Modeling, Simulation, and Characterization of Distributed Multi-agent Systems

Reed F. YOUNG
U.S. Army Yuma Proving Ground
Yuma, AZ 85365

and

Devendra P. GARG
Mechanical Engineering and Materials Science Department, Duke University
Durham, NC 27708 USA

ABSTRACT

A strategy is described that utilizes a novel application of a potential-force function that includes the tuning of coefficients to control mobile robots orchestrated as a distributed multi-agent system. Control system parameters are manipulated methodically via simulation and hardware experimentation to gain a better understanding of their impact upon mission performance of the multi-agent system as applied to a pre-determined task of area exploration and mapping. Also included are descriptions of experiment infrastructure components that afford convenient solutions to research challenges. These consist of a surrogate localization (position and orientation) function utilizing a novel MATLAB executable (MEX) function and a user datagram protocol (UDP)-based communications protocol that facilitates communication among network-based control computers.

Keywords: Control, Multi-agent System, Swarming Robots, Potential Function.

1. INTRODUCTION

From the inception of the development of robotic technologies, researchers have generally concluded that, with many if not most applications, there are significant advantages to be gained through the coordination and orchestration of multiple robotic entities [1-2]. This is not difficult at all to imagine by simply observing Darwinian development in biological systems whereby two or four legs proved superior to one for locomotion; or separate arms each with a hand, fingers, and an opposing thumb proved superior for grasping and cooperatively manipulating objects. Furthermore, it similarly is understandable that a small group of separate robotic entities (e.g., a small number of cooperating or coordinating unmanned ground vehicles) could also exhibit significantly desirable characteristics in certain circumstances. Five main motivations for using multi-robot systems include: task complexity; task distribution; resource distribution; parallel processing; and robustness through redundancy [3]. Some researchers have explored the advantages gained for specific tasks such as exploration where a specified task can be accomplished by a number of agents surveying the maneuver space versus a single robot tasked with exploring the entirety by itself [4]. Indeed, these become valuable against overarching task accomplishment; however, not without burden. Multi-robot systems add communications overhead and workload with the requirement for inter-robot cooperation and knowledge. They also add complexity to path planning and collision avoidance schemes since each robot's cooperating peer also exists as an obstacle to be avoided [5].

Research over the past years has centered on organizing several to tens of mobile robots (or agents) into coordinated multi-agent systems (MAS) [6-8]. Further, the quantity of these systems may reach into the hundreds or thousands that comprise a robotic swarm. In contrast to single mobile robots with their broad, comprehensive capabilities (vision, LIDAR, differential global positioning, ultrasonic proximity sensing, etc.), the agents comprising a MAS tend to have somewhat lesser organic capability due to a desired affordability or individual physical size. An example of one such capability is localization that could provide accurate agent position and orientation in a global reference akin to the Global Positioning System (GPS). While GPS receivers have become small and inexpensive, they cannot be used indoors where MAS experimentation and utilization typically exist. Research in the field has centered on understanding the root mechanisms and control features that can define and drive the behavior and capability of the MAS architectures in terms such as task allocation, flocking, or foraging. The study of analogous biological systems has also provided valuable insight. However, to date, there has been little work done with respect to understanding the capability of the MAS as a function of its composition given the option of pre-mission selection of the performance and capability of individual agents, as well as opportunity to include extra-system information sources or sensors.

Various attempts at defining MAS composition [9-11] explored heterogeneous versus homogeneous systems, highlighting concepts such as hierarchic social entropy, diversity, and cooperative localization. Heterogeneous MAS have been used to explore unknown environments for the purpose of mapping and exploration [12]. Here, each MAS featured varying types of single-mode sensor payloads including, for example, an infrared camera or sonar, coordinated in an occupancy grid Bayesian mapping algorithm. But again, the composition of the architecture was held constant.

An example of a MAS configuration is illustrated in Figure 1. Here, four differential-wheeled robotic agents featuring distributed control architectures orchestrate to conduct a mapping of an initially-unknown discrete arena to localize and identify targets and obstacles. Each agent has imaging and proximity sensing. They pass information to each other via a consolidated state matrix that receives data from each agent and simultaneously provides state updates to each agent.

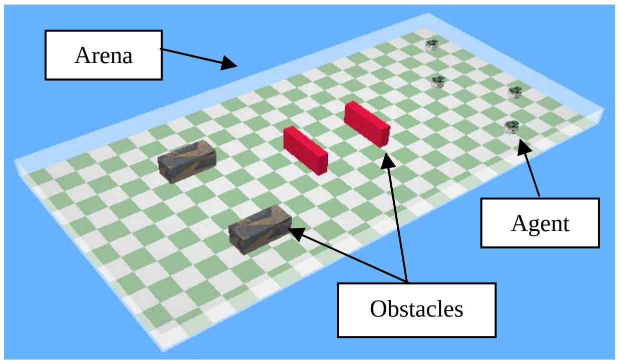


Figure 1: Representative Multi-agent System Arena.

