

Communication Scheme Comparison for a Distributed Multi-Agent System

Matthew Conforth and Yan Meng

Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ, USA.

Abstract - *In this paper, we investigate various communication schemes for a distributed swarm robotic system in a target-searching task under different environmental scenarios. Since multi-target searching using a multi-robot system is an NP-hard problem, it is difficult to find an optimal solution for different scenarios. Therefore, we aim to provide a comparison of different communication schemes for several scenarios. These comparisons will provide us some statistical references, which will make it easier to further investigate the optimized solutions for specific scenarios in the future.*

Keywords: multi-agent systems, communication scheme, distributed systems.

1 Introduction

Recently, swarming robot systems have attracted considerable attention due to their broad applications in the real world. Extensive research has been conducted on swarming robot systems, such as foraging [1], box-pushing [2], aggregation and segregation [3], formation forming [4], cooperative mapping [5], soccer tournaments [6], site preparation [7], sorting [8], and collective construction [9]. All of these systems consist of multiple robots acting autonomously based on their own individual decisions. Compared to centralized control of multi-robot systems, distributed control algorithms have many advantages such as flexibility, robustness, parallel processing, and adaptability, especially in dynamic environments,

Coordinating swarming robots working in a cooperative, distributed manner to achieve a global, coherent activity is a challenging problem. For a distributed multi-robot system, without any communication with other robots, each individual robot has to make its control decisions based only on its local view of the problem, which may lead to inappropriate decisions about its next action and more travel cost resulting in low-efficiency. It has been shown that agent communication is an essential factor for the emergence of cooperation in multi-agent systems [10].

To achieve coordinated actions, robots potentially need to know detailed information about the current and planned activities of other robots, and information these robots have acquired about the targets and the environment. It is costly to receive and analyze all this non-local data; more importantly,

usually only a small subset of information is necessary to coordinate actions in a specific situation. Thus, on one hand, a lack of sharing global data may cause a robot to execute redundant or unnecessary actions, leading to more redundant travel costs and more power consumption. On the other hand, the transmission of unnecessary information can contribute to overloading communication, and add significant computational activities to robots as they process this nonlocal information and update their local database. Therefore, the process of identifying the appropriate balance between the communication cost and the travel cost for the robots must be linked closely to the characteristics of the coordination situation and working environment.

In our previous work [11] on a multi-target searching task using a bio-inspired distributed multi-robot system, extensive communication loads were noticed, especially when the scale of the robot system become large or when the communication range was extended. Therefore, we have to make a tradeoff between the communication cost and travel cost of the multi-robot systems. From a power-efficiency point of view, travel, communication, and computation costs all consume different levels of power. Letting the robots learn what and when to communicate with their team members could significantly improve the flexibility and adaptability of a multi-agent system.

Most of the available multi-robot systems mainly focus on the decision making algorithms assuming that the inter-robot communication is predefined and fixed, and the agents achieve group behavior by reacting to these communications appropriately. However, due to the inherent uncertainty in the robots' activities under the dynamic working environment, and the cost of communication and computational processing, it would be desirable to have a flexible and adaptive communication mechanism, which can automatically change its communication protocol to adapt to the dynamic environments and various uncertainties.

To be able to develop an adaptive communication mechanism, first we need to understand which communication mechanisms are most effective in varying application and environmental scenarios. No such rule or protocol exists so far to be used as a reference for us. Therefore, in this paper, we will investigate several communication protocols for a distributed multi-agent system, where the tradeoff between the global task performance (such as finishing time) and the power cost (including the communication cost and the travel cost) will be investigated under various scenarios in a searching task.

The following sections are organized as follows. Related work is introduced in Section 2. Section 3 provides the four working scenarios we will investigate in this paper. The various communication mechanisms are described in Section 4. Extensive simulation results and results analysis are provided in Section 5. The conclusion and future work are given in Section 6.

2 Related Work

Very little work has been conducted on how communication can affect the behavior of a multi-agent system. Mackin and Tazaki [12] proposed an automatically defined function genetic programming (ADF-GP) method to allow agents to automatically learn effective agent communication messaging in order to achieve group behavior. Giles and Jim [13] proposed a general model of multi-agent communication in which all agents communicate simultaneously to a message board, and then a generic algorithm is used to learn multi-agent language for a predator-prey problem. Jang et al. [14] proposed an Actor Architecture (AA) to support large-scale open multi-agent communication, which tried to address two related problems: efficient message passing and service agent discovery. Basically, Agents in AA obey the operational semantics of actors. However, these papers mainly focus on the communication mechanisms for general multi-agent systems, which may not be appropriate for distributed multi-robot systems.

