one that operates using collected joint space trajectories (henceforth denoted as CAMO-JS), and one that uses Cartesian trajectories and does not assume access to joint trajectories (denoted as CAMO-CS).

The agent executes 10 tasks, producing as many observed trajectories, which we provide to the learner to estimate joint limits. We also sample 20 reachable random goals to construct a test set on which we test the predictive power of our method. We perform an experiment in which we

[1]For reproducibility and access to all parameter values, we provide the simulation code at the following address: `https://github.com/Sensible-Robots/learning-joint-limits`.

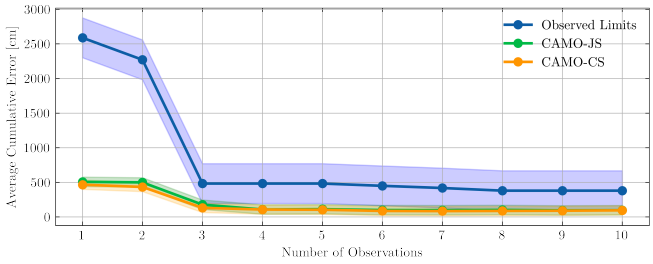

Fig. 2. The cumulative Cartesian Error across 20 unseen trajectories. The lines represent the average across 30 runs, and the shaded areas indicate the 95% confidence interval.

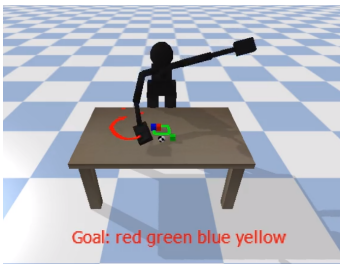

Fig. 4. The Human–Robot Collaboration domain. A human model and a floating robot gripper collaborate in a block stacking task in which the robot has to identify all the blocks the human cannot reach in order to help them to complete the tower.

calculate the error between actual test set trajectories and those predicted after each observation, to investigate the improvement of the estimation. We average these results over 30 runs.

Figure 2 shows that on average, our method is able to learn a better estimate of joint limits that produces lower error than the naïve approach at every task and with lower variance. The estimate converges, in this experiment, within 3 tasks. The performance of the method is similar irrespective of the nature of the observation space, which implies that predictions are equally good, whether based on joints or end-effector pose.

Next, we examine the total error after having observed all tasks. As we can see in Figure 3, the two variations of our method perform comparably, and both are significantly better than the naïve approach. We conclude that CAMO performs better when predicting trajectories for unseen goals compared to the naïve method, as the inferred joint limits produce lower error. Since trajectories have 100 points and the average cumulative error is around 25cm, the average error per trajectory point is 2.5mm, meaning that the method can identify parameters that induce accurate predictions.

B. Human–Robot Collaboration Scenario

We evaluate the utility of our method as a task allocation sub-routine in a collaborative task. We consider a block-stacking scenario with a human model (as seen in Fig. 5). Such block stacking tasks are common benchmarks in Task and Motion Planning [32], as well as a common platform for studying Human–Robot Collaboration.

The domain contains 4 coloured blocks $b = \{$`red`, `green`, `yellow`, `blue`$\}$, which are initialised in random positions at every trial. The goal is a stack of blocks, in a random order for each trial.

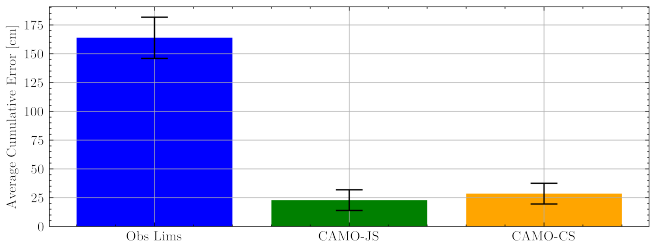

Fig. 3. Mean cumulative loss in joint trajectory space across 20 previously unseen goals, after observing 20 random goals. The bars represent the average error and error bars show 95% confidence interval.

The actions available in the domain are $U = \{$`pick-place-human(b)`, `pick-place-robot(b)`$\}$, where either the person or the robot can manipulate a block and add it to the tower. However, due to the joint limits, the person cannot reach or manipulate all the blocks, which means that the robot needs to recognize this limitation and assign to itself the manipulation of the blocks that the person cannot reach.

