A Distributed Foraging Algorithm Based on Artificial Potential Field

***Abstract*—Simple collections of agents that perform collectively and use distributed control algorithms constitute the interests of swarm robotics. A key issue to improve system performances is to effectively coordinate the team of agents. We present in this paper a multi-agent foraging algorithm called Cooperative-Color Marking Foraging Agents (C-CMFA). It uses the coordination rules of the S-MASA (Stigmergic Multi-Ant Search Area) algorithm to (i) speed up the search process and (ii) allow agents to build an optimal Artificial Potential Field (APF) simultaneously while exploring. To benefit from multiple robots, we add one cooperation rule in the algorithm to attract large number of agents to the found food. This algorithm constitutes a distributed and synchronous version of the c-marking algorithm. Simulation results in comparison with the c-marking one show the superiority of C-CMFA in different environment configurations.**

I. Introduction

Multi-agent approach is suitable for many real world applications: mine detecting [1] [2], search in damaged buildings [3] [4], fire fighting [5], and exploration of spaces [6] [7]. Simple, local and individual rules can provide complex, global and collective behaviors [8], as in social insects (ants, bees and termites). Such biological systems use stigmergic communication (pheromones for example), to coordinate their actions aiming to improve the performances of the team. Recently many researchers have investigated this kind of bio-inspired coordination methods [9] [10]. We focus in this work on the problem of foraging for robot swarm. Foraging algorithms allow a collection of agents to search their environment for a target ('food'), then return it to the nest [11]. Behavior-based algorithms are well-known in the swarm algorithms community [12] [13] [14]. These algorithms are based on a few 'sub-algorithms' which the collection of agents intelligently switches between in various combinations and at various periods. Gradient-based algorithms are well-known too in foraging. Agents in these algorithms create gradients leading from nest to food, while they explore. They use: physical marks [15] [16], pre-deployed sensor networks [17], and beacons [18]. In our work, we focus on behavior-based and gradient-based algorithms. Authors in [19] have developed behavior and gradient-based algorithm, in which agents can build gradients by writing integer values in the environment. The algorithm proposed in this paper is similar to the one in [19], in that it is behavior-based and agents can write integer values to create gradients. Differently from [19], our algorithm uses a new search strategy and include cooperation rule to accelerate the exploitation of found food.

We present in this paper, a distributed foraging algorithm called Cooperative-Color Marking Foraging Agents (C-CMFA), which is both behavior-based and gradient-based. To achieve the foraging task efficiently, agents need a way to find the food as fast as possible and a way to return to the nest. In the c-marking algorithm, the convergence of APF to the optimal values takes huge time. As we want to avoid such drawback in C-CMFA algorithm, we have used S-MASA algorithm [20] to accelerate the search time and to allow agents to build optimal paths by writing integer values while they explore. We also add one cooperation rule (diffusion of pheromone to neighbors when food is found) to attract agents in the neighborhood in order to accelerate the exploitation of food. We conclude from the quantitative comparisons of C-CMFA algorithm with c-marking algorithm, that C-CMFA algorithm is a distributed and synchronous version of the wavefront algorithm [21], which was enhanced initially by [19], to distributed and asynchronous algorithm for foraging problem by multiple agents in bounded grid environments.

The remainder of this paper is organized as follows: in Section I, we present the two previous works that we reused to develop our algorithm. In Section II, we describe the C-CMFA algorithm, specifically its finite state machine and some comparisons with c-marking algorithm. Section III presents the performance evaluation of our algorithm with respect to the c-marking algorithm. Finally, we conclude the paper and present some future works in Section IV.

II. Background

In this section we present the two algorithms that we have based on to develop our algorithm: the S-MASA algorithm [20] and the c-marking algorithm [19].

