Trust-based Performance Optimization for Human-Swarm Collaboration

R. Oliver Zanone [*] **Javad Mohammadpour Velni** [*]

[*] *Department of Mechanical Engineering, Clemson University, Clemson, SC, USA (e-mail: rzanone@clemson.edu, javadm@clemson.edu).*

Abstract: This work develops a trust inference model to address scenarios where agents in a swarm collaborate to achieve the coverage control task. To gather empirical data from human subjects for the probabilistic model development, we build various simulation tools and user interfaces. Using our visual training tool, we train a single-agent model and then extend that to create our multi-agent model. These models utilize a dynamic Bayesian network and produce stochastic predictions. We then apply these models to our Voronoi-based area coverage problem in real time, where agents adjust their behavior to maximize the team performance and hence human trust. As a result of this research, multi-agent teams will be able to increase their individual trust levels thereby enhancing team performance and efficiency.

Keywords: Trust modeling, multi-agent systems, dynamic Bayesian network, expectation maximization

1. INTRODUCTION

With recent advances in the field of artificial intelligence (AI) in combination with autonomous systems, we are witnessing an increasing number of their applications in various industries, such as manufacturing, healthcare, and transportation. These technologies have the potential to improve efficiency, reduce costs, and enhance safety in many different contexts. However, as these systems become more sophisticated and independent, it is important to address the issue of trust and how it affects the way humans interact with them. For example, the automotive industry has implemented various systems, including advanced cruise control that can maintain lane and proximity to other vehicles. Some car manufacturers have even released beta versions of full self-driving capabilities. However, as these autonomous systems are increasingly applied to technologies that have the power to directly impact human life, the question of when to trust them becomes more critical.

Lee and See (2004) propose a definition of trust in the context of human-robot trust/interaction as:

> The attitude that an agent will help achieve an individual's goals in a situation characterized by uncertainty and vulnerability.

However, it should be noted that the definition varies heavily depending on the context in which it is inferred (Khavas et al., 2020). At the outset, a human must invest a certain amount of trust in a system, and this level of trust can be increased or decreased by the subsequent actions of the system. It is crucial to assess the human's initial level of trust, as placing too much trust in a system can result in severe consequences, while not trusting it enough can lead to reduced task efficiency overall.

This paper presents a trust model for a team of agents that can reduce the need for an operator to monitor multiple robots, enabling them to oversee the entire team. The model determines the level of autonomy assigned to individual agents and identifies when intervention is necessary. We achieve this by expanding a single-agent stochastic trust model to the multi-agent scenario. Our models are trained using empirical data from human test subjects, and we develop a multi-agent coverage control simulation platform to test their effectiveness in real time. Agents utilize individual models of trust to adjust their behavior and maximize the model prediction for team trust.

2. RELATED WORK

Trust in literature varies from psychological studies to methods of modeling human/robot trust to applications of developed models. Hancock et al. (2011) look at the trust from the psychological standpoint performing a meta-analysis of factors in the development of human-robot trust while Freedy et al. (2007) look at attributes most critical when deciding upon a level of trust, and Hussein et al. (2020) look at properly calibrating trust levels based on reliability and transparency.

There is a vast literature focused on the single-agent and human interactions. Xu and Dudek (2015) present an asymmetric human-robot model of trust termed OPTIMo that assumes the human operator plays the role of a supervisor and the agent is a worker. They base their model off of a dynamic Bayesian network (DBN) structure and use casual reasoning and evidential factors to produce performance-centric belief distributions for inferring human trust. Our work uses a very similar single-agent model to that of OPTIMo. A bi-directional model is presented by Azevedo-Sa et al. (2021) which notes the differences between human trust and robotic trust, distinguishing it from many single-agent models. One question that needs to be answered in this model is how a robot can determine when to trust the human. The researchers collected data from 284 participants and demonstrated the model's potential applications in control authority allocation.

Wang et al. (2018) propose an application of OPTIMo, using the model on several robots for symbolic motion planning utilizing a human-in-the-loop. This work implements a monitoring and switching framework that gives control to the human when taking a path deemed risky. It uses the OPTIMo single-agent model to make that determination. They also incorporate additional algorithms to prevent deadlock/livelock in the case that agents are blocking each other's paths. More recently, Fooladi Mahani et al. (2021) extended OPTIMo to the multi-agent scenario. In this work, a team of robots are overseen by an operator. Robots are non-collaborative and each have assigned set areas in a search and rescue mission setting. They tune a linear model for performance using least squares and feed that into their model for trust. General distributions are modeled by a categorical Boltzmann machine (CBM) which uses a recurrent neural network. They then use expectation maximization for model parameter training. They assume at most only one agent may be in manual mode for the operator to intervene or manually control, while the rest must remain autonomous. This model, while trained for the multi-agent scenario, defines an individual measure of trust relative to other agents and uses evidential factors from the operator. ***Our work differs in that we look to define an overall model of trust for the team using weighted team performance and a weighted trust metric. Individual agents then have the goal of changing their behavior to maximize team trust***. We create simulations that apply these models in real-time and provide results that support our models.

