Workshop/Summer School on Evolutionary Computing
Lecture Series by Pioneers, August 18-22, 2008, Londonderry

Asynchronous Particle Swarm Optimization Based Search with a Multi-Robot System

Salih Burak Akat, Veysel Gazi, and Lino Marques

***Abstract*— In this article we consider a version of Particle Swarm Optimization (PSO) algorithm that is appropriate for the search tasks of multi-agent systems, which consist of small robots with limited sensing capability. The proposed method adopts asynchronous mechanism for information exchange and position (way point) updates of the agents. Moreover, at each (information exchange) step the agents communicate with only a possibly different subset of the other agents leading to a dynamic neighborhood topology. Simulations using the Player/Stage robotic simulator are performed on an experimentally collected realistic data of ethanol gas concentration to verify the effectiveness of the algorithm.**

I. Introduction

Searching for one or more targets in an unknown and possibly dangerous (for humans) environment is a task that can be performed by deploying multiple autonomous robots. Equipping the robots (referred to as agents in this article) with the necessary sensors and developing efficient navigation and cooperative search algorithms can lead to improving the performance of the system in terms of more effective exploration/coverage and decreasing the time of search.

There have been works on investigating search methods inspired from Particle Swarm Optimization for multi-agent systems. Doctor and his colleagues studied a multi-robot search application involving one or more than one targets [1]. Their study is focused on finding optimal parameters for the PSO algorithm, inertia weight parameter (w) and upper bounds of learning coefficients ($\bar{\varphi}_1$ and $\bar{\varphi}_2$), in order to perform a target search task efficiently. They use a 2-level hierarchy in which the PSO in the inner level solves the problem of finding the locations of target/targets, while the PSO in the outer level determines the optimal set of parameters for the inner level PSO. Hereford [2], [3] considered a distributed PSO for robot search application which is somehow similar to the algorithm we consider here. His emphasis was on simplicity and decreasing the communication burden in the system and to achieve scalability of the algorithm to large number of robots. Pugh and Martinoli [4] worked on adapting Particle Swarm Optimization algorithm to multi-agent search applications. They considered the cases in which: (i) agents know their global positions and; (ii) agents rely on their local knowledge. Since the agents arrive to their respective way-points at different times, those agents which have arrived earlier than the others wait and updates are performed after all the agents arrive to their respective positions. Marques and his colleagues [5] presented a PSO inspired search method in order to detect odor sources across large search spaces. They compared the PSO inspired search method with other strategies and observed that the PSO inspired search method is more successful in their setting with unstable environment with high turbulence, than the other search methods.

In [6], [7] a framework for PSO with dynamic neighborhood topology and distributed asynchronous operation with possible time delays was developed. Inspired by the works in [6], [7], in this article we consider a system consisting of multiple robots deployed in a search task using Particle Swarm Optimization based decision making process and position updates. The robots are allowed to operate asynchronously and to exchange information using dynamic neighborhood topology. We test the effectiveness of the method via simulations using the Player/Stage realistic robotic simulator on a realistic environment of experimentally collected ethanol gas concentration data. To the best of our knowledge PSO inspired robotic search in the manner we consider here (dynamic neighborhood and asynchronous operation) has not been considered so far in the literature.

II. Problem Formulation

In this article we consider a system consisting of N mobile robots (agents) moving in $\mathbb{R}^2$ with continuous-time unicycle. The robots are required to perform a search in an unknown environment. Each point in the environment is assumed to have a particular potential value which, in this article represents experimentally collected ethanol gas concentration. However, in general, depending on the problem under consideration, it can represent other entities such as different type of odor, chemical plume or smoke concentration, temperature or light intensity etc. We call this potential the resource profile and use PSO based optimization strategy to plan the motions of the robots with the objective to locate the extremum (minimum or maximum) points of the resource profile. Again, depending on the application, these points can represent the source of odor, smoke/fire, heat or light. We assume that the robot is equipped with the necessary sensors in order to be able to measure the value of the environmental resource profile.

Salih Burak Akat and Veysel Gazi are with TOBB University of Economics and Technology, Dept. Electrical and Electronics Engineering, Söğütözü Caddesi, No: 43, 06560 Ankara, Turkey (sbakat@etu.edu.tr,vgazi@etu.edu.tr). Their work was supported in part by TÜBİTAK (The Scientific and Technological Research Council of Turkey) under grant No. 106E122 and by the European Commission under the GUARDIANS project (FP6 contract No. 045269).

