Modeling Teamwork in Supervisory Control of Multiple Robots

Fei Gao, Member, IEEE, Mary L. Cummings, Senior Member, IEEE, and Erin Treacy Solovey, Member, IEEE

Abstract—Simultaneously controlling multiple robots requires multiple operators working together as a team. Determining how to construct the team to promote performance and reduce workload are critical questions that must be answered in these settings. To this end, we investigated the effect of team structure and scheduling notification on operators’ performance, subjective workload, work processes, and communication using a human-in-the-loop experiment. In an urban search and rescue setting, we compared a pooled condition, in which team members shared control of 24 robots, with a sector condition, in which each team member controlled half of all the robots. For scheduling notification, an alert was given when the operator spent too much time on one robot and either suggested or forced the operator to change to another robot. A discrete-event simulation model was constructed to model the teamwork in supervisory control of multiple robots. The model was significantly improved by the inclusion of a behavior termed as “backup.” Backup behavior is a critical coordination mechanism often observed in teams, but rarely explicitly modeled. Pooled teams showed an advantage when performing backup behaviors in both the experiment and the model. However, other factors must be considered when making a decision on what team structure to use.

Index Terms—Backup behavior, discrete-event simulation (DES), human supervisory control, robots, teamwork.

I. INTRODUCTION

Advances in technology have enabled increasingly sophisticated automated systems to be applied to a number of fields including manufacturing, aviation, command and control, search and rescue, air traffic control, and healthcare. Unlike autonomous systems designed primarily to take humans out of the loop, many future systems will support people and agents working together. Despite the benefits of such automation technologies, challenges exist for the successful integration of human operator and automation technologies.

In recent years, there is an increasing interest in enabling one operator controlling multiple agents with higher levels of autonomy. By releasing the operators from manual control, enhanced autonomy enables operators to work with multiple agents and perform a more diverse set of tasks requiring monitoring, coordination, and complex decision-making. However, the required cognitive load for working with multiple agents could easily exceed the capacity of a single operator, even with high levels of automation. Teams of humans are increasingly called upon to perform complex cognitive tasks that are less efficiently done or impossible to do by an individual. Operators in such teams typically have to communicate in order to make effective decisions including the distribution or assignment of tasks, updating of status, seeking help, maintaining coordination, and exchanging information. Although teamwork may impose extra workload related to coordination and communication, teams have the potential of offering greater adaptability, productivity, and creativity than any one individual can offer [1].

Backup behavior is a critical coordination mechanism that teams employ to reduce the risk of errors and maintain performance. Backup behavior refers to “the extent to which team members help each other perform their roles” [2]. Team members may provide different forms of backup, such as assisting the teammate who is behind in his or her work in performing a task, completing a task for the team member when an overload is detected, helping a fellow team member correct performance-related mistakes, and providing resources or supplies [2], [3].

Despite the importance of backup behavior, limited research has been devoted to quantitatively investigating its impact on overall team performance. In this study, we used discrete-event simulation (DES) to model the teamwork of operators during supervisory control of multiple robots, predict their performance and explore the role of backup behavior in team coordination. It is the first quantitative model of backup behavior.

DES has been used to model a single operator’s supervisory control of multiple robots in previous research, where the robots requesting assistance are thought of as customers waiting in queues and the operators are thought of as servers [4]–[6]. Many of the interesting teamwork problems cannot be solved analytically using the queueing theory since some of the strict assumptions necessary for closed-form solutions do not hold. However, it is possible to use DES to overcome the limitations of analytical models. More importantly, DES modeling has the advantage of capturing the process and dynamics of teamwork, which is lacking in previous research. With simulation models, we can test and compare proposed changes to the current system, or new designs of the system at a significantly lower cost than testing directly in real world.

This study modeled the teamwork in a dyad during supervisory control of multiple robots. Since the operators were not differentiated by role or task, the term team was not used in...
the traditional sense [7]. This paper is organized as follows. Section II reviews previous research on teamwork and backup behavior. Section III introduces the key model constructs. Section IV presents the methodology, main results, and observations from an experiment of teamwork during supervisory control of multiple robots. Section V describes the DES model built using the experiment data, as well as the comparison between model outputs and experiment results. Section VI explores the role of backup behavior using the DES model by varying task uncertainty, operator capability, and the level of individual effort. Section VII contains a discussion of the results and conclusions.

II. BACKGROUND

A. Backup Behavior

Backup behavior is critical to the effectiveness of teamwork. It is positively related to team performance when teams have a member with a large amount of workload [2]. Backup behavior can improve performance outcomes by redistributing the workload within the team. More importantly, backup behavior affects team processes to allow greater team adaptability in changing situations and environments. On the other hand, some research has found that backup behavior can be harmful when backup providers neglect their own tasks, especially when workload is evenly distributed [8].

In the behavioral markers of teamwork breakdowns proposed by Wilson et al. [9], backup behavior is identified through three aspects: 1) Did team members correct other team members’ errors? 2) Did team members provide and request assistance when needed? 3) Did team members recognize when one performed exceptionally well? In this study, we focused explicitly on the first two aspects of backup behavior.

Backup behavior is closely related to other factors affecting teamwork. First, whether team members can shift the workload within the team is largely determined by the team structure. Second, backup behavior usually happens together with mutual performance monitoring and communication. When team members detect an error made by their teammates through mutual performance monitoring, communication is often required for information exchange. Infrequent communication may not supply enough information to achieve desired levels of performance [12].

B. Team Structure

Team structure is an important factor that is hypothesized to affect team effectiveness [10]. One aspect of team structure is the “manner in which the task components are distributed among team members” [11]. How the team is structured is closely related to communication, coordination, and team performance.