Some researchers started working on the communication issues of distributed systems [15, 16, 17]. In [15], Klavins tried to introduce a notion of communication complexity as a means of investigating the scalability of multi-robot algorithms in terms of how much coordination they require and presented several communication schemes. However, this work didn't consider either a model of possible sensing modalities and costs, or the high-level control algorithms. Bandwidth-aware control and communication schemes in a distributed system were proposed in [16]. When using communication in multi-robot systems, it is often not desirable to choose an abstract form of communication that separates the messages from the physical environment. Therefore, Stoy [17] proposed a situated form of communication that exploits the physical properties of the signal transferring the message. For the multi-robot systems, Balch and Arkin [18] showed that communication can significantly improve the performance of the robots with little environmental communication, and more complex communication strategies provide little or no benefit over low-level communication.

3 Four Distributed Working Scenarios

In this paper, we will investigate a multi-robot system in a multi-target searching and processing task, where the

multiple targets and multiple homogeneous robots are randomly distributed in a searching area. It is assumed that the searching area is bounded and robots can detect the targets using their on-board sensors, such as camera systems. Once a robot detects a target, it starts to process the target, simulated as "eating", where the process time is proportional to the "size" of target. Each robot can only communicate with its neighbors which are within its pre-specified communication range. The objective of this searching and processing task is to find all the targets and process the targets as soon as possible. Processing the targets is abstracted as "eating" the targets until all of the target tasks are finished.

In accordance with the stated goal of generality and wide relevance, the team divided the study space into four broad categories that capture the essential nature of a wide variety of different swarm robot applications. Each scenario is a variation on our base setup of 50 robots, 10 targets, and 5 obstacles. The two variables we considered for the purpose of differentiating scenarios are swarm density and target complexity. A sparse swarm is one in which a robot is in direct communication range with only a couple other robots on average. A dense swarm is one in which a robot is in direct communication range with many other robots on average. Complex targets require the cooperation of several robots to process in a reasonable amount of time. Simple targets are either processed rapidly by an individual robot or do not benefit from cooperation.

1) *Sparse Swarm; Complex Targets*: In this scenario, the robots are tasked with targets that require a significant amount of time to process, even with multiple robots cooperating on an individual target. The small communication range simulates a robot swarm that is spread out over a large area, or operating in a high-interference environment. Analogous real world situations include search-and-rescue and military applications.

2) *Dense Swarm; Complex Targets*: In this scenario, the robots are tasked with the same type of targets as in 1). A longer communication range allows information to be spread further and more rapidly through the swarm. Industrial cleanup, factory assistance, transporting large objects, etc. are examples of real world tasks in this category.

3) *Sparse Swarm; Simple Targets*: In this scenario, the robots are tasked with targets that an individual robot can process quickly. The small communication range simulates a robot swarm that is spread out over a large area, or operating in a high-interference environment. Analogous real world situations include retrieval and delivery of small objects, residential cleanup, and routine inspection.

4) *Dense Swarm; Simple Targets*: In this scenario, the robots are tasked with the same type of targets as in 2). A longer communication range allows information to be spread much more rapidly through the swarm. Surveillance, indoor navigation, commercial or residential assistance, etc. are examples of real world tasks in this category.

4 Communication Mechanisms

All the algorithms discussed here are implemented for broadcast networking as opposed to point-to-point. The algorithms tested are not optimized to any particular task or situation. For the most part, they are not optimized at all. Instead they are implemented in the most general sense possible, essentially demonstrating different philosophies or archetypes of swarm communication. Testing these archetypes in simplest form puts them on a level playing field, so as to generate some metrics of initial performance. These initial metrics should be useful in evaluating which communication philosophy is appropriate to a particular task. This also helps to prevent the particulars of our simulator from introducing bias into the results. An optimized strategy is necessarily a “tighter fit” and concordantly actual behavior rapidly becomes less predictable the more one’s actual case diverges from the tested cases.

