SWARM ROBOTICS: RESEARCH EXPERIENCE FOR HIGH SCHOOL STUDENTS

B. Samanta[1]*, Chiraag Nataraj[2], Sanjeev Reddy[3], Mark Woods[4], C. Nataraj[5]

Abstract: The paper reports an exploratory project to provide research experiences on swarm robotics to high school students. A group of three simple mobile robots (Lego NXT) was used to study 'search and rescue' operation. A bio-inspired global optimization technique called particle swarm optimization (PSO) was used as the main algorithm. Each robot was placed in pre-defined positions with a target position corresponding to a single target. The robots would search in spirals till the target was found by any one of the robots. Once the target was detected the robots would reach the target using the PSO algorithm. Results of initial exploratory efforts were encouraging. The students got a first-hand experience of implementing swarm robotics as a real-time engineering application.

Key words: Swarm robotics; swarm intelligence; Lego NXT; school students' research experience; Java programming; demonstration.

I. Introduction

Robotics is viewed as an emerging field that has potential to significantly impact the nature of engineering and science education at all levels, from K-12 to graduate school [1-7]. A recent development in robotics is swarm robotics [8]. The use of a large group (swarm) of small, simple and cheaper robots with limited local processing capability in place of a large, powerful and expensive robot is being envisioned in many hazardous, unknown and dynamic environments. The advantages of using swarms instead of a single centralized robot include enhanced capabilities in terms of wider dynamic coverage, fault tolerance, self organization and emergence. Application areas of robot swarms include: autonomous search and rescue operation, decentralized autonomous systems for protection and damage control, among others. For successful implementation, both hardware and software issues of such co-operative robots need proper investigations [8-13].

[1] Department of Mechanical Engineering, Villanova University, Villanova, PA, * Corresponding author. E-mail: biswanath.samanta@villanova.edu
[2] Conestoga High School, Berwyn, PA
[3] Radnor High School, Radnor, PA
[4] Department of Mechanical Engineering, Villanova University, Villanova, PA
[5] Department of Mechanical Engineering, Villanova University, Villanova, PA

Villanova University has a structure of outreach to involve communities which are under-represented in Science and Engineering. Two main projects are the V.E.S.T.E.D. Academy and BEST. The V.E.S.T.E.D. Academy in its fourth year at Villanova University aims to promote academic achievement in mathematics, science, technology, and engineering for at-risk middle and high school students. BEST is a non-profit, volunteer-based organization whose mission is to inspire students to pursue careers in engineering, science, and technology through participation in a sports-like, science and engineering-based robotics competition. Villanova is also a participant for GEAR UP, a teacher training program to increase teacher and student understanding of math subjects and to help them use robotics to accomplish their math goals.

To provide educational and research experiences to high school students, an exploratory (pilot) project on swarm robotics was initiated in Summer 2008 in the Department of Mechanical Engineering at Villanova University, Villanova, PA with a team of two high school students and a senior Villanova Mechanical Engineering undergraduate under the supervision of two Faculty members. The plan was to test the feasibility with a small group. The program would later be extended to a larger group in line with the outreach programs of the University and with potential funding support from external agencies. In this paper, the activities and results related to this pilot project are reported. A group of three simple mobile robots (Lego NXT) was used to study 'search and rescue' operation in the context of swarm robotics. A bio-inspired global optimization technique called particle swarm optimization (PSO) was used as the main algorithm. PSO was proposed by Kennedy and Eberhart [14] as a population based stochastic optimization technique inspired by the social behavior of bird flocking. PSO is a computationally simple algorithm based on group (swarm) behavior. The algorithm searches for an optimal value by sharing cognitive and social information among the individuals (particles). PSO has many advantages over evolutionary computation techniques like genetic algorithms in terms of simpler implementation, faster convergence rate and fewer parameters to adjust [14, 15]. The popularity of PSO is growing with applications in diverse fields of engineering, biomedical and social sciences, among others [16-18].