The team structure that is suitable for a specific scenario largely depends on the task characteristics and resources available [12]. For a team of operators working together with multiple robots, two possible ways to organize the robots are as sectors or as a shared pool [13]. Under the sector condition, each operator exclusively controls a portion of all the robots. Under the shared pool condition, operators share the control of all the robots and service them as needed. Sector assignment, which is how modern day air traffic control is architected, can reduce the number of robots the operator must monitor and control. However, the shared pool condition offers a more flexible scheduling advantage of load balancing since any operator in the team can service any robot as needed, which is one important aspect of backup behavior. Previous research by Lewis and Wang et al. [10] investigated the effect of team structure in an urban search and rescue (USAR) setting. Although there was no significant difference on performance, teams that shared the control of all robots were found to have slightly lower workload. In addition, for monitoring applications, the shared pool offers a redundant observer advantage, such that a second operator with partially overlapping perceptual judgments may detect problems missed by the first operator.

C. Mutual Performance Monitoring

Mutual performance monitoring is the ability to develop common understandings of the team environment and apply appropriate task strategies to accurately monitor teammate performance [3]. It usually involves behaviors such as identifying mistakes and lapses in other team members’ actions, and providing feedback regarding team member actions to facilitate self-correction. Research has shown that individuals may not be aware of their own performance deficiencies [14], [15]. Salas et al. [3] proposed that it is the information gathered through mutual performance monitoring that affects team performance by identifying errors or lapses, and this information, expressed through communication and backup behavior, boosts the team performance.

D. Team Communication

Mutual performance monitoring and backup behavior are usually facilitated by communication within the team [3]. Research about group decision making [16] shows that in effective decision-making groups, communication serves both promotive functions that facilitate sound reasoning and critical thinking as well as counteractive functions that prevent a group from making errors. Communication relates to building an accurate understanding of team members’ needs, responsibilities, and expected actions [12], which is the foundation for mutual performance monitoring. In addition, when an error or overload is detected, communication is often required for information exchange. Infrequent communication may not supply enough information to achieve desired levels of performance.

On the other hand, communication takes time and carries a coordination cost. Research has investigated the negative effects of communication in terms of increased workload and decreased performance [12]. It contributes a part of process loss, which means team performance is lower than the combination of individual performance due to the extra work on team coordination. MacMillan et al. [12] investigated the cost of coordination and communication in a team of six persons that are performing a joint task force mission of air-based and sea-based operations. They concluded that a lower need for coordination and a lower communication rate were associated with better performance.
There is no simple answer to how much communication is appropriate, because it is impacted by factors such as task characteristics, team structure, level of workload, etc. [17], [18]. While frequency of communication about task and team rapport relates to superior team performance, excessive word usage has a negative association with team performance [19]. From the aspect of supporting backup behavior, the effectiveness of communication depends on whether the communication is needed at the time. Communication causes backup providers to dedicate resources on it, and thereby, reduces the amount of resources that are available for other tasks [8]. If the workload is evenly distributed, spending too much effort to communicate and provide backup could be harmful. If the workload is not evenly distributed, the benefits of backup behavior may outweigh the losses resulted from communication. To achieve effective team performance, teams should communicate adequately and effectively, using backup when needed. Conversely, teams should communicate relatively little when backup is not needed.

III. DISCRETE-EVENT SIMULATION MODEL OF MULTIHUMAN MULTIAGENT TEAM

The key constructs of DES models are events, arrival processes, services processes, and queuing network structure. The DES model for this effort was constructed under the assumption that operators are acting in a supervisory control mode and the robots in the team are highly autonomous. Robots should function independently of the human most of the time, and require human interaction only intermittently. Operators function as servers in the queuing model and serve the events generated from the robots. The overall framework is shown in Fig. 1.

The events generated from the robots enter the queue and wait to be served when the operators are busy. Operators select the next event to be served from the queue. This task assignment process is affected by the team structure of operators. After the events are served, the model generates performance outputs, which can be compared with empirical data. Communication between the teammates is modeled as two parts. Baseline communication happens with a random interarrival time. The other part of communication happens during task assignment and error correction.

A. Arrival Process of Robot-Generated Events

Robot-generated events arise due to the nature of the mission and robot capability [5], such as detecting a victim or getting stuck and needing operator intervention. An event arrives to the system and stays in a queue for a time $T$. An event is then either served by the operator or exits the queue without being served if it waits longer than $T$. Unlike independent arrival processes in many queuing systems, the arrival of robot-generated events usually depends on the system status. To model the dependent arrival process, we limit the number of active events in the system associated with each event stream to be one at any time. In other words, a new event is generated from this stream only if there is no event from the stream in the queue or being served. The interarrival times of events are between the completion of service/reneging from the queue and the arrival of the next event. These interarrival times are described by a random variable $\Lambda_i$, where $i$ stands for event stream $i$.

Sometimes, events generated are not identical. In this situation, a random variable $C$ following a multinomial distribution is used to describe the categories of events. New events are generated according to the interarrival time $\Lambda_i$ and assigned an event category from $C$.

B. Service Process of a Single Operator

Each event is served for a service time described by random variable $M$. The service process typically involves several steps. In this case, the service time $M = \mu_1 + \mu_2 + \cdots + \mu_\alpha$, with $\mu_i$ being the time required for step $i$. The time an operator spends working on an event is associated with an opportunity cost of missing other important events waiting in the queue. Limiting the service time on one event may result in an increase on the number of tasks processed. Although the error rate may increase, it is possible that the overall mission performance is improved. This could be modeled by making the event exit from the server when the time limit is reached. In this situation, an output from processing this event may not be generated due to the shortened service time.

The queuing policy determines the order in which multiple events that are waiting in the queue are served. Several common ways to pull an event from a queue include: first-come-first-serve, last-come-first-serve, and random selection. They can be used in the DES model according to characteristics of different task scenarios.