A. S-MASA Algorithm

The S-MASA (Stigmergic Multi-Ant Search Area) algorithm is a search algorithm based on the principles of the ant system and the water vortex dynamics. It have been used for multi-target search and coverage tasks in bounded grid environments [20]. The coordination principles depicted in Figure 1 and reported in Algorithm 1 are used to adjust the heading of the agents to choose the not visited cells. It will be applied in this work for multi-agent foraging in unbounded environments. S-MASA offer some advantages: (1) It accelerates the search process and (2) It helps to produce Optimal APF values without the need to revisit already visited cells because of its vortex turns around the central point and around agents; this produces optimal paths from food sources to the nest (if the task needs to return home as foraging for example).

B. c-marking Algorithm

c-marking algorithm [19] is an original approach for foraging. It is a parameter-free distributed and asynchronous version of wavefront algorithm [21]. Agents in c-marking algorithm

```
Data: APF Value
Result: Heading

if (Right cell is marked) then
    if (Heading = 270) then
        Set heading to 0
    else
        Set heading to heading + 90
    end
end
```

Algorithm 1: Coordination Principle in S-MASA Algorithm

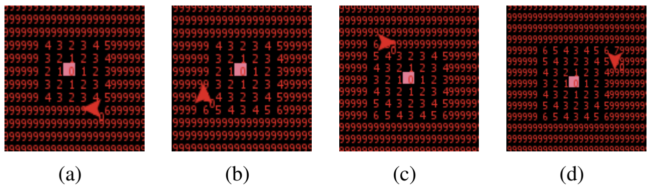


Fig. 1: S-MASA coordination principle: (a) Changing heading from 180 to 270 (b) Changing heading from 270 to 0 (c) Changing heading from 0 to 90 (d) Changing heading from 90 to 180

build paths simultaneously while exploring the environment, they avoid local minima by computing a wavefront expansion from the agent destination which involves building an ascending Artificial Potential Field. Agents follow the negative gradient at each iteration, it examines the potential values of the four neighboring cells and moves to the cell with the smallest value. Because agents use a pseudo random walk (choose the not marked yet cells) as a search strategy; the convergence of APF values takes huge time to be reached. They may, therefore need to visit the same cell several times before it reaches its optimal value specially when the number of agents is not sufficient to sustain the cells corresponding to the wavefront.

Some advantages of c-marking algorithm: (1) Agents build simultaneously paths when they explore and (2) It is very robust to agents' failure and to complex environments. However, it provides some drawbacks: (1) As a result of the pseudo random walk, the built paths are not optimal, and the length of the paths can increase dramatically the foraging time ; (2) The number of agents that contribute to foraging is in most of the cases very small especially when the environment is large.

III. C-CMFA Algorithm

We assume simple ant-like agents, that can move in an unbounded grid world in the four directions (up, down, right and left). The world is unbounded, with a nest, obstacles in some fixed positions and food locations distributed randomly. Agents can perceive only the four neighboring cells. They use indirect communication via writing APF values, to repulse agents from already visited cells (marked cells). The position of food locations is unknown to agents, and they must adjust their headings using the coordination principles introduced in section II-A and move in the environment. Once the food is located, agents can pick a limited quantity and deposit diffusible pheromone to attract the other agents to this food, then begin returning it to the nest following the negative gradient.

To achieve the foraging task efficiently, agents need a way to find as fast as possible the food and some way to return to the nest. In our proposed Algorithm C-CMFA (Algorithm 2 and Figure 2 represent the behavior executed by agents), agents:(1) use S-MASA to search for food, (2) follow the negative gradient of APF values to return home, and (3) diffuse pheromone to attract agents in order to cooperate in the exploitation of found food. With these three points, we can avoid the drawbacks of c-marking algorithm. With point (1), we speed up the search process and get optimal paths (optimal APF values) without the need to revisit already visited cells. While points (2) contributes to accelerate the homing process by using the optimal paths and point (3) contributes to speed up the exploitation time by the cooperation.