Lino Marques is with University of Coimbra, Dept. Electrical and Computer Engineering, Institute of Systems and Robotics, 3030-290 Coimbra, Portugal (lino@isr.uc.pt). His work was supported by the European Commission under the GUARDIANS project (FP6 contract No. 045269).

Let us denote with $p_i(t) = [x_i(t), y_i(t)]$ the position of robot i at time t in cartesian coordinates. From the perspective of PSO inspired search robot i constitutes particle i. Given the robot is at its k^{th} way point $p_i(t_k) = [x_i(t_k), y_i(t_k)] \in \mathbb{R}^2$ at time t_k (iteration k for the robot/particle), its (desired) next way point $p_i(t_{k+1})$ is calculated using the PSO algorithm. In other words, PSO is used for higher level path planning for determining the way points that the robot should visit. In order to move the robot from the k^{th} way point $p_i(t_k)$ to the $(k+1)^{th}$ way point $p_i(t_{k+1})$ in the interval $t \in [t_k, t_{k+1})$, we use artificial potential functions for low level control. In particular we require the robot to move along the vector $\bar{p}(t_k) = p_i(t_k) - p_i(t_{k+1})$ to reach $p_i(t_{k+1})$ (with collision avoidance). This path could be different from a straight path. With this objective, we use a quadratic attractive potential function and require the robot to move along its negative gradient. Similarly, in order to avoid collisions between robots we use a repulsive potential which is activated when the distance between two robots becomes less than a predefined constant value d.

III. Asynchronous PSO

There are various possible implementations of the Particle Swarm Optimization method. Some of them suffer from the so-called explosion problem and need to employ a bound on the particles velocities. In this article we use the PSO version that uses a "constriction coefficient" proposed by Clerk and Kennedy in [8] in which at the k^{th} iteration, which corresponds to time t_k for the robot, the update for the way points for particle (robot) i, $i = 1, \ldots, N$, is given by

$$\begin{aligned} v_i(t_{k+1}) &= \chi\Big[v_i(t_k) + \varphi_1^i(t_k)\Big(b_i(t_k) - p_i(t_k)\Big) \\ &\qquad + \varphi_2^i(t_k)\Big(g_i(t_k) - p_i(t_k)\Big)\Big] \\ p_i(t_{k+1}) &= p_i(t_k) + v_i(t_{k+1}) \end{aligned} \tag{1}$$

Here $p_i(t_k) \in \mathbb{R}^2$ represents the position (way point) of the i'th particle at time t_k (the estimation of this particle about the minimum/maximum point of the function being optimized at time t_k), $b_i(t_k) \in \mathbb{R}^2$ represents the best position of the i'th particle until time t_k, $g_i(t_k) \in \mathbb{R}^2$ represents the best position of the neighborhood of the i'th particle until time t_k. The value $p_i(t_{k+1}) \in \mathbb{R}^2$ which is calculated at the end of the iteration is the next (desired) way point to which the robot should move. The learning coefficients $\varphi_1^i(t_k) \in [0, \bar{\varphi}_1]^2$ and $\varphi_2^i(t_k) \in [0, \bar{\varphi}_2]^2$ are two dimensional uniform random vectors. At each iteration these random vectors respectively determine the relative significance/weight of the cognitive and social components in the iteration. The constant parameter $\chi > 0$ is the constriction parameter that prevents the explosion behavior, i.e., particles having high velocity values leading to their scattering in the search space. For efficient performance and prevention of the explosion behavior in (1) we choose the components of the $\varphi_1^i(t_k)$ and $\varphi_2^i(t_k)$ learning coefficient vectors as

$$\varphi_{1j}^i(t_k), \varphi_{2j}^i(t_k) \in [0, 2.05], j = 1, 2; i = 1, \ldots, N, \tag{2}$$

The constriction parameter $\chi > 0$ can be calculated using the relation (refer to [8])

$$\chi = \begin{cases} \frac{2\kappa}{\varphi - 2 + \sqrt{\varphi^2 - 4\varphi}}, & \text{if } \varphi > 4, \\ \kappa, & \text{else.} \end{cases} \tag{3}$$

Here $\varphi = \bar{\varphi}_1 + \bar{\varphi}_2$ and $\kappa \in [0, 1]$. Considering $\bar{\varphi}_1 = \bar{\varphi}_2 = 2.05$ and $\kappa = 1$, the constriction parameter is calculated as 0.7298 for this study.