Other works involving trust in human-swarms include inverse reinforcement learning (Nam et al., 2017), transparency modeling (Hepworth et al., 2020), and another bidirectional model (termed mutual trust or bilateral) using a qualitative time series model for trust (Wang and Wang, 2017).

3. BACKGROUND

Our model of trust can quantify reliability of the team as a whole, which is a function of agents' performance and inferred individual trust. In the collaborative task sharing, the ultimate goal is to handle the main task while ensuring each agent is taking a right share of the overall task based on their dynamics or their associated performance-oriented measures. Hence, the developed model for team trust should take the mutual trust levels of the agents into account. In essence, in addition to the evaluation of each agent by the human supervisor, our model should also be aware of relative trust measures within the team. Our proposed approach incorporates two models. Firstly, we develop a linear Gaussian model similar to that found in OPTIMo for individual agents. Then, we use several metrics, a weighted team performance and mutual trust metric as inputs to our team model for trust.

The single-agent model of trust is a performance-centric model meaning the inferred human trust state is highly dependent upon the current performance of the agent. The performance is scored on a normalized scale and directly correlates to task performance, which the operator can visually see during the data collection/simulations. This effectively means high task performance will lead to a higher trust score, and poor performance leads to a low trust score. Then, we incorporate intervention which can be thought of as an evidential factor showing a lapse in the current trust as the agent fails to effectively complete the current task. Now the question is how one can revise the trust inference model when dealing with multiple agents collaborating on given task(s). The idea is to account for agents mutual interaction through proposing a mutual trust score. This allows for further extension of the proposed trust models to the case where a high number of agents are operating in a distributed manner. The objective is to decrease the number of required agent interventions by human operator using the learned individual trust models and by maximizing the team trust output.

The end goal of this research is to extend the OPTIMo concept from an individual agent to a team of multiple agents. This capability will allow collaboration between the agents in order to complete coordinated autonomous operations. The team of agents will contain trust-seeking [1] adaptive robots: these robots will be able to sense when the human has low trust and adapt their behaviors in response to improve task performance and seek greater team trust. The human operator will therefore only need to analyze the team's overall performance as the individual agents will self-adjust their behaviors in attempt to reach optimal team trust. In this work, we first evaluate an agent's individual inferred trust state in order to quantify the human's trust state during interactions. We then introduce our approaches for modeling team trust, which use weighted performance metrics and mutual trust. The individual model dictates how an agent must adapt its behavior to improve the team's overall performance and trust metric.

The **main contribution** of this work is the development of a ***trust model that can accurately infer the trust relationship between a human operator and a team of autonomous agents in real time***. It also allows individual agents to coordinate their behaviors in order to increase their individual trust states and improve the team's performance and efficiency. This model can be trained through direct interactions between the human operator and the team allowing the agents to capture the operators trust tendencies. *By quantifying the operator's degree of trust, one can add a new degree of autonomy to coordinated multi-agent operations.*

3.1 Single-Agent Trust Model

The original OPTIMo paper presents an asymmetric human-robot performance-centric model for trust. This

[1] Trust-seeking can actually be problematic since the trust feedback provider may not be able to provide the optimal feedback and best intervention. It works fine under the assumption that the trust feedback provider is an expert in giving feedback.

model was chosen as a starting point for our work in that it is a time-series model able to dynamically infer the human trust state in near real-time and give probabilistic trust estimates. Our model uses an agent's individual task performance, human intervention, and trust feedback as the main factors to infer the trust state. The OPTIMo model considers a few more factors such as trust change and extraneous causes, which are unnecessary in our problem setup. Since the trust state is a "non-observable" state in the system, it is a hidden variable in the graph. The human operator perceives the performance of the robot agent and gives trust feedback and intervention. After human intervention, the agent adjusts its behavior based on the input from human.

