Neglect Benevolence in Human Control of Swarms in the Presence of Latency

Phillip Walker, Steven Nunnally, and Mike Lewis
School of Information Sciences
University of Pittsburgh
Pittsburgh, PA 15213, USA
pmw19@pitt.edu, smn34@pitt.edu, ml@sis.pitt.edu

Andreas Kolling, Nilanjan Chakraborty, and Katia Sycara
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
andreas.kolling@gmail.com, nilanjan@cs.cmu.edu, katia@cs.cmu.edu

***Abstract*— Autonomous swarm algorithms have been studied extensively in the past several years. However, there is little research on the effect of injecting human influence into a robot swarm—whether it be to update the swarm's current goals or reshape swarm behavior. While there has been growing research in the field of human-swarm interaction (HSI), no previous studies have investigated how humans interact with swarms under communication latency. We investigate the effects of latency both with and without a predictive display in a basic swarm foraging task to see if such a display can help mitigate the effects of delayed feedback of the swarm state. Furthermore, we introduce a new concept called *neglect benevolence* to represent how a human operator may need to give time for swarm algorithms to stabilize before issuing new commands, and we investigate it with respect to task performance. Our study shows that latency did affect a user's ability to control a swarm to find targets in the foraging task, and that the predictive display helped to remove these effects. We also found evidence for neglect benevolence, and that operators exploited neglect benevolence in different ways, leading to two different, but equally successful strategies in the target-searching task.**

I. INTRODUCTION

Robotic swarms are made up of small, homogeneous robots with limited capabilities that act through local interactions to collectively achieve a variety of behaviors including flocking [1], [2], [3], [4], deployment [5], [6], and foraging [7], [8]. The principal advantage of swarms is that, due to their large numbers and emergent behaviors, they are typically robust to failure of individual robots. For using swarm robotic systems in human-supervised missions, it is imperative to understand the basic tenets of human-swarm interaction (HSI) [9]. Key characteristics of swarm robotic systems that make HSI challenging are (a) the swarm behavior is self-stabilizing and takes some time to emerge (b) individual robots in a swarm are very simple units with limited communication hardware capabilities and (c) robots have poor localization capabilities leading to errors in interpreting the input of the operator. The communication latency and the input error along with the self-stabilizing nature of the emergent behaviors of robotic swarms create challenges for the operator in understanding the current state of the swarm and the effect of his or her command.

This research has been sponsored in part by AFOSR FA955008-10356 and ONR Grant N0001409-10680.

In this paper, we present what we believe to be the first user study in HSI under realistic assumptions of communication latency and localization heading errors, and study the effects of latency and error on human performance in controlling swarm robotic systems. We also study the use of predictive displays to mitigate the effect of latency.

The extant literature on HSI [8], [9], [10], [11], [12], [13], [14], [15], [16] have not studied the performance and behavior of human operators in the presence of delayed information transmission between the swarm and the human and vice versa (exceptions being [17], [18], [19], [20] for haptic control). In the haptic control studies, the operator uses a force feedback and (possibly) visual information about the robot state to apply a continuous control input and maintain a robot formation. In contrast, in our problem, the operator does not have any force feedback. Furthermore, the operator influences the swarm system intermittently, meaning that the timing difference between two commands issued by the operator is much larger than the time step used in the discrete state update equations of the robot.

A. Neglect Benevolence

Since swarms require time to stabilize after an operator command is issued, it is possible for operator commands to have different—sometimes adverse—effects depending on the state of the swarm. To capture the idea that humans may need to observe the evolution of the swarm state and wait for stabilization before acting, we investigate a novel concept called *neglect benevolence*. This concept is analogous to *neglect tolerance* [21], [22] in human-robot interaction (HRI), which is defined as the time a human can neglect a robot in a multi-robot system of independent, non-coordinating robots without degradation in system performance. For neglect tolerance, it is assumed that the performance of an individual robot degrades with time, and hence the attention of the operator needs to be scheduled so that the time between servicing robots is minimized [23], [24]. In contrast, neglect benevolence captures the concept that it may be beneficial to leave the swarm alone for a certain length of time after issuing an instruction to allow the behavior to stabilize (since the swarm state may not degrade monotonically with time).