In its basic form, the iteration in (1) is performed synchronously after all the particles have performed their function evaluations and exchanged information. Although even in its basic form the method seems to be appropriate enough to robot search applications, we propose some modifications on it to improve its performance and to overcome some shortcomings that might arise. These modifications are in the line of the works in [6], [7] and include totally asynchronous implementation (much different from the ones considered in the literature) and dynamic information exchange topology between the robots (i.e., particles) in the system.

First of all note that multi-robot systems are naturally distributed and decentralized decision making and operation are desired properties of them for higher levels of robustness and flexibility. Moreover, due to the decentralized/distributed nature, multi-robot systems operate in an asynchronous manner. Furthermore, usually the sensing and communication capabilities of the robots are limited, which results in time dependent interactions (since only the robots which are within each others sensing or communication range can interact). Therefore, direct implementation (without modification) of the PSO algorithm for search in multi-robot systems may result in unsatisfactory performance. This is because first of all the robots cannot instantaneously jump to their next way points and it may take different amounts of time for the robots to reach their respective way points. For this reason, the robots which arrive earlier than the others have to wait for all the robots to arrive to their respective way points before exchanging information with the objective to update the global best of the neighborhood ($gbest$) and to move to the next iteration of the PSO algorithm. Moreover, for large search areas the respective way points of the robots may be far away from each other and the distance between robots may become larger than the communication range. This may lead to the fact that some of the robots may not be able to communicate with each other and therefore a system operating with standard PSO may stall since in order to move to the next iteration the robots need to obtain information from all its neighbors. The situation may become even worse in presence of temporary or permanent communication or agent failures. To overcome these shortcomings, motivated by the works on asynchronous PSO and PSO with dynamic neighborhood in [6], [7], we modify the PSO algorithm as described in the pseudocode given in Table I.

As one can see from the algorithm in Table I, while a robot is moving towards its respective way-point in the

TABLE I

PSEUDOCODE OF THE ALGORITHM

```
Initialize of pbest and gbest (and all other variables)
Calculate the first way point using Equation (1)
while (Target not found and iterations have not expired) do
   while (Agent has not arrived to its way point) do
      Move towards the desired way point
      Update pbest
      if (Information received from other agents) then
         Update pbest_other
      end if
   end while
   Broadcast own pbest
   if (pbest_other > gbest or pbest > gbest) then
      Update gbest
   else
      Use previous gbest
   end if
   Calculate a new way point using Equation (1)
end while
```

search area, it continuously listens for information from other robots (since if other robots arrive to their respective way-points earlier they broadcast their best fitness values achieved until that time) and later uses this information for updating the global best $gbest$. After arriving at the desired way-point, the robot broadcasts its best fitness value achieved until arriving to the desired way-point $pbest$. Then using the information gathered by itself and received from others, it performs an update of its next desired position (way-point) based on its current velocity, its best position, the global best position (which is extracted from the information obtained from the other robots) - see equation (1). If there are no other robots which have arrived at their respective way-points earlier than the robot under consideration (meaning that it has not received any information from the other robots yet), the robot continues its update based on its old information (that it had obtained from the other robots in the past). In other words, it does update the global best based only the information that it has sensed during its motion and continues its operation. Since the robots may arrive at their respective way-points at different time instants, the set of robots that the robot receives information from at each iteration of the PSO algorithm may change leading to a time-varying (dynamic) neighborhood topology and the robots perform asynchronous way-point updates. In the basic form of the PSO algorithm the neighborhoods of the particles are static, i.e., particle neighborhoods remain stationary throughout all iterations of the algorithm.

IV. SIMULATION RESULTS

The Asynchronous Particle Swarm Optimization based search method discussed in the preceding sections is tested using the Player/Stage realistic robot simulator. Player provides an interface to the robot's sensors and actuators over an IP network. Client program send commands to Player over TCP sockets and reads data from sensors. Stage provides a simulation environment in order to test the developed algorithm. The simulation environment provides movement of mobile robots in a two dimensional environment and various sensor models.