The rest of the paper is organized as follows. The basic aspects of the lego NXT platform are discussed briefly in section II. In section III, basics of PSO are presented. The implementation issues and results are presented in section IV. Conclusions and future steps are summarized in section V.

II. LEGO NXT Robot platform

Hardware

Lego Mindstorms NXT robotics kit was used as the robotic platform for the project. Four kits were procured, at any time three were used and the fourth was kept as a stand-by. The main component of a NXT kit is the NXT intelligent brick. There are two microcontrollers embedded inside the brick. One brick can take inputs from four sensors and control up to three motors at once with four input ports for sensors and three output ports for motors. The NXT kit comes with four sensors, namely, light, sound, touch and ultrasonic sensors. In this project only two of these sensors, touch and ultrasonic sensors were used. Two of the motors were used for driving the robot. The third motor was used to rotate the ultrasonic sensor through a geared mechanism. Figure 1 shows the photograph of two assembled NXT mobile robots. For communication with a PC laptop, the wireless Bluetooth connection was used. There is a speaker for sound and a LCD display for the status of NXT.

NXT Brick

Figure 2 shows the photograph of the NXT intelligent brick. The NXT brick contains an Atmel 32-bit ARM7 processor running at 48 MHz with 64 KB of RAM and 256 KB bytes flash memory. The other processor is an Atmel 8-bit AVR running at 8 MHz with 512 KB RAM and 4KB of flash memory. The AVR controls the peripherals while ARM7 has the main processing power. The 100 x 64 LCD display is used to navigate through the NXT menu and shows the status of the brick. The NXT is connected to its peripherals through 6-wire cable digital platform.

Servo Motors

Figure 3 shows the external and internal views of a servo motor. The gears help reduce the wheel speed and increase its torque. Each motor has a built-in tachometer to keep track of the motor rotation. Each motor is also equipped with a servo loop for velocity and position control.

Touch Sensor

Figure 4 shows a photograph of the touch sensor used in the NXT kit. The touch sensor senses when it is pressed and when it is released. This signals to the robot that it has contacted another object. The sensor was activated using a link in front of it. When the robot hits an object with the link activating the touch sensor, the NXT was programmed to stop moving forward.

Ultrasonic Sensor

Figure 5 shows photograph of the ultrasonic sensor mounted on a geared platform driven by one motor. The ultrasonic sensor was used for the robot to avoid obstacles and measure distance. The rotating platform was used programmed so that the ultrasonic sensor can cover the range of -90° to 90° in front of the robot. The sensor measures a distance upto 250 cm. It is sensitive the shape and distance of the object.

Bluetooth

The communication between any NXT robot and the PC laptop (host) was implemented using a D-Link DBT-120, wireless Bluetooth 2.0 USB Adapter. It is compatible with Windows 2000/XP, follows the IEEE 802.15.1 standard, uses USB 2.0 interface, and sends signals at 2.1Mb/s. [19]. The Bluetooth USB Adapter supports the Microsoft Service Pack 2 Bluetooth stack.

Java Software Platform for NXT

The NXT needs to have a firmware installed in order to be usable. The default firmware and software, NXT-G, are adequate for normal users. However, for greater flexibility, an alternate firmware and software system for the NXT, called leJOS NXJ, was adopted for the project. It interfaces with the NXT hardware and allows users to program in Java. The PC laptop was using leJOS JVM (Java Virtual Machine) under Linux operating system. An open source integrated development environment (IDE) suitable for leJOS NXJ, called Eclipse, was used in this project.