We implement the scenario in PyBullet [33], where we instantiate an URDF-based human model generated from collected kinematic data of a real person [34]. We restrict the model to 9 Degrees of Freedom in total — 7 in the right arm, and 2 in the torso. We use a differential kinematics controller as described in [28], which we parameterize with joint limits of the model. The controller is used to produce trajectories given the location of a block the person intends to reach. We make the assumption that only positive examples are available, as the person would not attempt to pick a block they know they cannot reach.

We formulate and analyse the problem of task allocation between the two entities in the domain as a binary classification problem, where for every given block in the domain, the robot needs to decide whether the person is able to reach it or not. The classification labels are therefore the assignments of tasks to either the person or the robot, and the ground truth is produced by the human's true ability.

As around 75% of the labels belong to the majority class of assigning the task to the robot, we measure precision and recall. Precision, in this case, captures all the cases in which the robot erroneously assigned the object to the person even though the person cannot reach it. Recall, on the other hand, describes the instances in which the robot mistakenly assigned the object to itself even when the person can reach it as well. For completeness, we calculate the F1 score, which is the harmonic mean of the two metrics. We contrast our approach with a classifier based on the naïve method introduced in the previous section.

We use a sequence of 10 tasks for training, and 20 unseen tasks for testing. After each observed training task, we test the classifiers on the test set. We use two priors in the

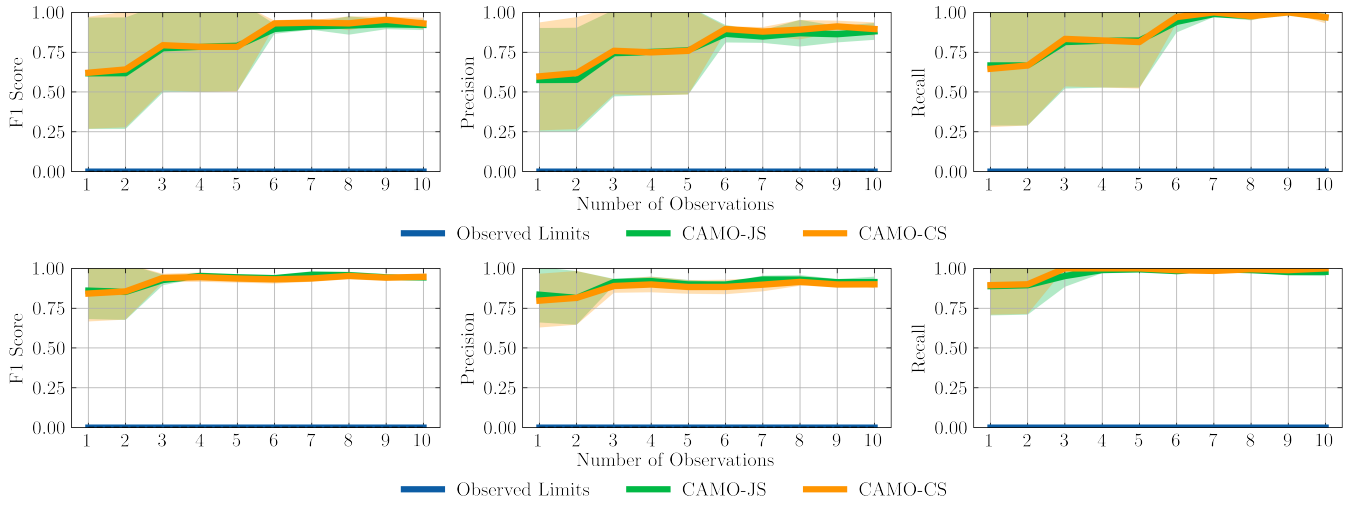

Fig. 5. CAMO correctly assigns a higher percentage of tasks to the human compared to the naive method of just observing the joint limits produced in the task. Top: strong prior, bottom: weak prior. The lines represent an average over 10 runs and the shaded areas represent the 95% Confidence Interval.

experiment: a strong prior in which only the shoulder joint limits and elbow joint limits are allowed to vary and the wrist limits are fixed and known, and a weaker uniform prior. We bound the parameter space to the joint limits of a person with a full range of motion. To ensure sufficient coverage of the sampling space, we employ Latin Hypercube Sampling, and we generate 5000 samples per task. For computational speedup reasons, we only keep at most 500 best samples from each task in our solution set Q_n to calculate the joint probability of these samples being the solution.