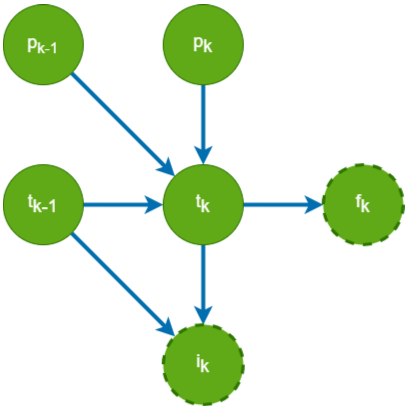


Fig. 1. Dynamic Bayesian network model for a single-agent.

Figure 1 illustrates our DBN structure. This diagram shows how factors such as performance, p_k, affects the human's latent trust state, t_k. Intervention, i_k, and trust feedback, f_k, provide evidence to support a change in trust. We look at a continuous sequence of interactions $k \in K$. Performance is recorded for all k, while intervention and trust feedback occur at a set interval, or x steps. Data for intervention/trust feedback may also be recorded if the user chooses to pause at any point and provide data. All data is normalized $\in [0, 1]$, with the exception of intervention which is a binary variable $\in \{0, 1\}$.

Like OPTIMo, we use conditional probability distributions (CPDs) to model the relationship between each factor. Human trust is modeled after a Gaussian distribution expressed in terms of prior trust, t_{k-1}, and current/prior performance, p_k, p_{k-1}:

$$\begin{aligned} &Prob(t_k|t_{k-1}, p_k, p_{k-1}) \\ &\quad = \mathcal{N}(t_k; t_{k-1} + \omega_{tb} + \omega_{tp} p_k + \omega_{td}(p_k - p_{k-1}), \sigma_t), \end{aligned} \tag{1}$$

where $\mathcal{N}(x; \mu, \sigma)$ is a Gaussian distribution for the random variable x, mean μ, and standard deviation σ. Parameters w_{tb}, w_{tp}, and w_{td} are learned during training from collected data and correspond to bias, performance, and performance difference weightings.

Intervention is modeled in terms of current/prior trust, t_k, t_{k-1}, and uses a sigmoid distribution $\mathcal{S}(x)$ since i_k is a binary variable.

$$\begin{aligned} &Prob(i_k = 1|t_k, t_k - 1) \\ &\qquad = \mathcal{S}(\omega_{ib} + \omega_{it} t_k + \omega_{id}(t_k - t_{k-1})). \end{aligned} \tag{2}$$

Again, parameters ω_{ib}, $\omega_{it} t_k$, and ω_{id} are learned during training and correspond to bias, intervention, and difference weightings. It is understood that $Prob(i_k = 0|...) = 1 - Prob(i_k = 1|...)$.

A Gaussian CPD can be used to model the relationship of trust feedback (given by the human) to the latent trust state as

$$Prob(f_k|t_k) = \mathcal{N}(f_k; t_k, \sigma_f). \tag{3}$$

The single-agent model is trained using the data collected from human subjects. To perform training, hard assignment expectation maximization is utilized. Full process details will be discussed in a future section.

4. DEVELOPMENT OF MULTI-AGENT TRUST MODEL

We propose two structures for expanding the single-agent trust model to the multi-agent scenario:

4.1 One-layer DBN Structure

The first model structure we propose is shown in Fig. 2. It is assumed that the performance data p_k^i is available at all time steps, intervention data i_k^i is available at some time steps, and team trust feedback F_k is available at some time steps. We denote each agent as i, where $i = 1, 2, ..., N$ (N is the total number of agents). Using this network structure, we will assume we only have recorded data for F_k, or trust feedback at the team level, and f_k^i, or trust feedback at the agent level, remains unknown.

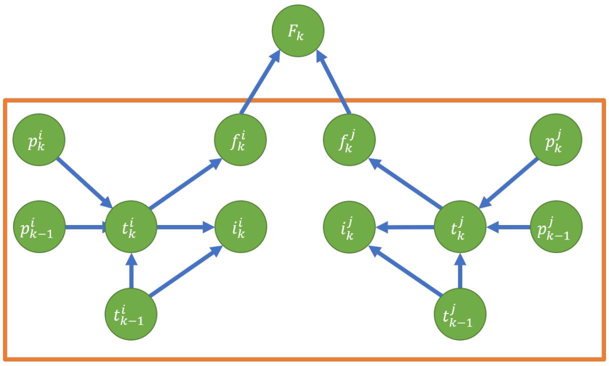


Fig. 2. An illustrative example of a dynamic Bayesian network for each agent within the team's Bayesian structure. In this structure, t_k^i, p_k^i, i_k^i, and f_k^i represent the i-th agent's trust, performance, human intervention, and decomposed trust feedback at time step k, respectively; F_k is the human trust feedback at the team level.