The underlying control architecture plays an absolutely critical role in the efficiencies observed in the multi-agent versus single-robot architectures [13-16]. Clearly, beyond the inherent capability of the electro-mechanical configuration as well as the sensor payloads, the single greatest contributor to mission efficiencies and task capabilities will exist in the capability, adaptability, and comprehensiveness of the control system. A wide variety of algorithms and schemes have been explored including feedback laws, leader follower, task-oriented mission planning, task reallocation and reordering, and deliberative and reactive behavior modeling. Beyond these, potential-function-based robot control strategies exhibit significant promise to control payload-agile MAS.

This paper is organized as follows: Section 2 presents a potential-function based robot control strategy that uses parametric tuning factors to increase the efficiency with which the control strategy can accomplish an exploration task with no a priori information of the search space. Section 3 describes results from the research's simulation phase. Section 4 describes the creation of a novel surrogate global positioning system based on a task-specific MATLAB MEX function sampling an optical motion capture system broadcasting over a network socket. Additionally, this section summarizes a UDP-based data transmission architecture that enabled inter-agent sharing of realized occupancy map elements. Sections 5 offer the results from the research's experimentation phase whereby increasing quantities of agents with various capabilities were placed in the arena and evaluated for task accomplishment metrics such as time to complete. The last sections summarize the research results and conclusions.

2. POTENTIAL-FUNCTION-BASED ROBOT CONTROL STRATEGY

Control via potential-function theory has proven to be a valuable basis upon which more complex control feedback influence can be implemented. In particular, potential-function theory affords a comprehensive and simple method to add factors that can be scaled either as independent input functions or those dependent upon selected system features. The basis of the potential function is to represent system control influences as independent or dependent forces [17] akin to an electrical potential field. Each force is calculated based on physical parameters (such as real distances) or on representative values (such as arbitrary repulsion to facilitate obstacle avoidance or attraction to a desired target location). In conjunction with the potential function, it is convenient to represent the operational space as an occupancy map, i.e., a two- or three-dimensional grid that sub-divides the space into identifiable cells. Each cell would have characteristics such as a potential related to its occupancy status, or status as a target destination. Since the potential function depends upon proximity, the relative location of the cells would also be cataloged.

The goal of the control function for the MAS is to calculate the agent-unique potential force witnessed by each of the agents comprising the MAS. Force defines the resultant heading and speed that the individual agent would follow. The potential force existent on the ith agent is given by:

$$\hat{F}_i = Q_i\left(\sum_{j=1}^{N_C}\frac{\mu_1 q_j}{\|rad_j\|^2}\widehat{rad_j} + \sum_{j=1}^{N_A-1}\frac{\mu_2 Q_j}{\|rad_j\|^2}\widehat{rad_j}\right) + \sum_{j=1}^{L}\mu_3 P_j \quad (1)$$

where,

Q_i = inherent charge of the ith agent,
Q_j = inherent charge of the jth agent,
q_j = inherent charge of the jth cell,
rad_j = radius vector from the ith agent to the jth agent or jth cell,
N_C = number of cells in the occupancy map,
N_A= total number of agents,
P = calculated force contribution from specified payloads, and
μ_x = tuning coefficients.

Then, the resultant heading angle (θ_i) of the ith agent is given by:

$$\theta_i = \tan^{-1}\left(\frac{F_{Z\,i}}{F_{X\,i}}\right) \quad (2)$$

The calculation of P will vary with payload type as the value and weighting of each payload type can and should have varying degrees of influence on the resultant potential force calculation.

Tuning coefficients (μ_x) provide a mechanism whereby the relative contribution of each of the potential forces can be directly tuned either as an independent variable, or dynamically tuned based on a feedback mechanism. This feature is particularly important in multi-agent systems due to scalability. Since the potential forces contributed by elements within the system are cumulative, it becomes very important to gain an awareness of the proportional contributions and to subsequently be able to control them proactively. To illustrate, assume an occupancy grid of 120 by 240 cells or a total of 28,800 total cells. Each cell might offer a contributing charge value from 0 to 1 meaning a span of 28,800 charge units. If other contributors merely summed to a maximum of 1,000 charge units, the cells' contribution would easily overwhelm any potential contribution calculation.

The occupancy map cell characteristic vector represents the probability that the cell is occupied. A value of 1.0 indicates that there is a 100 percent chance that the cell is occupied. A value of 0.0 indicates that there is a 100 percent chance the cell is unoccupied. A value of 0.5 would indicate an equal probability that the cell is either occupied or unoccupied. Separate variables that provide valuable influence in the force calculation are the number of times that a cell has been sampled by a sensor, referred herein as the "cell visit count," and the number of times that a fraction of the entire occupancy map has been visited. The usefulness here is to accommodate the dynamic nature of agents moving in the arena as well as the potential for moving targets in the field of regard. As a result, the charge associated with each cell is given by:

$$q_i = \begin{cases} P_O q_{max} & \text{for } 0<V<n \\ 0 & \text{for } V>n \end{cases} \tag{3}$$

where,

P_O = cell occupancy probability,

q_{max} = maximum possible cell charge,

V = cell visit count defined as the number of visits to the ith cell, and

n = number of times that a percentage of the cells have been visited.