III. Particle Swarm Optimization (PSO)

A. Standard Particle Swarm Optimization (PSO)

In this section, a brief introduction to PSO algorithm is presented, for details text [15] can be referred to. Recent overviews of PSO and its variants are presented in [16-18]. For a problem with *n*-variables, each possible solution can be thought of as a *particle* with a position vector of dimension *n*. The population of *m* such individuals (particles) can be grouped as the *swarm*. Let x_{ij} and v_{ij} represent respectively the current position and the velocity of *i*th particle ($i=1,m$) in the *j*th direction ($j=1, n$). The fitness of a particle is assessed by calculating the value of the target or objective function for the current position of the particle. If the value of the objective function for the current position of the particle is better than its previous best value then the current position is designated as the new *best individual* (*personal*) location *pbest*, p_{bij}. The best current positions of all particles are compared with the historical best position of the whole swarm (*global* or *neighborhood*) *gbest*, p_{bgj}, in terms of the fitness function and the global best position is accordingly updated if any of the particle individual best (*pbest*, p_{bij}) is better than the previous global best (*gbest*, p_{bgj}). The current position and the velocity decide the trajectory of the particle. The velocity of the particle is influenced by three components, namely, inertial, cognitive and social. The inertial component controls the behavior of the particle in the current direction. The cognitive and the social components represent the particle's memory of its personal best position (*pbest*) and the global best position (*gbest*). The velocity and the position of the particle are updated for the next iteration step ($k+1$) from its values at current step k as follows:

$$v_{ij}(k+1) = v_{ij}(k) + c_1 U(0,1)(p_{bij}(k) - x_{ij}(k)) + c_2 U(0,1)(p_{bgj}(k) - x_{ij}(k)), \quad (1)$$

$$x_{ij}(k+1) = x_{ij}(k) + v_{ij}(k+1), \quad (2)$$

where r_1 and r_2 represent uniformly distributed random numbers in the range of (0,1). These random numbers present the stochastic nature of the search algorithm. The constants c_1 and c_2 define the magnitudes of the influences on the particle velocity in the direction of the individual and the global optima. N represents the maximum number of iterations (epoch). In many early applications, good results were obtained when the inertia term ω was decreased from 0.9 to 0.4 linearly enhancing the exploration at the beginning and exploitation towards the end of the solution process. In this work, c_1=2.0, c_2= 2.0 were used.

B. Distributed Particle Swarm Optimization (DPSO)

In the present work, each robot was assumed to be a particle in the swarm. Each robot was equipped with touch and ultrasonic sensors in addition to the two servo motors for keeping track of the robots position. Each robot would measure its position, update its velocity and position, update its personal best value (*pbest*) and its personal best position. Each robot would send to the PC the values of its current position, and its personal best. The PC will receive the *pbest* values of all robots and transmit to them the global best (*gbest*) to be used by each robot for updating its velocity and position for the next move. This arrangement would allow each robot to make its calculations in parallel locally and sharing the information of *pbest* and *gbest* with the host PC minimizing the communication overhead and the delay.

C. DPSO Based

The aim of the present approach is to select the robot positions such that an objective function representing the sum of squared error of the robot position relative to the target is minimized.

$$J = \sum_{l=1}^{N}((x_d - x_l)^2 + (y_d - y_l)^2) \quad (3)$$

where *l* is the robot index, N represents the total number of robots, (x_d, y_d) is the desired position and (x_l, y_l) represents the current position of the robot '*l*'. In the present work, DPSO was used from a user-given range for each particle position [-400, 400] mm to cover the entire search space.

IV. Implementation Results and Discussions

The project objectives were (i) to get familiarized with the Lego Mindstorms NXT hardware, (ii) to interface the sensors and actuators (motors), (iii) to program the NXT brick using leJOS NXJ, (iv) to establish communication between NXT robots and a PC laptop using Bluetooth, (iv) to program each NXT for uniform spiral coverage using Archimedes' curve, and (iv) to implement the distributed particle swarm optimization (DPSO) algorithm for 'search and rescue' operation using a swarm of NXT mobile robots. Four NXT robot kits were procured for the project. Each robot was assembled using two servo motors for driving, the other servo motor for rotating the base of the ultrasonic sensor for the total front view of the robot and the touch sensor with a link in the front for avoiding obstacles. At any time three of these robots were used and the fourth one was kept as a stand-by.