Figure 2, shows the finite state machine of our foraging agent. According to surrounding events, the agent switches between several states: *LOOK-FOR-FOOD*, *CHOOSE-NEXT-PATCH*, *PICK-FOOD*, *RETURN-AND-COLOR*, *RETURN-TO-NEST*, *AT-HOME*, *CLIMB* and *REMOVE-TRAIL*. It starts with looking for food (*LOOK-FOR-FOOD*). If there is no food, it moves in its environment using *CHOOSE-NEXT-PATCH* rules and deposits pheromones that evaporates with the time. It changes automatically to *LOOK-FOR-FOOD* state, else it picks a quantity of food (*PICK-FOOD*), and returns home by following the colored trail, if there exists one (*RETURN-TO-NEST*). If there exits no trail, it diffuses the information to its neighbors by depositing pheromones that do not evaporate (brown color) and creates one trail by depositing pheromones that do not evaporate too (yellow color)(*RETURN-AND-COLOR*) while returning to home. When the home is reached, it unloads the food (*AT-HOME*) and if the food is exhausted it removes the existing trail (*REMOVE-TRAIL*), else it climbs the trail to the food location (*CLIMB*).

C-CMFA algorithm is similar to c-marking algorithm in using APF values to return home and in coloring trails to keep track of found food. However, some differences exist between them: (1) The search sub-task in C-CMFA algorithm is achieved by S-MASA algorithm, while in c-marking it is achieved by pseudo-random walk, (2) The diffusion of pheromone to attract agents to cooperate in exploitation of food does not exist, the only way of cooperation in c-marking is climbing existing trails by different agents, and (3) The remove trail in C-CMFA is the opposite of c-marking one, it starts from the nest to the food.

IV. Performance Evaluation

We discuss here the performance of the C-CMFA algorithm and we compare it with the c-marking algorithm in obstacle-free and obstacle unbounded environment. We present the parameters, metrics and scenarios used to evaluate the algorithms. We present, compare and discuss the obtained results.

A. *Parameters and Metrics*

Three metrics have been used to test the performance of the algorithms:

```
LOOK-FOR-FOOD
if (a resource is found) then go to PICK-FOOD;
else go to CHOOSE-NEXT-PATCH;
PICK-FOOD
Pick up a quantity $Q_{max}$ of food;
if (trail exist) then go to RETURN-TO-NEST;
else go to RETURN-AND-COLOR;
CHOOSE-NEXT-PATCH
if (obstacle detected) then Avoid_Obstacle();
else
    Update_Value();
    if (Brown pheromone here) and (food not
    exhausted) then Diffuse_Pheromone();
    go to food location using brown cells;
    else if (Brown pheromone here) and (food
    exhausted) then Remove brown trail;
    else Diffuse_Pheromone();
    Detect_Pheromone_Adjust_Heading();
    Update_Value();
    Move();
end
RETURN-TO-NEST
if (Home is reached) then go to AT-HOME;
else move to a neighboring cell with the smallest APF
value;
RETURN-AND-COLOR
if (Home is reached) then go to AT-HOME;
else
    move to a neighboring cell with the smallest APF
    value;
    Color that cell to a specific trail color (yellow);
end
AT-HOME
Unload Food;
if (Trail exists) and ($food > 0$) then  go to CLIMB;
else if (Trail exists) and ($food = 0$) then  go to
REMOVE-TRAIL;
else go to LOOK-FOR-FOOD
CLIMB
Move to neighboring colored cell with the greatest APF
value;
if the color here is yellow then go to CLIMB;
else go to LOOK-FOR-FOOD;
REMOVE-TRAIL
Move to neighboring colored cell with the greatest APF
value;
Update its color to the default one (black);
if the color here is brown then go to
LOOK-FOR-FOOD else go to REMOVE-TRAIL;
Update_Value()
Val = 1 + min (4 neighboring values);
write Val in current cell;
```