3. INITIAL OBSERVATIONS THROUGH SIMULATION

Initially, the Cyberbotics' Webots simulation package was used to represent a 120 by 240 centimeter arena (Figure 1) occupied by e-puck differential-wheeled robots. The arena is sub-divided into a representative occupancy mapping grid consisting of 1 cm by 1 cm cells each having an associated vector representing dynamic characteristics such as the probability of occupancy or potential charge. The MATLAB scripting language was utilized to provide high-level potential-function based path planning as well as map databasing.

A comprehensive set of state and evaluation metrics are collected at every control time step that provide an opportunity to observe the progression of the exploration task. Agent trajectories (Figure 2) provide insight to the energy consumed by each individual agent in terms of the distance travelled. Trajectories also illustrate the qualitative uniformity and dispersion with which the team of agents uniquely maneuver in and canvas the arena.

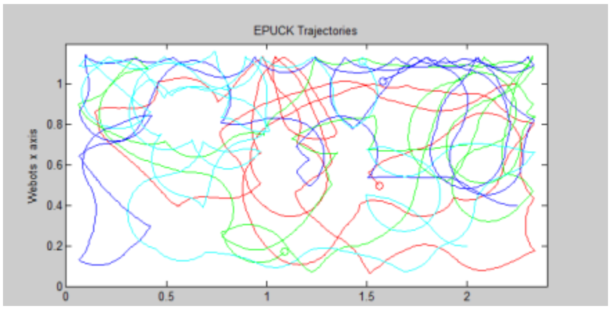


Figure 2: Agent Trajectories.

The occupancy map data vector (Figure 3) at a specific time step is displayed to show the occupancy probability, cell charge, and cell visit count. Observed individually and sequentially over time, these vectors provide insight to the progression of understanding of the heretofore unknown arena and resolution with which obstacles and targets appear. One can also conclude trends such as the extent of exploration and behaviors proximate to obstacles.

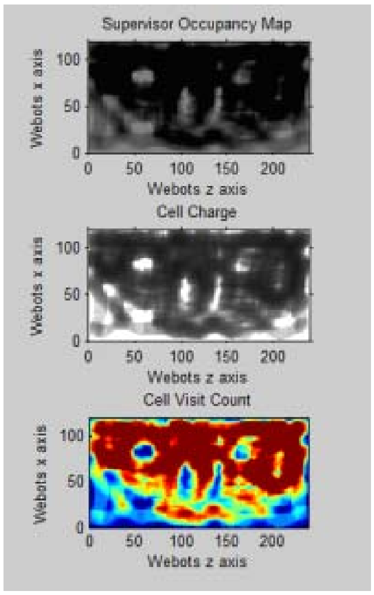


Figure 3: Occupancy Map Data Vectors.

Finally, mean characteristic values graphed over time (Figure 4) quantify the performance of the multi-agent system and illustrate its ability to progress toward threshold goals. The average occupancy map value shows the time progression of the average across each cell's probability of occupancy. Since there is no a priori knowledge afforded, the initial value is 0.5 meaning all cells have equal probability of being occupied or unoccupied. Over time, as the agents explore the arena and determine free and occupied space, the curve progresses as expected decreasing over time with the free space determined and step increasing periodically with the sensor-enabled detection of obstacles, i.e., occupied space. The curve is non-linear due to the proximity of unexplored space diminishing over time resulting in the necessity to travel longer distances (and consuming more time) to sample those unexplored spaces. The average cell charge plots the average of all the cells' calculated charge values over time. This curve initially decreases due to the fact that as cells are explored, their charge decreases so as to have less impact in subsequent path planning loops. However, in deference to the dynamic state of the arena, the average cell charge begins to increase in correlation with the map coverage count due to the forcing relationship in Equation (3). The average cell visit count tracks the average number of times that any agent samples individual cells over time. This curve has a constant slope as expected knowing that the agents' speed varies very little perhaps only to slow for short period of times while maneuvering around obstacles. Finally, the map coverage count (n, Equation 3) is an evolving parameter that tracks the number of times that a pre-defined percentage of the arena has been sampled at least the count's number of times. For example, the count starts at zero. As soon as 75 percent of the arena has been sampled once, the count increments to 1. As soon as 75 percent of the arena has been sampled twice, the count increments to 2. This parameter is used within the control loop to influence the charge calculation in order to accommodate the dynamic nature of the state representation. The shape of this curve provides qualitative clues as to the efficiency of the search.

