A Predictive Autonomous Decision Aid for Calibrating Human-Autonomy Reliance in Multi-Agent Task Assignment

Larkin Heintzman and Ryan K. Williams

Abstract—In this work, we develop a game-theoretic modeling of the interaction between a human operator and an autonomous decision aid when they collaborate in a multi-agent task allocation setting. In this setting, we propose a decision aid that is designed to calibrate the operator's reliance on the aid through a sequence of interactions to improve overall human-autonomy team performance. The autonomous decision aid employs a long short-term memory (LSTM) neural network for human action prediction and a Bayesian parameter filtering method to improve future interactions, resulting in an aid that can adapt to the dynamics of human reliance. The proposed method is then tested against a large set of simulated human operators from the choice prediction competition (CPC18) data set, and shown to significantly improve human-autonomy interactions when compared to a myopic decision aid that only suggests predicted human actions without an understanding of reliance.

I. INTRODUCTION

Recently there has been much attention given to how machines can better interact with humans [1], [2]. Indeed the idea that reliance on autonomy must be calibrated is not new [3], [4], however with an increasingly automated world there has been more focus given to how reliance can be affected by the machine itself [5], [6], as well as how to better link human and machine decision making [7]. As detailed in the seminal work [3], using automation to assist with a task can result in sub-optimal outcomes if the operator is either under *or* over weighting the effectiveness of the automation. For our running example in this paper, consider an *autonomous* multi-agent search and rescue (SAR) mission [8], [9] where the goal is to locate a lost person by searching areas of land (referred to as sectors) and a human mission operator is using a decision aid to help assign tasks to each searching agent (unmanned aerial vehicles (UAVs) or human searchers). A decision aid in a SAR context can certainly take many forms, such as computing likely lost person trajectories [10], likelihood of survival given environment [11], determining ideal search tasking/plans [9], [12], [13], and so on. The risk involved in such a scenario is quite high, thus the decision aid and operator would need to be calibrated for such an environment. If the operator were to under weight the decision aid's information then critical information may be being ignored, mitigating the effectiveness of the multi-agent team. Similarly, if the operator were *over* weighting the aid's information then the indispensable first-hand experience of SAR professionals may be being inhibited.

Larkin Heintzman and Ryan K. Williams are with the Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA USA, E-mail: {*hlarkin3, rywilli1*}@*vt.edu*.
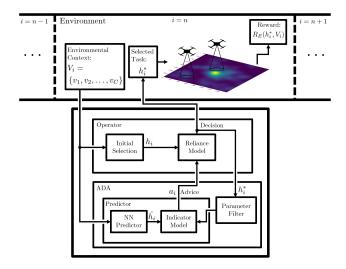
Fig. 1. Showing a simplified overall system diagram for the proposed work.

As detailed in [3], no automated system is free from faults thus the danger of an operator becoming complacent should be considered. Readers are referred to [10] and [14] for additional information on state-of-the-art SAR operations.

In this work, we consider a game-theoretic modeling of the interaction between a human operator and an autonomous decision aid (ADA) in a multi-agent task allocation setting. In this setting, we assume the operator has ultimate control over the final tasks selected, however it is the goal of the aid to correctly calibrate the operator's reliance upon the aid and to improve long-term team performance. In order to accomplish this goal, an ADA must consider several pieces of information related to the operator's usage of the aid. Namely, the decision aid must consider both what *task* the operator may select (action or selection prediction), as well as the operator's likely preference towards the aid itself (reliance indication). We approach this problem while assuming a generic multi-agent task allocation scenario, with the SAR context as inspiration. To the best of the authors' knowledge, our design of a predictive decision aid for human-autonomy interactions in a multi-agent task allocation context, with the reliance model used therein, is the first of it's kind.

Several recent works have considered decision aid design as it relates to trust/reliance of multi-agent systems [15], [16] as well as active reliance calibration [2], [6]. In [15], the authors investigate human trust dynamics when interacting with a semi-autonomous aerial multi-agent swarm carrying out various exploration tasks. An earlier work [16] also considers a similar problem, though with underwater vehicles, and a

simple switching control method. In [15], the autonomy and human operator are working in concert to drive the swarm to various locations in an unknown environment, while the operator uses a sliding scale to give trust evaluations. Similar to our work, [15] uses an adaptive predictor to estimate trust dynamics which is updated as new information becomes available. However, the predictor has input requirements specific to swarm dynamics, such as agent heading variance and convex hull area, where as we design a more general LSTM-based predictive aid that requires only prior interaction data.

In [2], the authors propose to adjust a human operator's trust via so-called trust calibration cues (TCCs), which are designed to notify the operator of under/over reliance during a simulated UAV piloting task. The main focus of [2] is in the design of a trust calibration AI which controls the frequency and type of calibration cues given to the operator. We differ from this work in our problem setting of discrete sequential interactions between operator and aid, and our purposeful limiting of aid suggestions.