Algorithm 2: C-CMFA Algorithm

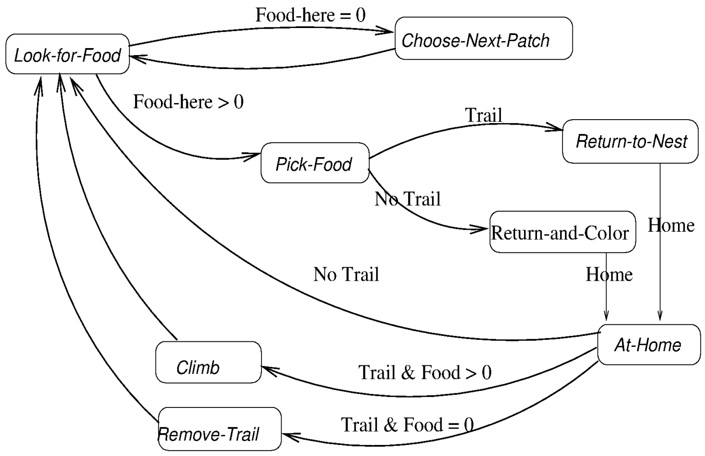


Fig. 2: Finite state machine of an autonomous foraging agent

- *Search time* – Represents the number of steps needed to discover a food.
- *Foraging time* – Represents the time needed to exhaust a food, it is measured in steps (ticks).
- *Total food returned* – The total amount of food that have been returned to the nest after given elapsed time steps.

The simulations were carried out using agent-based modeling within Netlogo [22], in two world configurations: obstacle-free and obstacle environment (see Figure 3). The environment in the two configurations is unbounded, the nest and the obstacles take fixed positions, and the food is distributed randomly in a limited region of the environment. Parameters of the three scenarios used are presented in Table I, where *Food density* is the number of food locations, each food location contains a limited amount of food (*Food concentration*) and each agent can carry a limited amount of food at each time (unit) *Agent's capacity*.

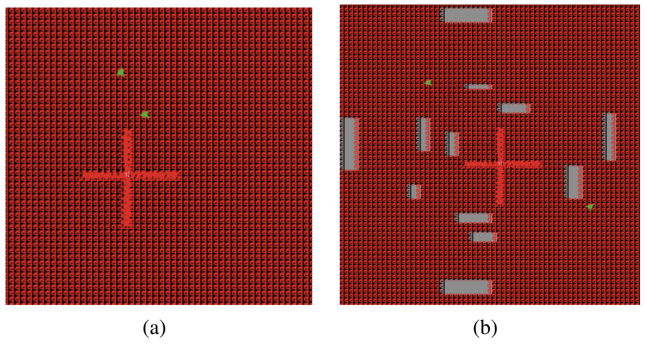


Fig. 3: World setups used in simulations (a) Unbounded obstacle-free environment (b) Unbounded obstacle environment, where red arrows in the center are agents, the pink cell in the center is the nest, the green arrows are food and the gray clusters are obstacles.

To test the effect of the search strategy on the performance of the proposed algorithm, we have used scenario 1, scenario

TABLE I: Parameters of scenario 1, scenario 2 and scenario 3

	Parameter	Value
Scenario 1		*Search Time Analysis*
	Number of agents	5–30
	Food density	1 site
	Food concentration	30 units
	Agent's capacity	5 unit
Scenario 2		*Exploitation Time Analysis*
	Number of agents	5–30
	Food density	1 site
	Food concentration	60 units
	Agent's capacity	5 unit
Scenario 3		*Returned Food Analysis*
	Number of ticks	600–4000
	Number of agents	10
	Food density	2 sites
	Food concentration	30 units
	Agent's capacity	5 unit

2 and scenario 3. We test in scenario 1 the efficiency of the search strategy, we vary in it the number of agents from 5 to 30 to test the effect of the agents number on the search strategy, we use one food location with 30 units (concentration) and each agent can carry 5 units at each time. To prove the efficiency of cooperation in multi-agent foraging, we have used scenario 2, in which we vary the number of agents from 5 to 30, the environment contain one food site with 60 units each one, with 5 units as agent's capacity. In scenario 3, we measure the amount of food returned over an elapsed time. We vary in this scenario, the number of ticks from 600–4000, the number of agents is 10 with 5 units (agent's capacity) 2 sites were used each with 30 units each one.

B. Simulation Results and Comparison

In this section, we present, compare and discuss the results obtained by the two algorithms in the three scenarios showed in Table I.