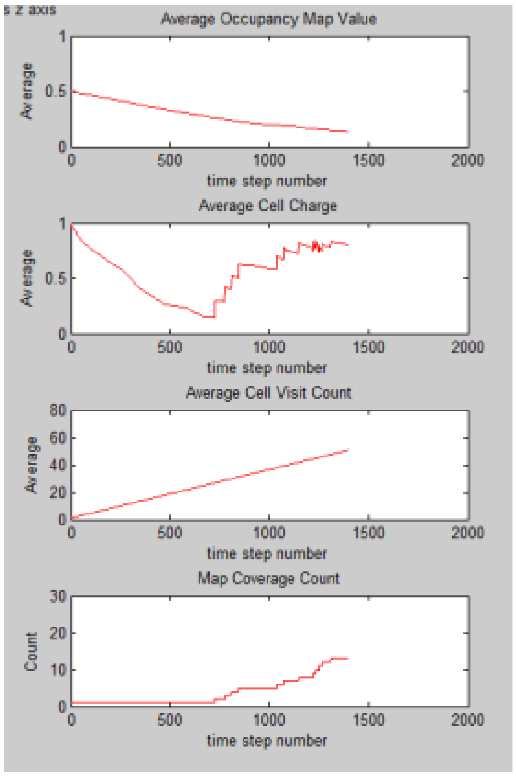


Figure 4: Characteristic Value Histories.

4. DISTRIBUTED MAS EXPERIMENTATION

The transition from simulation to hardware experimentation is a non-trivial task. There are many real-world challenges in localization and communications that can be inaccurately represented as perfection in the simulation. In support of on-going MAS research at Duke University, an arena (Figure 5) has been created consisting of a 120 by 240 centimeter walled stage hosting various numbers of e-puck robots [18], obstacles, and targets. The e-puck robot is approximately 7 cm in diameter. It features a Microchip dsPIC processor running at 60MHz or about 15 MIPS. Mobility is provided by two stepper motors in a differential-wheel drive configuration. The robot has eight infra-red sensors positioned around the robot's perimeter that provide both proximity sensing and light sensing. There is a 640x480 pixel camera on-board that can provide video and imagery in either color or black and white modes. Bandwidth limitations force pragmatic collection to 52x39 or 640x1 pixel images at up to three frames per second. There are three omni-directional microphones for sound localization, a three-axis accelerometer, and on-board speaker. Communications is provided by Bluetooth. Each robot has a dedicated desktop computer that remotely controls the robot via Bluetooth communications. This dedicated computer allows for substantial processing capability, image processing in particular, that simply isn't supportable via the e-puck limited on-board processing capability. The control software is written in MATLAB programming language. The e-puck robots have no inherent localization capability.

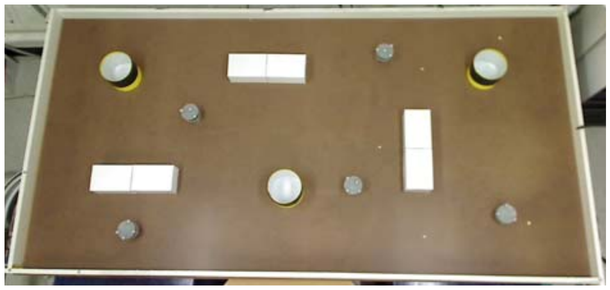


Figure 5: MAS Experimental Arena.

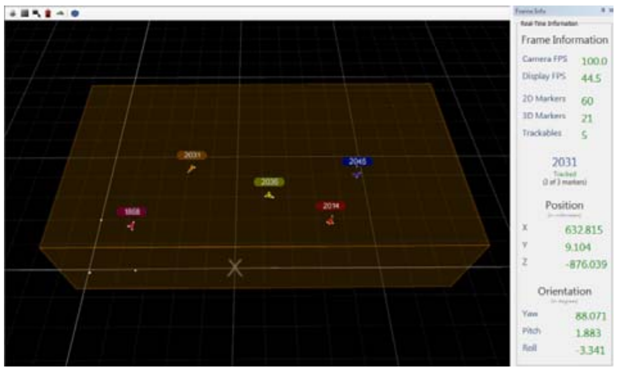
Two novel components of this experimentation infrastructure required to fully replicate the simulation environment include a surrogate global positioning system (GPS) and data packet transfer between agent controllers. Both capabilities, developed under the auspices of this research, are described in the following subparagraphs.

Surrogate GPS via External Optical Motion Capture and Tracking

Much research effort thus far has been focused on the simultaneous localization and mapping (SLAM) problem [19]. SLAM acknowledges that externally-provided localization information is not typically available, and also acknowledges a scenario of exploration of heretofore unknown space. Therefore, clever schemes are explored that use available modalities (e.g., inter-agent communications) to determine relative localization among the agents. That, combined with robot-sensor-enabled, continually-updated mapping databases, generates a comprehensive representation of the spatial and spectral environment. Many times though, inherent SLAM functionality is not germane to the research problem being pursued. In this case, an extra-MAS localization capability is most desirable to minimize the experiment's independent variables.

There are several materiel possibilities for an extra-agent surrogate GPS capability. Fricke et al [20] utilized a combination of a Cognex camera and Sick LIDARs to calculate localization. The Vicon motion capture system [21] is a popular solution that uses connected high-speed digital cameras viewing the MAS experiment workspace with redundant coverage to provide localization of tracked objects.