B. Contributions

Our experimental results indicate that, as expected, there is a degradation of performance due to latency. However, when using predictive display, the performance of the operators can be as well as it was in the absence of delay (control conditions). Our results also indicate that, by exploiting neglect benevolence in different ways, users came up with different strategies to control the swarm robotic system (see Section III).

II. TASK DESCRIPTION

Our study is designed to look at the ability of a human operator to effectively influence a swarm operating under algorithms that require time to exhibit emergent behaviors. We created a simple foraging task that requires users to direct a swarm around an open environment using instructions to change swarm heading and flocking constraints. We also use this study to look at the effect of communication latency in the swarm-operator channel on this ability. Latency can be caused by multiple factors, including limited radio power of the robots and communication channel properties such as limited bandwidth, which requires the communication of the robots to be scheduled.

A. The Environment

We use three different environments of size 100x100 meters divided into six regions (Figure 1), with each region containing one of three target frequencies: *low* (1-4), *medium* (5-9), or *high* (10+). The target distribution is different across the search missions that each participant solves, i.e. different regions may have different frequencies between missions, but there are always 40 targets in total. The interface that participants use to issue commands to the swarm does not show either the region boundaries or their target frequencies; however, participants do receive a worksheet for each trial showing the regions and their target frequencies.

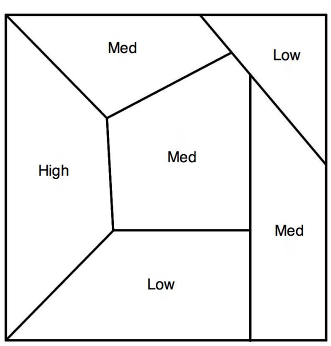

Fig. 1. An example worksheet the participant receives before a condition. The worksheet shows each of the six regions with the approximated number of targets the robots could expect to find in each. $Low = 1-4, Med = 5-10, High = 10+$.

We use Stage v. 3.2.2 [25] to simulate the environment, the targets, and a swarm of 40 differential drive P2AT robots. Robot controllers are implemented using the Robot Operating System (ROS) [26]. Each robot is equipped with a color sensor, allowing it to detect the colored targets, and an additional, simulated sensor that allows the robots to sense the location of a neighbor. Both have a 4 meter range, and the latter allows each robot to estimate the direction of motion of its neighbors from repeated observations of their location.

The graphical user interface is also implemented in ROS. Each robot transmits its position and observations from its color sensor to the user interface. When six or more robots detect a target simultaneously, it is displayed on the map at the centroid of the robots that sensed it. The total number found is displayed on the side of the screen.

B. Human Influence

Users can influence the swarm with three commands: *stop*, *heading*, and *apply-constraints*. The *stop* command simply instructs the robots to stop their motors. The *heading* command broadcasts a global heading to the swarm. To simulate a localization error, every robot interprets the global heading with respect to a local coordinate frame to compute its goal heading. The orientation of this local coordinate frame differs from the true orientation of the robot by an error sampled from the Gaussian distribution $\mathcal{N}(0, \frac{4\pi}{9})$.

Upon receiving the command, the robots turn toward their respective goal heading and begin moving (Figure 2a). In order to correct for the erroneous interpretations of the global heading, each robot also executes a consensus algorithm at a frequency of 0.5 Hertz. Robots sense the direction of motion of their neighbors and update their goal heading to the average goal heading of their neighbors and themselves.