C. Team Structure and the Shift of Workload

Queuing networks are systems with multiple queues and service centers that are connected by customer routing. By connecting queues and service centers in different ways, various team structures can be modeled. For teams organized under the Sector condition, each operator has his or her own queue. Robot-generated events enter the queue of one operator according to...
the task assignment, which is determined before the start of mission. Since operators only pull events from their own queue to serve, shifting of workload is impossible. For teams organized under the shared pool condition, events generated from the robots all enter the same queue, and operators pull events from this single common queue to serve. In this case, shifting of workload is possible.

In the literature of the queuing theory, whether to have a single common queue or multiple separate queues has been investigated. For a queuing system of $s$ servers with Poisson arrival process and exponentially distributed service time, comparison has been made between having $s$ queues and one single common queue based on the steady-state average waiting time. It was proved that, with the same arrival rate and service rate, the system with multiple servers and a single common queue has a shorter average waiting time than assigning a separate queue to each server [20]. The main reason for this advantage is the shifting of workload between multiple servers enabled by the single queue. Pooling multiple queues into a single queue may not always be beneficial [21]–[23]. In operator teams that supervise multiple robots, factors such as differences in individual capability, uncertainty in task load and individual level of effort may impact the choice of team structure.

D. Mutual Performance Monitoring and Communication

In our DES model, mutual performance monitoring is modeled as a higher probability of correcting an error when there is communication. With communication, an operator is able to correct his teammate’s errors, in addition to his own errors. However, as discussed previously, communication has been shown to have both a positive and negative influence on team performance.

We modeled communication by separating its impact into positive and negative aspects. Theoretically, performance improves when positive impact outweighs the negative impact and vice versa. From the negative aspect, communication time is modeled as process loss. When there is communication during the service process, the service time is extended by the duration of communication. If the communication is too often or too long, number of services completed within a certain time period is decreased. From the positive aspect, the benefit of communication is modeled together with mutual performance monitoring. When an error is detected, it is corrected with a probability $P(\text{Correction}) = p$ without communication. If a communication event happened at the time, the probability is increased so that $P(\text{Correction}) = p + p'$.  

IV. MULTI-HUMAN MULTI-ROBOT TEAM EXPERIMENT

This experiment investigated the effect of team structure and scheduling notification on participants’ team performance, workload and communication in an USAR task. Participants supervised multiple simulated robots, manipulating and viewing the imagery the robots provided in order to detect and mark the locations of victims. Empirical data were collected to obtain insight into teamwork during supervisory control of multiple robots and support the development of a simulation model.

A. Participants

The study, IRB approved, adhered to ethical guidelines for the treatment of human participants. A total of 48 participants, 19–47 years old, participated in the experiment. The average age was 26.6 years, with a standard deviation (SD) of 5.5. Among them, 19 were female and 29 were male. Thirty-three of the participants were undergraduate or graduate students, and 15 had other occupations. Twenty-two of the participants did not play video games regularly. The average time playing video game per week for the remaining 26 participants was 4.1 h (SD = 4.9). The correlation between hours spent on video games and average individual performance was not significant ($r = 0.129, p = 0.354$). Of all 24 teams formed by the 48 participants, team members in four teams knew each other before the experiment. The other 20 teams were formed by strangers.

B. Independent Variables

A $2 \times 3$ mixed design was used to evaluate team structure (two levels) and scheduling notification (three levels). Team structure was a between-subject variable with 24 participants assigned to one of the two types of team structure.

1) In Sector ($S$) teams, each participant controlled half of all the robots, for a total of 12 robots each. Locations of their teammates’ robots were shown on the map, but video feed from their teammates’ robots could not be seen.

2) In Pool (P) teams, two participants shared the control of all the robots. They were able to see the video feed of all robots and could switch control of any robot not under the control of a teammate.

The level of interdependence was not high in both team structures. Independent works of team members were combined to represent team output [24]. However, Pool teams allowed more coordination and communication between the team members compared with Sector teams.

The second scheduling variable evaluated the utility of a cue indicating that attention should be switched to a different robot. This variable is of interest because previous work in automated visual search task allocation for single operator-multiple unmanned vehicle environment has shown that automated scheduling notification can improve operator performance in terms of probability of detection for overall mission and decrease workload by influencing switching times [22]. This form of scheduling notification was hypothesized to be beneficial also in the context of team scenarios, where resources were distributed across operators, who could benefit from recommendations for when to switch to new search tasks. For this experiment, scheduling notification was employed that issued text notification layered over the video panel and a beeping sound at an appropriate time to cue the participant to interrupt the current search pattern for one robot and switch to a different robot.

Each participant completed one session with each level of the scheduling notification. The order these three levels were experienced was counterbalanced across participants. The three levels of scheduling notification were the following.

1) Off condition, no cue was administered. No decision support was provided.
2) **Suggested** condition gave a text notification on the interface with a beeping sound when the participant spent more than 30 s on a robot. Participants were trained that the cue signaled that attention should be switched to a different robot. However, this cue can also be ignored.

3) **Enforced** condition also gave the cue when the participant spent more than 30 s on the same robot, but also, after another 5 s had elapsed with the same robot, automatically switched to another randomly selected robot. Thirty seconds was chosen as the threshold criteria based on a previous study [10], [25], [26]. In previous studies on visual search tasks [25], the possibility of finding a target was shown to decrease as more time was spent on the visual search task. The probability was estimated to be 0.8 for 26 s spent on searching, and then, declined. In another experiment for USAR tasks [10], the mean time from a robot being selected to a victim being marked under autonomous control was approximately 35 s. We selected 30 s as the threshold so that the participant was given a reasonable amount of time to finish the task if a victim was successfully located and yet was prevented from spending too much time on a low probability search task if the participant failed to locate the victim. This threshold was validated in pilot tests as well.

