Robot Swarming over the Internet

Jérôme Gilles, Balaji R. Sharma, Will Ferenc, Hannah Kastein, Lauren Lieu, Ryan Wilson, Yuan Rick Huang, Andrea L. Bertozzi, Baisravan HomChaudhuri, Subramanian Ramakrishnan, Manish Kumar

***Abstract*— We consider cooperative control of robots involving two different testbed systems in remote locations in different time zones, with communication on the internet. The goal is to have all robots properly follow a leader defined on one of the testbeds, while maintaining non-overlapping positions within each swarm and between swarms, assuming they are superimposed in the same virtual space. A dual-testbed design is developed involving real robots and remote network communication, performing a cooperative swarming algorithm based on a modified Morse Potential. Extensive experimental results were obtained with real internet communication and virtual testbeds running in each lab. The communication protocol was designed to minimize loss of packets, and average transfer delays are within tolerance limits for practical applications. We ran several experiments, with intentional packet loss, that illustrate the degradation of the results in the case of modest and severe packet loss. The novelty of this work is its experimental aspect involving long range network communication across a large distance via the internet. The work raises a series of interesting theoretical problems.**

I. INTRODUCTION

The efficient cooperation between multiple agents situated at distinct locations while pursuing common objectives is an important aspect of multi-agent systems. While the topic raises fundamental questions related to a variety of fields such as communication systems and distributed co-operative control, it is of immense practical interest as well. For example, as schematically represented in Fig. 1, collaborating unmanned vehicles can be used in combat or hazardous environment zones to reduce the risks for human lives. An important kind of cooperation is when agents communicate through a long-range network such as the Internet. In this article we report on our efforts to develop and implement algorithms designed to manipulate two sets of mobile robots, situated in two geographically distinct locations, in order to cooperatively realize a common task with information exchange through the World Wide Web.

Ryan Wilson and Jérôme Gilles and Andrea Bertozzi are with the Department of Mathematics, University of California Los Angeles, 520 Portola Plaza Los Angeles, CA 90095-1555, U.S.A. jegilles@math.ucla.edu

Will Ferenc and Hannah Kastein and Lauren Lieu are with the Department of Engineering, Harvey Mudd College, 301 Platt Boulevard, Claremont, CA 91711, U.S.A. Will_Ferenc@hmc.edu

Yuan Rick Huang is with Anteros Labs, Inc, Torrance, CA 90505, U.S.A. rhuang@anteroslab.com

Balaji R. Sharma, Baisravan HomChaudhuri and Manish Kumar are with the School of Dynamical Systems, 629 Rhodes Hall, University of Cincinnati, Cincinnati, OH 45221, U.S.A. sharmabr@mail.uc.edu, homchabn@mail.uc.edu, kumarmu@ucmail.uc.edu

Subramanian Ramakrishnan is currently with the Center for Nonlinear Dynamics and Control, Villanova University, Villanova PA 19085, U.S.A. s.ramakrishnan@uc.edu

Fig. 1. Concept of cooperative dual-platforms of unmanned vehicles.

While there have been many laboratory testbeds built to explore swarm algorithms, they all use local communications [1]. Our situation is novel because we use internet, with its inherent delays, as the support of our communications.

In order to experimentally explore algorithms for robotic applications such as the one described above, the University of California Los Angeles (UCLA) Applied Mathematics Department and the University of Cincinnati (UC) School of Dynamic Systems have built their own laboratory testbeds [2], [3]. The testbeds allow the study of single and multiple robots tasked with missions such as path planning, target searching and environmental exploration. The primary objective of the work reported in this article was to establish a connection through the Internet between the two testbeds aimed at exploring cooperative algorithms. It was determined that a robot swarming algorithm would be an ideal candidate to test the different aspects of the communication and co-operative control mechanisms. The experiments reported in this article may be summarized as follows. In one of the testbeds, one robot was designated as the leader and was programmed to follow a predefined path. On both testbeds, the remaining robots were asked to follow the leader based upon a swarming algorithm. Each vehicle (leader and followers) broadcast their positions, speeds and headings to the others through both local radio features and the Internet. Our purpose was to test different configurations: pure simulations between the computers located at UCLA and UC, and a combination of simulations and actual robots, as steps towards meeting the larger objective of implementation of the cooperative control algorithm on actual robots on each testbed communicating