We compose the relation between f_k^i and F_k as the following two equations

$$f_k^i = \frac{2}{\pi} \arctan \left[\frac{p_k^i}{\frac{1}{N} \sum\limits_{i=1}^{N} p_k^i} \tan(\frac{\pi}{2} F_k) \right], \tag{4}$$

$$F_k = \frac{2}{\pi} \arctan \left[\frac{1}{N} \sum_{i=1}^{N} \tan(\frac{\pi}{2} f_k^i) \right]. \tag{5}$$

From (4), we can obtain the following equation by multiplying $\frac{\pi}{2}$ and taking the tangent on both sides

$$\tan(\frac{\pi}{2} f_k^i) = \frac{p_k^i}{\frac{1}{N}\sum_{i=1}^{N} p_k^i} \tan(\frac{\pi}{2} F_k). \tag{6}$$

Then, summation over all $i = 1, \ldots N$ on both sides, we obtain

$$\sum_{i=1}^{N} \tan(\frac{\pi}{2} f_k^i) = N \tan(\frac{\pi}{2} F_k). \tag{7}$$

We now use F_k to generate f_k^i using (4). The single-agent model is then trained on data p_k^i, i_k^i, and the generated f_k^i. We can then use the trained model to infer f_k^i on new data, or during real-time simulations. This inferred data can then be used to generate F_k using (5) to obtain a team trust metric.

It is worth mentioning that F_k and f_k^i are strictly in the range $(0, 1)$. If some instances of F_k and f_k^i are coincidentally exactly 1, these get reduced by a number very close to 1, e.g., 0.999, due to $\tan(\pi/2)$ being undefined.

4.2 Two-layer DBN Structure

Another approach is a two-layer DBN structure. The graphical model is shown in Figure 3. We decompose this network into two levels: the team level and the agent level. The two levels are relatively independent. The team level includes factors such as team performance P_k, team trust T_k, team trust feedback F_k, and mutual trust M_k. We assume the distributions linking these factors are normal and are as follows

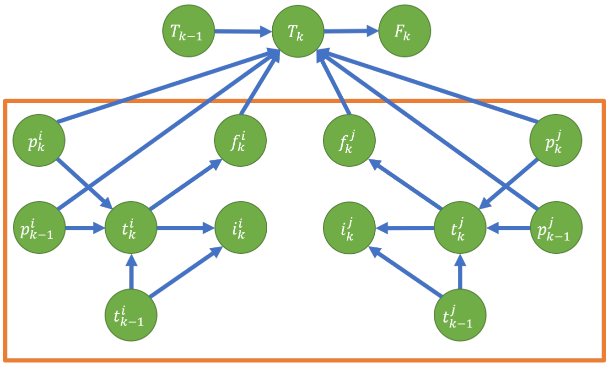


Fig. 3. An illustrative example of a dynamic Bayesian network for each agent within the team's Bayesian structure. In this structure, t_k^i, p_k^i, i_k^i, and f_k^i represent the i-th agent's trust, performance, human intervention, and trust feedback at time step k, respectively; T_k and F_k are the trust and human trust feedback at the team level, respectively.

$$Prob(T_k | T_{k-1}, P_k, P_{k-1}, M_k) = \mathcal{N}(T_k; \mu_T, \sigma_T) \tag{8}$$

$$Prob(F_k | T_k) = \mathcal{N}(F_k; T_k, \sigma_F), \tag{9}$$

where

$$\mu_T = T_{k-1} + \omega_{Tb} T + \omega_{Tp} P_k + \omega_{Td}(P_k - P_{k-1}) + \omega_{Tm} M_k, \tag{10}$$

$$P_k = \sum_{i=1}^{N} \frac{A_k^i}{A} p_k^i, \tag{11}$$

$$M_k = \sum_{i=1}^{N} \frac{A_k^i}{A} t_k^i. \tag{12}$$

We train at the single-agent level using p_k^i, i_k^i, and f_k^i to acquire t_k^i and all the relevant parameters. Then, at the team level, we first generate P_k and M_k and use F_k to train for T_k and acquire all relevant parameters. Since this model was better aligned with our objectives, we decided to adopt this approach going forward.