Also dealing in multi-agent trust calibration, the authors of [5] consider interaction between a user and a robot working together on a table-clearing task. The robot employs a trust POMDP to inform its decisions with a goal of improving team performance in the long term. The human user exists in a supervisorial role and only intervenes when they feel the robot may not succeed in its chosen task, which is then used to inform the trust POMDP model. The resultant robot behaviors primarily attempted to build trust, though interestingly the robot would sometimes intentionally fail to correct an inappropriate level of trust. However, whereas [5] requires a numerical evaluation of trust directly from each user, we seek to create a reliance calibration method that does not require such evaluations. Also we pose the interaction as both the human and ADA having input to cooperative problem, in our case multi-agent task allocation.

II. PRELIMINARIES AND PROBLEM FORMULATION

A. Game Structure

We model the interaction between the human operator and the autonomy (ADA) in a game-theoretic sense. Taking nomenclature from the field of game theory, the interaction is modeled as a two player sequential imperfect information game. The result of an interaction between human and autonomy is a task for a single agent $s \in S_i$, where S_i is the task set available to the multi-agent team at iteration i, to execute given some reward associated with the selection. In this work, we will consider tasks related to our SAR example, however it is important to point out the generality of a task allocation model. Indeed, the allocation of tasks to multi-agent teams can be mapped to a wealth of both theoretical efforts in multi-agent systems [17], [18], and to application-oriented domains such as target tracking [19]–[21], agricultural monitoring [22]–[25], etc.

Upon selecting a task, s, an additive reward is gained based on $R_E(s, V_i) \in \mathbb{R}$ which is a stochastic function of the selected task and the current environment state. Here, $V_i \in \mathbb{R}^L$ is an environmental context vector. Going back to our running multi-agent SAR example, the context vector would represent the state of the search such as lost person location probabilities [14], and the reward would represent information gain from executing the selected task potentially affecting later decisions. The game is repeated for a known number of iterations, K.

The order of interaction between operator and ADA proceeds as follows:

- The operator decides which task to select without ADA input, label this task h_i.
- The ADA has the opportunity to make a suggestion of a particular task to select, potentially different from the task previously selected by the operator, label the suggestion a_i.
- The human operator considers the given suggestion and decides whether to keep their original selection or to switch and agree with the ADA, resulting in task h_i^*.
- The selected task is carried out according to the operator's decision and reward $R_E(h_i^*, V_i)$ is gained.

For a visual model of the proposed work, consider Figure 1 where the process begins with the context vector V_i which informs both the predictor (Section III-B) as well as the operator. Once the initial selection, h_i, and prediction, $\hat{h}_i$, are made, the indicator model (Section III-D) is used to determine the suggestion provided, a_i, which is then returned to the operator. The final task, h_i^*, is selected based on the reliance decision of the operator which in turn is returned to the parameter filter to improve the indicator model.

B. Operator Reliance Model

We model the operator as using a stochastic reliance model that determines ADA reliance during an interaction. Reliance upon, or *trust of*, an autonomous aid can change quickly and responds to certain conditions, thus the reliance model used in this work aims to capture the relevant behaviors [26], [27]. We base our operator reliance model on the well known decision field theory (DFT) models from [4], [27]. The definition of *preference*, which directly determines reliance, is given as:

$$P(n) = (1 - s)P(n - 1) + sB_C(n) + \epsilon(n) \qquad (1)$$

where $P(n) \in \mathbb{R}$ is the reliance value at time step n, $s \in [0, 1]$ is the reliance inertia, and ϵ is a sequence of i.i.d zero-mean Gaussian random variables as noise. The *belief of autonomy capability*, $B_C(n) \in \mathbb{R}$ is derived as:

$$B_C(n) =$$
$$\begin{cases} B_C(n-1) + b_1\left(C(n-1) - B_C(n-1)\right), & \text{if } I_C = 1 \\ B_C(n-1) + b_0(B_{C_{\text{ini}}} - B_C(n-1)), & \text{if } I_C = 0 \end{cases}$$
$$(2)$$

where $I_C \in {0, 1}$ defines different autonomy types by describing when capability information is available to the operator, $b_1 \in [0, 1]$ describes the system interface transparency, and $b_0 \in [0, 1]$ describes to what degree the operator's initial belief

matters when system capability information is not available (i.e. when $I_C = 0$). Lastly, $C(n) \in \mathbb{R}^{\geq 0}$, and $C_{\text{ini}} \in \mathbb{R}^{\geq 0}$, are the true capability of the autonomy given the task, and its initial value, respectively. The value of $C(n)$ is defined in a later section. In this work, we will be focusing on situations with $I_C = 1$, meaning that the operator has access to the aid's capability information at all time steps (e.g., through some form of interface).

Remark 1. One may incorrectly assume that to increase the number of reliance decisions made by a human operator, we need only to artificially increase the capability value provided to the reliance model. However, doing so would certainly violate the assumptions and reasoning behind the chosen reliance model. As we design for *appropriate reliance*, modifying the true capability information of the autonomy would undermine that goal.