Preprint submitted to 2012 American Control Conference.
Received September 25, 2011.

over the internet. This paper describes the design of the dual-platform including both internet communication and real robotic testbeds. We present detailed experimental results involving real internet communications and virtual robots running on the two remote locations. Section II gives an overview of each testbed and describe how network communications are set up. Section III describes the swarming algorithm used in the experiments. Section IV presents representative results from the simulations, and Section V presents the conclusions and outlines future research directions.

II. TESTBEDS CONFIGURATIONS AND COMMUNICATIONS

A. UCLA Testbed

The UCLA testbed is currently in its third generation and is composed of three main subsystems: (1) two overhead-cameras associated with a PC tracking system which detects each robot's position, speed and orientation (it mimicks the functionality of a GPS unit), (2) a remote terminal PC which can communicate with the robots and (3) several micro-car robotic vehicles (see Fig. 2 for a schematic of the test bed and Fig. 3 for a typical micro-car robotic vehicle). Physically, the testbed is a 1.5m $\times$ 2.0m rectangular area. Each car carries its own identification tag on its top surface in order to be recognized by the tracking system, and is equiped with two serial radio modules (one receives the "GPS" information and the other is used to communicate with the remote terminal). Onboard each vehicle are two electronic boards. The lower board has all features to drive the motor, the steering servo and receive GPS data. The upper board has a Virtex4 FX-2 FPGA which is used to program the different algorithms that are being tested. More details about vehicles and a complete description of the testbed can be found in [2], [3].

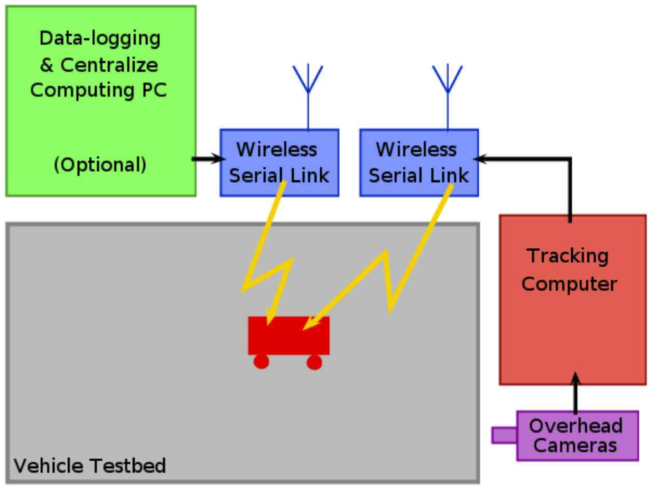

Fig. 2. UCLA testbed configuration.

B. UC Testbed

The UC testbed is based on Khepera III robots [4], [5] with embeded an XScale processor, infrared and ultrasonic

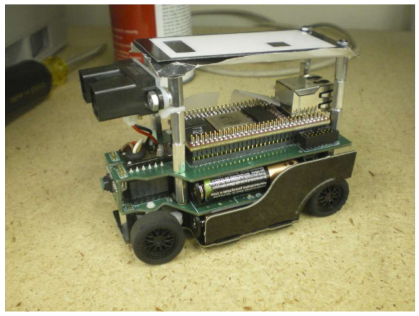

Fig. 3. UCLA micro-car vehicle. Its size is $50.8mm \times 101.6mm \times 45mm$.

sensors and wireless communication capabilities. The open-source Player/Stage software is used in order to communicate with and control the robots [6]. The Player server is installed on each Khepera robot and provides access to each sensor via TCP/IP protocol to interact with the robots. Like the UCLA's testbed, an overhead camera is connected to a tracking system installed on a remote terminal PC which mimicks the GPS function and relays global position information to the robots. This computer can also be used to exchange data with the robots by the previously mentioned TCP/IP protocol. An illustration of UC's testbed is depicted in Fig. 4.