C. **Testbed**

**USARSim**, a robotic simulation performing USAR tasks [27], was used to provide the underlying simulation for the testbed. Multirobot control system (MrCS), a multirobot communication and control infrastructure with an accompanying user interface was used as the control interface. MrCS provided facilities to start and control robots in the simulation, displaying camera, and laser range finder output, and supporting interrobot communication through Machinetta, a distributed multiagent system developed at Carnegie Mellon University Pittsburgh, PA, USA [28].

In MrCS, each robot was capable of updating a map, planning its routing, and sending video feed to participants. An occupancy grid was used to represent the joint robot team knowledge of the environment and available information about the planned paths of other robots, as shown in Fig. 2. Possible locations were generated and filtered based on the expected information gain for being at that location. Edges were generated between locations if there was a sufficiently high possibility to move between the locations. A branch-and-bound search was performed across the network of possible locations and edges for the path that maximized the expected information gain. Plans were allowed to backtrack with no additional value added for visiting a location multiple times. When a robot finished planning, it shared its planned path with some nearby robots to allow them to both avoid collisions and search different areas.

MrCS was displayed on a dual display computer as shown in Fig. 3. The robot camera list on the left screen shows thumbnails of camera feeds. Video panel shows a video feed of interest. The teleoperation panel allows teleoperation and camera pan and tilt. The right shows the current area map with the positions of robots, and allows participants to mark the location of victims.

In this experiment, robots were started automatically in different regions and explored the environment based on an autonomous path planner. The participants’ tasks were to identify as many victims as possible and mark their locations on a map. When a victim appeared in the camera of a robot and was detected by the participant, he or she could select a robot either from the robot camera list or by clicking the icon of the robot on the map. When a robot was selected, the thumbnail of its camera was highlighted with a thick black border, the video feed from its camera was shown in the video panel, and its field of laser was highlighted on the map. The participant could stop the selected robot and move the robot manually using the teleoperation panel to bring the victim back into the camera view or fine tune the robot’s position if necessary. The participant could then double click on the map to mark the position of this victim.

If the participant wanted to delete the mark, he or she clicked the mark and pressed the Del button on the keyboard. When no robot was under direct control, the participant continued monitoring all the robots while exploring the environment until a new victim in a camera view was noticed. Most of the time robots navigated using autonomous path planning by default, and the participant only needed to monitor the thumbnails of video feeds. However, the participant could also choose to manually control the robots using the teleoperation panel to send them to a specific unexplored place.

The team members were located in the same room, each with one display station. The stations were located so that it was difficult to view the teammates’ display in detail, although quick glances were allowed. For **Pool** teams, participants could see video feeds and locations of all robots. The status of the robot was shown on top of each thumbnail of camera view, as highlighted in Fig. 3.

The default status was AUTONOMOUS, which means the robot was navigating automatically. When one of the team members was teleoperating the robot, its status was changed to TELEOP on both team members’ display. As a result, the other team member would know that this robot was controlled by his or her teammate. Conflict may happen when two operators tried to teleoperate the same robot, which then required verbal communication between team members. In **Sector** teams, video feeds and locations of 12 robots were shown on the display. Under both team structure conditions, participants could
see their marks and their teammates’ on the map in different shades of red. Participants could communicate with their teammates verbally with no restrictions.

Scheduling notification was administered by layering text on top of the video feed in the video panel together with a beeping sound generated by repeating the Windows system default beep sound for 5 s. Under Off condition, there was no scheduling notification. In Suggested condition, participants could choose to follow the notification and move on to a new robot, or to ignore it and stay with the current robot. If ignored, the text and the beeping sound lasted for 5 s, and then, disappeared. No more notification would be administrated afterwards. For the Enforced condition, the system would switch to another randomly selected robot 5 s after the original notification. Whenever a new robot was selected, either by the participant or the system, the previous robot would continue to navigate using autonomous path planning. The new robot selected was in autonomous mode by default. The autonomous path planning would stop only if the participants chose to teleoperate.

### D. Procedure

The experiment began with a 15-min training session prior to three 25-min test sessions. A training session allowed the participants to practice the operation of GUI, especially teleoperation. Enforced scheduling notification was used during this training session because it was the most complex one among three conditions. No training for communication strategy was provided. Participants were tested in groups of two in the same room. Each participant controlled either 12 (Sector) or all 24 (Pool) robots, depending on their team structure assignment. Each pair of participants performed all three scheduling notification conditions. The three conditions were randomized and counterbalanced to limit any learning effect. Audio and screen recordings were collected during the experiment.

### E. Dependent Variables

Dependent variables included task performance metrics, operation measures, communication as a team measure, and subjective workload. All the dependent variables are summarized in Table I, along with their definitions.

![Fig. 3. Interface for controlling robots.](image)

**Table I**

**Dependent Variables**

<table>
<thead>
<tr>
<th>Category</th>
<th>Dependent Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task performance metrics</td>
<td>Found: number of victims marked in the correct position</td>
</tr>
<tr>
<td></td>
<td>Error: number of marks in the wrong position</td>
</tr>
<tr>
<td></td>
<td>Deletes: number of marks deleted</td>
</tr>
<tr>
<td></td>
<td>Missed: number of victims that appeared in the camera but were not marked</td>
</tr>
<tr>
<td>Operation Measures</td>
<td>Teleoperation duration: length of teleoperation period before marking victim or robot selection</td>
</tr>
<tr>
<td></td>
<td>Teleoperation frequency: number of teleoperations</td>
</tr>
<tr>
<td></td>
<td>Total teleoperation time: total amount of time spent on teleoperation</td>
</tr>
<tr>
<td></td>
<td>Display-to-mark time: time from victim appearing in the camera to being marked</td>
</tr>
<tr>
<td></td>
<td>Select-to-mark time: time from robot selection to victim being marked</td>
</tr>
<tr>
<td>Team Measure: Communication</td>
<td>Communication time: total time spent communicating with team member</td>
</tr>
<tr>
<td>Workload</td>
<td>NASA-T LX rating</td>
</tr>
</tbody>
</table>
when a victim was not marked accurately or was marked more than once. Participants’ actions of marking, deleting, as well as teleoperation were recorded in the system log, and used to calculate the dependent variables. Total communication time, average duration, and frequency were measured in both team structures. Subjective workload ratings were obtained through the NASA-TLX [29], which is a rating along six subdimensions. The dependent variables were analyzed using either analysis of variance (ANOVA) or nonparametric tests if they did not satisfy the ANOVA assumptions of normality and/or homogeneity.