4.1 No Communication

No communication is the simplest of all communication algorithms. No information is shared between robots. Each robot must operate only based on its own direct observation. Each robot chooses a random initial heading and continues on that path until a target is reached. If the robot encounters an obstacle, it will change its heading to avoid the obstacle. If the robot reaches one of the simulation boundaries, it chooses a new random heading away from that boundary. When the robot reaches a target, it will stop to process the target if the target is active, or ignore it if the target is finished. This algorithm is included because it establishes a baseline of performance for the simulation scenarios, to which the other algorithms can be compared.

4.2 Low Communication

This algorithm seeks to improve on the no communication approach by adding only a little communication. Each robot maintains two lists of target positions. T_{Alive} contains the positions of targets that have not been serviced completely yet. T_{Dead} contains the positions of targets that have been completed. A robot adds a target t_{new} to the appropriate list when either 1) it directly observes the target, or 2) it receives a message from a neighbor informing it about the target. A robot only sends a message when it directly observes a target that is not in one of its lists. The message transmitted is simply the contents of T_{Alive} and T_{Dead} .

$$\text{send iff } t_{\text{observed}} \notin (T_{\text{Alive}} \cup T_{\text{Dead}}). \quad (1)$$

4.3 High Communication

This algorithm seeks to improve on the no communication approach by adding frequent communication. Each robot maintains two lists of target positions just as described for the low communication algorithm. In this case,

however, a robot sends a message containing its lists any time one or more of its lists changes, regardless of the source of the new information.

$$\text{send iff } \begin{cases} t_{\text{new Alive}} \notin T_{\text{Alive}} \\ t_{\text{new Dead}} \notin T_{\text{Dead}} \end{cases}. \quad (2)$$

4.4 Request-Based Communication

This algorithm seeks to improve on the no communication approach by allowing a robot to request information from its neighbors. Each robot maintains two lists of known target positions as described previously. A robot that does not know of any active targets will send a message containing its KnownDead list. Any neighbor that knows of one or more active targets will respond with a message containing its lists.

$$\begin{aligned} \text{send}_{\text{request}} \text{ iff } |T_{\text{Alive}}| = 0 \\ \text{send}_{\text{reply}} \text{ iff } \text{received}_{\text{request}} \wedge |T_{\text{Alive}}| \geq 1 \end{aligned} \quad (3)$$

4.5 Pheromone-Edge-Pair Communication for PSO-based Coordination

A pheromone-edge-pair (PEP) communication mechanism was proposed for a virtual pheromone based PSO coordination method in our previous work [11]. Basically, two coordination processes among the robots are established. One is a virtual stigmergy based algorithm to guide the robots’ movements for targets, where each robot has its own virtual stigmergy matrix that can be created, enhanced, evaporated, and propagated to its neighboring robots. The other one is Particle Swarm Optimization (PSO)’s cognitive capabilities through local interaction, which aims to achieve the balance for each robot between the exploration and exploitation through the interactions among the robots.

The target utility function is a function of the target weight (how big the target is), stigmergy intensity and the local robot redundancy (how many robots have started processing the targets). The stigmergy density is an indication of the number of robots who will potentially process the corresponding target at location (i, j) . It can be updated by the following equation to emulate the stigmergy enhancement and elimination procedure:

$$\tau_{ij}^k(t+1) = \rho * (\tau_{ij}^k(t) + T_{ij}^k) - (1 - \rho) * m * \tau_{ij}^k(t) \quad (4)$$

Where $0 < \rho < 1$ is the enhancement factor of the stigmergy density. m represents the elimination factor. To slow down the elimination relative to enhancement, we set $m < 1$. T_{ij}^k is the stigmergy interaction intensity received from the neighboring robots for a target at (i, j) , which is defined as

$$T_{ij} = \begin{cases} \alpha, & \text{if source pheromone} \\ \beta, & \text{otherwise} \end{cases} \quad (5)$$

The target visibility is defined as a function of the robot local detection range and the distance between the robot and the interested target. Then a Particle Swarm Optimization (PSO) based coordination method is used to integrate the target utility and target visibility into a population-based optimization method to search for the optimal coordination strategy. Each robot controls its movement behaviors by following these equations:

$$v^k(t+1) = \psi_e * rand_e() * v^k(t) + \psi_c * rand_c() * (p_c - x^k(t)) + \psi_s * rand_s() * (p_s - x^k(t)) \quad (6)$$

$$x^k(t+1) = x^k(t) + v^k(t+1) \quad (7)$$

Where $v^k(t)$ and $x^k(t)$ is the velocity and position of robot k at time step t , respectively. $\psi_e, \psi_c,$ and ψ_s represent the propensity constraint factors for explosive, cognitive, and social behaviors, respectively, $0 \leq rand_\theta() < 1$ where $\theta = e, c,$ or s , and $x_{ij}^k(t)$ represents the position of robot k at time t .