4.3 Training Approach

We apply hard-assignment expectation maximization to acquire the optimized parameters. This is a well known approach for performing maximum likelihood estimation in the presence of latent variables (trust). First, initial parameters Θ_0 are chosen, and the expectation is then computed to find missing values. Next, the maximization step finds optimized parameters. This process then repeats until objective function convergences or a certain threshold is reached. The interested reader is referred to (Koller and Friedman, 2009) for more details on the process.

For the expectation calculation, we use a filtered then smoothed belief to maintain stochastic properties. We then use the resulting distribution to find a deterministic/discrete values for expectation. These deterministic values are then used for the maximization step. The expectation steps are derived from the OPTIMo model of trust and are as follows:

We first calculate the filtered trust belief $bel_f(T_k)$ as

$$bel_f(T_k) = \frac{\int \overline{bel}(T_k, T_{k-1}) dT_{k-1}}{\iint \overline{bel}(T_k, T_{k-1}) dT_{k-1} dT_k} . \tag{13}$$

This is calculated forward in time assuming a uniform initial trust belief $bel_f(T_0) = Prob(T_0) = 1$. Initialized parameters Θ_0 are chosen and used for the first iteration. We use the joint distribution of T_k and T_{k-1} or $\overline{bel}(T_k, T_{k-1})$ within $bel_f(T_k)$ as

$$\overline{bel}(T_k, T_{k-1}) = Prob(F_k) Prob(T_k) bel_f(T_{k-1}). \tag{14}$$

We can marginalize the distributions to perform the calculations of the numerator and denominator as

$$\int \overline{bel}(T_k, T_{k-1}) dT_{k-1} \rightarrow \sum_{T_{k-1}} \overline{bel}(T_k, T_{k-1}), \tag{15}$$

$$\iint \overline{bel}(T_k, T_{k-1}) dT_{k-1} dT_k \rightarrow \sum_{T_k} \sum_{T_{k-1}} \overline{bel}(T_k, T_{k-1}). \tag{16}$$

We discretize the space of $T_{k-1} \times T_k$ to equally spaced bins, e.g., 10×10. This allows us to calculate $\overline{bel}(T_k, T_{k-1})$ and then perform the summations.

We now calculate the smoothed trust belief $bel_s(T_k)$ using (17) for all data backward in time given that $bel_s(T_{final}) = bel_f(T_{final})$.

$$bel_s(T_{k-1}) = \int \frac{\overline{bel}(T_k, T_{k-1})}{\int \overline{bel}(T_k, T_{k-1}) dT_{k-1}} bel_s(T_k) dT_k \tag{17}$$

We use marginal distributions to calculate the numerator/denominator in a similar fashion to the filtered belief step. Lastly, we take the expectation for each smoothed trust belief distribution to obtain a single sequence of trust states. These values are then used for the maximization step.

For the maximization step, we employ our log likelihood function, which takes into account the collected data and values from the expectation steps. We utilize an optimization function from the SciPy optimization library, specifically employing the Nelder Mead algorithm, which enables us to incorporate constraints and bounds within its parameters. Our objective becomes finding the parameters that optimize our joint probability across all time steps, or

$$\Theta^* = \arg\max_{\Theta} \prod_{k=1}^{K} \mathcal{P}(F_k|T_k)\mathcal{P}(T_k|T_{k-1}, P_{k-1}, P_k, M_k), \tag{18}$$

where $\Theta = (\omega_{Tb}, \omega_{Tp}, ...)$. Our objective function can then be converted into a log likelihood function so that a summing operation can be applied to simplify iterations:

$$\arg\max_{\Theta} \left[\sum_{k=1}^{K} \ln P(f_k|t_k) + \sum_{k=1}^{K} \ln P(t_k|p_{k-1}, p_k, t_{k-1}) \right]. \tag{19}$$

5. DATA COLLECTION FOR THE MULTI-AGENT TRUST MODEL LEARNING

In order to perform analysis and training of our model, human subject testing data had to be collected. To do this, a user interface was built in Python. This simulation looked at a multi-agent coverage problem with $\mathcal{N}$ agents and utilized Voronoi partitioning in an environment containing obstacles. An illustrative example is visualized in Fig. 4.