F. Results
This section introduces the main results of the experiment, including team performance and evidence of backup behavior, as well as communication and error correction. Detailed results on the experiment can be found in a previous paper [30]. This paper extends the previous one by building a DES model based on the experiment data and exploring several scenarios using the model, presented in Sections III, V, and VI. Data from the training session were not included in the analyses. A significance level of 0.05 was used for the analyses.

1) Task Performance and Operation Measures: Team structure had no significant impact on task performance in terms of number of victims found, number of errors, and number of victims missed. For number of deletes, Pool teams (Mean = 8.3, SD = 5.49) tended to delete more than Sector teams (Mean = 6.0, SD = 3.02), although the effect of team structure was not significant (Z = −1.838, p = 0.066, r = −0.217). This indicates that Pool teams corrected themselves more often, because the marks were in wrong locations or duplicated marks were made for the same victim.

Team structure had a significant effect on the total time of teleoperation (F(1, 138) = 10.68, p = 0.001, n² = 0.072). Sector teams (Mean = 1166.7, SD = 194.20) spent more time on teleoperation than Pool teams (Mean = 1055.4, SD = 236.66) on average. No significant effect was found for teleoperation duration, teleoperation frequency, display-to-mark time or select-to-mark time. The interaction effect of team structure and scheduling notification was not significant on any of the dependent variables.

Scheduling notification did not improve or decrease performance, but had an influence on working process. Scheduling notification had a significant effect on duration (F(2,138) = 21.64, p < 0.001, n² = 0.239) and frequency of teleoperation (F(2,138) = 16.62, p < 0.001, n² = 0.194), due to the way scheduling notification was implemented. With scheduling notification, the duration of teleoperation dropped and the frequency increased. The Enforced condition resulted in the shortest duration and highest frequency of teleoperation, followed by the Suggested and Off conditions. The effect of scheduling notification on total time of teleoperation was not significant (F(2,138) = 2.61, p = 0.078, n² = 0.036).

Scheduling notification helped the participants to notice and mark victims faster when they appeared in the camera, which is important for such a time-critical task environment.

In Sector teams, scheduling notification had a significant effect on mean display-to-mark time (F(2, 69) = 3.91, p = 0.024, n² = 0.102). The teams under Suggested condition had the lowest mean display-to-mark time (Mean = 88.0 s, SD = 58.9 s), followed by Off condition (Mean = 103.2 s, SD = 59.1 s) and Enforced condition (Mean = 128.6 s, SD = 70.8 s). The increase in time under the Enforced condition may be due to the interruption in the current operation and extra time to regain situation awareness.

In Pool teams, the effect of scheduling notification was insignificant, which suggest that display-to-mark time was affected by team process. One team member could start working on a robot with a victim in view when the other was busy. For select-to-mark time, scheduling notification was found to have a significant effect (F(2, 138) = 24.77, p < 0.001, n² = 0.264), shortening the time to finish a task.

2) Evidence of Backup Behavior: Subjective workload using NASA-TLX was analyzed using nonparametric tests. Box plots of subjective workload under different conditions are shown in Fig. 4. Mann–Whitney tests for the effect of team structure showed a significant effect on workload (Z = 2.036, p = 0.042, r = 0.170). Pool teams demonstrated lower workload on average than Sector teams. When analyzing each dimension of workload (mental demand, physical demand, temporal demand, performance, effort, and frustration) separately, Pool teams had a significant lower rating on effort (Z = 2.148, p = 0.032, r = 0.179) and frustration (Z = 2.799, p = 0.005, r = 0.233).

This was consistent with a previous study [10], in which a slight advantage in workload was observed favoring the Pool structure. One reason may be that in the Sector teams, there was no opportunity for backup. Furthermore, in Pool teams, it was possible to balance the workload according to operators’ individual abilities. When one operator was better at finding victims, it was possible he/she could share the burden of the less skilled teammate and did not report excessive workload. We analyzed the maximum, minimum, and averaged workload of each team based on two team members’ individual workload ratings. This showed that maximum workload of the team members in Pool teams was significantly lower than in Sector teams (F(1, 68) = 6.6, p = 0.012, n² = 0.089), while minimum workload did not differ significantly, as shown in Fig. 4.
This result, combined with the significantly larger difference on individual performance (number of victims found) in Pool teams ($F(1,68) = 4.72, p = 0.033, \eta^2_p = 0.065$) as shown in Fig. 5, suggests workload balancing processes or backup behaviors in Pool teams.

3) Communication and Error Correction: During the experiment, participants were allowed to talk with each other. In such a high workload scenario, almost all the communication was mission related. Some teams discussed what strategies to use when exploring the area, updated their status with the teammate, requested their teammates’ status or shared experiences about robot control. In contrast, some teams did not communicate at all. An analysis of the time (seconds) spent on communication showed that team structure had a significant effect ($F(1, 66) = 12.53, p < 0.001, \eta^2_p = 0.160$). Pool teams (Mean = 177.7, SD = 198.74) expectedly communicated more than Sector teams (Mean = 53.44, SD = 80.97), on average.