We now begin to extend existing work by modifying the reliance model given in (2) to include an *agreement* bias in the autonomy capability belief portion of the model. While this particular modification is certainly novel, the idea that agreement with an autonomous aid can sway human opinions is known [6], [28]. This modification models situations wherein the human operator's opinion of the aid is swayed by the ADA confirming the operator's initial selection. The agreement-adjusted belief is as follows:

$$
\begin{aligned}
B_C(n) = {} & B_C(n-1) + b_1 \left(C(n-1) - B_C(n-1) \right) \\
& + A_C b_2 \left(1 - B_C(n-1) \right)
\end{aligned} \tag{3}
$$

where $b_2 \in [0, 1]$ describes the degree to which aid agreement affects the operator's belief in autonomy capability, and $A_C \in \{-1, 1\}$ indicates whether the operator and ADA are in agreement ($A_C = -1$ if in disagreement). The agreement adjusted model in (3) remains based on DFT and incorporates the agreement bias into the operator's belief of capability, in keeping with the rest of the reliance model.

Following [27], preference is converted to a reliance state $d \in \{0, 1\}$, where $d = 1$ indicates the operator is relying upon the ADA to select a task and $d = 0$ indicates the operator is selecting a task manually, by comparing the preference value with a threshold value $\theta \in \mathbb{R}^+$. Specifically, if $P(n) < \theta$ at time step n then d is set to 0 and if $P(n) \geq \theta$ then d is set to 1. In this way we convert an internal continuous preference value to a concrete reliance decision by the operator, and d can then be used later to inform the indicator model.

Note that the reliance model described above may not apply to all operators. As shown by [4], [26], there is significant variance in how an individual interacts with a decision aid that depends on many personal characteristics. As such the parameters (see Table I) are assumed to be randomly selected from a corresponding distribution. Further, not all possible realizations from the parameter set would result in a reasonable reliance model, however given the relatively tight ranges mentioned in [4] we can assume that the chosen parameter distributions generate a set of reliance models that is a super

TABLE I
VARIABLE DESCRIPTIONS

$b_1 \in [0, 1]$	System interface transparency weighting
$b_2 \in [0, 1]$	ADA agreement weighting
$s \in [0, 1]$	Controls reliance inertia
$C(n) \in \mathbb{R}^{\geq 0}$	True ADA capability
$I_C \in \{0, 1\}$	Capability information indicator
$A_C \in \{-1, 1\}$	Operator agreement indicator
$B_C(n) \in \mathbb{R}$	Operator belief over capability
$P(n) \in \mathbb{R}$	Operator preference value
$\theta \in \mathbb{R}^{\geq 0}$	Operator reliance threshold
$d \in \{0, 1\}$	Operator reliance state
S_i	Task set available at time step i
$h_i \in S_i$	Initial task selection
$a_i \in S_i$	Decision aid task suggestion/advice
$a_{\text{opt}} \in S_i$	Optimal task in expectation
$h_i^* \in S_i$	Final task selection
$K \in \mathbb{R}^{\geq 0}$	Total number of time steps
$V_i \in \mathbb{R}^L$	Environmental context vector at time step i
$R_E(\cdot) \in \mathbb{R}$	Reward function, context dependent
$P_{\text{ind}}(n) \in \mathbb{R}$	Indicator model preference value
$d_{\text{ind}} \in \{0, 1\}$	Indicator model reliance state

set of realistic human reliance models. For completeness, all model parameters are described and defined in Table I.

C. Problem Statement

Problem 1. Given a human operator, applying a reliance model $P(n)$ with unknown stochastic parameters, interacting with an autonomous decision aid which results in a selected task $h_i^* \in S_i$, an environmental context vector V_i, multi-agent task set S_i, and a reward function $R_E(\cdot)$, approximately optimize the total reward received over K iterations:

$$
S^* = \underset{a_i \in S_i}{\arg\max} \sum_{i=0}^{K} R_E \left(h_i^*, V_i \right) \tag{4}
$$

where h_i^* is selected subject to the operator reliance model $P(n)$, and $R_E(h_i^*, V_i)$ is a reward function based on the selected task and current environment state as described by the context vector. The result of solving (4) is a set of suggestions, S^*, for each time step to approximately optimize total reward received. Note that the ADA does *not* have direct control over the final task selected, h_i^*, thus selecting what task to suggest is the only method of control.

III. DECISION AID STRUCTURE

Here we detail the structure of the ADA and all of its component parts, which references the system flow diagram in Figure 1 as well as the game structure discussed in Section II-A. The ADA can begin once the operator has made an initial selection, h_i, which in our case is derived from the 2018 choice prediction competition (CPC18) data set [29] (detailed in the next subsection). The next concern is whether the operator will be relying upon the ADA or not, which is the purpose of the indicator model. Given h_i, the indicator model determines which suggestion the ADA makes, if reliance is

indicated then the numerically optimal task is suggested, if reliance is *not* indicated then the task predictor is used to make the suggestion. Using the task predictor at the specified times allows the ADA to take advantage of the agreement bias inherent in (3) to improve the overall reward received over K iterations. In our considered problem, improvement is certainly possible as the human operators often do not select the optimal task due a variety of factors, such as the reward received in the previous trial biasing their reasoning [29].