By using the consensus algorithm, robots will change their heading to the average heading of their neighbors, and all robots in a connected component of the swarm will eventually move in the same direction (Figure 2b). The amount of time needed to reach consensus depends on the spectral properties of the connectivity graph of the robots [27]. If each robot is connected to every other robot, then the consensus happens in one cycle. However, in general, the robot connectivity graph is not complete and may not even be connected as the robots move. In such cases, there will usually be a bias in the heading of the swarms when the headings converge.

Finally, the user can issue an *apply-constraints* command, which applies biologically-inspired flocking constraints similar to those in [1], [2], and [16]. These constraints force robots to repel from each other if they are closer than 1.5 meters, and otherwise attract to neighbors further than 3 meters. Only the closest 5 neighbors are considered for these constraints. If a robot is between 1.5 and 3 meters from each of it's neighbors, it proceeds toward the goal heading dictated by the consensus algorithm (Figure 2c).

Applying the constraints serves two necessary functions: the repulsive force spaces out the swarm to give better coverage, and the attractive force helps prevent the swarm from splitting into many disconnected subgroups. However, these constraints were not automatically on for the duration of the study because of the need for a swarm to have time to reach consensus. Because the consensus algorithm required robots to sense the positions of their neighbors over time

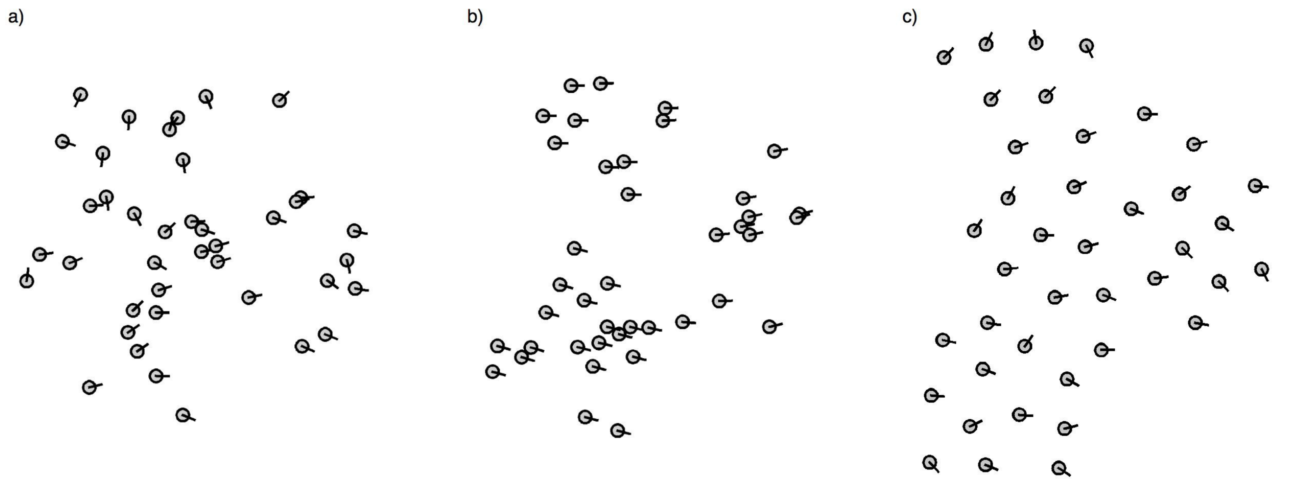

Fig. 2. The swarm in each of the three possible states. After the user issues a heading command, each robot moves toward their estimated goal heading (a). After enough rounds of the consensus algorithm, the robots have all converged on an approximately identical heading (b). Finally, after the user issues the flocking constraints, the robots attempt to stay between 1.5 and 3m of each other, and thus cover more area (c).

in order to get accurate heading estimates, if constraints were applied automatically following a heading command, much of this movement would be due to enforcing the 1.5 to 3m separation between robots. This introduces significant noise to the consensus algorithm and increases the error dramatically. Therefore, allowing the operator to activate constraints allows for the opportunity to observe the swarm and decide when the benefits given by the constraints are more important than further consensus.