The Natural Point OptiTrack™ system [22] utilizing its 'Tracking Tools' (TT) software application (Figure 6) provides a similar capability. The OptiTrack™ cameras are installed and calibrated for the desired experiment workspace. Each robot is given a unique spatial configuration of reflective markers (Figure 7) that enables individual robot identification during the tracking process. A critical functionality important to supported experimentation that TT offers is the ability to multicast tracked-object localization in real time through a designated network port on the computer hosting the Tracking Tools application. If connected to a local-area network, any other computer (e.g., one hosting an agent controller) could also have near-instantaneous access to the real time streaming position information of itself, and of any other agent in the workspace with update rates approaching 100 Hertz. This is particularly desirable as this streaming broadcast scheme significantly reduces the inter-agent communications requirement. TT offers three formats for broadcast: industry-standard Virtual Reality Peripheral Network (VRPN) and Trackd®, and NaturalPoint's own NatNet format. VRPN, provided by the University of North Carolina's Department of Computer Science, offers several advantages including a comprehensive C/C++ dynamic-linked library (DLL) and public domain license [23]. For illustration, NaturalPoint provides sample C++ code titled "VRPN-Listener" [24] that affords a simple capability for sampling to the TT port and output the received localization information to the screen.



Frame Information
Camera FPS 100.0
Display FPS 44.5
2D Markers 60
3D Markers 21
Trackables 5
2031
Position
X 632.815
Y 9.104
Z -876.039
Orientation
Yaw 88.071
Pitch 1.883
Roll -3.341

Figure 6: Screen Shot of NaturalPoint Tracking Tools in Use.

Control can be coded in a wide variety of software languages, each offering unique benefits and shortcomings. MATLAB offers significant advantages in terms of comprehensive image processing routines and robust matrix manipulation routines to support important functionality such as volume mapping. In implementation with the VRPN construct though, a significant challenge arises in that there are no MATLAB native commands or functions available to interpret the VRPN broadcast as the DLL is C/C++ based. Here we have realized a solution and its experimental implementation by creating a MATLAB-based MEX-function [25] that allows the robots' MATLAB-based control algorithms to access and call the C++ DLL native to VRPN and thereby provide a comprehensive surrogate localization function.

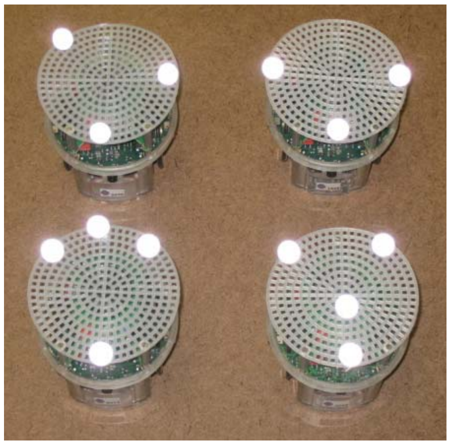


Figure 7: Optical Tracking Reflectors on E-puck Robots.

MEX-functions have an organic capability offered by MATLAB that allows a programmer to compile software routines written in C/C++ and linked to C/C++ DLLs resulting in a function callable from within MATLAB programs just as one would call a MATLAB built-in function. The specific C++ code used in this research is a task-specific modification of the Natural Point-provided "VRPN-Listener" code written in C++. The capability of the created MEX-function is twofold. First, given identification of the specific agent of interest, the MEX-function interrogates the VRPN broadcast port and obtains the most current localization information. Secondly, the MEX-function converts the provided localization information into the desired coordinate system and units. For the cited experiment, this consisted of converting the VRPN-provided orientation in the three-dimensional quaternion rotation space in radians to roll-pitch-yaw parameters in degrees.

UDP Communications for Data Sharing

Multi-agent systems can be characterized as having either distributed or centralized control with the distinction being the level to which each individual agent in the system governs its own motion. Fully distributed control is characterized by a complete absence of data or control-command exchange among the agents whereby each agent relies upon its own sensors to understand the environment, and its own processing capability to determine controlled actions. On the other end of the spectrum, centralized control is characterized by a single processor calculating the collective motion of the system and disseminating commands to each agent for the orchestration movement. As a result, distributed control architectures require robust on-agent processing capabilities, but minimal communications. In contrast, centralized control architectures are accomplished with a single robust processor (the central controller) and relatively minimal agent processing. However, a robust inter-agent communication capability is required to facilitate distribution of necessary control commands, sensor information, and other data such as state and localization information. As described control is a spectrum, one can imagine that there are varying levels of processing and communications required among the various control schemes ranging from fully distributed to fully autonomous.

The agents comprising the system in this research are constructed and defined as being fully distributed with each agent and its associated controlling computer fully responsible for calculating motion commands. However, there is a slight variation in that a separate "state" computer exists to act as a surrogate global positioning system (via the OptiTrack™ Tracking Tools system and software); and to collect, maintain, update, and distribute state information (position and orientation of each agent as well as occupancy mapping of the operating space). Because of this, a comprehensive inter-computer communications infrastructure and protocol was created for passing of data among the state computer and controller computers.