$p_s = \max(\mu_{ij}^k(t))$ represents the global best from the neighbors, which is the target with the maximum value of the target utility. $p_c = \max(\eta_{ij}^k(t))$ represents the local cognitive best, which is the target with the maximum value of the target visibility.

To reduce the communication propagation, a pheromone-edge-pair (PEP) propagation funneling method was proposed in [11]. The basis idea of this PEP method is that the robot would ignore the target information if it is received from the same neighbor again. Otherwise, the robot would update its local database and propagate the new target related information to its neighbors. This communication mechanism is obviously not efficient if the scale of the robots becomes large because the propagated package would be increased exponentially due to the dynamic characteristics of the robot movements. But it can provide a good reference point for the communication scheme comparison.

5 Simulation Results

Extensive simulations have been conducted. The simulator utilized is a significantly modified version of the Java-based simulator described in [11]. The simulation environment is a 2-dimensional area 640 pixels by 480 pixels, as shown in Fig.1. Five grey obstacles, 100 by 20 pixels in size are randomly placed in this environment. Next, ten randomly generated red targets between 10 and 20 pixels in radius are randomly placed in the environment with the restriction that they cannot be placed inside obstacles. Finally, 50 robots 5 pixels in diameter are randomly placed in the environment with the restriction that they cannot be placed inside obstacles. Robots change color to indicate their current state of behavior. Blue indicates the robot is engaged in exploration. Pink indicates that the robot has

chosen a target and is navigating to it. Green indicating that the robot is processing the target to which it is attached. In the PSO-based case, black and blue are used to indicate swarm intelligence behaviors corresponding to dispersing and converging respectively.

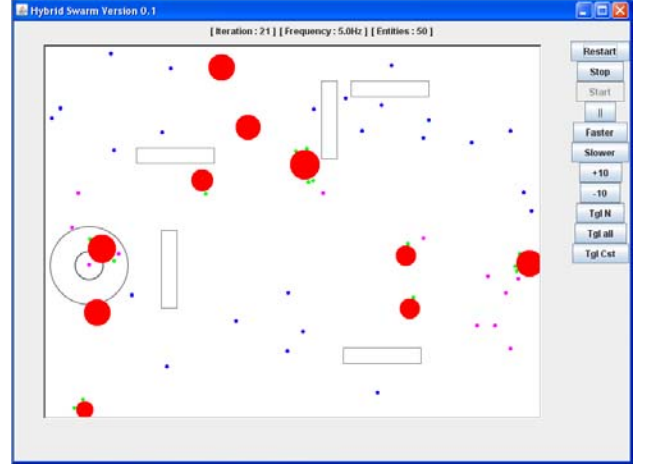


Fig.1. Java-based simulator GUI.

Each robot can detect the targets within 20 pixels of its current position. For the sparse scenarios, the robots have a communication range of 50 pixels. This is extended to 150 pixels for the dense scenarios. The simulator can pause, start, stop, restart, speed up, slow down, add robots, and remove robots. There is also functionality for displaying other useful information, such as communication links, obstacle avoidance, and so on.

The simulator was run 1000 times for each algorithm in each scenario, for a total of 20,000 simulations. The simulations were conducted using Sun Java SE Runtime Environment (build 1.6.0_02-b06) in Microsoft Windows XP SP2.

5.1 Sparse Swarm; Complex Targets (SSCT)

As can be seen in Fig. 2, communication and information propagation is quite advantageous in this scenario. More information sharing corresponds to less movement. Fig. 3 shows that this information advantage, in turn, leads to shorter completion times. While the request-based communication algorithm yielded the minimum average completion time, it does so by only a small margin and at the cost of substantial power usage. The high communication algorithm seems the clear winner here, since it required the least total power and was just slightly slower than the request based communication algorithm.

A quick note on units is in order here. For the sake of simplicity, time is measured in simulation cycles, distance is measured in pixels, and power is measured in a made-up unit, the simWatt. 1 simWatt corresponds to the power required to move one robot one pixel, and in the case of Fig. 3, is also equal to the power required to send one communication packet. The reason being that robots vary extensively in size, weight, speed, etc. such that anyone trying to apply these

results would have to convert all the units as appropriate to their particular robots anyway. However, we realize that in a real robot, the cost of movement is probably quite different from the cost of communication.