C. Experimental Design

The experiment consists of three conditions—the *control*, *latency*, and *predictive* conditions. In all conditions, the operator begins with an open environment and the swarm of 40 robots positioned randomly in a 10x10 meter box at the center.

In the *control* condition, there is no latency in either of the human-to-swarm or swarm-to-human channels. In the *latency* condition, however, each channel has a latency of 10 seconds, which provides a realistic latency that is easily noticeable by a participant and forces them to make accurate predictions if they wish to influence the swarm effectively. This means that operator-issued commands will not reach the robots until 10 seconds after issuing, and the state of the swarm displayed in the interface for the user is 10 seconds old. In the *predictive* condition, the latency remains present; however, the interface gives the user a prediction of where each member of the swarm will be in 20 seconds (the time it takes for an operator's command to travel to the swarm and the result to return to the operator) by taking the heading and speed (which is a constant 0.5 m/s) of each swarm member and extrapolating the robot's position that far in the future (Figure 3). The prediction does not extend past the point the robots would perform the user's command. In other words, if the user issued a command 3s ago, the prediction will only show the swarm state 17s further into the future (which is 7 seconds ahead of the actual state).

Each participant has a different environment for each of these conditions, and the order of both the conditions and the maps are randomized for each participant in order to remove any learning biases. 21 participants (8 men and 13 women) were recruited from the University of Pittsburgh and surrounding areas to participate in the study. Each participant was given a short explanation of the controls and goals of the study and a 10-minute training session to familiarize themselves with the interface, after which they completed each of the three conditions.

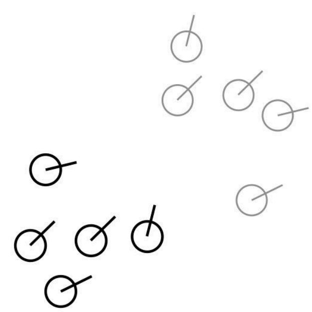

Fig. 3. An illustration of the predictive display condition. The interface projected a lighter shadow ahead of each robot to predict for the user where the robot would be in 10 seconds, or when the next command is received.

III. RESULTS AND DISCUSSION

All evaluation measures were compared using analysis of variance (ANOVA) tests, unless otherwise specified. We used total coverage and number of targets found out of 40 as global measures of success for a participant. Total coverage is defined as the area of the environment that was visible by 6 sensors simultaneously at some point during the mission. Furthermore, we looked for evidence of neglect benevolence directly by investigating how the state of consensus at the time of operator commands interacted with the swarm's progress along the goal heading, and measures of swarm cohesion such as the number of connected components and the average communication graph degree (i.e., average number of neighbors for the robots in the swarm).

The overall mission performance for each participant is measured in terms of the number of targets found. In the *control* condition participants found 19.86 targets on average. In the *latency* condition participants found 16.71 targets on average, significantly fewer than in the *control* condition ($p = .021$). Finally, in the *predictive* condition participants found 18.86 targets on average, not significantly different from the *control* condition ($p = .467$), see Figure 4. These results show that the latency of 10 seconds impedes operator performance in the *latency* condition, and that the predictive display in the *predictive* condition prevents this impediment.

The behavior of operators can be distinguished by the frequency, duration, and timing of the *heading* and *apply-constraints* instructions. The frequency is given by the total number of instructions during a condition, and the two durations we analyzed were the average time between a *heading* and a subsequent *apply-constraints* command (hereafter referred to as time to constraints), or the next *heading* command (duration). Because only one heading instruction could be active at a time, frequency and duration of heading commands have an obvious relationship. The timing of instructions is an important concept, since it relates to the state of the swarm at the time of the instruction—meaning two commands of the same type can lead to drastically different effects depending on when they were issued.