From a hardware perspective, the seven computers (one state computer plus the six controller computers) achieve full gigabit Ethernet inter-computer communications throughput capability by connection to a single DLink DGS-2208 eight-port gigabit Ethernet switch. That switch in turn, is connected to a Netgear Prosafe FVS336G gigabit Ethernet router. That router provides connectivity to the external internet service provider, but more importantly, is set up to statically assign internet-protocol addresses to each of the experiment's computers. With this architecture, each computer can be uniquely identified by its statically-assigned IP address facilitating unique computer programming.

The second aspect of providing a robust information exchange infrastructure is to select the underlying data protocol used. There are several protocols available that can be used in MATLAB and C programming including notably transmission control protocol/internet protocol (TCP/IP) and user datagram protocol (UDP). In fact, MATLAB's Instrument Control Toolbox provides a very comprehensive set of TCP/IP and UDP command sets. Each offers its own advantages and disadvantages. For example, TCP/IP offers a strong recovery feature that consists of error-checking that will initiate the resending of data packets if they are not accurately received. UDP offers greater throughput potential due to its use of minimal data fields for error-checking overhead. However, it can suffer some packet losses. Ultimately, UDP was utilized based on the construct of the validation experiment. In fact, MATLAB's TCP/IP protocol cannot be used to communicate between disparate MATLAB instantiations, i.e., communications between MATLAB running on computer "X" and a separate instantiation of MATLAB running on computer "Y." The concern here, however, is that while it offers the greatest throughput, UDP does not offer any organic error checking. This turned out not to be an issue in this research as error checking in the authored code itself was able to accommodate any transmission errors.

5. RESULTS

Simulation

Digital simulations showed that the potential-function path planning control architecture provides a robust and complete methodology to explore and map an initially unknown operating space. One representative experiment included a series of excursions in which the time allowed per mission was held constant. The number of agents comprising the MAS was varied from one to four. These variant MAS were then evaluated in terms of the resulting state of knowledge such as map coverage count or average occupancy probability. Increases in number of agents utilized in sequential experiments consistently showed a non-linear relationship between number of agents and perforamance criteria, for example, map coverage count (Figure 8). This likely is due to an inherent efficiency gained in terms of distance required to travel to unexplored spaces by having additional agents in the arena.

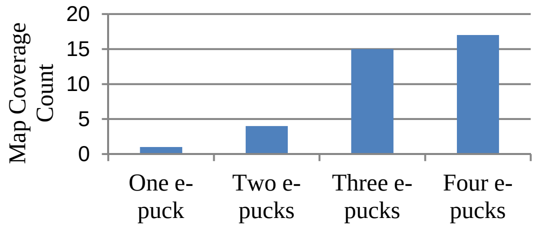


Figure 8: Map Coverage Count versus Number of Agents.

The physics of potential energy functions dictate that the force is inversely proportional to the square of the distance. In this research, the energy physics were replicated as such. However, Figure 9 shows that this tends to greatly magnify the influence of proximate cells and greatly diminish distant cells. As a result, calculated path planning was heavily weighted in a more local versus global sense. This clearly will influence the efficiency of the MAS' ability to exhaustively search the entire arena acknowledging that understanding the entirety of unexplored space and accommodating in the path planning scheme is likely more effective than merely considering the radius to unexplored space. Nonetheless, the shortcoming of this inefficiency is overwhelmed by the potential function's ability to accomplish exhaustive search. Ultimately, each cell will influence the path planning calculation insuring that each cell ultimately is explored. An enhancement to consider is to normalize cell charges to and gain better responsiveness over the entirety of the stage versus proximate cells. Effectively, this means breaking the laws of physics and making the cell charge vary as reduced exponents or even linearly as a function of distance. Clearly, this deviates from classic potential theory, but might nonetheless offer some interesting advantages in path planning efficiencies.

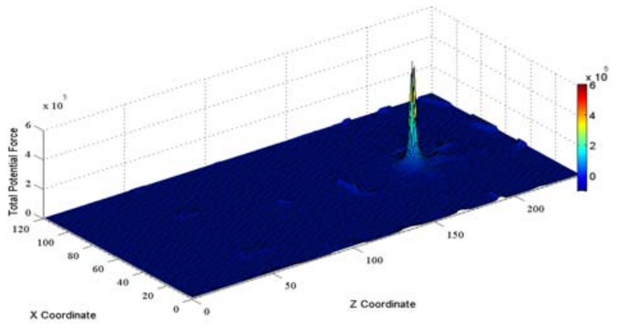


Figure 9: Total Potential Force Witnessed by an Individual Agent in the Arena.