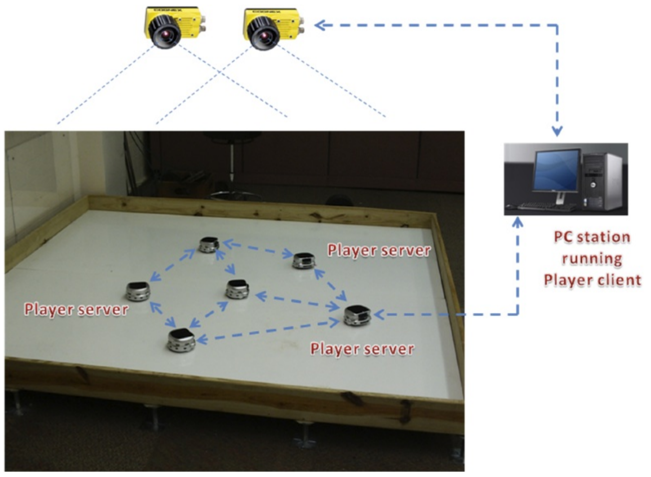

Fig. 4. UC testbed configuration. A Khepera III robot has a size of $130mm$ diameter and a height of $70mm$.

C. Communication between testbeds

We first note that in [7], the authors discuss the use of TCP/IP communication between robots and a user interface while [8] provides details about remote simulation using internet. Based on these papers, we determined that two different types of communication are needed for our purposes, viz. local and nonlocal. Local communications are supported by radio/wireless functions embedded in each robot. This allows each vehicle to broadcast the necessary informations (position, speed and orientation) to the others and to the remote terminal PC (RTPC). It also permits RTPC to send informations to the robots and eventually send instructions

to the leader.
Network communications are established between each testbed RTPC located in each university by using the TCP/IP protocol over a VPN (Virtual Private Network) through the Internet. See Fig. 5 for a complete configuration view. The testbeds can thus communicate with each other over the established network. In addition, each robot belonging to each testbed can be updated on the status of the others (including the leader) using this link.
For practical reasons, we chose to use the Instrument Control Toolbox in Matlab for establishing communication. The R2011a release of Matlab allows for Matlab to act as a server, a feature previously nonexistant. This toolbox permits us both to drive the wireless communication between robots and RTPC, and to establish the TCP/IP connection through the Internet between the two RTPCs.

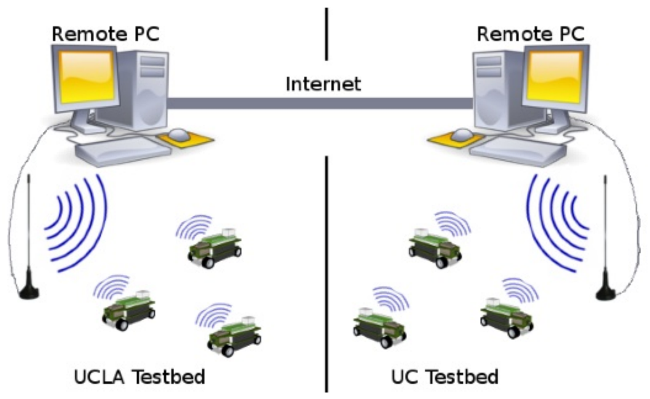

Fig. 5. Communication configuration between testbeds.

We conclude this section by observing that each swarming code has its own matrix containing the information regarding position, orientation and speed for all the robots (both at UCLA and UC) under consideration. One testbed then sends an updated version of the information about its robots to the other and also listens for the updated data from the other robots. Specifically, if $X_i, Y_i, V_{x_i}, V_{y_i}$ represents the two dimensional Cartesian co-ordinates of $i-th$ robot and its velocity components respectively, each message sent by a testbed has the following format:

$$(X_1\ X_2\ X_{N_p}\ Y_1\ Y_2\ Y_{N_p}\ V_{x_1}\ V_{x_2}\ V_{x_{N_p}}\ V_{y_1}\ V_{y_2}\ V_{y_{N_p}}) \quad (1)$$

where $p \in \{UCLA, UC\}$ and N_p is the number of robots evolving on the corresponding testbed.