The four teams with members that knew each other before the experiment tended to communicate more compared with other teams. The effect of team structure on communication time was still significant when this factor was controlled. Pool teams also had significantly longer communication duration (Mean = 58.0, SD = 5.04, $F(1,66) = 5.85, p = 0.018, \eta^2_p = 0.081$) than Sector teams (Mean = 3.37, SD = 3.00), and higher frequency (Mean = 31.17, SD = 41.50, $F(1,66) = 8.66, p = 0.004, \eta^2_p = 0.116$) than Sector teams (Mean = 9.5, SD = 13.46). Scheduling notification did not have a significant effect on communication time ($F(2,66) = 0.01, p = 0.986, \eta^2_p < 0.001$).

Further analyses on the correlation between communication with team performance and subjective workload revealed that communication time was moderately negatively correlated with errors ($r = -0.309, p = 0.008$). In other words, teams that communicated more tended to make fewer errors. This correlation existed even if we controlled for whether the teammates knew each other ($r = -0.280, p = 0.018$). This negative correlation between communication time and number of errors existed in Pool teams but not in Sector teams. This result, combined with the larger number of deletes in Pool teams, suggests that these participants engaged more in mutual performance monitoring, facilitated by communication. No significant correlation was found between communication time and number of victims found, number of deletes, number of victims missed, or subjective workload ratings.

V. DISCRETE-EVENT SIMULATION MODEL REPLICATION

A DES model was built based on the process data and observations from the experiment to simulate team performance in these search and rescue tasks. In order to determine the model’s ability to describe the observed data, we compared the DES model outputs with the experimental results. Several datasets were recorded in the experiment and used to fit probability distributions applied in the model, as shown in Table II.

Robot-generated events occurred when victims appeared in the robot camera. Although all the robots were the same type, they were started in locations with different victim density. We modeled the interarrival time of each robot individually to account for this difference. Another attribute of the robot-generated events was event identity (ID), which corresponded to different victims in the experiment. A victim already marked may appear in the camera again, which may be ignored or reprocessed to check for an error. This was modeled by filtering the arrived events by their ID. Event ID was generated from a Multinomial distribution with $n = 34$.

Performance measures generated from the model were number of victims found, number of errors, number of deletes, and number of victims missed. In this model, the event when a victim appeared in the camera was defined as a robot-generated event. These events enter the queue, and were then, served by operators. Events that exit the queue without being served were measured as number of victims missed. Events processed were separated as victims found or errors based on a Bernoulli distribution. An event arrived with its ID already in the error list was defined as error detected in the model. In Sector teams, operators corrected one’s own error by a probability $P(\text{Correction}) = p$. In Pool teams, the common queue made it possible to detect and correct the teammate’s error as well as one’s own, during which communication was often required. This was modeled by an increase of $P(\text{Correction})$ to $p + p'$ when communication happened during error correction. To correct an error, operators deleted a wrong mark and added a correct one, which was measured by number of deletes. The error correction process is presented in Fig. 6.

Sector teams were modeled by using two different queues, each for one operator. Operator 1 served events generated by robot 1–12, and operator 2 served events generated by robot 13–24. Pool teams were modeled by using a single common queue for the two operators. To model the scheduling notification, we added a time limit of 30 s to the service time. For Suggested condition, service was stopped at 30 s if the operator followed the system recommendation with a probability 0.7. For the Enforced condition, service was stopped at 30 s. If the service was stopped, it was possible that the service did not generate a victim found or an error, the probability of which is set to be 0.6.

We compared the team performance measures generated by the model with those collected in the experiment. One thousand trials were conducted using the DES model under each combination of team structure and scheduling notification mode. Fig. 7 shows the comparison between simulation outputs and experiment results of Sector [see Fig. 7(a)] and Pool [see Fig. 7(b)] teams under Off notification condition with their standard error,
Table II

Model Parameters and Data Recorded During the Experiment

<table>
<thead>
<tr>
<th>Parameters in the Model</th>
<th>Distributions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrival of Robot Generated Events</td>
<td>I(G(112.76, 5.07), Exp(43.21), Exp(65.13), Exp(85.76), I(G(101.55, 3.66), I(G(116.56, 8.03))</td>
</tr>
<tr>
<td>Interarrival Time of Each Robot</td>
<td>I(G(142.08, 5.34), Exp(51.81), Exp(72.00), Exp(92.27), I(G(88.20, 4.38), I(G(106.79, 10.09))</td>
</tr>
<tr>
<td>Duration</td>
<td>Exp(2: 15.31)</td>
</tr>
<tr>
<td>Event ID</td>
<td>Multinomial (0, 0.1, 0.1, 0.1, 0.2, 0.2, 0.2, 0.3, 0.3, 0.5, 0.6, 0.7, 0.8, 1.1, 1.1, 1.3, 1.4, 1.4, 1.5, 1.8, 1.9, 2, 2.2, 2.4, 2.9, 3, 5, 6, 15, 17, 17.7)</td>
</tr>
<tr>
<td>Service</td>
<td>Weibull (λ: 30.06, k: 0.87)</td>
</tr>
<tr>
<td>Telesoperaion Time</td>
<td>Bernoulli (p: 0.38)</td>
</tr>
<tr>
<td>Probability of doing telesoperaion</td>
<td></td>
</tr>
<tr>
<td>Communication</td>
<td>Gamma (k: 0.23, θ: 139.74)</td>
</tr>
<tr>
<td>Interarrival time</td>
<td>Exp (λ: 50.88)</td>
</tr>
<tr>
<td>Duration</td>
<td>Pool: Bernoulli (p: 0.82); Sector: Bernoulli (p: 0.56)</td>
</tr>
<tr>
<td>Probability to communication</td>
<td></td>
</tr>
<tr>
<td>Error and Correction</td>
<td>Pool: Bernoulli (p: 0.5); Sector: Bernoulli (p: 0.4)</td>
</tr>
<tr>
<td>Probability to make an error</td>
<td>Bernoulli (p: 0.4)</td>
</tr>
<tr>
<td>Probability to correct an error</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 6. Error correction process.