Remark 2. Depending upon the application, attempting to match the operator's selection at every iteration may not always be the best path forward for an ADA. Indeed if the operator were using the ADA in an unsafe fashion, it may be advantageous to attempt to *decrease* reliance where appropriate. While our formulation certainly allows this functionality, due to limitations of the selected data set and human testing requirements, we leave it to future work.

A. Human Decision-Making Data Set

To both validate and train our predictor model we require a source of human decision data under some form of risk. To this end, consider the 2018 choice prediction competition (CPC18) data set [29], which tabulates data from a set of participants, sourced from Amazon Mechanical Turk, making sequential selection decisions with stochastic rewards based on the selection. This data set is uniquely suited to our problem due to its sequential nature, a rare property in human decision-making data sets, which allows us to implement the predictor portion of the ADA. Participants in the CPC18 data set played a set of 30 *games*, each game consisting of 25 sequential decisions called trials. Each game consists of two gambles, each with their own reward distribution, where the participant was asked to make a selection between the two. After each trial the participants were presented with the reward gained/lost as well as the foregone reward. Each participant played 30 different games drawn from a population of 270 different paired distributions. A total of 926 individuals participated, which generated ≈ 0.694M data points of human decision-making under reward-based risk.

As an example of the contents of the CPC18 data set, a single game could be displayed as option "A" having a 25% chance of generating 3 reward and a 75% chance of generating 0 reward, and option "B" having a 20% chance of generating 4 reward and a 80% chance of generating 0. Typically, in the CPC18 data set option "B" is the more risky of the two. Readers are referred to [29], [30] for more details on the data set and the construction of individual games.

To map this data set to our context of multi-agent task allocation, we first assume that each option represents a task or set of tasks in S_i. When the participant makes a selection we can interpret that the operator assigning a task to a single agent, and the reward generated is a function of the selected task. Going back to our running SAR example, each option in the data set might correspond to a method of search. One method covering a large area of land quickly, at a high risk of
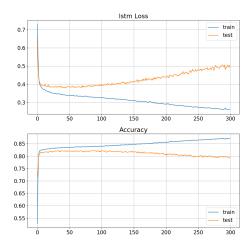

Fig. 2. Showing the training and validation curves of predictor.

failing to locate a target, and the other method being a slower more methodical search, with a low risk of failing to locate a target.

B. Task Predictor

Given this data set, we can design and build a neural network to predict the next selection a human is likely to pick given the previous $k \in \mathbb{N}^+$ selections as well as a context vector. In our case, the context vector contains the game information from the data set, with rewards normalized to be within the interval $[0, 1]$. We opted to use an LSTM-based network with 128 recurrent neurons per layer, 4 hidden layers, and a single dense output layer with a Softmax activation. LSTM layers were used, with an input size of k, as they are designed for use with sequence-to-sequence prediction problems [31]. The network was trained using randomized sequences drawn from CPC18, where the target was chosen to be the next selection by the participant.

The training was run for 75 epochs, as this was seen to be the point at which performance stopped increasing and over-fitting began, see Figure 2 for illustration. After training the prediction accuracy in the validation set is $\approx 83\%$. Note that the prediction problem considered is quite difficult due to the natural variance between participants and their personal risk analysis, though well-suited to machine learning as creating a closed form model to predict human selections would be challenging. To the authors' knowledge this is the first time a sequential action predictor of this kind has been applied to the CPC18 data set and integrated into a human-autonomy teaming context.

C. Autonomous Capability

We can calculate the capability of the autonomy, given the context of the CPC18 data set, as the probability of selecting the gamble that generates more reward overall, see Section III-A for details. Calculating capability is necessary to correctly implement the operator reliance model, this method also serves to determine the theoretical optimal task to select

regardless of any interaction constraints. Given two binomial gamble reward random variables, g_A and g_B, the capability of the autonomy, as we have posed it, is given by:

$$Pr(g_A < g_B) = \sum_{i=0}^{n_c} \sum_{j=0}^{i} P(g_B = i) P(g_A = j)$$

$$C(n) = Pr\left(g_A > \left\lceil \frac{n_c(L_B - L_A) + g_B(H_B - L_B)}{H_A - L_A} \right\rceil\right) \quad (5)$$

where $g_A \sim \mathcal{B}(n_c, p_A)$ and $g_B \sim \mathcal{B}(n_c, p_B)$, n_c is the number of selections left before the end of the game ($n_c = 25 - n$ in CPC18), H_A, L_A, H_B, and L_B are the high and low, rewards from selecting option A and B respectively. We use $\mathcal{B}(n, p)$ to indicate a binomial distribution with n samples and a probability p. The parameters controlling reward distributions form our environmental context vector V_i, here the size of the context vector would be $L = 6$. Here we are assuming, without loss of generality, that option A is the option more likely to generate maximum reward. If we evaluate $C(n)$ and find that it is < 0.5 we can simply take the opposite option and let g_B be the maximum expected reward option. Let this optimal task be a_{opt}. Note that this formulation of capability is specific to the task options in the chosen data set, but could represent a variety of multi-agent task allocation scenarios.