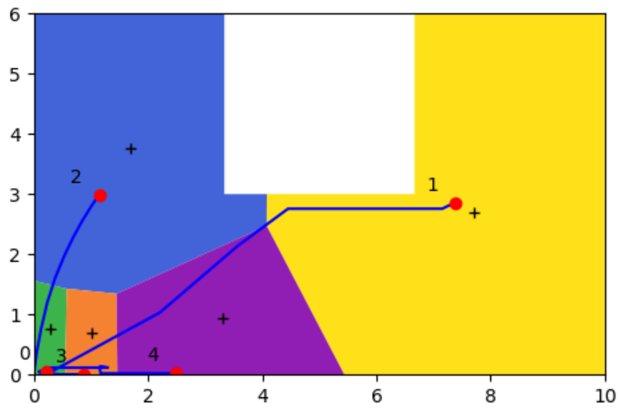
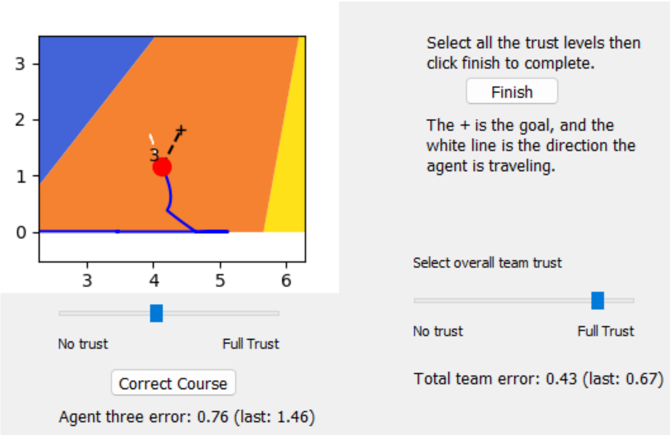


Fig. 4. A coverage simulation utilizing Voronoi partitioning and five agents.

The agents goals, i.e., their current task headings, are denoted by the plus sign "+" and they use an iterative approach to move towards them. Upon initialization of the simulation, each agent is assigned a random error based on a seed. The heading is calculated, or a normalized vector from the agent's position to its goal, and the error is added to this vector. This error can be thought of as a different constant wind applied to each agent. Each agent's speed is then multiplied by its heading (which includes the error component) to move one step forward in time. After a predetermined amount of time passes, the human subject is presented with an intervention GUI. Alternatively, the subject can pause the simulation at any time and intervene/provide feedback. The intervention GUI queries the subject about the trust level of each individual agent. In addition, the distance-to-goal error and the heading error are displayed to aid the user in their decision, and the subject can decide if they would like to intervene by pressing the corresponding "correct course" button. This makes adjustments to the heading error. Finally, the subject is asked to provide feedback for the overall team trust. A snapshot of the developed GUI is shown in Fig. 5.


Select all the trust levels then click finish to complete.

The + is the goal, and the white line is the direction the agent is traveling.

Select overall team trust

Agent three error: 0.76 (last: 1.46)

Total team error: 0.43 (last: 0.67)

Fig. 5. The intervention window where the subject can offer feedback. A single agent is shown, but the user is queried for all remaining agents. The agents error can be seen by the white dashed line. This is what we call heading error, i.e., the actual direction the agent will move after the error is applied (an example is the wind applied to the heading).

Data was collected from a sample size of 10 people. Each participant was asked to run through two simulation sessions. This generated a total of approximately 8,000 data points to train upon. All relevant data was saved to a CSV file for further processing such as trust feedback, intervention (course correction), distance error, coordinates, and coverage area size[2] .

6. TRAINING TOOL FOR THE MULTI-AGENT TRUST MODEL

6.1 preprocessing

The collected data from the human subject study is loaded into our training visualization tool, a GUI created in Python. The distance error is converted to a normalized metric of performance using

$$f(|z|) = \frac{2}{1 + e^{|z|}} \, . \tag{20}$$

This function was chosen as it best fits our dataset, but other functions can be used as well. In addition, as a larger

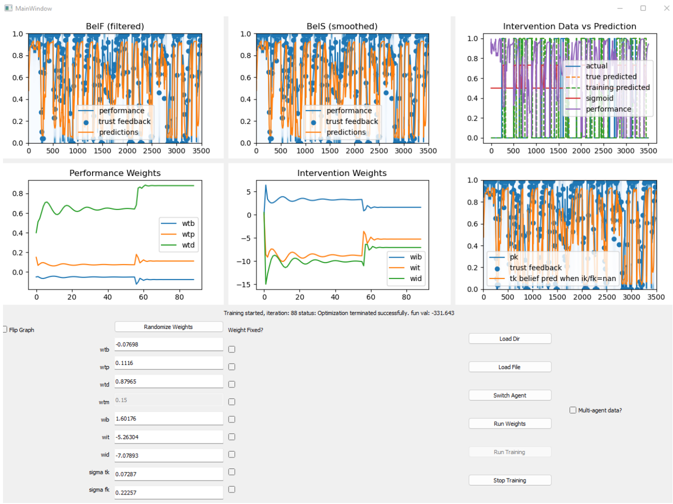
[2] More information on this simulation and other codes for this paper can be found at https://Github.com/OliverZanone.

distance corresponds with a smaller performance score, functions that had inverse relationships were needed.