Two NXT robots were powered up with rechargeable battery packs and the other two were AA batteries. LeJOS NXJ codes were written for acquiring data from the sensors and motors and giving signals to the motors. The sensors were calibrated and the motors were tested. The program was written using classes for the corresponding functionalities of the sensors and the motors. The communication between the NXT robots and the PC laptop was implemented using leJOS JVM with the laptop running under Linux operating system and D-Link DBT-120 Bluetooth wireless adapter. The Windows operating system was giving some problem with the setup. The Linux was preferred for the current application as the wireless communication was quite smooth.

The coverage and target reaching was implemented in a decoupled manner. The uniform spiral coverage using Archimedes' curve was implemented in leJOS NXJ. The pseudocode for DPSO, shown in Table 1, was implemented in leJOS NXJ. The codes were developed in Eclipse IDE and the firmware was uploaded to each NXT. The NXT robots were placed in three known positions in the lab for this project. In the first phase, the robots would search for an object in covering the respective local area in spirals. Figure 7 shows a typical coverage plot for one NXT. In the next phase, the NXT would be required to go the desired (target) position using DPSO. Figure 8 shows the paths of the robots in a typical run. It shows that all robots reach quite close to the target. The separation between the robots shows the effect of touch sensors.

V. Conclusions

The paper presents an exploratory project on swarm robotics involving high school students. The project provides educational and research experiences to the students covering a wide range of areas like sensing and actuation, control, swarm intelligence, hardware and software implementation. The students got a first-hand experience of implementing software codes in real-time engineering applications. However, some hardware and software issues would need closer and more thorough investigations for further enhancements. These are currently under consideration for the next phase. For example, the seamless implementation of coverage, search and rescue would be considered using a multimodal approach where the operation would switch from one mode (coverage) to the other (search and rescue). The involvement of larger groups of students with potential external funding is also envisioned in the future.

Acknowledgment

The work was carried out with partial support from Naval Sea Systems Command (NAVSEA) grant N00024-07-C-4212 (Monitors: Marc Steinberg, R. Wagner and John Metzer). This support is gratefully acknowledged.

References:

1. Matari´c, M. J. *Robotics education for all ages*. *Proceedings of AAAI Spring Symposium on Accessible, Hands-on AI and Robotics Education*, Palo Alto, CA, Mar 22-24, 2004
2. Korpela, C. M. and Adams, W. J. *Robotics in multidiscipline multicultural projects*. *Proceedings of the 2007 Middle Atlantic Section Fall Conference of the American Society for Engineering Education.*
3. You, Y. *A project-oriented approach in teaching robotics application engineering. Proceedings of the 2009 American Society for Engineering Education Conference.*
4. Kuc R. and Kuc, A. *Teaching introductory autonomous robotics with JavaScript simulations and actual robots*. *IEEE Transactions on Education*. Vol. 47, 2004, pp. 74-82.
5. Greenwald, L. and Kopena, *J. Mobile robot labs*. *IEEE Robotics and Automation Magazine*, 2003, pp. 25-32.
6. Galvan S., Botturi, D., Castellani, A., and Fiorini, P. *Innovative robotics teaching using lego sets*. *Proc. IEEE International Conf. on Robotics and Automation*, 2006, pp. 721-726.
7. Bagnall, B. *Maximum LEGO NXT Building Robots with Java Brains*. Variant Press. 2007.
8. Sahin E. *Swarm Robotics: from sources of inspiration to domains of applications. Swarm Robotics WS* 2004, Sahin, E. and Spears, W.M. (Eds.), *LNCS* 3342, 2005, pp. 10-20.
9. Garcia, E., Antonia, M., De Santos, P. G., and Armada, M. *The evolution of robotics research from industrial robotics to field and service robotics. IEEE Robotics and Automation Magazine*, 2007, pp. 90-103.
10. Ercan, M. F., Partawijya L., and Fung, Y-F. *Collective search and exploration with a robot swarm*. *Proc. IEEE International Conf. on Robotics and Automation*, 2006.
11. Cheng. K. and Dasgupta, P. *Dynamic area coverage using faulty multi-agent swarms. Proc. IEEE/WIC/ACM International Conf. on Intelligent Agent Technology*, 2007, pp.17-23.
12. Hereford, J. *A distributed particle swarm optimization algorithm for swarm robotic applications. Proc. IEEE Congress on Evolutionary Computation*, 2006, pp. 1676-1685.
13. Pugh, J. and Martinoli, A. *Distributed adaptation in multi-robot search using particle swarm optimization*. *SAB* 2008, M. Asada (Eds.), LNAI 5040, pp. 393-402.
14. Kennedy, J. and Eberhart R. C. *Particle swarm optimization*. *Proc. IEEE Intl. Conf. on Neural Networks IV*, Piscataway, NJ: IEEE Service Center, 1995, 1942–1948.
15. Kennedy, J., Eberhart, R.C., and Shi, Y. *Swarm Intelligence*, Morgan Kaufmann Publishers, San Francisco, CA, 2001.
16. Poli, R., Kennedy, J., and Blackwell, T. *Particle swarm optimization an overview*. *Swarm Intelligence*, vol. 1, 2007, pp. 33-57.
17. Samanta, B. and Nataraj, C. *Use of particle swarm optimization for detection of machine condition. Engineering Applications of Artificial Intelligence*, vol.22, 2009, pp. 308-316.
18. Samanta, B. and Nataraj, C. *Prognostics of machine condition using soft computing. Robotics and Computer-Integrated Manufacturing*, vol. 24, 2008, pp. 816-823.
19. ftp://ftp10.dlink.com/pdfs/products/DBT-120/DBT-120_ds.pdf.

Table 1: The pseudo code for distributed PSO (DPSO) algorithm

```
Initialize the swarm
While (mission on)
     get sensor readings
  If (no object is found) Then
       move in spiral
   Else
     evaluate the target position
     report the target position to the host
     set dPSO on
         While (dPSO on)
           get sensor readings
           update local best
           update global best;
           report to host the current position and
           get global best
           move to the next best position
           avoid obstacle(s), if present
                 If (target reached)  set dPSO off
         End /end while (dPSO off)
      Endif /endif/
   If (mission completed) set mission off
End /end while/completion of mission/
```