Results in scenario 1 prove the efficiency of the search strategy used (S-MASA algorithm) (see Table II and Figure 4(a), 4(b)). At each increase in agent's number, the search time decreases in the two algorithms. In C-CMFA algorithm, even with small number of agents we got shorter search time (5 agents – 346 ticks), while in c-marking algorithm the time is much longer(9832 tick with 5 agents too). C-CMFA algorithm is five times faster than c-marking algorithm when the number of agents is small (5 – 10) and two to three times faster when the number of agents is greater (15 – 30). We obtain the same performance of c-marking with 25 agent, with only 10 agents in C-CMFA.

Scenario 2 was chosen to test the efficiency of the exploitation time of found food which is function of the path length and number of agents cooperating in the exploitation. We got best results with C-CMFA algorithm in obstacle-free and obstacle unbounded environments. The exploitation time decrease with

TABLE II: Results in scenario 1: Search time

	5	**10**	**15**	**20**	**25**	**30**
Unbounded obstacle-free						
C-CMFA	346.4	285.6	187.4	157.7	118.7	89.4
c-marking	9832.7	1551.2	541.7	412.3	277.8	202.4
Unbounded obstacle						
C-CMFA	515.7	414.1	287.3	178.5	149.6	114.8
c-marking	11081.2	2310.5	661.2	526.4	303.5	261.8

the increase of agents' number in the two algorithms. C-CMFA algorithm is faster two to four times than the c-marking one. We obtain the same performances in c-marking (25 agents) with only 10 agents. This quantitative results prove the benefit of cooperation in multi-agent foraging.

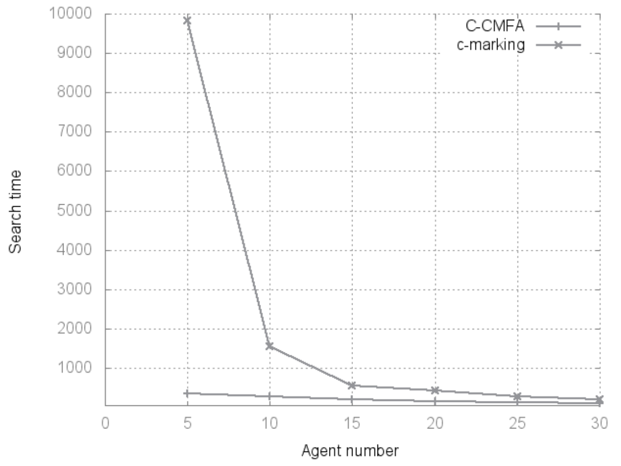
TABLE III: Results in scenario 2: Exploitation time

	5	**10**	**15**	**20**	**25**	**30**
Unbounded obstacle-free						
C-CMFA	291.7	243	172.5	140.4	125.8	92.2
c-marking	1231.5	894.1	656.7	471.6	284.1	213.7
Unbounded obstacle						
C-CMFA	335.2	271.8	206.6	192.3	157.8	132.6
c-marking	1404.1	1055	847.8	529.6	390.6	294.4

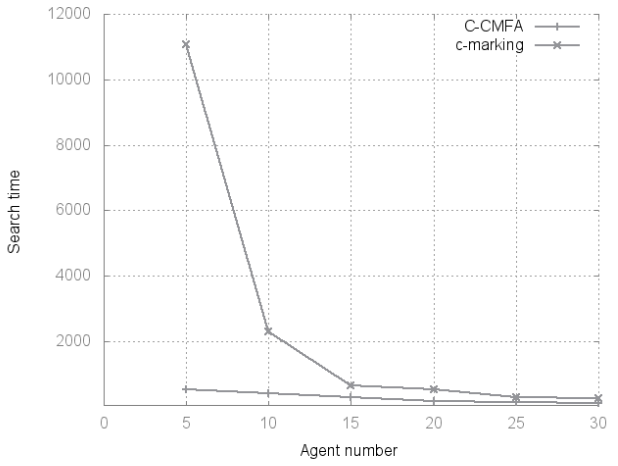
In scenario 3, we test the performance of the two algorithms by computing the amount of food returned by agents in some elapsed time (ticks). The total amount of food deposited in the environment is 60 units divided between two food sources. With the increase of ticks, the amount of food returned increase in the two algorithms in obstacle-free and obstacle environment. While C-CMFA algorithm obtain the total amount of food in 1000 ticks in obstacle-free environment (1400 ticks in obstacle environment), c-marking algorithm could not obtain it, because the environment is unbounded, the number of agents is not enough to sustain the cells corresponding to the wavefront and the pseudo random walk can drive agents away from food. However, when we increase the number of agents from 10 to 40 in c-marking, it obtains the total amount of food in 1500 ticks in obstacle-free environment (1800 ticks in obstacle environment)