III. SWARMING ALGORITHM

The goal of the swarming algorithm is for a group of robots to follow a leader while avoiding collisions and maintaining formation. For simplicity we assume that all robots on each testbed occupy the same 'set space' and thus avoid collisions within each testbed and between members of the different testbeds. We note that in [9] a swarming algorithm without communication is proposed while in [10] simulation results based on an interesting potential model are reported. A decentralized control algorithm based on differential artificial potential is discussed in [11], in the context of self-sorting in a group of heterogeneous agents. Another work [12] uses three-dimensional mapping and graph theory to design swarming algorithms. In [13], the question of controlling large swarms of robots is addressed. After a survey of the literature keeping in mind the fact that only limited computing resources are available on robots, we chose the algorithm developed in [14] where an easily implementable [15], exponential potential model is discussed. Precisely, the following differential equations govern the dynamics of each robot except the leader which operates independently of the swarm. Practically, the velocity v_i and the movement direction (resulting from x_i coordinates) are provided to the motion controller of the robot.

$$\frac{dx_i}{dt} = v_i \quad (2)$$

$$\frac{dv_i}{dt} = \left(\alpha - \beta\|v_i\|^2\right) v_i - \nabla U(x_i) + \sum_{j=1}^{N} C_0(v_j - v_i) \quad (3)$$

$$U(x_i) = \frac{1}{2}C_l(x_i - y)^2 + \sum_{j=1}^{N}\left(C_r e^{\frac{\|x_i - x_j\|}{l_r}} - C_a e^{\frac{\|x_i - x_j\|}{l_a}}\right) \quad (4)$$

In the above system of equations, U is the potential function, N is the number of robots on the testbed (including both the UCLA robots and the UC robots) and y is the position of the leader robot. The constants associated with the model are $m, C_0, C_l, C_r, C_a, l_r$ and l_a. Here m refers to the robot mass, C_0 is the velocity alignment coefficient which dictates that the robots have identical velocities or otherwise, C_l is the leader potential coefficient and represents the strength of attraction the followers experience towards the leader, C_a and C_r are respectively the robot attraction and repulsion coefficients that ensure collision avoidance while keeping a compact formation and l_a and l_r are the robot attraction and repulsion lengths, respectively. The reader is referred to [14] for a more detailed analytical evaluation of this control model.

IV. EXPERIMENTS

A. Pure simulations

We first checked the consistency of the swarming code without network communications, simulating a variety of scenarios with different pre-programmed paths for the leader, for optimal selection of gain values for the control algorithm being implemented. Next, we set up the network connection between UCLA and UC, with the UC RTPC acting as the server and the UCLA RTPC as the client. A leader was defined on UC's side and ten virtual robots were assigned to each virtual testbed (see Fig. 6). For the sake of simplicity, the leader is programmed to follow a circular path in the virtual testbed. A TCP/IP socket is created connecting the RTPCs at the two ends, with I/O buffers facilitating the data exchange on a First-In-First-Out (FIFO) basis. The swarming code is implemented both sides, governing the dynamics of the virtual robots in such a manner as to form a swarm that follows the leader.

Swarming algorithms for cooperative control of agents over the internet are very sensitive to network performance, and must pay close attention to delays in data transmission over the network, critically so in relation to the rate at which control commands are sent to agents. If control commands from each RTPC to its local agents can be sent at a rate low enough to accommodate any network delays while maintaining smooth continous actuation of the agents, data exchange - and the resulting cooperative behavior between the two sets of agents - can be effectively implemented without any loss in data. Such an implementation, if permitted by actuation constraints on individual agents and network characteristics, would be most desired and would allow for establishing baseline performance for comparisons against more constrained cases. We begin by presenting results from such a case, where actuation rates for the agents are low enough to accommodate any network delays, ensuring control action in the absence of any loss of information during data exchange over the network.