The duration of *heading* instructions differs significantly between conditions. The *control* condition, with a mean duration of 26.6 seconds, differs significantly from the *latency* ($p = 0.002$) and *predictive* conditions ($p = 0.004$), which have means of 42.4 and 40.2 respectively. The *latency* and *predictive* conditions do not differ significantly from each other ($p = 0.68$), see Figure 5.

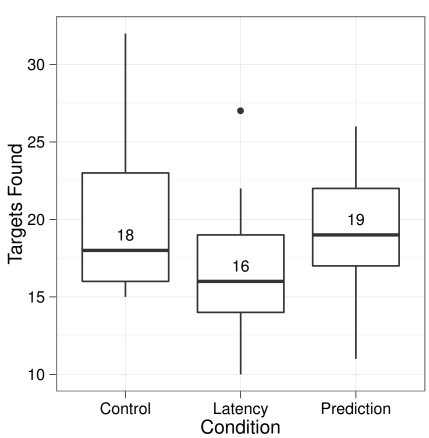

Fig. 4. A boxplot of the number of targets found across conditions. The median number of targets is shown above the median line.

The *apply-constraints* instruction, however, can be more

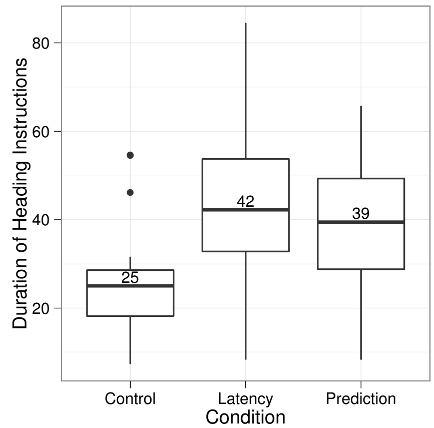

Fig. 5. A boxplot of heading command durations across conditions. The median duration time in seconds is shown above the median line.

flexible, as operators may decide to issue constraints at any point in time after a *heading* command, or issue a new *heading* command without activating constraints at all. On average, a participant did not use any constraints for 39.42% (std. deviation of 30% and median of 39.41%) of their heading instructions, with no differences ($p > 0.7$) across conditions or between any two conditions. There are, however, significant differences across participants. In fact, combining all conditions, participants cover the entire spectrum, from using constraints for only 9% of *heading* commands up to using constraints after every one. Exactly 2 participants used constraints for 0% to 25% of *heading* commands, 6 for 25% to 50%, 6 for 50% to 75%, and 7 for 75% to 100%. To investigate the effect of the application of constraints, we grouped all missions with a high (100% to 78%), medium (78% to 45%), or low (45% to 0%) number of headings commands for which constraints were applied. The boundaries were chosen to obtain equal sized groups (across conditions and participants). Table I summarizes the main results comparing these groups. Performance in terms of targets found does not differ between these groups, but the total area swept is significantly different, with fewer constraints (low) leading to less overall area that is covered by at least six sensors. On the other hand, many constraints (high) lead to a larger error between the heading of the swarm and the operator's goal heading, and fewer constraints (low) have more heading instructions leading to a consensus (74%) and more robots in a consensed state when the next instruction is issued (69%). In addition, more constraints lead to fewer connected components (2.12) and more neighbors for each robot (4.79) on average.

These results suggest that operators employ different strategies to find targets. Some operators use constraints often and earlier to cover a wider area at the expense of higher heading errors, while others prefer to give the consensus more time, leading to a smaller deviation from the operators goal heading at the expense of coverage and swarm cohesion.