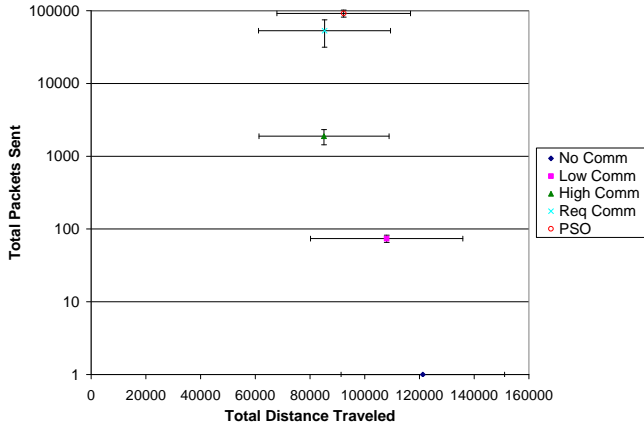


Fig.2. Total packets sent vs. total distance traveled for SSCT scenario. (Notes: the plotted points are the mean values from the simulation runs, and the error bars show the standard deviation of the results. The same is true for the following figures).

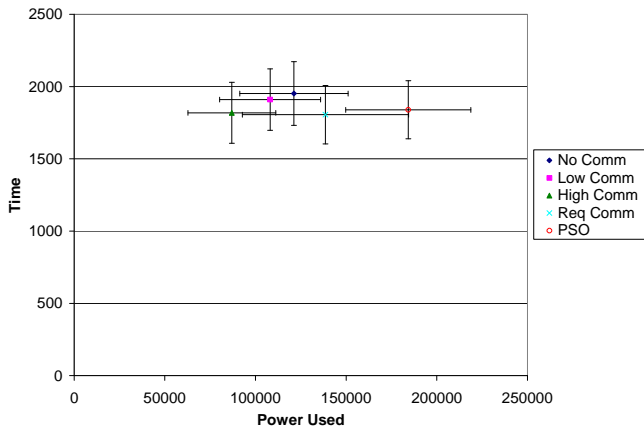


Fig.3. Completion time vs. power used for SSCT scenario.

5.2 Dense Swarm; Complex Targets (DSCT)

In this scenario we see from Fig. 4 that with communication you can indeed have too much of a good thing. The low communication and high communication algorithms both offer substantial improvement over the no communication case. However, the request-based communication and PSO-based swarm intelligence algorithms both begin to suffer here. There are two reasons for this. The first is that information acquisition is definitely a diminishing-returns proposition in a swarm. A robot that knows the location of the nearest target obviously will outperform one that does not. However, each additional target the robot knows about is progressively less useful. The second cause is the spatially limited utility of information. Having robots near a target stop exploring to process it is generally a good thing. However, further away robots will be more useful overall if they continue exploring. Fig.5 shows that the low communication algorithm is the winner in this scenario,

though the high communication algorithm performs very nearly as well.

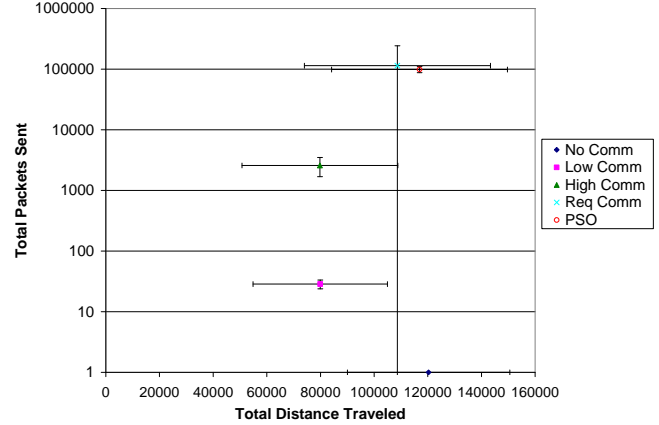


Fig.4. Total packets sent vs. total distance traveled for DSCT scenario. (Note: the standard deviation bars are not equal in length because the y-axis is logarithmic scale).