At the beginning of the experiment all robots are initiated in a random non-overlapping configuration. The leader, positioned on the UC platform, follows a prescribed circular trajectory. Results of the simulation are shown in Fig. 7 with the leader marked by a star and the agents marked by circles - the red ones denoting the UC agents and the blue ones denoting the UCLA agents. Note that the agents assemble into a circular region with non-overlapping positions despite being located on remote sites. The swarm is observed to closely follow the trajectory of the leader on the UC platform. Average transfer delays across the length of the simulation were observed to be 0.1240 and 0.1154 seconds at the UC and UCLA ends respectively, with a maximum of about 0.3-0.35 seconds. The data exchange was synchronous with no loss of data packets during the exchange. The effective convergence of the magnitudes of individual agent velocities to a common value can be seen in Fig. 8.

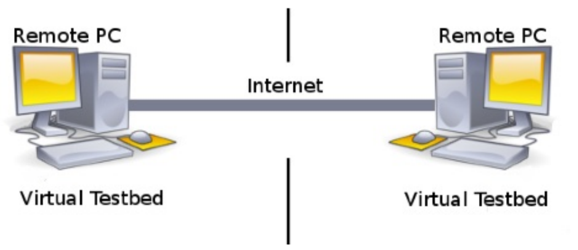

Fig. 6. Virtual testbeds experiment.

Subsequently, three control scenarios, referred to as cases (1), (2) and (3), are implemented in simulation to explore swarming behavior under varying network performances relative to agent actuation rates. Specific actuation rates are defined for the agent groups in each case, resulting in data

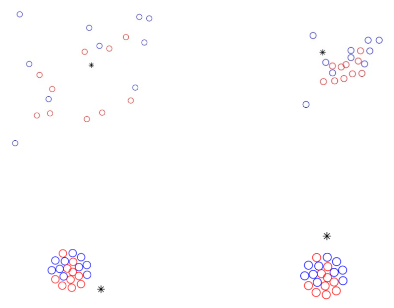

Fig. 7. Complete virtual swarming. The leader moves on a circular trajectory. UCLA and UC agents are represented in blue and red respectively. Frames show the evolution of the robots during the experiment at iteration times 107, 170, 1475, and 1999.

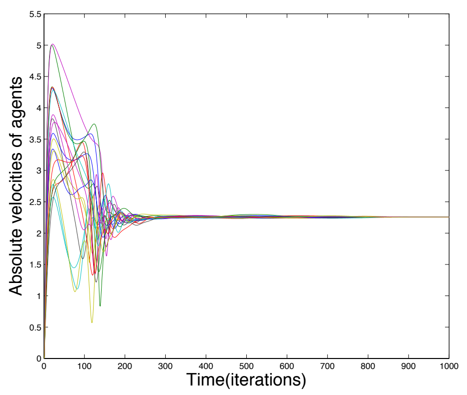

Fig. 8. Convergence of velocities of individual agents in the experiment show in Fig. 7, to a common magnitude.

exchanges across the network taking longer than the rate at which control commands are sent from the RTPC to the local agent group. The result is a loss of data from the other agent group. In the event of such network losses, the control algorithm reconfigures instantaneously to implement swarming behavior based on information from local agents only. The availability of the virtual leader information to both sets of agents, however, is assumed at all times. Assuming fairly consistent network behavior across all cases, a very tight constraint is placed in Case (1) on the acceptable delay in data exchange over the network, resulting in significantly high losses (in excess of 90% at both ends) in the exchange of position and velocity information between the two groups of agents over the length of the simulation. An increased tolerance for network delays in Case (2) effects a reduction in data losses to around 5-6% at both ends. The constraints are relaxed further in Case (3), allowing for simulating control behavior under minimal network losses.