Fig. 7. Comparison between model outputs and experiment results for performance in (a) Sector teams with Off scheduling notification, and (b) Pool teams with Off scheduling notification.

and 95% confidence intervals with a modified degree of freedom [31] for the difference between simulation outputs and experiment results are included. These confidence intervals contain zero, indicating no significant differences between simulation outputs and experimental results. The comparison under Suggested and Enforced conditions showed similar results, as listed in Table III. These indicate that the model can successfully capture the essential elements of teamwork and replicate the experimental results on team performance under all team structure and notification conditions. The match between the model and the experiment provides a foundation for further exploration using the model.

Communication time as an important team process measure was also compared. Communication is difficult to model. Although there is much research about communication, it is unclear when people will communicate, what they will communicate, and how that will impact the team performance. In the DES model, we simplified the communication as one special type of event. As observed in the experiment, Sector teams communicated less than Pool teams because the operators were less interdependent. Based on this observation, we modeled the baseline communication as an exogenous process with an interarrival time estimated based on communication data in the Sector teams collected in the experiment. In Pool teams, we modeled another two components of communication in addition to the baseline communication: communication during task assignment and during error correction. Communication duration was modeled with an Exponential distribution estimated from experiment data. The comparison between simulation outputs and experiments results for communication time is shown...
TABLE III

<table>
<thead>
<tr>
<th>Measures</th>
<th>Experiment</th>
<th>Simulation</th>
<th>Confidence Interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>Suggested</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pool</td>
<td>Found</td>
<td>18.67</td>
<td>1.82</td>
</tr>
<tr>
<td></td>
<td>Error</td>
<td>8.17</td>
<td>1.34</td>
</tr>
<tr>
<td></td>
<td>Deletes</td>
<td>8.42</td>
<td>1.88</td>
</tr>
<tr>
<td></td>
<td>Missed</td>
<td>2.67</td>
<td>0.62</td>
</tr>
<tr>
<td>Suggested</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sector</td>
<td>Found</td>
<td>18.33</td>
<td>1.02</td>
</tr>
<tr>
<td></td>
<td>Error</td>
<td>7.00</td>
<td>1.27</td>
</tr>
<tr>
<td></td>
<td>Deletes</td>
<td>6.00</td>
<td>0.93</td>
</tr>
<tr>
<td></td>
<td>Missed</td>
<td>3.17</td>
<td>0.56</td>
</tr>
<tr>
<td>Enforced</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pool</td>
<td>Found</td>
<td>19.58</td>
<td>1.68</td>
</tr>
<tr>
<td></td>
<td>Error</td>
<td>6.75</td>
<td>1.34</td>
</tr>
<tr>
<td></td>
<td>Deletes</td>
<td>6.83</td>
<td>1.24</td>
</tr>
<tr>
<td></td>
<td>Missed</td>
<td>2.83</td>
<td>0.71</td>
</tr>
<tr>
<td>Enforced</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sector</td>
<td>Found</td>
<td>19.08</td>
<td>1.28</td>
</tr>
<tr>
<td></td>
<td>Error</td>
<td>7.33</td>
<td>1.50</td>
</tr>
<tr>
<td></td>
<td>Deletes</td>
<td>7.17</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>Missed</td>
<td>2.00</td>
<td>0.46</td>
</tr>
</tbody>
</table>

Fig. 8. Comparison between model outputs and experiment results for communication time.

in Fig. 8, which shows the DES model replicated the experiment results of communication time.

VI. EXPLORING BACKUP BEHAVIOR USING THE DISCRETE-EVENT SIMULATION MODEL

With confidence in the DES model after the comparison, we simulated three scenarios using the DES model to further investigate backup behavior: the uncertainty in task load, the difference in individual capability and the level of individual effort. We wanted to observe whether the team members could back up each other and adapt to the uncertainty by providing assistance when needed. We analyzed the shift of workload in Pool and Sector teams under these scenarios.

A. Uncertainty in Task Load

Uncertainty in task load is an important factor that affects the balance of workload within the team. In the real world, tasks are rarely evenly assigned to team members. In the search and rescue scenario, operators do not know where victims are beforehand in order to make a plan for the search. As a result, teams have to adapt during the execution of tasks.

In the original DES model, half of all the victims would appear in the camera of robot 1–12, and the other half in the camera of robot 13–24. To bring more uncertainty to the task load, \( m \) victims appeared in the camera of robot 1–12 where \( m \) is generated from a uniform distribution, and the remaining victims appeared in the camera of robot 13–24. One thousand simulation trials were run.

The percentage of events processed by operator 1, and the ratio of busy time of operators were compared in Pool and Sector teams (see Fig. 9). The ratio was calculated as the measure on operator 1 divided by the sum of two operators. A ratio close to either zero or one indicates the overload of an operator. As shown in Fig. 9, the percentage of events processed by the two operators is scattered in Pool teams. In Sector teams, the percentage increased as the number of victims appeared in robot 1–12 increased. The ratio of busy time showed a similar pattern with a larger deviation.

The simulation results can be interpreted from two aspects. First, since operators share the control of all robots, it does not matter whether one group of robots found more victims. Backup behavior can be easily performed to balance the workload. In other words, Pool teams show adaptability with the uncertainty in task load. On the contrary, in Sector teams, an operator experiences more workload if the robots find more victims. His or her teammate cannot offer much help even if idle.