Experimental Validation

The novel surrogate-localization and inter-agent data-communications solutions developed for experimentation infrastructure proved to be robust and highly conducive to support control algorithm cycle rates. Based on the data collection capability and mobility of the individual agents, governing control schemes strive for cycle times less than 200 milliseconds to insure exhibition of responsive behaviors. The surrogate localization function created only minimal processing time burden consuming only low-single-digit milliseconds and with near-real-time latencies. This proved extremely supportive of the control scheme with the agents being able to accurately represent the locations of themselves, other agents, targets, and obstacles - certainly well within the 1 cm^2 resolution of the state map. A significant enabler to this was an ability to calibrate the accuracy of the OptiTrack^{TM} system itself to sub-millimeter levels. Additionally, this function proved a robust capability to provide the variety of network-connected controllers immediate access to their own controlled agent, but also to the peer agents operating in the arena, all in near real time, and without having to burden the inter-agent communications architecture. The UDP-based data-communications function provided a reliable capability to pass large datasets of state information among the networked computers. A single state map consists of 28,800 cells (120 by 240) each represented by a 64 bit data element. Therefore, the single full mapping approaches 1.85 megabits of data. Over the course of an individual experiment, hundreds of state maps are transmitted to the agent controllers along with thousands of state map updates from the agent controllers. This means tens of thousands of packets navigating the network among the state and controller computers. Notwithstanding UDP's lack of an error correction capability, the data-communication solution herein resulted in zero packet loss realized across more than 300 individual experiments. Certainly, this was enabled by isolating the network on a single 8-port network switch; however, also enabled by careful consideration and definition of ports and buffers.

The resulting experiment infrastructure enabled a comprehensive series of individual vignettes and excursions that explored varying MAS parameters such as the number of imaging payloads, number of agents comprising the MAS, and relative agent and cell charge values. These were evaluated against mission performance criteria such as time to explore a selected percentage of the identified space or the number of positive detections collected by proximity sensing or imaging. With these parameter and metric sets, methodical experiments can be conducted sequentially to result in datasets that can be reduced and interpreted for qualitative and quantitative characterization of the relative and absolute performance of the various MAS configurations.

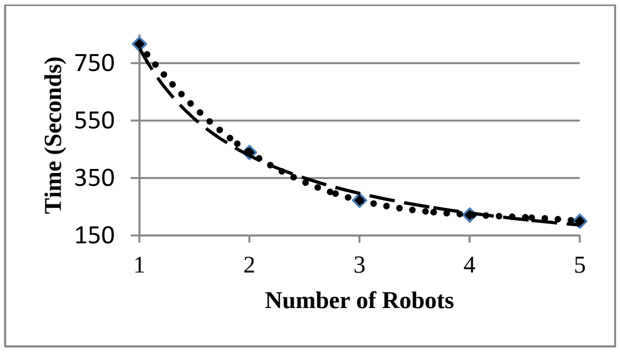


Figure 10: Time to Complete the Mission versus Number of Robots.

The most interesting conclusion drawn was that the relative performance of a MAS as a function of the number of agents comprising the MAS faded non-linearly (Figure 10). In these series of experiments, mission success was defined as exploring 75% of the arena's cells. Further, each agent was capable to use its on-board camera and associated image processing capability to detect and localize targets in the arena. Each agent could also use its proximity sensing for local collision avoidance and for obstacle detection and localization. Excursions were run whereby the number of agents comprising the MAS was varied from one to five with ten experiment runs per excursion. Mean values were taken across the ten runs to conclude the excursion results. In an attempt to curve fit the data (annotated by the diamonds), both polynomial and power types were evaluated. A third-order polynomial curve (dotted line) provided a better fit (R^2=1) within the bounds of one to five agents; however, did not extrapolate as values above six agents drove the time to completion to negative numbers. A power curve fit (dashed line) also resulted in an excellent fit (R^2=0.9898) and additionally provided better extrapolation beyond five robots with 6- through 8-agent MAS resulting in 158, 137, and 122 seconds, respectively. Additionally, this curve fit has an asymptote at 0 seconds as one would expect. Logically, this is the case whereby the entirety of the arena is covered by the concatenation of agents' sensors such that that entirety is exhaustively sensed in the very first control cycle’s data collection.

6. CONCLUSION

A solution is described that utilizes a novel application of a potential-force function that includes tuning coefficients to control mobile robots orchestrated as a distributed multi-agent system. Further, MAS and individual agent parameters were methodically explored to evaluate and characterize resulting overarching performance of the MAS. Experimentation began via a software simulation. Then, with the creation of novel localization and data-transfer capabilities, the experimentation migrated to a hardware implementation. Results provide valuable insight to the behavior of the various MAS configurations such that a priori mission planning can be optimized as a function of prioritized and weighted criteria and planning factors. Notably, there is a distinct tradeoff between MAS capability and consumption such that optimal configuration can indeed be selected a priori to conduct of missions.

7. ACKNOWLEDGEMENT

The research reported in this paper was supported by the U.S. Army Research Office under grant number W911NF-08-0106.

8. REFERENCES

[1] D. Garg and R. Young, “Coordinated Control of Cooperating Robotic Manipulators”, *International Journal of Systems Science*, Vol 21, Issue 11, November 1990 , pp. 2161 - 2176.