The intermittence in the availability of state information among the two sets of agents, over the length of the simulation run, results in variations in the control action for each set of agents at each instance of loss of data. Consequently, the convergence of agent velocities to a common value is perturbed at such instances. The instantaneous agent velocities then proceed to converge towards the common velocity from the disturbed states until further disturbed by data loss.

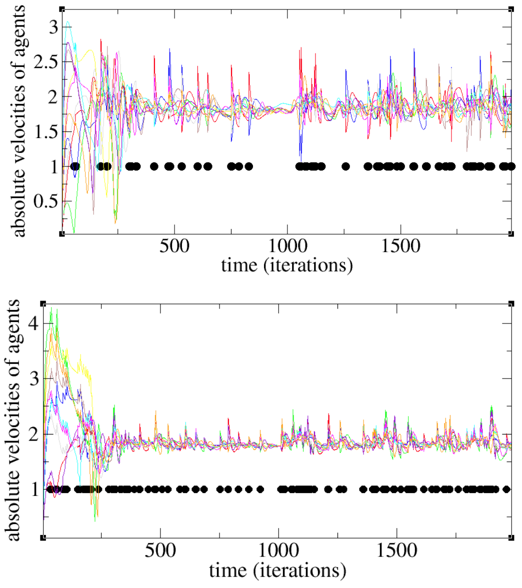

Fig. 9. Case (1): Agent velocities (UC) top; Agent velocities (UCLA) bottom. Large packet loss case, the packets transmitted are shown as black dots. Only 57 packets out of 2000 are transmitted to the UC system whereas only 53 packets out of 2000 are transmitted to the UCLA system.

In Case (1), the data losses are high and result in long periods of unavailable data from the non-local agents, resulting in agent velocities being continuously disturbed with significant variations about the expected velocity of convergence. This is demonstrated, for agent groups at UC and UCLA respectively, in Fig. 9. Note that the significant spikes in the velocity data occur during periods of packet transmission (denoted by black dots) rather than packet loss, suggesting that the quiescent periods are marked by independent and separate control of the swarms with packet transmission causing a disruption of this independence. Data losses are less frequent in Case (2), as can be observed in Fig. 10 for the agents at UC and UCLA respectively, with the instances of *data loss* represented by black dots. Bounded disturbances to the convergence of agent velocities occur at each instance of data loss, and tending to die down till another instance of data loss is encountered. Case (3) represents a highly desirable control scenario where data losses over the network are expected to be minimal, without significantly affecting the overall swarm behavior. This is shown in Fig. 11, where data losses induce variations in agent velocities which in the absence of further losses progressively die down, resulting in the reconvergence of agent velocities to the common velocity. In all time-series plots, we show the sparser case, of either transmitted packets or lost packets, as black dots for ease of visualization of this large data set.

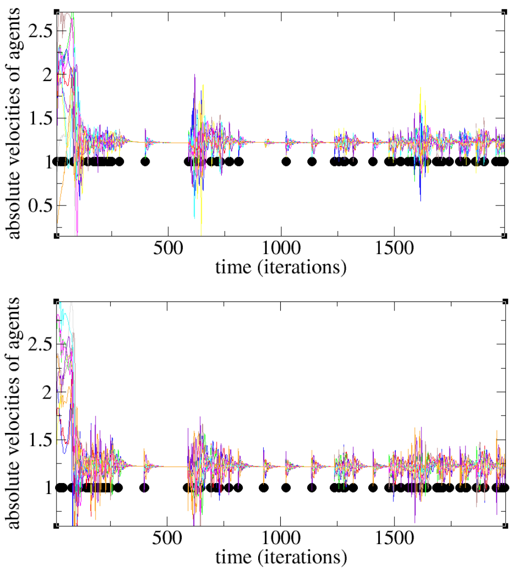

Fig. 10. Case (2): Agent velocities (UC) are shown on top and for (UCLA) on the bottom. Packet loss is 103/2000 for UC and 109/2000 for UCLA. The black dots are the lost packets, shown as a time series in each figure.