As seen in Fig. 5, both versions of CAMO achieve precision of around 85% and near perfect recall in the case of both strong and weak priors. Furthermore, the naïve method always predicts the majority class in all cases.

We therefore conclude that the predictions made through the parameters learned with CAMO can be used by the robot to predict the person's plans in the demonstrated human–robot collaboration scenario.

VI. Discussion

Based on the results of the experiments described in section V, we can conclude that CAMO learns useful representations of the person's ability irrespective of the observation space. This is especially useful as it means that CAMO does not depend on availability of joint tracking to learn a representation that helps it predict the future behaviour of the person based on past observations.

It is important that samples cover a large amount of the parameter space, to find the largest possible set of solutions. One downside of such an approach, however, is that the spacing between samples increases with the increase in dimensions. We expect that, similarly to sample-based black-box optimization, this method would not be effective in high-dimensional parameter estimation.

Kernel density and the threshold at which a sample is accepted into the posterior are important hyperparameters. We determined those empirically, but a fine balance is needed to ensure that enough samples are accepted without inflating the error tolerance (the ϵ parameter) unnecessarily.

CAMO is not limited to estimating joint limits. The formulation is general enough to infer any continuous variable that influences the observable behaviour and for which we have a causal model linking the variable to said behaviour.

Lastly, the parameterized models used in this work are exact, that is, they correspond to the system that generates the trajectories. We could, therefore, focus on the accuracy of the predicted trajectories due only to our estimation method. However, accurately modeling human trajectories is an active research areas, and errors in the model will compound with parameter estimation error in ways that we did not investigate.

VII. Conclusions and Future Work

We presented CAMO, a model-based multi-task method to estimate human parameters in human–robot collaboration, in order for robots to adapt to the particular user they are working with and their capabilities. We assessed the prediction error on stand-alone observations, and integrated the observation in a collaborative manipulation task in a realistic simulator. We showed that using planning on known goals, CAMO enables the observer to estimate parameters more accurately than from direct observation only. Furthermore, the observer can use CAMO to adapt the subtask assignments in a collaborative task with a simulated person with joint range of motion deficiency, adapting to their capabilities. In this work, we considered only stationary parameters. Extending the estimation to tracking changing parameters is an interesting future work, for instance, in scenarios that involve human fatigue estimation.

ACKNOWLEDGMENT

The authors would like to thank C. Christakou for help with preliminary experiments.

REFERENCES

[1] N. Vahrenkamp, H. Arnst, M. Wächter, D. Schiebener, P. Sotiropoulos, M. Kowalik, and T. Asfour, "Workspace analysis for planning human-robot interaction tasks," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, 2016.

[2] E. A. Sisbot and R. Alami, "A human-aware manipulation planner," *IEEE Transactions on Robotics*, vol. 28, no. 5, 2012.

[3] J. Mainprice and D. Berenson, "Human-robot collaborative manipulation planning using early prediction of human motion," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013.

[4] S. Pellegrinelli, H. Admoni, S. Javdani, and S. Srinivasa, "Human-robot shared workspace collaboration via hindsight optimization," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[5] E. Lamon, A. De Franco, L. Peternel, and A. Ajoudani, "A capability-aware role allocation approach to industrial assembly tasks," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, 2019.

[6] E. Merlo, E. Lamon, F. Fusaro, M. Lorenzini, A. Carfì, F. Mastrogiovanni, and A. Ajoudani, "An ergonomic role allocation framework for dynamic human–robot collaborative tasks," *Journal of Manufacturing Systems*, vol. 67, 2023.

[7] S. Zhang, Y. Chen, J. Zhang, and Y. Jia, "Real-time adaptive assembly scheduling in human-multi-robot collaboration according to human capability," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020.

[8] E. Matheson, R. Minto, E. G. Zampieri, M. Faccio, and G. Rosati, "Human–robot collaboration in manufacturing applications: A review," *Robotics*, vol. 8, no. 4, 2019.

[9] K. M. Shah, B. R. Clark, J. B. McGill, C. E. Lang, and M. J. Mueller, "Shoulder limited joint mobility in people with diabetes mellitus," *Clinical Biomechanics*, vol. 30, no. 3, 2015.

[10] L. Blomqvist, B. Stark, N. Engler, and M. Malm, "Evaluation of arm and shoulder mobility and strength after modified radical mastectomy and radiotherapy," *Acta Oncologica*, vol. 43, no. 3, 2004.