Second, Pool teams have a larger standard deviation for the ratio of events processed and ratio of busy time comparing with Sector teams. Pool teams could have an advantage in maintaining a reasonable workload balance when the task load has little variability. However, when there is large uncertainty in task load, Pool teams have an advantage because of the adaptability enabled by backup behaviors.
B. Difference in Individual Capability

Difference in individual capability was simulated by varying the service time of operators. The distribution for operator 1 was unchanged. The service time of operator 2, originally the same for both, was increased from two to six times of the original. The percentage of change on the number of victims found, percentage of change on the number of victims missed, the ratio of events processed by operators, and the ratio of operator busy time were compared. These ratios were calculated as the measure of operator 2 divided by operator 1. The busy time refers to the total service time. Although operators were sometimes actively searching instead of just monitoring during their idle time, this free searching time is not included in the model output. We assumed that the two operators put similar effort into free searching. Results generated from the model are shown in Fig. 10.

From the results, we can see Sector [see Fig. 10(a)] teams and Pool [see Fig. 10(b)] teams react differently. In Sector teams, the busy time of operator 2 increased rapidly to around three times of operator 1, while the number of events processed by operator 2 was a little bit less than that of operator 1 with the increase of service time. This means operator 2 had to work longer because of the slow service and the lack of help from operator 1.

In Pool teams, because the two operators shared the control of all robots, it was easier to shift the workload within the team. This was confirmed by the DES model outputs. In Pool teams, the busy time of operator 2 increased only to around 1.5 times of operator 1, while the number of events processed by operator 2 was much less than that of operator 1 with the increase of service time. This suggested a shift of workload in Pool teams when one operator was overloaded due to his/her individual capability. The shift of workload also had an impact on team performance. The percentage of change on the number of victims found was small in both types of teams. However, Pool teams had a slower increase on the number of victims missed comparing with Sector teams. Pool teams had an advantage through backup behaviors, which could shift the workload within the team when one operator was slower.

C. Individual Level of Effort

Although Pool teams have an advantage through backup behaviors, they may be affected when some individuals in the team expend less effort when working collectively. In addition to idle and busy states, we simulated individual level of effort by adding a lazy state for operator 2, during which he or she was neither working on tasks nor responding to tasks in the queue. If the operator is idle, he or she enters the lazy state with probability $P(\text{lazy})$. The lazy state lasts for 5 s. Fig. 11(a) shows the impact on the average percentage of events processed by operator 1 in
Pool teams when $P(lazy)$ of operator 2 changes from 0.2 to 0.6. As $P(lazy)$ increases, the curve of operator 1 is shifted upward. In other words, operator 1 processes more events on average if operator 2 is lazy.

With reduced individual level of effort, the degree to which Pool teams are better than Sector teams in terms of balancing workload depends on how the tasks arrive. Fig. 11(b) is a simplification of Fig. 11(a) to illustrate the concept. We represent average percentage of events processed by operator 1 in Pool teams as $r_p$, and in Sector teams as $r_S$. In the shadowed area of Fig. 11(b), $\max(r_p, 1 - r_p) > \max(r_S, 1 - r_S)$, which means workload is more evenly distributed in Sector teams. Beyond this range, $\max(r_p, 1 - r_p) < \max(r_S, 1 - r_S)$, which means workload is better balanced in Pool teams. Based on this, we can conclude that Sector teams are better when the tasks arrive to the two operators evenly, especially when there is reduced individual level of effort in Pool teams. However, Pool teams deal with extreme difference in task load better than Sector teams, even with reduced individual level of effort.

VII. DISCUSSION AND CONCLUSION

In the experiment, participants’ mean ratings indicated lower workload with the Pool structure as compared with the Sector structure, even though task performance was similar across the two types of team structure. Pool teams also communicated more and balanced workload among team members. These conclusions were supported by the lower maximum workload and larger difference on individual performance in Pool teams. This suggests the reduced subjective workload under the Pool condition occurred because teammates could provide backup if needed. In addition, the shared control of robots promoted communication in teams under the Pool structure, which was also good for task performance since teams with more communication tend to make fewer errors. The reason may be that they corrected each other via communication, which led to fewer errors.

A DES model that is based on the queuing theory was constructed to simulate the teamwork in supervisory control of multiple robots. The outputs from the model replicated the experiment results on team performance measures. Backup behavior was investigated by varying the uncertainty in task load, the individual capability and the individual level of effort. In all scenarios, Pool teams show an advantage on balancing workload through backup behaviors. Although Pool teams have an advantage on balancing workload as suggested by both the experiment and the DES model, we must consider other factors, like team strategies and coordination cost when deciding which team structure to use. Pool teams gain the advantage of balancing workload with the cost of increased coordination on task assignment. In our simulation, we found that the advantage of backup behaviors is meaningful only when the task load is unevenly distributed. This conclusion based on simulation results is consistent with several empirical research on backup behaviors [8], [32]. If the task load is evenly distributed with low uncertainty, backup behaviors are not necessary.

Team members also employed certain team strategies to cope with the increased coordination cost. In the experiment, we observed that some operators in Pool teams would preplan on which robots to control via verbal communication, even if the plan changed during the task execution. For example, some divided the robots by robot ID, and some divided by robot location. Other operators reported the robot ID to their teammates whenever they started on a new robot. These team strategies reduced the effort for team coordination while still leaving a possibility for balancing workload. However, as the team size increases, we would expect an increase on the cost and difficulty of coordination. In addition, reduced individual level of effort is easier in Pool teams than in Sector teams. With reduced individual level of effort, the advantage of Pool teams is diminished. To better reflect the tradeoffs of these factors, these factors will be modeled in future research to support the design of teams.

The DES model can be used as a useful and efficient tool to assess the impact of change on performance and share of workload under different team structures. In this study, a dyadic team was modeled. In the future, we would like to investigate whether this model structure can be extended for teams of three or more members.

ACKNOWLEDGMENT

The authors would like to thank MIT undergraduates M. R Redmond and R. Malik for their programming contributions, H. Siu and V. Palay for their help with conducting the experiment, and Prof. M. Lewis and his students at University of Pittsburgh for providing the testbed.

REFERENCES