After loading the dataset, initial weights can be set and visualized, and training can then commence. There are many useful features in the training GUI to allow for a flexible on-the-fly training setup. For example, certain training weights can be set as static to allow only a certain set of parameters to be trained for. This is helpful in guiding the parameters to convergence. The training tool can be seen in figure 6.

6.2 Training Results

The results of the single-agent training can be seen in Fig. 6. For both our single-agent model and multi-agent model, training results show the highest emphasis on the difference weighting. Weights show a convergence around 60 iterations when starting with reasonable initial parameters. For the single-agent tool, we visualize the filtered belief, smoothed belief and intervention predictions. In addition, we show weight convergence and print out the current optimizer iteration and function output. The multi-agent tool shows the same information but adds the performance of all individual agents and their corresponding weighted team performance/feedback. These multiple visualizations allow for real-time monitoring of the training process to ensure proper convergence and allowing on the fly changes to parameters.



Training started, iteration: 88 status: Optimization terminated successfully. fun val: -331.643

wtb -0.07698 □
wtp 0.1116 □
wtd 0.87965 □
wtm 0.15 □
wib 1.60176 □
wit -5.26304 □
wid -7.07893 □
sigma tk 0.07287 □
sigma fk 0.22257 □

Fig. 6. The training results for our single-agent model.

7. APPLICATIONS OF THE DEVELOPED TEAM TRUST MODEL FOR PERFORMANCE IMPROVEMENT

Our data collection tool/simulation involved the coverage control problem utilizing Voronoi partitioning. Agents started in a cluster, Voronoi partitioning was applied, and agents moved towards their partition's centroid by calculating a heading and multiplying it by a speed. An error is randomly calculated at the start of the simulation that applies a constant error vector to the heading. This can be thought of as a constant wind that will cause agents to stray from their desired course. During human trials, subjects were given the ability to intervene. When performing an intervention, the simulation then reduced the error vector. We use this same simulation but slightly modified to implement our multi-agent model of trust. Our goal now is to maximize the team trust prediction in real time, and to do this, we propose two applications. We set a constant error seed to compare the methods and establish a baseline result, or where neither method is used (see Fig. 7).

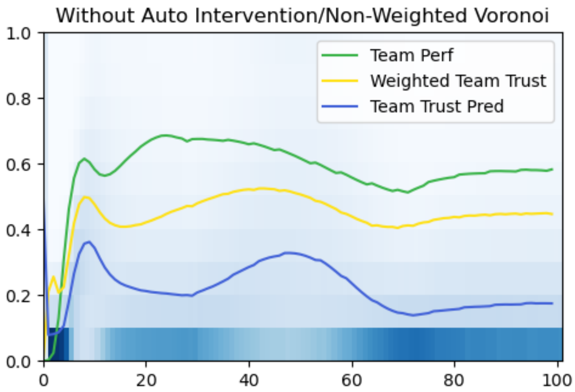


Fig. 7. The resulting team trust prediction when neither approach is used. The background represents the trust belief distribution.

7.1 Approach 1: Auto Intervention

Firstly, we use the single-agent model to predict individual trusts. This is then fed into our model for intervention, which decides whether an agent requires outside intervention. As the simulation runs, the model is queried at the same rate the human subject was as to whether or not an agent needs an intervention. If the model predicts an agent requires intervention, the simulation applies the same error reduction technique to the agent. We term this "auto intervention." The simulation is then run with auto intervention disabled to establish a baseline and then again with it enabled. As team trust performance is highly correlated to individual performance, the results show a maximization of the team trust level.

7.2 Approach 2: Weighted Voronoi

In this approach, we utilize a multiplicatively weighted Voronoi diagram (see Dobrin (2005)). As each individual agent is scored a trust metric, this can be proportionately applied to a weighting for individual agent's Voronoi cell. Put simply, a low trust score leads to a smaller area to be covered by the agent and vice versa. The data from each individual agent is used as an input to the multi-agent trust model for final results. Weighted Voronoi works in a similar fashion to a non-weighted Voronoi partitioning. The primary difference is how the distance function is defined (see the section on Voronoi partitioning for further information). The results of both auto-intervention and the weighted Voronoi method can be seen in Fig. 8.