D. Indicator Model

Certainly we do not have access to each operator's internal reliance model, thus we require a method of determining whether or not the operator is likely to take the ADA's suggestion in any given trial. With this capability, we can modulate suggestions in response to operator actions to improve overall team performance. Note that estimating whether the operator will rely upon the ADA is different from *predicting* which task the operator would select.

We begin by assuming we have been given a *disturbed* version of the parameters used in the reliance model from (1). This disturbed model can then be used as an indicator model for actual operator reliance decisions, we label it as $P_{\text{ind}}(n)$ along with its reliance state d_{ind}. To use the indicator model, we must provide the same inputs as for the real reliance model, such as capability and agreement, then we can use information from operator interactions to update the parameters. We use approximate Bayesian computation (ABC) [32] to update indicator model parameters as new interaction data becomes available. The mechanism of ABC is to compute a large set of realizations from a randomized model and compare it to observed data, through a set of summary statistics[1] and a Euclidean distance metric. Each realization that is within some threshold of the observed data is added to a set of accepted samples. The set of parameters used to generate each accepted sample become the priors for the parameters used to generate observed data.

The main advantage to using ABC, as opposed to for example a direct Bayesian or other common filtering paradigm,

ABC samples	10^4 via rejection
ABC batch size	10^5
ABC threshold	0.5
Reset interval	30 iterations
b_1 distribution	$\mathcal{U}(0.01, 0.04)$
b_2 distribution	$\mathcal{U}(0.01, 0.04)$
s distribution	$\mathcal{U}(0.10, 0.80)$
θ distribution	$\mathcal{U}(0.50, 0.20)$

is that we retain freedom over all possible reliance models without the need to derive likelihoods based on the specific model in use. In addition, ABC provides convenient methods to update parameters iteratively as new data is observed but without recomputing large amounts of data. Readers are referred to [32], [33] for more detailed information on ABC.

E. Update Intervals

Due to the fact that more interaction data becomes available as trials pass, we must periodically update the indicator model's parameters using ABC to increase accuracy. The interval to wait is certainly application specific, however in this work we use a period of 10 complete game iterations, or 250 individual trials, between updates. Selecting the update interval is a trade off between indicator model accuracy and computation time with diminishing returns for smaller values due to the lack of new observed data.

In addition, as previously mentioned, a large set of participants were used to obtain the data. Specifically in CPC18, every participant played 30 game iterations. To reflect this fact in simulation, we reinitialize the reliance model every 30 games with new realizations of stochastic parameters to simulate a new operator arriving. Of course refreshing the operator's parameters also means that we must reset the indicator model accordingly, otherwise the indicator model would be using inaccurate parameters to inform suggestions. The complete statement of the decision aid algorithm is given in Algorithm 1.

IV. SIMULATIONS

A. Implementation

The simulation pipeline was implemented in Python 3.7.7 and makes use of several common open source packages such as NumPy and PyTorch, the complete codebase used to generate results can be found here[2]. Prior to simulation, the task predictor was trained on a portion of the CPC18 data set which resulted in a prediction accuracy of $\approx 80\%$ in the validation set. The training set is kept entirely separate from the data used in the simulation pipeline to prevent bias in the predictor's accuracy. The training process must be done beforehand due to the time required, as is common in machine learning applications.

---

[1]We use the sample mean, variance, and skew as our summary statistics

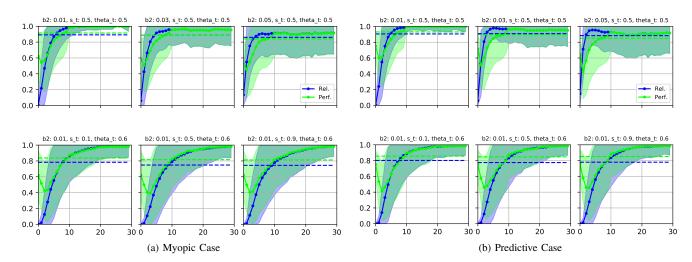[2]Link text: https://git.caslab.ece.vt.edu/hlarkin3/reliancecontroller

Fig. 3. Showing a set of Monte Carlo simulations where each point corresponds to the mean reliance/performance observed during a complete game where each new operator is randomly initialized according to the parameters at the top of each plot. The blue and green lines represent reliance (d) and performance (ρ) respectively, and the shaded regions indicate ± 1 standard deviation. The dashed lines represent the mean of their corresponding value. The x-axis is $[0, 30]$ since this is the number of games each participant played in the CPC18 data set.