Simulations are performed in an obstacle free environment using experimentally collected gas concentration data. The data was gathered inside an enclosed environment with 3-by-4 meters area and 0.5 meters height. This environment was weakly ventilated through an opening in one corner and a system composed by four 12 centimetres diameter fans able to generate a flow ranging from 0 to 1500 lpm each in the other corner. There were three gas sources in the set-up placed on the ceiling of the environment in the locations $(2.25, 0.625)$, $(0.5, 2)$, and $(2.25, 3.45)$. These gas sources were obtained passing a controlled airflow through ethanol bubblers. The flow was controlled by individual SMC proportional valves. The ethanol concentration inside the environment was acquired with a sensor network composed by twelve Figaro TGS2600 gas sensors distributed uniformly into the enclosed environment. These sensors can detect ethanol vapour starting from few parts per million. Each gas sensor was mounted on a small PCB board with the necessary signal conditioning circuit. The output from these sensors was acquired by two Microchip PIC18F4431 micro-controllers interfacing to a PC through an RS485 shared bus. The continuous distribution of concentrations was estimated with a kriging estimator.

Given the experimentally collected data, the objective is to determine the areas of high gas concentration. The first plot on Figure 1 shows the plot of the experimentally collected concentration (the environmental gas profile) used in this study. This profile is the instantaneous odour concentration at a given time instant in the 3-by-4 bounded area mentioned above. For the simulations 6 Pioneer2dx robots are used.[1]

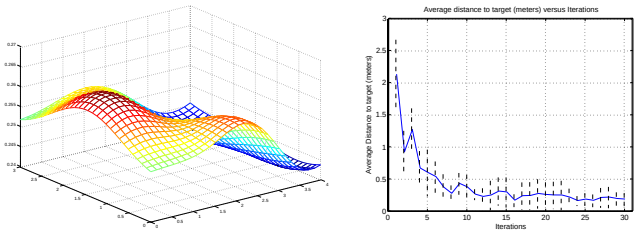

Fig. 1. The profile and average distance to the maximum.

We specify the maximum speed that robots can reach as $0.8m/s$ and the maximum acceleration as $1m/s^2$. Robots are equipped with 16 ultrasonic range detectors located at their front, left and right sides and and their back. However, they are not equipped with the devices that provide global position information to them and rely on their position odometry information only throughout the search. It is also assumed that robots are equipped with necessary sensors to get the potential (i.e., ethanol concentration) value information of each grid in the search area.

Initially the robots are located near the area entrance, point (0,0) in the cartesian plane. The first way-points in the search area are generated randomly for the robots and each robot

[1]In the Player/Stage realistic simulator Pioneer2dx robots are already modelled (i.e., have drivers for).

starts to move towards to its initial way-point. This helps the robots to localize themselves (with some errors due to the fact that all of the robots cannot start their search at exactly the same entrance point but they start rather close to each other and this will cause positioning errors at the beginning in terms of global positions) without a need for global positioning information. Moreover, it represents a realistic situation in which the area to be searched has only single entrance (such as a building to be searched for example). The ultrasonic sensors, which the robots are equipped with, are used to measure/determine inter-robot distances and to prevent robot to robot collisions. The ultrasonic sensor readings are used in calculating repulsive potential which is activated only at a smaller distance than predefined d which is chosen as $d = 0.5m$.

The plots in Figure 2 show the way-points that the robots visited and the trajectories of the robots at the search space respectively. The "X" symbol represents the random first way-points of the robots (which are effectively the initial positions of the particles of the PSO algorithm), the "O" symbol represents the final position of each robot, and the curves represent the trajectories of the robots. Stopping criteria for each robot is determined as the velocity vector, from (1), getting smaller values than a predetermined threshold which implies that robot will visit way-points which is close to its current way-point and eventually stops, as the velocity vector is close to zero and there is no improvement in position vector according to (1). The way-points of the robots are shown as "dots" on the contour maps of the profile. Due to robots physical dimensions, position odometry errors, initial robot location errors in the search area, and repulsion forces between them (which are used to prevent robot to robot collisions), it is a fact that all the robots congregate around the target and take position values close to the target (instead of all the robots reaching exactly the same point). Since each robot updates its position and velocity vectors asynchronously and there is no global iteration counter that synchronizes these updates, each robot finds the target in different number of iterations. The second plot on Figure 1 shows the average distance of robots to the target, global maximum point of resource profile, for each iteration. The vertical lines represent the standard deviation of the distance of robots to the target. The average distance to target and standard deviation decreases as the robots perform their PSO updates for the next iterations. Since each robot finishes its search task at different iterations, it is considered that each robot remains in the same position after it satisfied the stopping criteria mentioned above. The largest iteration number that a robot finished its task is 30 iterations, so the iteration axis is considered up to 30 iterations. From the graph, it is observed that the average number of iterations that robots could find target is about 20 iterations.