8. CONCLUDING REMARKS

In conclusion, trust is a critical factor in the development and deployment of autonomous systems. With advance-

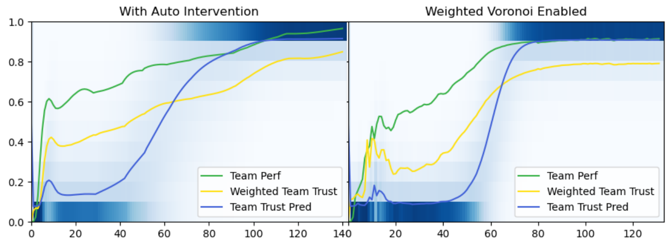


Fig. 8. The resulting team trust prediction when using both methods, i.e. auto intervention and weighted Voronoi partitioning proportionately weighted to trust metrics. The background represents the trust belief distribution.

ments in AI and autonomous systems technology, it is important to understand how trust is defined and how it can be modeled to facilitate effective human-robot interactions. The context in which trust is inferred and the initial level of trust are crucial in determining the level of autonomy a robot should be given and when intervention is necessary. The multi-agent trust models developed in this paper have the potential to greatly increase the efficiency and performance of robotic swarms, and the use of simulations and empirical data collection can aid in the training and application of these models. While there are still challenges to be overcome, the continued research into human-robot trust will undoubtedly lead to the development of more capable and trustworthy autonomous systems in the future.

REFERENCES

Azevedo-Sa, H., Yang, X.J., Robert, L.P., and Tilbury, D.M. (2021). A unified bi-directional model for natural and artificial trust in human–robot collaboration. *IEEE robotics and automation letters*, 6(3), 5913–5920.

Dobrin, A. (2005). A review of properties and variations of voronoi diagrams. *Whitman College*, 10(1.453), 9156.

Fooladi Mahani, M., Jiang, L., and Wang, Y. (2021). A bayesian trust inference model for human-multi-robot teams. *International Journal of Social Robotics*, 13(8), 1951–1965.

Freedy, A., DeVisser, E., Weltman, G., and Coeyman, N. (2007). Measurement of trust in human-robot collaboration. In *2007 International symposium on collaborative technologies and systems*, 106–114. Ieee.

Hancock, P.A., Billings, D.R., Schaefer, K.E., Chen, J.Y., De Visser, E.J., and Parasuraman, R. (2011). A meta-analysis of factors affecting trust in human-robot interaction. *Human factors*, 53(5), 517–527.

Hepworth, A.J., Baxter, D.P., Hussein, A., Yaxley, K.J., Debie, E., and Abbass, H.A. (2020). Human-swarm-teaming transparency and trust architecture. *IEEE/CAA Journal of Automatica Sinica*, 8(7), 1281–1295.

Hussein, A., Elsawah, S., and Abbass, H.A. (2020). The reliability and transparency bases of trust in human-swarm interaction: principles and implications. *Ergonomics*, 63(9), 1116–1132.

Khavas, Z.R., Ahmadzadeh, S.R., and Robinette, P. (2020). Modeling trust in human-robot interaction: A survey. In A.R. Wagner, D. Feil-Seifer, K.S. Haring, S. Rossi, T. Williams, H. He, and S. Sam Ge (eds.), *Social Robotics*, 529–541. Springer International Publishing, Cham.

Koller, D. and Friedman, N. (2009). *Probabilistic graphical models: principles and techniques*. MIT press.

Lee, J.D. and See, K.A. (2004). Trust in automation: Designing for appropriate reliance. *Human factors*, 46(1), 50–80.

Nam, C., Walker, P., Lewis, M., and Sycara, K. (2017). Predicting trust in human control of swarms via inverse reinforcement learning. In *2017 26th ieee international symposium on robot and human interactive communication (ro-man)*, 528–533. IEEE.

Wang, X. and Wang, Y. (2017). Co-design of control and scheduling for human–swarm collaboration systems based on mutual trust. *Trends in Control and Decision-Making for Human–Robot Collaboration Systems*, 387–413.

Wang, Y., Humphrey, L.R., Liao, Z., and Zheng, H. (2018). Trust-based multi-robot symbolic motion planning with a human-in-the-loop. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 8(4), 1–33.

Xu, A. and Dudek, G. (2015). Optimo: Online probabilistic trust inference model for asymmetric human-robot collaborations. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*, 221–228.