TABLE IV: Results in scenario 3: Returned food over ticks

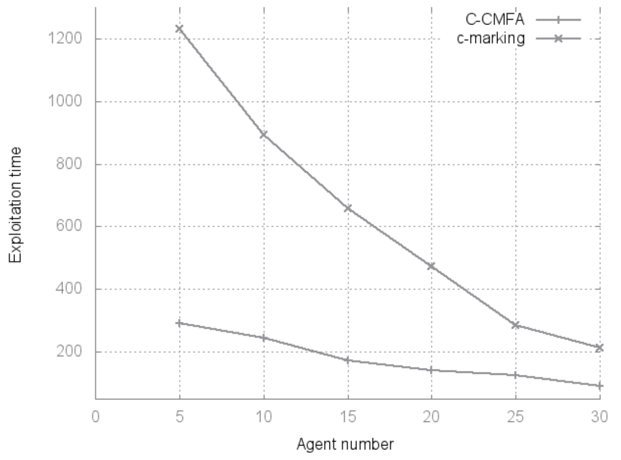
	600	**900**	**1000**	**1400**	**2100**	**2500**	**3000**	**3500**	**4000**
Unbounded obstacle-free									
C-CMFA	43.7	50.7	60	60	60	60	60	60	60
c-marking	19.2	29.5	33.5	40	49.7	57	50.7	52.5	43.5
Unbounded obstacle									
C-CMFA	31.2	40.7	47	60	60	60	60	60	60
c-marking	15.7	22.3	25.2	34	43.2	47.5	45.7	36.2	40.7



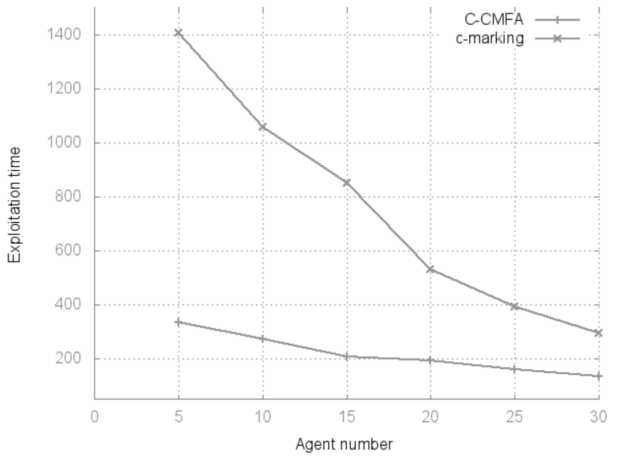
(a)



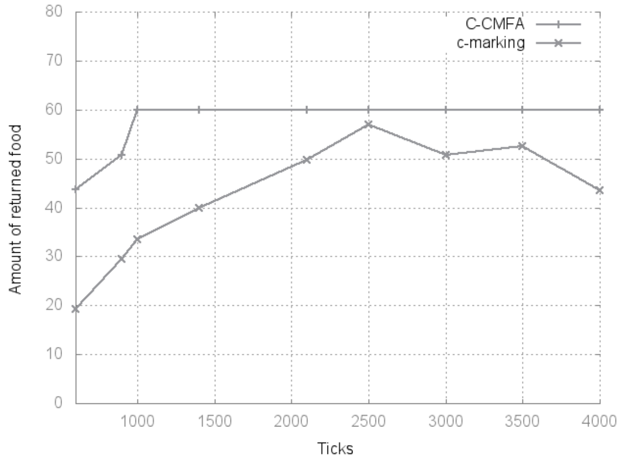
(b)