The differences in operator behavior between the *control*,

Measure	high	medium	low	p
Duration	42.9	34.3	31.8	0.085
Time until constraints	4.78	17.2	25.9	<0.001
Consensus reached by	14%	54%	74%	<0.001
Number of neighbors	4.79	4.16	3.75	<0.001
Connected components	2.12	3.23	3.92	<0.001
Robots in largest component	27.2	23	20.3	0.006
Heading interrupted	0.795	0.43	0.26	<0.001
Heading error	0.496	0.381	0.38	0.011
Total area swept	1949	1761	1330	0.040
Targets found	18.7	18.5	18.2	0.933

TABLE I

TABLE COMPARING THREE GROUPS WITH 100% TO 78% (HIGH), 78% TO 45% (MEDIUM), 45% TO 0% (LOW) APPLICATION OF CONSTRAINTS.

Measure	short	medium	long	p
Duration	19.3s	33.9s	56.0s	<0.001
Time until constraints	11.25	18.53	17.68	0.141
Constraints activated	52%	54%	76%	0.012
Consensus reached by	11%	19%	18%	0.141
Number of neighbors	3.76	4.14	4.82	<0.001
Connected components	3.60	3.19	2.45	0.005
Robots in largest component	21.97	23.14	25.54	0.267
Heading interrupted	0.580	0.400	0.514	0.138
Heading error	0.353	0.409	0.496	0.005
Total area swept	1204	1707	2149	<0.001
Targets found	19.57	17.95	17.90	0.378

TABLE II

TABLE COMPARING THREE GROUPS WITH SHORT, MEDIUM, OR LONG DURATIONS FOR HEADING COMMANDS .

latency, and *predictive* conditions are significant for heading duration, but not the average time to constraints. A difference between either of the two conditions with a 10 second latency and the control was expected, since operators have to wait 10 seconds to obtain information in order to decide whether constraints are needed, and since the activation of constraints takes 10 seconds to arrive at the swarm. In fact, across all instructions, only 27% in the latency condition and 30% in the *predictive* condition have constraints activated later than 20 seconds, meaning that, unlike with *heading* commands, operators often issued the constraints prior to seeing the effect of the heading instruction on the swarm. The *predictive* condition has a significantly higher time to constraints ($p = 0.023$) than the *control* condition, with a mean of 19.39 and median of 19, see Figure 6. However, the *latency* condition is not significantly different ($p = 0.12$) from the *control* condition with means of 16.91 and 11.15 and medians of 13 and 12. This indicates that there is some adaptation of strategies to conditions and that operators are often using the activation of constraints regardless of whether they have information about the swarm state. Our results above, however, show that operators already employ a wide variety of strategies that differ across participants.

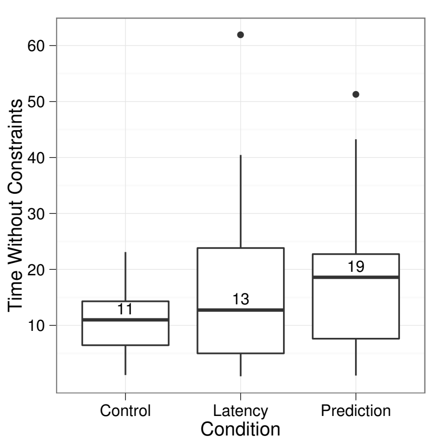

Fig. 6. The average time to the activation of constraints. The two black dots represent outlier trials, where two participants issued constraints considerably later on average in the *latency* and *prediction* conditions, respectively.

A closer look at the duration of heading instructions can cast some light on the effects of different behaviors on the swarm state. Participants with either short, medium or long heading instructions, see Table II, show a similar pattern than the activation of constraints in Table I. The targets found does not differ significantly, but the area covered with at least 6 sensors does ($p < .001$), demonstrating that the short duration group compensates for less area swept by using fewer constraints activated and thus a smaller heading error.