Algorithm 1. ADA Process

```
1:  procedure INTERACT(D)                 ▷ Interact given data D
2:      d_list = {}                       ▷ Observed data for filter
3:      for (h_i, V_i) in D do
4:          Predict selection given V_i              ▷ Gives ĥ_i
5:          if d_ind = 1 then
6:              a_i ← a_opt               ▷ Reliance is indicated
7:          else
8:              a_i ← ĥ_i            ▷ Reliance is not indicated
9:          end if
10:         A_C ← 1 if h_i = a_i, −1 otherwise
11:         if d = 1 then
12:             h_i* ← a_i      ▷ Operator takes suggestion
13:         else
14:             h_i* ← h_i     ▷ Operator ignores suggestion
15:         end if
16:         d_list = {d_list, (d, A_C)}         ▷ Append new data
17:         Receive reward R_E(h_i*, V_i)
18:         Step P(n) and P_ind(n)             ▷ A_C used here
19:         if Update required then
20:             P_ind ← ABC(d_list)        ▷ Periodic update
21:         end if
22:         if New operator then
23:             Reinitialize both P_ind(n) and P(n)
24:         end if
25:     end for
26: end procedure
```

In addition to the predictor, the indicator model parameter filter was implemented using an engine for likelihood-free inference (ELFI) [33] packaged for use with Python. There are many different variations on the standard idea of ABC, however in this work we select rejection sampling, discussed in Section III-D, as it is well suited to our case of periodically updating the indicator model. Other sampling methods are certainly applicable, sequential Monte Carlo sampling [32] was also considered and found to be unnecessary given the relative simplicity of the reliance model considered and frequency of updates.

In the interest of examining behaviors from a large set of reliance models simultaneously, each operator's reliance model parameters are randomized according to the distributions in Table II. Note the distributions used are uniform distributions of the form $\mathcal{U}(a, b)$ where b is the distribution width. The type and width of each distribution was inspired by values used in [4] as well as by experimentation. Note that a reliance model based on (3) will have significantly different behaviors with small changes in its parameters. For example, decreasing the preference inertia value, s, will result in an operator that tends to switch reliance states often. Similarly, increasing the reliance threshold value, θ, results in less overall reliance due to the high value of preference, from (1), required. We claim that the parameters and ranges selected are expressive enough to represent a wide range of human operators.

Later in this section, we will be comparing the proposed method, including the task predictor and indicator model, to the case of myopic suggestions. Myopic here meaning that the decision aid *only* suggests the numerically optimal task a_{opt} at every iteration regardless of the indicator model. Comparing to the myopic case allows us to better quantify the potential benefits of the task predictor and indicator model system, as opposed to the widely used advice-only decision aid [2], [26], [34].

In addition to reliance as an indicator of success, we will also be using indicator model and prediction *performance* as a metric, which we label $\rho(n)$. Specifically, we use the

TABLE III
METHOD COMPARISON

$\theta = 0.5$				$\theta = 0.6$				$\theta = 0.7$		
b_2 / s	0.01	0.03	0.05	b_2 / s	0.01	0.03	0.05	b_2 / s	0.01	0.03	0.05
0.1	4.14	3.19	4.52	0.1	6.96	11.34	8.86	0.1	25.01	20.68	12.50
0.5	7.56	5.83	5.57	0.5	8.96	7.27	8.02	0.5	19.77	27.75	28.28
0.9	7.21	7.84	6.98	0.9	11.35	11.62	10.61	0.9	7.98	15.21	10.50

following metric for performance:

$$\rho(n) = \begin{cases} 1 & \text{if } d_{\text{ind}} = d = 1 \\ 1 & \text{if } d_{\text{ind}} = d = 0 \text{ and } a_i = h_i \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

where $\rho(n)$ is evaluated at each time step n and quantifies the joint indicator model and task selection prediction accuracy. Essentially, if the indicator model is correct *and*, in the case of non-reliance, the ADA makes a correct prediction, then $\rho(n) = 1$. We do not allow $\rho(n) = 1$ in the case of $d_{\text{ind}} \neq d$, nor $d_{\text{ind}} = d = 0$ with $a_i \neq h_i$, since while the indicator model may be correct the suggested task is not. Taking the mean across all iterations in the data set allows us to examine the mean accuracy of the ADA.

B. Results

1) Qualitative Results: Here we present results of simulating the proposed method using the CPC18 data set. Shown in Figure 3 is the mean reliance and performance values across the data set for the two cases. At the top of each plot is the parameter distribution information used to randomize each new operator reliance model, where the value shown is the center of a uniform distribution of width 0.005. We limit the stochasticity of parameters in this way to better examine the expected effect of the ADA in different circumstances.