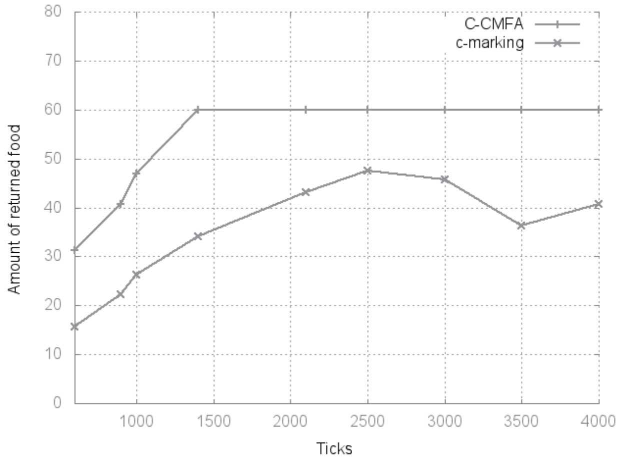
(c)



(d)



(e)



(f)

Fig. 4: Simulation results : (a), (b) Results of scenario 1 in unbounded obstacle-free and unbounded obstacle environment. (c), (d) Results of scenario 2 in unbounded obstacle-free and unbounded obstacle environment. (e), (f) Results of scenario 3 in unbounded obstacle-free and unbounded obstacle environment.

V. Conclusion

We presented in this paper a distributed foraging algorithm (C-CMFA), which uses S-MASA algorithm that allows agents to build optimal paths simultaneously and synchronously while exploring, which results in accelerating the search process. To accelerate the exploitation time, agents diffuse pheromones to attract others agents to cooperate in exploiting the found food. Three scenarios have been used in simulations: the first scenario aimed to test the efficiency of the search strategy used (S-MASA algorithm), the second scenario tested the efficiency of the homing strategy (paths are optimal or not) and the third one focused on testing the benefit of cooperation and its effect on the performances. In the three scenarios, C-CMFA algorithm gave the best results in obstacle-free and obstacle unbounded environments with respect to c-marking algorithm.

It is worth pointing out that the c-marking algorithm is a distributed and asynchronous version of Barraquand et al. wavefront algorithm [21]. Our algorithm is a distributed and synchronous version of Barraquand et al. wavefront algorithm [21]. c-marking is asynchronous because the wavefront is not created at the same time (the optimal values of cells create the wavefront). The number of agents is not sufficient to sustain all the cells corresponding to the wavefront. As a consequence the written values are not optimal from the first visit and agents need to revisit already visited cells in order to make them optimal. While the C-CMFA algorithm is synchronous, because the wavefront is created at the same time whatever the number of agents and the values are optimal from the first visit.

The assumption of writing integer values (APF values) in the environment by agents provide basic solutions for foraging, however, it is still far from real world applications. Pheromone is an example of real implementation of stigmergic communication in real world experiments and it can replace the integer values. This is one of our future direction to enhance C-CMFA algorithm. In addition, we intend to improve the obstacle-avoidance module of this algorithm to deal with complex environments.

References

[1] E. Acar, H. Choset, Y. Zhang, and M. Schervish, "Path planning for robotic demining: Robust sensor-based coverage of unstructured environments and probabilistic methods," *International Journal of Robotics Research 22(7-8)*, pp. 441–466, 2003.

[2] D. Gage, "Many-robot mcm search systems," in *Autonomous Vehicles in Mine Countermeasures Symposium, vol. 9*. DOI: 10.1.1.38.771, 1995, pp. 56–64.

[3] G. Kantor, S. Singh, R. Peterson, D. Rus, A. Das, V. Kumar, and G. Pereira, "Distributed search and rescue with robot and sensor teams," in *Field and Service Robotics*. DOI: 10.1007/10991459-51, Springer Berlin Heidelberg, 2006, pp. 529–538.

[4] J. Jennings, G. Whelan, and W. Evans, "Cooperative search and rescue with a team of mobile robots," in *8th International Conference on Advanced Robotics, ICAR*. DOI: 10.1109/ICAR.1997.620182, IEEE.