A. Support for Neglect Benevolence

The above results clearly show support for neglect benevolence. Early activation of constraints leads to larger heading errors and a mismatch between the operators desired direction and the direction of the swarm. However, letting a swarm move with constraints active for a long time, without interrupting with new heading instructions, improves its cohesion and ability to cover area with its sensors. Frequent and short commands, on the other hand, may provide an operator more control over the direction by providing new inputs more frequently, but this leads to an increase in the number of connected components and disturbs the swarm's cohesion by frequently deactivating constraints. In other words, we have shown that operators developed two equally successful strategies around the neglect benevolence of the swarm with respect to these two processes that either stabilize the consensus and lower the heading error, or stabilize the flocking formation and improve coverage. While we found some behavioral differences across *control*, *latency*, and *predictive* conditions, there is no direct evidence that latency affects some strategies differently than others.

B. Support for the Predictive Display

We also demonstrated that a predictive display helps participants overcome the latency issues and find the same number of targets as they would without any latency present. There are two possible explanations for this finding. First, the predictive display gave the operators the ability to time their new *heading* instructions appropriately. If, for instance, an operator wanted to explore a region entirely, the prediction allowed them to see when the swarm would reach the edge of the region or map and issue a new *heading* command so that the swarm would receive it at the proper time. Second, the predictive display also allowed users to easily identify groups that may break off, and to identify the state of consensus of the swarm. If a swarm of robots is about to split in two because two subgroups are moving at different headings and may not reach consensus before splitting, the predictive display will show these two subgroups 20 seconds in the future—at which point their splitting, if unaltered by constraints or consensus, is readily apparent. Similarly, if the swarm has not reached consensus, the predictive display will show them significantly more detached and spread out in 20 seconds time, whereas the prediction for a swarm at consensus will look more or less similar to the current state.

IV. CONCLUSIONS AND FUTURE WORKS

Overall, this study provides support for neglect benevolence. The possible commands in the study provided both costs and benefits depending on the state of the swarm at the time the commands were issued. Frequent redirections of the swarm gave the user more control over the direction, and thus location, of the swarm, but sacrificed other characteristics necessary for a foraging or exploration task performed with a swarm, including coverage and swarm cohesion.

This led to two basic types of operators. Some preferred a higher accuracy for the heading of the swarm, while others preferred a constrained motion in a more spread out swarm. Due to the nature of the swarm algorithms and localization errors, high heading accuracy and high coverage using constraints are not possible simultaneously. Therefore, participants had to decide which characteristics were more important. For the present study, both strategies achieved success; however, other tasks may be better achieved with one or the other. Future research could help determine for which tasks each of the above strategies is most suitable.

Furthermore, latency had a negative effect on the number of targets found, but only if the operator was not supported by a predictive display, demonstrating that the prediction enabled operators to regain some of the original performance. Latency also seemed to significantly impact the frequency with which operators issued commands. As this is the first study to investigate latency in HSI, future work should address latency issues for human control of other tasks and swarm algorithms, varying latency times, and using different methods of predicting future swarm states.

REFERENCES

[1] C. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4. ACM, 1987, pp. 25–34.

[2] I. Couzin, J. Krause, R. James, G. Ruxton, and N. Franks, "Collective memory and spatial sorting in animal groups," *Journal of theoretical biology*, vol. 218, no. 1, pp. 1–11, 2002.

[3] W. Spears and D. Spears, *Physicomimetics: Physics-Based Swarm Intelligence*. Springer-Verlag New York Inc, 2012.

[4] D. Bruemmer, "A robotic swarm for spill finding and perimeter formation," DTIC Document, Tech. Rep., 2002.

[5] R. Morlok and M. Gini, "Dispersing robots in an unknown environment," *Distributed Autonomous Robotic Systems 6*, pp. 253–262, 2007.

[6] H. Choset, "Coverage for robotics–a survey of recent results," *Annals of Mathematics and Artificial Intelligence*, vol. 31, no. 1, pp. 113–126, 2001.

[7] F. Ducatelle, G. Di Caro, and L. Gambardella, "Cooperative self-organization in a heterogeneous swarm robotic system," in *Proceedings of the 12th annual conference on Genetic and evolutionary computation*. ACM, 2010, pp. 87–94.