[2] R. Young, “The Coordination of Multiple Robotic Manipulators”, Master’s thesis, Duke University, 1987.

[3] L. Parker, “Multiple Mobile Robot Systems”, *Springer Handbook of Robotics*, 2008, pp. 921-941.

[4] W. Burgard, M. Moors, C. Stachniss, and F. Schneider, “Coordinated Multi-robot Exploration”, *IEEE Transactions on Robotics*, Vol. 21, No. 3, June 3, 2005, pp. 376-386.

[5] K. Baghaei, and A. Agah, “Task Allocation Methodologies for Multi-Robot Systems”, *Technical Report ITTC-FY2003-TR-20272-01*, November 2002.

[6] G. Fricke, G. Zhang, A. Caccavale, W. Li, and D. Garg, "An Intelligent Sensing Network of Distributed Swarming Agents for Perimeter Detection and Surveillance”, Paper No. DSCC 2010-4256, *Proceedings of the ASME 2010 Dynamic Systems and Control Conference*, September 12-15, 2010, Cambridge, MA.

[7] D. Atta, B. Subudhi, and M. Gupta, “Cohesive Motion Control Algorithm for Formation of Multiple Autonomous Agents,” *Journal of Robotics*, May 31, 2010, pp. 1-13.

[8] R. Williams and J. Wu, “Dynamic Obstacle Avoidance for an Omnidirectional Mobile Robot,” *Journal of Robotics*, September 14, 2010, pp. 1-14.

[9] L. Parker, “Guest Editorial, Special Issue on Heterogeneous Multi-robot Systems”, *Autonomous Robots*, Vol 8, Issue 3, June 1, 2000, pp. 207-208.

[10] L. Parker, “The Effect of Heterogeneity in Teams Of 100+ Mobile Robots”, *Multi-Robot Systems: From Swarms to Intelligent Automata, Proceedings of the 2003 International Workshop on Multi-robot Systems*, Vol II, March, 2003, pp. 205-218.

[11] M. Potter, L. Meeden, and A. Schultz, “Heterogeneity in the Coevolved Behaviors of Mobile Robots: The Emergence of Specialists”, *International Joint Conference on Artificial Intelligence*, Vol 17, Part 1, 2001, pp. 1337-1343.

[12] R. Grabowski, L. Navarro-Serment, C. Paredis, and P. Khosla, “Heterogeneous Teams of Modular Robots for Mapping and Exploration”, *Autonomous Robotics*, Vol 8, Number 3, June, 2000, pp. 293-308.

[13] J. Desai, J. Ostrowski, and V. Kumar, “Controlling Formations of Multiple Mobile Robots”, *Proceedings of the 1998 IEEE International Conference on Robotics & Automation*, Vol 4, May 1998, pp. 2864-2869.

[14] R. Alur, A. Das, J. Esposito, R. Fierro, G. Grudic, Y. Hur, V. Kumar, J. Ostrowski, G. Pappas, B. Southall, J. Spletzer, and C. Taylor, “A Framework and Architecture for Multirobot Coordination”, *Experimental Robotics VII*, 2001, pp. 303-322.

[15] J. Kim, D. Shim, S. Rashid, and S. Sastry, “Hierarchical System for Multiple-agent Scenarios”, *DTIC Report A794314*, January 1, 2002.

[16] B. M. Dias, B. Browning, M. Veloso, and A. Stentz, “Dynamic Heterogeneous Robot Teams Engaged in Adversarial Tasks”, *Technical Report CMU-RI-TR-05-14*, Robotics Institute, Carnegie Mellon University, April, 2005.

[17] A. Scott, and C. Yu, “Cooperative Multi-agent Mapping and Exploration in Webots”, *Proceedings of the 4th International Conference on Autonomous Robots and Agents, Canberra, Australia*, Feb 2009, pp.56-61.

[18] e-puck Educational Robot. Available:www. e-puck.org.

[19] X. Hou, C. Yu, and T. Summers, “A Virtual Framework of Robotic SWARM Testbed”, *Proceedings of the 21st Annual International Conference on Chinese Control and Decision Conference*, June 17-19, 2009, pp. 930-935.

[20] G. Fricke, D. Milutinovic, D. Garg, "Sensing and Estimation on a Modular Testbed for Swarm Robotics", *Proceedings of the 2nd Annual Dynamic Systems and Control Conference*, October 12-14, 2009, Hollywood, CA, paper TuBT4.4.

[21] Vicon website. http://www.vicon.com.

[22] NaturalPoint website. http://www.naturalpoint.com.

[23] Virtual Reality Peripheral Network website. http://www.cs.unc.edu/Research/vrpn/.

[24] Natural Point’s VRPN Streaming Sample Distribution Files. Available: http://media.naturalpoint.com/software/OptiTrack_files/VRPN-Tracking-v3.zip.

[25] MathWorks MATLAB “MEX-file Guide.” Available: http://www.mathworks.com/support/tech-notes/1600/1605.html#example5.