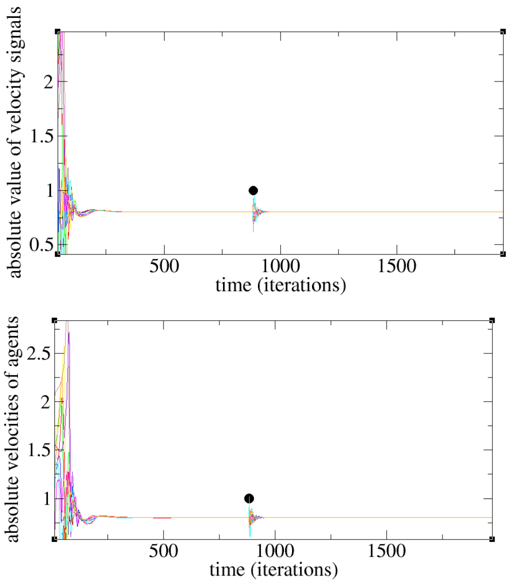

Fig. 11. Case (3): Velocities of UC agents are shown on top and the velocities of the UCLA agents are shown on the bottom. A single packet loss (shown as a black dot) occurs in the middle of the simulation.

These series of simulations allow for an understanding of the applicability and constraints associated with data exchange through the Internet for multi-agent swarming in non-local robot groups. The simulations also serve to provide an estimate of delays expected in the data exchange and allow for the determination of optimal actuation rates on physical robots for effective implementation of the control algorithms on each agent group. The experiments also allow for a strong understanding of the capabilities and limitations of data exchange over the internet for real-time applications such as is presented in this work. Further work is intended in the exploration of control strategies towards minimizing the effect of network delays and losses on swarming behavior. The possibility of a dynamically evolving control strategy based on a continuous evaluation of the network performance is also under investigation.

V. CONCLUSIONS AND FUTURE WORKS

This article investigates the possibility of using networks such as the Internet with their own communication constraints (time-lag in transmissions, imposed protocols, etc) in order to control the motion of two groups of robots physically living in distinct geographical locations. We established a connection between the Applied Math Laboratory of University of California at Los Angeles and the Cooperative Distributed Systems Laboratory of the University of Cincinnati over the Internet. This connection allowed us run a swarming algorithm on both sides where only information on positions, speeds and orientations were exchanged. Results obtained from the different experimental configurations demonstrate that the concept of using the Internet as a link between robotic platforms is valid and has immense potential for a variety of applications. The experimental influence of network delays and loss of data during simulations on the swarming behavior, particularly on the convergence of agent velocities to a common value, is presented. In addition to the experiments presented above, using virtual robots and real internet communications, we simulated a remote RTPC on a second computer plugged on the local network which interacted with the real testbed's RTPC in the UCLA Applied Mathematics Laboratory. This experiment allowed us to observe the influence of different latent times inherent to radio communications and network transmissions and investigate time-lag issues. We observed that for a small number of robots, the latent times were not significant constraints.

Work on implementation of the swarming algorithm across the IP network (demonstrated in simulation here), on physical robots is in progress. While simulations permit certain assumptions on the similarity in physical capabilities and actuation rates such assumptions are not always valid for physical robots, and the network exchanges would have to be adapted around such constraints. In future work, more complex algorithms involving cooperative behavior through the Internet such as target searching could be tested. The influence of heterogeneous control gains in the interacting swarms to offset physical constraints on the robots could be investigated too. Another interesting experiment would be to consider that the two testbeds represent the same area but at different scales. In this case, one will have two categories of robots: the ones who have a global view of the situation and local ones that have access to more details and could broadcast finer aspects of information. Applications are also possible in remote coordination among robots in geographically separated manufacturing units for networked production and conveyance. With growing internet coverage even in remote regions, large scale terrain exploration could also benefit from such networked coordination, with local swarms at various regions of the unexplored terrain interacting over the internet for more efficient mapping at a global level.

The experiments involving packet loss pose some interesting theoretical questions. Presumably there is a threshold for packet loss above which the cooperative algorithm performs well and below which it breaks down. It would be interesting to derive rigorous bounds on such rates as a function of the choice of swarming model and the number of agents.