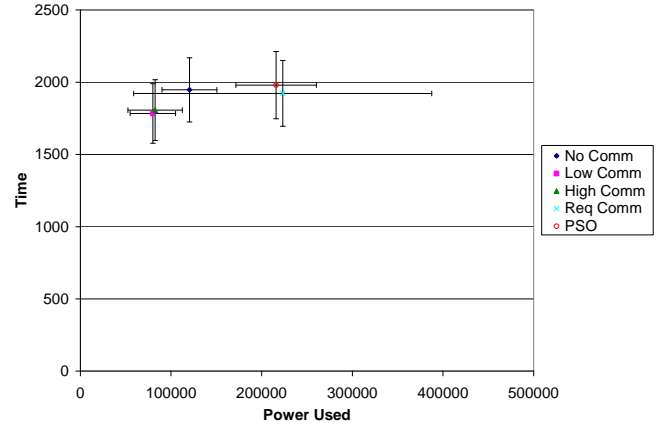


Fig.5. Completion time vs. power used for DSCT scenario.

5.3 Sparse Swarm; Simple Targets (SSST)

This could accurately be called the futility scenario. As seen in Fig. 6 and Fig.7, information sharing proportionally diminishes performance in this scenario. The reason for this is simple. The first robot to locate a target will process that target at the maximum rate possible. Additional robots cannot assist with that particular target, and since the locations of targets are random, knowing the locations of some targets does not help in predicting the locations of others.

5.4 Dense Swarm; Simple Targets (DSST)

This is quite similar to the Sparse Swarm; Simple Targets scenario, except even worse, as shown in Fig. 8 and Fig. 9. More extensive communication leads to even more proliferation of useless information. It is interesting to note that the swarm intelligence algorithm took a much bigger performance hit here than the others, though once again the clear winner is the no communication algorithm. For this scenario, since the coordination is not necessary at all for the

target processing, the information sharing and complex decision making in the PSO-based swarm intelligence algorithm only costs more communication and travel cost without improving the target processing.

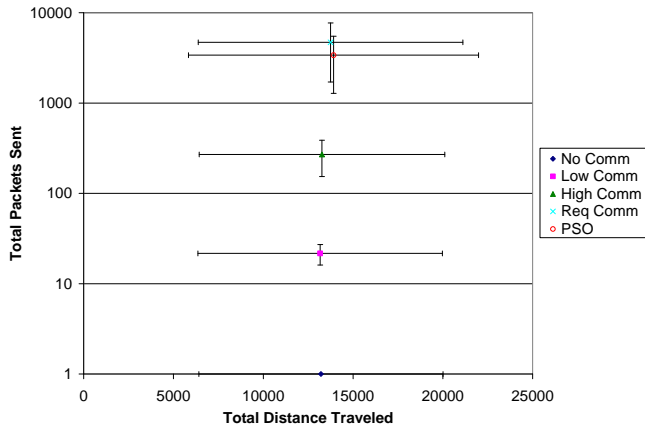


Fig.6. Total packets sent vs. total distance traveled for SSST scenario.

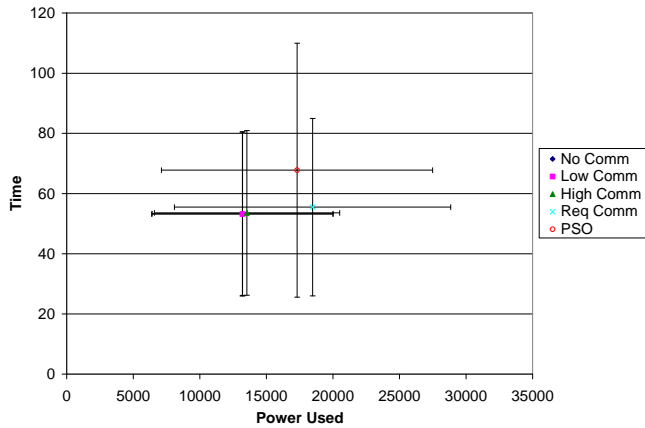


Fig.7. Completion time vs. power used for SSST scenario.