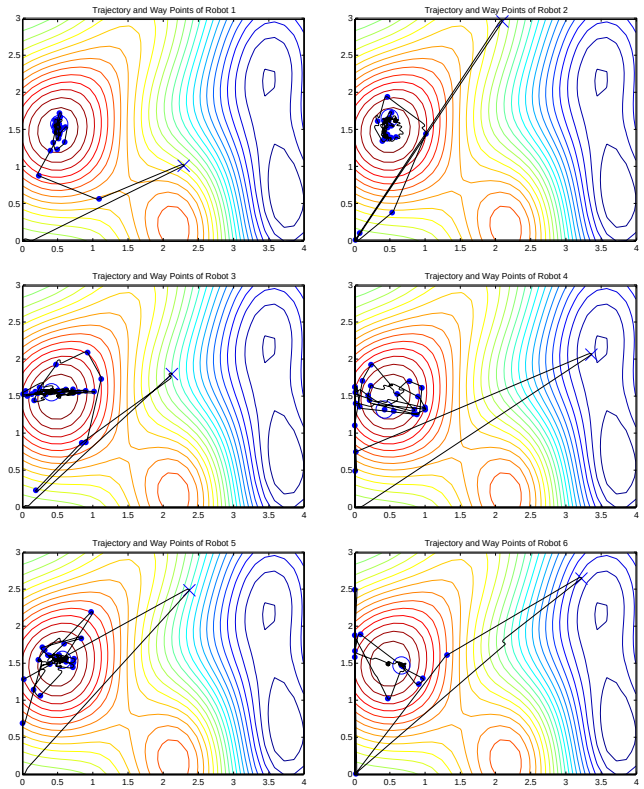

Fig. 2. Trajectories and way points of the robots.

V. Concluding Remarks

In this study Asynchronous Particle Swarm Optimization based robotic search method is tested in a simulation environment. The method is based on distributed asynchronous operation with dynamic information exchange topology. Future work can focus on implementing the search method in a dynamically changing environment and implementation of the method on real robot platforms.

References

[1] S. Doctor, G. Venayagamoorthy, and A. Gudise, "Optimal pso for collective robotic search applications," in *Proceedings of IEEE Congress on Evolutionary Computation (CEC-2004)*, vol. 2, 2004, pp. 1390–1395.

[2] J. Hereford, "A distributed particle swarm algorithm for swarm robotic applications," in *Proceedings of IEEE Congress on Evolutionary Computation (CEC-2006)*, vol. 2, 2006, pp. 1678–1685.

[3] J. Hereford, M. Siebold, and S. Nichols, "Using the particle swarm optimization algorithm for robotic search applications," in *Proceedings of IEEE Symposium on Swarm Intelligence (SIS-2007)*, 2007, pp. 53–59.

[4] J. Pugh and A. Martinoli, "Inspiring and modelling multi-robot search with particle swarm optimization," in *Proceedings of IEEE Congress on Evolutionary Computation (CEC-2002)*, vol. 2, 2002, pp. 1666–1670.

[5] L. Marques, U. Nunes, and A. de Almedia, "Particle swarm-based olfactory guided search," *Autonomous Robots*, vol. 20, no. 3, pp. 277–287, May 2006.

[6] S. B. Akat and V. Gazi, "Particle swarm optimization with dynamic neighborhood topology: Three neighborhood strategies and preliminary," in *IEEE Swarm Intelligence Symposium (SIS-2008)*, St. Louis, Missouri, September 2008.

[7] ——, "Decentralized asynchronous particle swarm optimization," in *IEEE Swarm Intelligence Symposium (SIS-2008)*, St. Louis, Missouri, September 2008.

[8] M. Clerc and J. Kennedy, "The particle swarm—explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, February 2002.