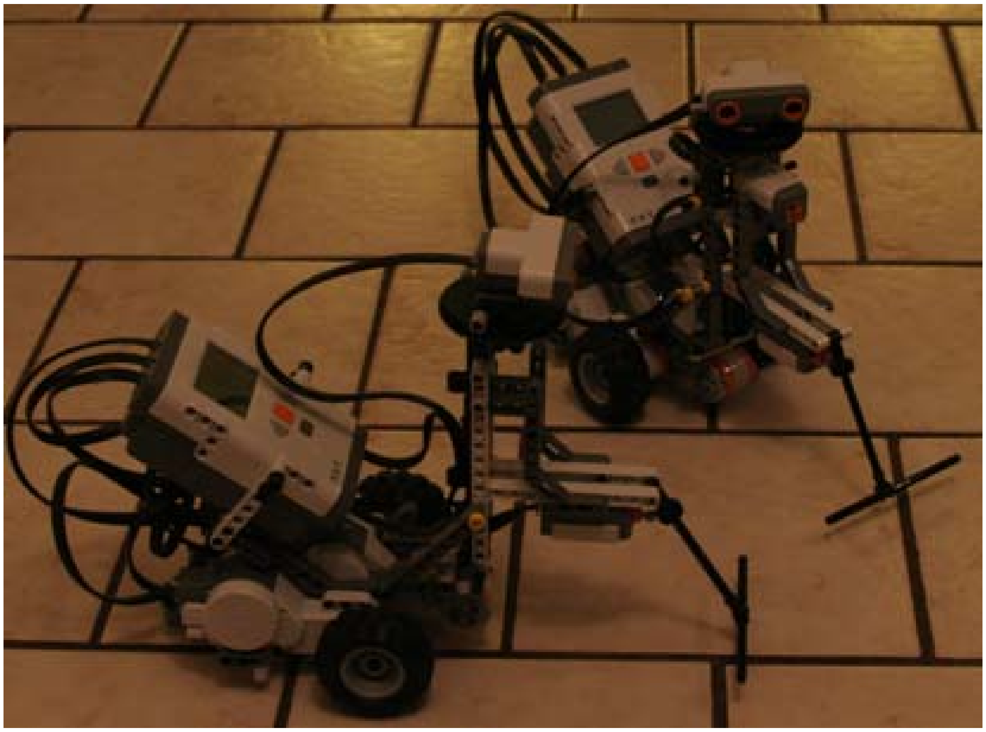
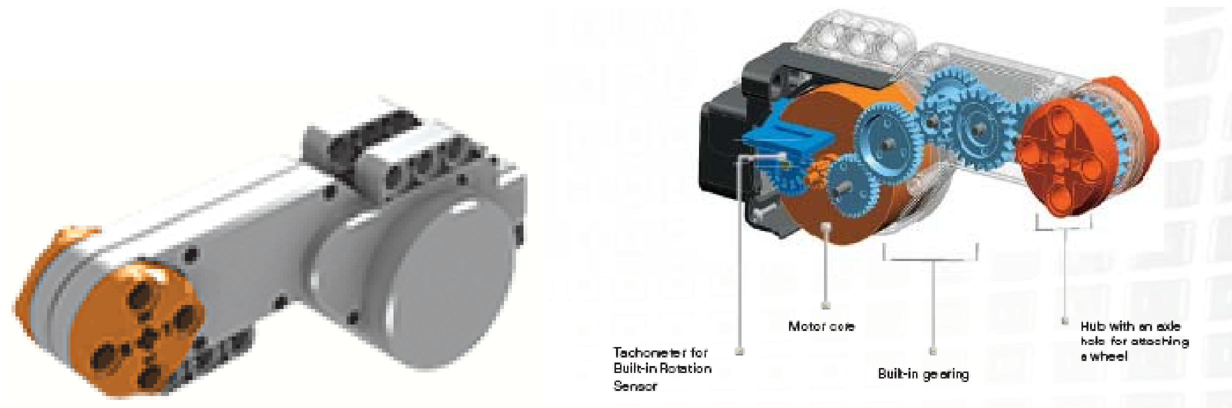


Fig. 1 Two assembled Lego NXT mobile robots

Fig. 2 NXT Intelligent brick



(a) (b)

Fig.3 Views of a servo motor (a) External view, (b) Internal view [Lego NXT website]

