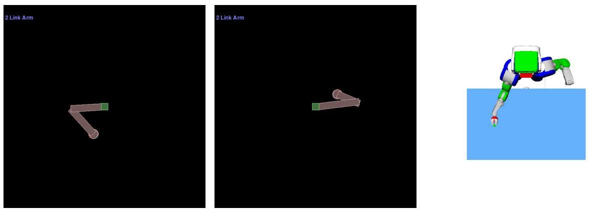
Show Me What You Can Do: Capability Calibration on Reachable Workspace for Human-Robot Collaboration (Supplementary)

Xiaofeng Gao[1], Luyao Yuan[1], Tianmin Shu[2], Hongjing Lu[3], Song-Chun Zhu[1]



(a) Robot A. (b) Robot B. (c) Robot C.

Fig. 1: Robots we use in this paper.

S.I. Implementation Details

Robots. In this paper, we simulated three robots: (i) Robot A is a 2-link arm where each link is of equal length (0.1), (ii) Robot B is a 2-link arm where the length of its first link (0.13) is larger than the length of the second (0.07), (iii) Robot C is a PR2 robot. Simulations are implemented in OpenRave. Robot appearances are displayed in Figure 1.

Hyperparameters. The hyperparameters for REMP are listed in Table I. We use grid search to find parameters that result in the best reachability estimation in simulation.

Random baseline. We implemented the *random* baseline by sampling waypoints in the robot's reachable workspace. From the starting position, the end effector moves to the waypoints one by one to demonstrate its reachability. The number of waypoints we sampled corresponds to the number of trajectories in the *belief* and *static* conditions.

S.II. User study on Reachability Estimation

Prior to the user study described in the main text, we carried out a pre-study to understand how a human user would estimate a robot's reachable workspace given an observed trajectory. We recruited 23 participants (78.3% female, age median: 19.0) from University Subject Pool and they were compensated course credits for their participation. None of the participant had experience with robot manipulations. After arriving at the study room, each participant is first asked to watch several trajectories of a robot reaching a target object. They are asked to draw the estimated reachable workspace of the robot after seeing each trajectory. The trajectories are manually designed, ranging from a direct path to the target to randomly moving in the workspace before reaching the target. To reduce the learning effect, the orders of trajectories are randomized. The example reachability estimation result is shown in Figure 2.

[1] Center for Vision, Cognition, Learning, and Autonomy, UCLA. Emails: {xfgao, yuanluyao}@ucla.edu, sczhu@stat.ucla.edu.
[2] Massachusetts Institute of Technology. Email: tshu@mit.edu.
[3] Department of Psychology, UCLA. Email: hongjing@ucla.edu.

TABLE I: REMP hyperparameters

	α	β	γ	N	λ
PR2	0.01	2.0	0.5	60	1.0
2link	0.02	2.0	0.5	70	1.0

Note that our pre-study is an in-person study, as we require users to provide accurate reachability estimation by interacting with the user interface. We used Amazon Mechanical Turk (AMT) for our main study. We acknowledge that there are some limitations using AMT, e.g. we are unable to work with the 3D reachability of robots because it is hard to demonstrate and communicate 3D positions online. Nevertheless, we argue that AMT is a decent choice for evaluating our framework. First, our goal is to help novice users better understand the robot's reachability, and AMT allows us to recruit a large number of subjects with various backgrounds, most of which lack knowledge of robotics. Second, since our evaluation does not require physical manipulation from humans, there is no significant difference between AMT and in person studies. Finally, in person studies are nearly impossible to carry out due to COVID-19 at the moment. We'd like to conduct additional in-person study if conditions permit in our future work.

S.III. Simulated Human Belief Update Model

We use a Gibbs distribution to model human's belief conditioned on observed trajectories. Notice that for the human belief update model we use in the simulation, instead of treating each voxel independently, we model the distribution of the entire reachability map, so that the proximity among voxels can be considered. We added local constraints to encourage voxels close to each other to have similar reachabilities. We denote reachability as $f : \mathscr{X}_{ws} \longrightarrow \{0,1\}$.

$$b_h(f|\xi_{1:t}) \propto \exp\Big(\sum_{x\in\mathscr{X}_{ws}} - \sum_{y\in N_x} \big(f(x)-f(y)\big)^2/\sigma_N - f(x)\sum_{i=1}^{t} \eta^{t-i} h(x,\xi_i)\Big), \tag{S.1}$$

where

$$h(x,\xi_i) = \begin{cases} d(x,\xi_i)/\sigma_d, & d(x,\xi_i) > \tau_d \\ C, & \text{otherwise} \end{cases} \tag{S.2}$$

Equation (S.1) accommodates two factors into human's belief about the reachability. The first is that neighboring voxels are more likely to have the same reachability. The second is that voxel closer to an observed trajectory are more likely to be reachable. Here N_x represents the neighbor voxels

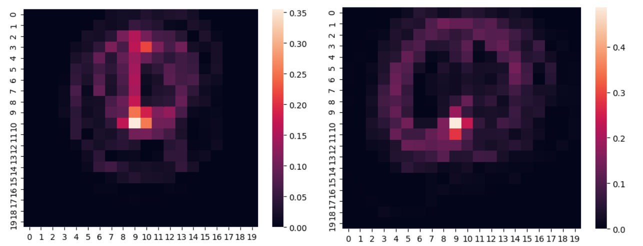


Fig. 2: Heatmap visualization of the reachability estimation provided by users in the pre-study. Higher value indicates the area is more likely to be the boundary of the robot's reachable workspace.

of x, σ_N and σ_d are two coefficients control the relative scale of the energy contributed by the two factors. τ_d is a threshold for the distance. All voxels fall within τ_d from ξ_i will have the same positive energy C from ξ_i. η is a factor simulating the forgetting effect of the user. The earlier the trajectory first presented to the user, the smaller its effect has to the current belief. The distance function is defined in Equation (4) in the main text. Values of these hyperparameters are listed in Table II. We used grid-search to find the set that best fit the human data from our pre-study.