[5] A. Marjovi, J. unes, L. Marques, and A. de Almeida, "Multi-robot exploration and fire searching," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*. DOI: 10.1109/IROS.2009.5354598, IEEE, 2009, pp. 1929–1934.

[6] Landis and A. Geoffrey, "Robots and humans: Synergy in planetary exploration," in *SPACE TECHNOLOGY AND APPLICATIONS INT. FORUM-STAIF 2003: Conf. on Thermophysics in Microgravity; Commercial/Civil Next Generation Space Transportation; Human Space Exploration; Symps. on Space Nuclear Power and Propulsion (20th); Space Colonization (1st)*, vol. 654, no. 1. DOI: org/10.1063/1.1541377, AIP Publishing, 2003, pp. 853–860.

[7] K. Schilling and C. Jungius, "Mobile robots for planetary exploration," *Control Engineering Practice*, vol. 4, no. 4, pp. 513–524, 1996.

[8] Z. Meng, B. Zou, and Y. Zeng, "Considering direct interaction of artificial ant colony foraging simulation and animation," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 24, no. 1, pp. 95–107, 2012.

[9] R. L. Stewart and R. A. Russell, "A distributed feedback mechanism to regulate wall construction by a robotic swarm," *Adaptive Behavior*, vol. 14, no. 1, pp. 21–51, 2006.

[10] M. J. B. Krieger, J. B. Billeter, and L. Keller, "Ant-like task allocation and recruitment in cooperative robots," *Nature*, vol. 406, no. 6799, pp. 992–995, 2000.

[11] A. F. Winfield, "Foraging robots," in *Encyclopedia of Complexity and Systems Science*. Springer, 2009, pp. 3682–3700.

[12] S. Nouyan, R. Groß, M. Bonani, F. Mondada, and M. Dorigo, "Teamwork in self-organized robot colonies," *Evolutionary Computation, IEEE Transactions on*, vol. 13, no. 4, pp. 695–711, 2009.

[13] R. C. Arkin, *Behavior-based robotics*. MIT press, 1998.

[14] C. Jones and M. Mataric, "Behavior-based coordination in multi-robot systems. autonomous mobile robots: Sensing, control," *Decision-Making, Applications*, 2005.

[15] J. Svennebring and S. Koenig, "Building terrain-covering ant robots: A feasibility study," *Autonomous Robots*, vol. 16, no. 3, pp. 313–332, 2004.

[16] M. Mamei and F. Zambonelli, "Spreading pheromones in everyday environments via rfid technologies," in *Proceedings of 2nd IEEE Symposium on Swarm Intelligence*, 2005.

[17] K. J. Ohara, D. B. Walker, and T. R. Balch, "The gnatslow-cost embedded networks for supporting mobile robots," in *Multi-Robot Systems. From Swarms to Intelligent Automata Volume III*. Springer, 2005, pp. 277–282.

[18] E. J. Barth, "A dynamic programming approach to robotic swarm navigation using relay markers," in *American Control Conference, 2003. Proceedings of the 2003*, vol. 6. IEEE, 2003, pp. 5264–5269.

[19] O. Simonin, F. Charpillet, and E. Thierry, "Revisiting wavefront construction with collective agents: an approach to foraging," *Swarm Intelligence*, pp. 113–138, 2014.

[20] O. Zedadra, N. Jouandeau, H. Seridi, and G. Fortino, "S-MASA: A stigmergy based algorithm for multi-target search," in *Proceedings of the 2014 Federated Conference on Computer Science and Information Systems*, ser. Annals of Computer Science and Information Systems, M. P. M. Ganzha, L. Maciaszek, Ed., vol. 2. IEEE, 2014, pp. pages 1477–1485.

[21] J. Barraquand, B. Langlois, and J.-C. Latombe, "Numerical potential field techniques for robot path planning," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 22, no. 2, pp. 224–241, 1992.

[22] U. Wilensky, "Netlogo. http://ccl.northwestern.edu/netlogo/,," in *Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL*, 1999.