The performance, from (6), is plotted in Figures 3b and 3a which shows the average indicator model and predictor accuracy as human-autonomy interaction proceeds. There is a clear convergence towards the reliance graph as the indicator model increases in accuracy, the predictor accuracy does not improve over time after the first $k = 5$ interactions as this is the input length of the predictor. In certain plots a clear jump in performance after 10 games is seen due to the ABC parameter update, where we are using prior interaction data to improve the indicator model. A smaller jump can also be seen after 20 games as the second ABC update occurs, but the refinement is minimal by this point.

Along the top row of Figure 3a, the b_2 parameter distribution center increases from left to right demonstrating the effect of different ranges of the agreement bias parameter. Here demonstrating once again that even slight parameter changes can generate significantly different operator profiles. Note that with higher values of b_2 the reliance tends to reach a lower value overall, this is likely due to the predictor's inaccuracies being weighted higher than other factors such as capability.

Along the bottom row of Figure 3a we instead vary the s parameter, as expected we observe a relatively slight delay in convergence with higher values of s but without much change to the shape. Further, the bottom row uses $\theta = 0.6$ and the result is a much decreased rate of reliance increase.

Moving on to Figure 3b, where in this case we are using the method described in Algorithm 1 to generate suggestions. The same parameter ranges as in Figure 3a are used to permit direct comparisons. Most notably, the speed of reliance increase is higher in the plotted cases, indicating the value of providing carefully selected suggestions. As a result the mean reliance achieved in the predictive case is higher in most cases shown. In addition, with higher ranges of b_2 there is some amount of reliance overshoot, caused by inertia present in (1) and a high weight placed upon predictions compared to autonomy capability.

2) Quantitative Results: Shown in Table III are results of comparing the previously mentioned myopic method to the predictive case. Specifically, in each cell we take the percentage difference between the overall mean reliance values of the two methods, where the reliance models are being driven by the parameter distribution centers shown on the left and top of the tables. Here we are examining the reliance value achieved over 30 games and taking the mean across the participants, which corresponds to the dashed blue lines in Figure 3. With Table III, we are able to test a wide set of parameter ranges and see that the proposed method improves significantly upon the myopic method in nearly all cases tested, with some exceptions where the percentage improvement is $\leq 5\%$ and considered negligible.

Interestingly in Table III, as θ increases the margin of improvement seems to increase as well. In general terms, a higher preference threshold value results in an operator much less likely to rely on autonomy for task selection, thus it is encouraging to see the effectiveness of the proposed method in such cases. There are a few cases which dispute that trend however, such as $\theta = 0.7$ with a high s value, which may be due to the limited number of trials used. That is, with higher θ reliance typically takes longer to build up, an effect worsened by a high inertia parameter as well.

V. CONCLUSIONS

In this work we developed a game-theoretic modeling of the interaction between a human operator and an ADA in a multi-agent task allocation setting. In this setting, the

ADA as designed to correctly calibrate the operator's reliance upon the aid and improve long-term team performance. We approached this via a combination of human action prediction and parameter fitting, resulting in a decision aid that adapts to human reliance dynamics. The proposed method was tested against a large set of simulated operators, and shown to substantially improve human-autonomy interactions compared to a myopic decision aid.

References

[1] S. Musić and S. Hirche, "Control sharing in human-robot team interaction," *Annual Reviews in Control*, vol. 44, pp. 342–354, 2017.

[2] K. Okamura and S. Yamada, "Empirical evaluations of framework for adaptive trust calibration in human-ai cooperation," *IEEE Access*, vol. 8, pp. 220 335–220 351, 2020.

[3] J. D. Lee and K. A. See, "Trust in automation: Designing for appropriate reliance," *Human factors*, vol. 46, no. 1, pp. 50–80, 2004.

[4] J. Gao and J. D. Lee, "Extending the decision field theory to model operators' reliance on automation in supervisory control situations," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 36, no. 5, pp. 943–959, 2006.

[5] M. Chen, S. Nikolaidis, H. Soh, D. Hsu, and S. Srinivasa, "Trust-aware decision making for human-robot collaboration: Model learning and planning," *ACM Transactions on Human-Robot Interaction (THRI)*, vol. 9, no. 2, pp. 1–23, 2020.

[6] D. P. Losey and D. Sadigh, "Robots that take advantage of human trust," *arXiv preprint arXiv:1909.05777*, 2019.

[7] J. W. Burton, M.-K. Stein, and T. B. Jensen, "A systematic review of algorithm aversion in augmented decision making," *Journal of Behavioral Decision Making*, vol. 33, no. 2, pp. 220–239, 2020.

[8] R. K. Williams, N. Abaid, J. McClure, N. Lau, L. Heintzman, A. Hashimoto, T. Wang, C. Patnayak, and A. Kumar, "Collaborative multi-robot multi-human teams in search and rescue," *Proceedings of the International ISCRAM Conference*, vol. 17, Apr. 2020.

[9] L. Heintzman, A. Hashimoto, N. Abaid, and R. K. Williams, "Anticipatory planning and dynamic lost person models for Human-Robot search and rescue," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. ieeexplore.ieee.org, May 2021, pp. 8252–8258.