S.IV. Collaborative Table Cleaning Task

We formulate the table clearing task in Section IV-A as a sequential human-robot collaboration using MDP. Formally, an MDP is represented as a tuple $\langle S, \mathscr{A}_h, \mathscr{A}_r, T, R\rangle$, where S is the set of environmental states, $\mathscr{A}_h$ and $\mathscr{A}_r$ are the sets of actions available to the robot and the human respectively. The transition function $T(s^{t+1}|s^t, a_h^t, a_r^t)$ models the probability of transitioning to state s^{t+1} after the agents taking action $a_{h/r}^t \in \mathscr{A}_{h/r}$ in state s^t. At each time step, the team receives a reward $R(s^t, a_h^t, a_r^t)$ based on the world state $s^t \in S$ and their actions. In particular, we define the collaborative table clearing task with MDP as the following.

State space and action space. Given a workspace $\mathscr{X}_{ws}$, we define $s \in S = \{0,1\}^{|\mathscr{X}_{ws}|}$. That is, for any position $x \in \mathscr{X}_{ws}$, $s_x = 1$ if there is an object to be cleared at x, otherwise, $s_x = 0$. The human action space is defined as $\mathscr{A}_h = \mathscr{X}_{ws}$, where $a \in \mathscr{A}_h$ means collecting the object at position a. The robot action space, $\mathscr{A}_r = \mathscr{X}_{ws} \bigcup \{\Xi\}$, is the same as the human's, but with an additional action Ξ to represent that the robot would do nothing when there is no reachable object in the workspace.

In reality, it is common to assume that both the human and the robot only reach a position when there is an object there. Thus, at state s^t, we have $s_h^t = \{x \in \mathscr{X}_{ws} | s_x^t = 1\}$ and $s_r^t = \{x \in \mathscr{X}_{rs} | s_x^t = 1\}$ as the set of positions where the human or the robot will possibly pick from. Let A_r^t be the possible actions available to the robot at time t. Then, the robot action space is picking up one reachable object (when there is any):

$$A_r^t = \begin{cases} s_r^t, & s_r^t \neq \varnothing \\ \{\Xi\}, & \text{otherwise} \end{cases}. \tag{S.3}$$

Transition model. We define a deterministic transition model as:

TABLE II: Simulated belief update model hyperparameters

	σ_N	η	σ_d	τ_d	C
PR2	1	0.95	0.01	0.1	20
2link	1	0.95	0.03	0.02	20

$$T(s^{t+1}|s^t, a_h^t, a_r^t) = \prod_{\substack{x \in \mathscr{X}_{ws}, \\ x \neq a_h^t, a_r^t}} \mathbf{1}(s_x^{t+1} = s_x^t) \prod_{\substack{a \in \{a_h^t, a_r^t\}, \\ a \neq \Xi}} \mathbf{1}(s_a^{t+1} = 0), \tag{S.4}$$

where $\mathbf{1}$ is the indicator function. The first product means the untouched positions remains the same between s^{t+1} and s^t. The second product means the human or robot action of collecting the object would change the state at corresponding positions.

Reward. We define the reward function as

$$R(s^t, a_h^t, a_r^t) = \lambda(|s_h^t| - |s_h^{t+1}|) - c, \tag{S.5}$$

where the cardinality difference of s_h^t is the number of objects picked up, λ is the reward for each collected object and c is the time penalty. The task is completed when $s_h^t = \varnothing$, i.e. when all objects have been collected. In our simulation experiments, we set $\lambda = 1.5, c = 2$.

Human Policy. We use a Boltzmann noisily-rational decision model [32], [33] based on a Q value, assuming the human is more likely to help the robot with its unreachable objects:

$$\pi_h(a|s^t) \sim \frac{\exp\left(k\mathrm{E}_{y\sim b_h^t(a)}[Q_y]\right)}{\sum_{a' \in s_h^t} \exp\left(k\mathrm{E}_{y\sim b_h^t(a')}[Q_y]\right)}. \tag{S.6}$$

Here we slightly abuse the notation and use a to denote both the action and the position in the workspace from which the object is collected by performing a. The expectation is calculated w.r.t. the human belief $b_h(a)$ that position a is reachable by the robot. $y = (0)1$ means picking up (un)reachable objects. We set $Q_0 > Q_1$ to represent human's tendency to help the robot with unreachable objects. We don't discriminate distinct objects and use a constant Q_0 and Q_1 for all the objects. $k \geq 0$ controls the extent of rationality, with $k \to \infty$ approximates to fully rationality and $k = 0$ is random policy. We set $Q_0 = 1, Q_1 = -1$ and $k = 1$ in simulations.

Robot policy. Since we want to emphasize the effect of the calibration, we use a simple uniform robot policy in the simulation, i.e. it would randomly pick up objects it can reach, and do nothing if no objects are reachable. Hence, we can have the robot policy:

$$\pi_r(a|s^t) \sim U(A_r^t) \tag{S.7}$$

where U stands for a uniform distribution.

Experiment details. We ran simulations using *belief*, *static* and *random* conditions. To evaluate the collaboration performance, we run 5000 games with objects randomly placed in the workspace. In every game, exactly half of the objects are reachable by the robot. As there is stochasticity in the *random* baseline, we simulated the belief of 100 users after each has observed the specified number of trajectories and calculated their mean task performance.