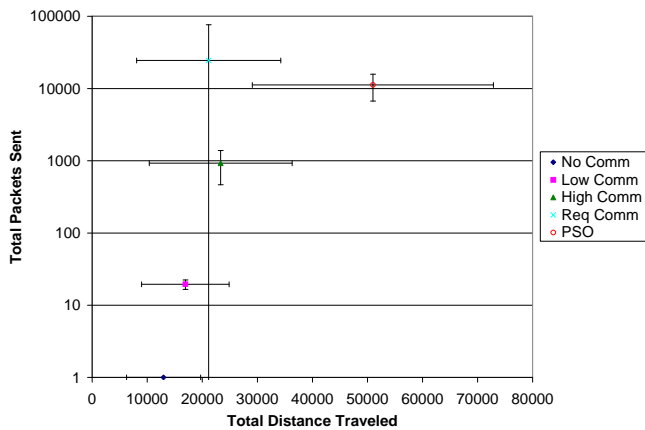


Fig.8. Total packets sent vs. total distance traveled for DSST scenario. (Notes: the std bars are not equal in length because the y-axis is logarithmic scale.)

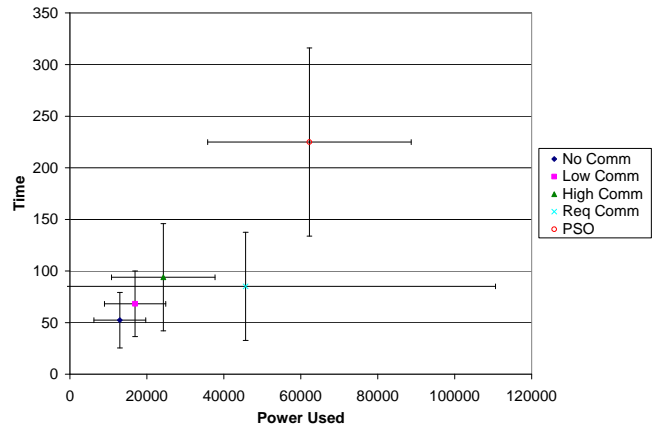


Fig.9. Completion time vs. power used for DSST scenario.

5.5 Results Discussion and Analysis

It is interesting that the no communication mechanism performed the best for those simple targets scenarios, where the coordination is not necessary for the target task processing. On the other hand, for the complex targets scenarios, information sharing shows better performance and simple communication mechanisms perform better. This is supported by the bio-inspired swarm philosophies that emphasize the use of simple rules for agents to construct efficient emergent behaviors in the swarm. However, further testing is needed to determine whether the more complex communication algorithms simply need different, more complicated, or more dynamic environments to demonstrate their advantages.

The Complex Target tests are probably the more important ones to consider, because they are more indicative of the applications where multi-robot coordination is necessary and good communication offers significant advantages. In both of those cases, the high communication algorithm wins or performs almost equivalently to the winner, depending on which performance metrics one considers most important. Intuitively, it makes sense that the low communication shines as the communication range increases, whereas its effectiveness dwindles as communication range decreases. Conversely, the high communication algorithm propagates new information throughout the swarm, which guarantees solid performance across the board.

The request-based communication has real potential (except for very small robots). The main issue that needs to be overcome is the scenario where no robot in the vicinity is aware of an active target, and so a whole group of robots is sending requests every simulation cycle. If this can be overcome, then it might evolve into the best performer in the Sparse Swarm; Complex Targets scenario.

The PSO-based swarm intelligence algorithm has built in tweaking capability, where different parameters have to be adjusted for different scenarios to achieve the optimized performance. It is hard to determine some ideal “general purpose” values for various scenarios. Further investigation is needed. It is interesting that whereas the Low and High

Comm swarms perform better when the communication range is increased, the Request Comm and PSO swarms actually perform worse. The main reason behind this is that when more robots are within a robot's neighbor area, more information sharing and communication exchanges occur. On one hand, a robot would benefit from this informative environment to make its own decision.

On the other hand, the decision making procedure becomes more complex. If the balance of the exploration and exploitation are not well adjusted, robots may end up wandering around between the targets for a longer time before eventually picking one. In addition, more communication and processing overheads are also involved in the diminished performance. The Simple Target tests demonstrate the brutal effects of random goals on swarm coordination. This is an important lesson because it emphasizes the importance in designing a robot system of minimizing the level of randomness allowed in the model. This can be best explained with an example. Suppose you want a swarm of robots to guard an area against intruders. The initial reaction is to treat the intruder detection as a random event. However, it's probably not completely random—or at least, not uniformly random. The intruder must have some patterns or objectives. Considering what that might be and making predictions based on that will allow you to reclaim the usefulness of your communication information. In a truly random environment, no one has anything useful to say.