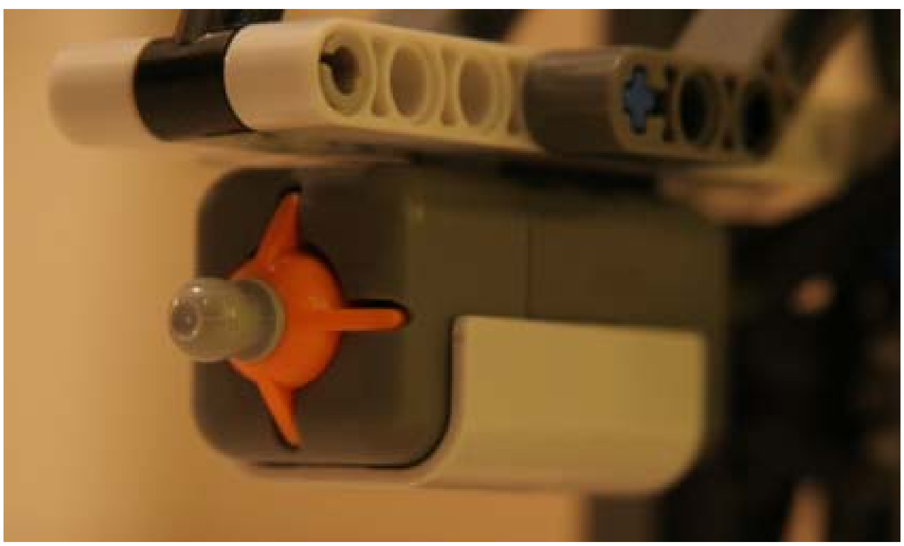


Fig. 4 Touch sensor of NXT

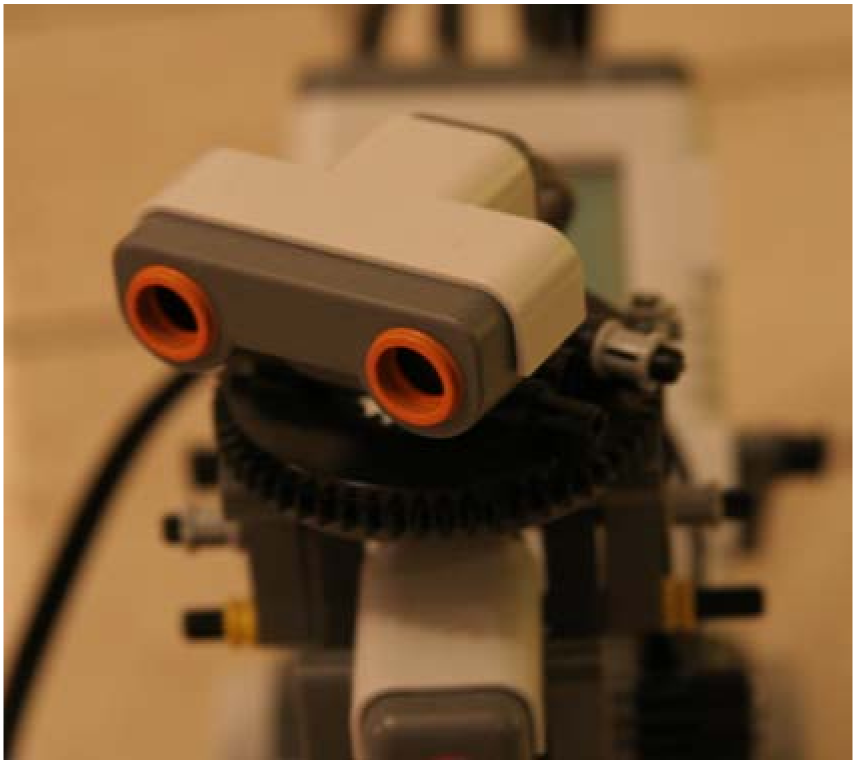


Fig. 5 Ultrasonic sensor of NXT

Fig. 6 D-Link DBT-120 Bluetooth adapter []