[10] A. Hashimoto and N. Abaid, "An agent-based model of lost person dynamics for enabling wilderness search and rescue," in *Dynamic Systems and Control Conference*, vol. 59155. American Society of Mechanical Engineers, 2019, p. V002T13A005.

[11] X. Xu, M. Amin, and W. Santee, "Usariem technical report t08-05: Probability of survival decision aid (psda)," 2008.

[12] L. Heintzman and R. K. Williams, "Nonlinear observability of unicycle multi-robot teams subject to nonuniform environmental disturbances," *Auton. Robots*, vol. 44, no. 7, pp. 1149–1166, Sep. 2020.

[13] ——, "Multi-agent intermittent interaction planning via sequential greedy selections over position samples," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 534–541, Apr. 2021.

[14] R. J. Koester, *Lost Person Behavior: A search and rescue guide on where to look - for land, air and water*. dbs Productions LLC, 2008.

[15] C. Nam, P. Walker, H. Li, M. Lewis, and K. Sycara, "Models of trust in human control of swarms with varied levels of autonomy," *IEEE Transactions on Human-Machine Systems*, vol. 50, no. 3, pp. 194–204, 2019.

[16] Y. Wang, Z. Shi, C. Wang, and F. Zhang, "Human-robot mutual trust in (semi) autonomous underwater robots," in *Cooperative Robots and Sensor Networks 2014*. Springer, 2014, pp. 115–137.

[17] R. K. Williams, A. Gasparri, and G. Ulivi, "Decentralized matroid optimization for topology constraints in multi-robot allocation problems," *2017 IEEE International Conference on Robotics and Automation*, 2017.

[18] J. Liu and R. K. Williams, "Submodular optimization for coupled task allocation and intermittent deployment problems," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3169–3176, 2019.

[19] Y. Sung, A. K. Budhiraja, R. K. Williams, and P. Tokekar, "Distributed assignment with limited communication for multi-robot multi-target tracking," *Auton. Robots*, 2020.

[20] J. Liu, L. Zhou, P. Tokekar, and R. K. Williams, "Distributed resilient submodular action selection in adversarial environments," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5832–5839, Jul. 2021.

[21] Y. Sung, A. K. Budhiraja, R. K. Williams, and P. Tokekar, "Distributed simultaneous action and target assignment for Multi-Robot Multi-Target tracking," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 1–9.

[22] J. Liu and R. K. Williams, "Data-driven models with expert influence: A hybrid approach to spatiotemporal process estimation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 2467–2473.

[23] ——, "Coupled temporal and spatial environment monitoring for multi-agent teams in precision farming," in *Proceedings of the IEEE Conference on Control Technology and Applications*, 2020, pp. 273–278.

[24] ——, "Optimal intermittent deployment and sensor selection for environmental sensing with multi-robot teams," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2018, pp. 1078–1083.

[25] ——, "Monitoring over the long term: Intermittent deployment and sensing strategies for multi-robot teams," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2020, pp. 7733–7739.

[26] K. Akash, W.-L. Hu, N. Jain, and T. Reid, "A classification model for sensing human trust in machines using eeg and gsr," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 8, no. 4, pp. 1–20, 2018.

[27] C. Dubois and J. Le Ny, "Adaptive task allocation in human-machine teams with trust and workload cognitive models," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2020, pp. 3241–3246.

[28] G. M. Beck, R. Limor, V. Arunachalam, and P. R. Wheeler, "The effect of changes in decision aid bias on learning: Evidence of functional fixation," *Journal of Information Systems*, vol. 28, no. 1, pp. 19–42, 2014.

[29] D. D. Bourgin, J. C. Peterson, D. Reichman, S. J. Russell, and T. L. Griffiths, "Cognitive model priors for predicting human decisions," in *International conference on machine learning*. PMLR, 2019, pp. 5133–5141.

[30] O. Plonsky, R. Apel, E. Ert, M. Tennenholtz, D. Bourgin, J. C. Peterson, D. Reichman, T. L. Griffiths, S. J. Russell, E. C. Carter *et al.*, "Predicting human decisions with behavioral theories and machine learning," *arXiv preprint arXiv:1904.06866*, 2019.

[31] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[32] J. Lintusaari, M. U. Gutmann, R. Dutta, S. Kaski, and J. Corander, "Fundamentals and recent developments in approximate bayesian computation," *Systematic biology*, vol. 66, no. 1, pp. e66–e82, 2017.

[33] J. Lintusaari, H. Vuollekoski, A. Kangasrääsiö, K. Skytén, M. Järvenpää, P. Marttinen, M. U. Gutmann, A. Vehtari, J. Corander, and S. Kaski, "Elfi: Engine for likelihood-free inference," *Journal of Machine Learning Research*, vol. 19, no. 16, pp. 1–7, 2018. [Online]. Available: http://jmlr.org/papers/v19/17-374.html

[34] G. M. Beck, *The effects of decision aid bias and fixation on user performance*. University of Missouri-Columbia, 2004.