VI. ACKNOWLEDGMENTS

The authors thank their respective IT staff for assistance in establishing the different network configurations. The University of Cincinnati researchers thank Timothy Arnett, Kevin McClellan and Kaitlyn Fleming. This work is supported by the following grants: NSF-DMS-1045536, NSF-EFRI-1024765, NSF-EFRI-1024608, NSF-DMS-0914856, and ONR N000141010641.

REFERENCES

[1] M. Li, K. Lu, H. Zhu, M. Chen, S. Mao, and B. Prabhakaran, "Robot swarm communication networks: Architectures, protocols, and applications," in *Communications and Networking in China*, Aug. 2008, pp. 162–166.

[2] M. Gonzalez, X. Huang, B. Irvine, D. H. Martinez, C. Hsieh, Y. Huang, M. Short, and A. Bertozzi, "A third generation micro-vehicle testbed for cooperative control and sensing strategies," in *Proceedings of the 8th International Conference on Informatics in Control, Automation and Robotics(ICINCO)*, 2011.

[3] J. Irvine, M. Gonzalez, and E. Huang, "Integration of third generation micro-cars into the UCLA Applied Mathematics Laboratory testbed," UCLA REU summer, Tech. Rep., August 2010.

[4] [Online]. Available: http://www.k-team.com/mobile-robotics-products/khepera-iii

[5] F. Mondada, E. Franzi, and A. Guignard, "The Development of Khepera," in *Experiments with the Mini-Robot Khepera, Proceedings of the First International Khepera Workshop*, ser. HNI-Verlagsschriftenreihe, Heinz Nixdorf Institut, 1999, pp. 7–14.

[6] [Online]. Available: http://playerstage.sourceforge.net/

[7] M. Sugisaka and D. Hazry, "User interface in web based communication for internet robot control," in *Proceedings of the International Conference on Control, Automation and Systems (ICCAS)*, 2005.

[8] A. Turan, S. Bogosyan, and M. Gokasan, "Development of a client-server communication method for matlab/simulink based remote robotics experiments," in *Proceedings of the IEEE International Symposium on Industrial Electronics*, 2006.

[9] C. Moeslinger, T. Schmickl, and K. Crailsheim, "Emergent flocking with low-end swarm robots," in *Proceedings of the 7th IEEE International Conference on Swarm Intelligence ANTS'10*, 2010.

[10] A. Scott and C. Yu, "Cooperative multi-agent mapping and exploration in webots," in *Proceedings of the 4th International Conference on Autonomous Robots and Agents*, 2009.

[11] M. Kumar, D. Garg, and V. Kumar, "Segregation of heterogeneous units in a swarm of robotic agents," in *IEEE Transactions on Automatic Control*, vol. 55, no. 3, Mar. 2010, pp. 743 –748.

[12] K. Cao, G. Xiang, and H. Yang, "Formation control of multiple nonholonomic mobile robots," in *Proceedings of the 7th IEEE International Conference on Information Science and Technology*, 2011.

[13] J. McLurkin, J. Smith, J. Frankel, D. Sotkowitz, D. Blau, and B. Schmidt, "Speaking swarmish: Human-robot interface design for large swarms of autonomous mobile robots," in *Proceedings of the AAAI Spring Symposium*, Stanford, CA, USA, 2006.

[14] W. Liu, Y. Taima, M. Short, and A. Bertozzi, "Multi-scale collaborative searching and swarming," in *Proceedings of the 7th International Conference on Informatics in Control, Automation, and Robotics (ICINCO)*, Madeira, Portugal, 2010.

[15] B. Nguyen, Y. Chuang, D. Tung, C. Hsieh, Z. Jin, L. Shi, D. Marthaler, A. Bertozzi, and R. Murray, "Virtual attractive-repulsive potentials for cooperative control of second order dynamic vehicles on the Caltech MVWT," in *Proceedings of the American Control Conference*, 2005.