6 Conclusions

In this paper, various communication mechanisms have been investigated for a swarm robot system under four different working scenarios, SSCT, DSCT, SSST, and DSST. Extensive simulation results demonstrate that it is necessary to select different communication mechanism for different scenarios. The research results presented in this paper should provide a solid, albeit limited, basis for early design decisions about communication algorithms in the development of swarm robot systems for particular tasks.

7 References

[1] M.J.B. Krieger, J. B. Billeter, and L. Keller, "Ant-like Task Allocation and Recruitment in Cooperative Robots," *Nature*, Vol. 406, 2000, pp. 992-995.

[2] M. J. Mataric, M. Nilsson, and K. T. Simsarian, "Cooperative Multi-robot Box-Pushing", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1995.

[3] A. Martinoli, A. J. Ijspeert, and F. Mondada, "Understanding Collective Aggregation Mechanisms: From Probabilistic Modeling to Experiments with Real Robots", *Robotics and Autonomous Systems*, Vol. 29, pp. 51-63, 1999.

[4] T. Balch and R. C. Arkin, "Behavior-based Formation Control for Multi-robot Teams", *IEEE Trans. on Robotics and Automation*, 1999.

[5] B. Yamauchi, "Decentralized Coordination for Multi-robot Exploration", *Robotics and Autonomous Systems*, Vol. 29, No. 1, pp. 111-118, 1999.

[6] T. Weigel, J. S. Gutmann, m. Dietl, A. Kleiner, and B. Nebel, "CS Freiburg: Coordinating Robots for Successful Soccer Playing", *Special Issue on Advances in Multi-Robot Systems*, T. Arai, E. Pagello, and L. E. Parker, Editors, *IEEE Trans. on Robotics and Automation*, Vol. 18, No.5, pp. 685-699, 2002.

[7] C.A.C. Parker and H. Zhang, "Collective Robotic Site Preparation". *Adaptive Behavior*. Vol.14, No. 1, 2006, pp. 5-19.

[8] O.E. Holland and C. Melhuish, "Stigmergy, Self-Organization, and Sorting in Collective Robotics", *Artificial Life*, Vol. 5, pp. 173-202, 1999.

[9] R. L. Stewart and R. A. Russell. "A Distributed Feedback Mechanism to Regulate Wall Construction by a Robotic Swarm". *Adaptive Behavior*. 14(1):21-51, 2006.

[10] H. Iba, T. Nozoe, and K. Ueda, "Evolving Communication Agents based on Genetic Programming", *Proceedings of 1997 IEEE International Conference on Evolutionary Computation*, 1997, pp. 297-302.

[11] Y. Meng, O. Kazeem, and J. Muller, "A Hybrid ACO/PSO Control Algorithm for Distributed Swarm Robots, 2007 IEEE Swarm Intelligence Symposium (SIS 2007), April 1-5, 2007, Honolulu, Hawaii, USA

[12] K. J. Mackin and E. Tazaki, "Emergent Agent Communication in Multiagent Systems using Automatically Defined Function Genetic Programming (ADF-GP)", *1999 IEEE International Conference on Systems, Man, and Cybernetics*, Tokyo, Japan, pp.138-142.

[13] C. L. Giles and K. Jim, "Learning Communication for Multi-Agent Systems", *Workshop on Radical Agent Concepts (WRAC) 2002*, pp. 377-390.

[14] M. Jang, A. A. Momen, and G. Agha, "Efficient Agent Communication in Multi-Agent Systems", *Software Engineering for Multi-Agent Systems III: Research Issues and Practical Applications*, LNCS 3390, 2005.

[15] E. Klavins, "Communication Complexity of Multi-Robot Systems", *WAFR 2002*, Nice, France.

[16] J. F. Yook, D. M. Tilbury, and N. R. Soparkar. "Trading Computation for Bandwidth: Reducing Communication in Distributed Control Systems using State Estimators", *IEEE Trans. On Control Systems Technology*, 10(5), July, 2002.

[17] K. Stoy, "Using Situated Communication in Distributed Autonomous Mobile Robotics", *Proceedings of the 7th Scandinavian Conference on Artificial Intelligence*, 2001.

[18] T. Balch and R. C. Arkin, "Communication in Reactive Multiagent Robotic Systems", *Autonomous Robots*, 1, 1994, pp.27-52.