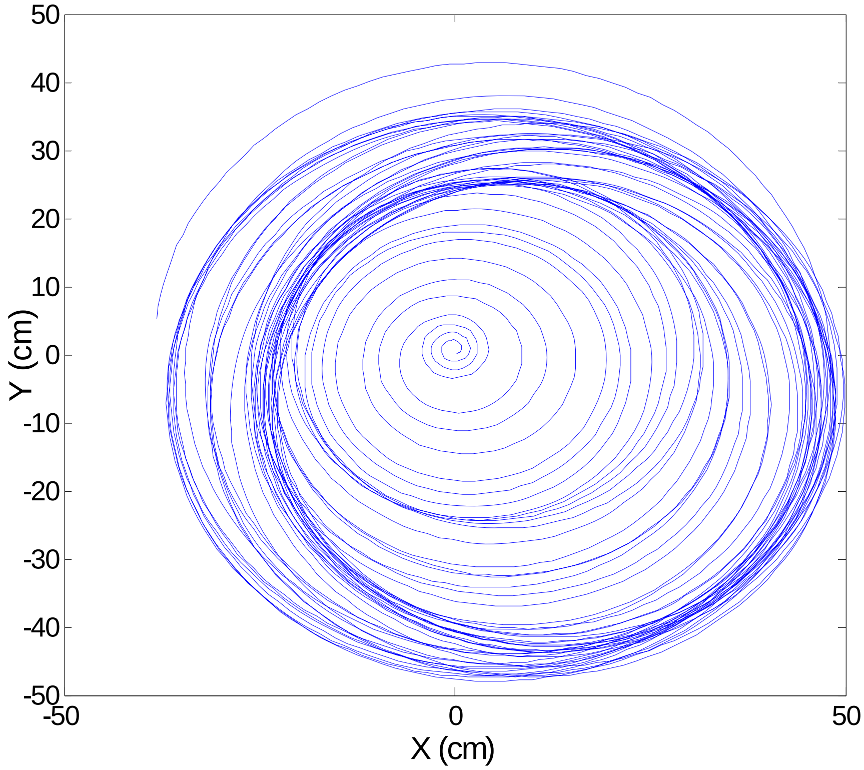


Fig. 7. Area coverage (spiral)

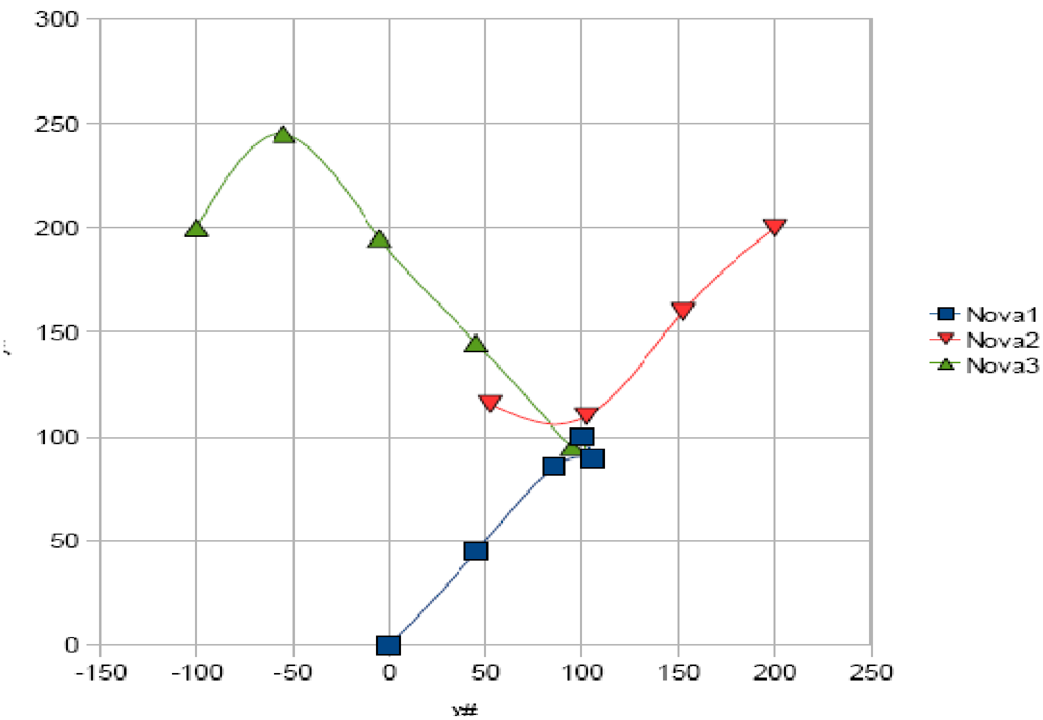


Fig. 8. Paths of robot swarm