[11] J. H. de Groot, S. M. Angulo, C. G. Meskers, H. C. van der Heijden-Maessen, and J. H. Arendzen, "Reduced elbow mobility affects the flexion or extension domain in activities of daily living," *Clinical Biomechanics*, vol. 26, no. 7, 2011.

[12] A. M. Zanchettin and P. Rocco, "Probabilistic inference of human arm reaching target for effective human-robot collaboration," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017.

[13] J. Z.-Y. He, Z. Erickson, D. S. Brown, A. Raghunathan, and A. Dragan, "Learning representations that enable generalization in assistive tasks," in *Conference on Robot Learning*. PMLR, 2023.

[14] H. S. Koppula and A. Saxena, "Anticipating human activities using object affordances for reactive robotic response," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 1, 2015.

[15] C.-M. Huang and B. Mutlu, "Anticipatory robot control for efficient human-robot collaboration," in *2016 11th ACM/IEEE international conference on human-robot interaction (HRI)*. IEEE, 2016.

[16] J. Mainprice, R. Hayne, and D. Berenson, "Predicting human reaching motion in collaborative tasks using inverse optimal control and iterative re-planning," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015.

[17] ——, "Goal set inverse optimal control and iterative replanning for predicting human reaching motions in shared workspaces," *IEEE Transactions on Robotics*, vol. 32, no. 4, 2016.

[18] M. Gharbi, R. Lallement, and R. Alami, "Combining symbolic and geometric planning to synthesize human-aware plans: toward more efficient combined search," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[19] E. Merlo, E. Lamon, F. Fusaro, M. Lorenzini, A. Carfi, F. Mastrogiovanni, and A. Ajoudani, "Dynamic human-robot role allocation based on human ergonomics risk prediction and robot actions adaptation," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022.

[20] A. Bestick, R. Pandya, R. Bajcsy, and A. D. Dragan, "Learning human ergonomic preferences for handovers," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[21] S. Nikolaidis, R. Ramakrishnan, K. Gu, and J. Shah, "Efficient model learning from joint-action demonstrations for human-robot collaborative tasks," in *Proceedings of the tenth annual ACM/IEEE international conference on human-robot interaction*, 2015.

[22] O. C. Görür, B. Rosman, and S. Albayrak, "Anticipatory bayesian policy selection for online adaptation of collaborative robots to unknown human types," in *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, 2019.

[23] M. Fiore, A. Clodic, and R. Alami, "On planning and task achievement modalities for human-robot collaboration," in *Experimental Robotics: The 14th International Symposium on Experimental Robotics*. Springer, 2016.

[24] T. Munzer, M. Toussaint, and M. Lopes, "Efficient behavior learning in human–robot collaboration," *Autonomous Robots*, vol. 42, 2018.

[25] S. Izquierdo-Badiola, G. Alenyà, and C. Rizzo, "Adaptive human-robot collaboration: Evolutionary learning of action costs using an action outcome simulator," in *2023 32nd IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, 2023.

[26] M. A. Beaumont, "Approximate bayesian computation," *Annual review of statistics and its application*, vol. 6, 2019.

[27] J. Felip, N. Ahuja, D. Gómez-Gutiérrez, O. Tickoo, and V. Mansinghka, "Real-time approximate bayesian computation for scene understanding," *arXiv preprint arXiv:1905.13307*, 2019.

[28] J. Felip, D. Gonzalez-Aguirre, and L. Nachman, "Intuitive & efficient human-robot collaboration via real-time approximate bayesian inference," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022.

[29] C. Sun, A. G. Cohn, and M. Leonetti, "Online human capability estimation through reinforcement learning and interaction," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[30] K. Wang and T. Gasser, "Alignment of curves by dynamic time warping," *The annals of Statistics*, vol. 25, no. 3, 1997.

[31] P. Corke and J. Haviland, "Not your grandmother's toolbox–the robotics toolbox reinvented for python," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021.

[32] F. Lagriffoul, N. T. Dantam, C. Garrett, A. Akbari, S. Srivastava, and L. E. Kavraki, "Platform-independent benchmarks for task and motion planning," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, 2018.

[33] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," 2016.

[34] C. Latella, S. Traversaro, D. Ferigo, Y. Tirupachuri, L. Rapetti, F. J. Andrade Chavez, F. Nori, and D. Pucci, "Simultaneous floating-base estimation of human kinematics and joint torques," *Sensors*, vol. 19, no. 12.