[8] S. Bashyal and G. Venayagamoorthy, "Human swarm interaction for radiation source search and localization," in *Swarm Intelligence Symposium, 2008. SIS 2008. IEEE*. IEEE, 2008, pp. 1–8.

[9] M. Cummings, "Human supervisory control of swarming networks," in *2nd Annual Swarming: Autonomous Intelligent Networked Systems Conference*, 2004.

[10] A. Kolling, S. Nunnally, and M. Lewis, "Towards human control of robot swarms," in *Proceedings of the 7th international conference on Human-robot interaction*. ACM, 2012, pp. 89–96.

[11] G. Coppin and F. Legras, "Autonomy spectrum and performance perception issues in swarm supervisory control," *Proceedings of the IEEE*, no. 99, pp. 590–603, 2012.

[12] P. Klarer, "Flocking small smart machines: An experiment in cooperative, multi-machine control," Sandia National Labs., Albuquerque, NM (United States), Tech. Rep., 1998.

[13] Z. Kira and M. Potter, "Exerting human control over decentralized robot swarms," in *Autonomous Robots and Agents, 2009. ICARA 2009. 4th International Conference on*. IEEE, 2009, pp. 566–571.

[14] M. Fields, E. Haas, S. Hill, C. Stachowiak, and L. Barnes, "Effective robot team control methodologies for battlefield applications," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE, 2009, pp. 5862–5867.

[15] A. Naghsh, J. Gancet, A. Tanoto, and C. Roast, "Analysis and design of human-robot swarm interaction in firefighting," in *Robot and Human Interactive Communication, 2008. RO-MAN 2008. The 17th IEEE International Symposium on*. IEEE, 2008, pp. 255–260.

[16] M. Goodrich, B. Pendleton, P. Sujit, and J. Pinto, "Toward human interaction with bio-inspired robot teams," in *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2859–2864.

[17] H. Son, L. Chuang, A. Franchi, J. Kim, D. Lee, S. Lee, H. Bulthoff, and P. Robuffo Giordano, "Measuring an operator's maneuverability performance in the haptic teleoperation of multiple robots," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 3039–3046.

[18] P. Giordano, A. Franchi, C. Secchi, and H. Bulthoff, "Experiments of passivity-based bilateral aerial teleoperation of a group of uavs with decentralized velocity synchronization," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 163–170.

[19] D. Lee, A. Franchi, P. Giordano, H. Son, and H. Bulthoff, "Haptic teleoperation of multiple unmanned aerial vehicles over the internet," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1341–1347.

[20] C. Secchi, A. Franchi, H. Bülthoff, and P. Giordano, "Bilateral teleoperation of a group of uavs with communication delays and switching topology," *Proceedings of the IEEE International Conference on Robotics and Automation*, 2012.

[21] D. Olsen and M. Goodrich, "Metrics for evaluating human-robot interactions," in *Proceedings of PERMIS*, vol. 2003, 2003.

[22] A. Steinfeld, T. Fong, D. Kaber, M. Lewis, J. Scholtz, A. Schultz, and M. Goodrich, "Common metrics for human-robot interaction," in *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*. ACM, 2006, pp. 33–40.

[23] P. Mitchell and M. Cummings, "Management of multiple dynamic human supervisory control tasks," in *10th International Command and Control Research and Technology Symposium*, 2005.

[24] S. Mau and J. Dolan, "Scheduling for humans in multirobot supervisory control," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*. IEEE, 2007, pp. 1637–1643.

[25] B. Gerkey, R. Vaughan, and A. Howard, "The player/stage project: Tools for multi-robot and distributed sensor systems," in *Proceedings of the 11th international conference on advanced robotics*. Portugal, 2003, pp. 317–323.

[26] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, vol. 3, no. 3.2, 2009.

[27] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*. Ieee, 2005, pp. 63–70.