swarmFSTaxis: Borrowing a Swarm Communication Mechanism from Fireflies and Slime Mold

Joshua Cherian Varughese[1,2], Daniel Moser[1], Ronald Thenius[1], Franz Wotawa[2], and Thomas Schmickl[1]

[1] Institut für Zoologie, Karl Franzens Universität Graz, Austria
[2] Institut für Software Technologie, Technische Universität Graz, Austria

Abstract. One main motivation for studying swarm intelligence comes from observing the resilience of living systems in nature. Swarm intelligence has provided important inspirations for the engineering of technical systems. The swarmtaxis algorithm and the FSTaxis algorithm are such swarm intelligent algorithms that aim to move a group of agents from a starting point to a predefined goal. The swarmtaxis algorithm bases its state transition on a voting like mechanism in which the agents count the number of "pings" they get from their surroundings. In contrast, the FSTaxis algorithm uses a scroll wave based communication mechanism inspired by slime mold and fireflies. The scroll wave based communication is expected to be more resilient than the voting like mechanism of the swarmtaxis algorithm. In this paper, we borrow the communication mechanism used in FSTaxis algorithm to improve the swarmtaxis algorithm. We will also discuss how this modified algorithm performs in comparison to the parent algorithm.

Keywords: swarm intelligence, bio-inspiration, signaling, taxis

1 Introduction

Nature has been a perpetual source of inspiration in scientific research and engineering. Swarm intelligence is one of the fields that relies heavily on nature as its main source of inspiration. Task allocation [12], flocking [3], navigation [2], communication [8] are few examples of the areas swarm intelligence has drawn inspiration from nature. The FSTaxis algorithm [8] implements a combination of the communication mechanisms of fireflies and slime mold for gradient ascent of a group of mobile autonomous agents. In this paper we present an improved algorithm, borrowing from this communication mechanism used by fireflies and slime mold as well as an existing swarm intelligent algorithm known as swarmtaxis [1]. This paper intends to demonstrate how resilient communication methods such as that of slime mold and fireflies can substantially improve existing algorithms.

This study is part of project subCULTron [6,7] which aims to develop a swarm of autonomous underwater robots, which perform measurements and monitoring

of the underwater environment. In contrast to traditional underwater communication using acoustics, project subCULTron uses a local communication method, blue-light communications [7], where the robots exchange small packets of modulated blue-light signals. The range of blue-light communication is around one meter under water [7]. Considering the cost and complexity of underwater swarm communication, the swarmtaxis and the FSTaxis algorithms are suitable candidates among other considered algorithms [4,9,2] for swarm navigation. During our recent study on resilience (which is under review for publication), we established that the communication mechanism of both FSTaxis and swarmtaxis algorithm have non-zero resilience to agent-to-agent communication failures while the FSTaxis algorithm exhibited a significantly higher resilience. In this paper, we will present a modified version of swarmtaxis algorithm which borrows the resilient communication behavior of the FSTaxis algorithm. Such a work aims to make the swarmtaxis algorithm more resilient to agent-to-agent communication failures.

In the following sections of the paper, we will describe briefly the algorithms that we draw inspiration from in Section 2. After which in Section 3, we will describe our approach by which we test our algorithm in simulation and present the simulation results. Subsequently, discuss the results and briefly compare it with the original swarmtaxis algorithm in Section 4 before concluding the paper.

2 Algorithms

In this section, we will briefly describe the swarmtaxis algorithm and then define the modified swarmtaxis algorithm.

2.1 The swarmtaxis Algorithm

In [1], the authors present the algorithm to move a group of robots from a starting point to a predefined goal. Each agent executing the swarmtaxis algorithm is required to have the ability to move, communicate with other robots within a communication range, sense other robots in its immediate surroundings and also the distance to a nearby robot. Apart from these capabilities, each robot is equipped with a long range sensor with which the robot perceive the direction of the goal. In the swarmtaxis scenario, the goal can be occluded from an agent if another agent positions itself in between the goal and the agent. If an agent is occluded from the goal, it is said to be "shadowed" and otherwise the agents are said to be "illuminated". Each agent broadcasts a single bit ping to its surroundings and every agent receiving this ping will use a polling mechanism to add it to the number of pings previously received. We will refer to this mechanism as a poll based mechanism henceforth. Depending on the pings received, the agents may assume one of the following states:"forward", "coherence", "avoid" and "random". By default, all agents are in "forward" state in which the agents moves forward at a preset velocity. If the number of pings received increases beyond a preset threshold α, the agents senses that it is at the center of the

swarm and executes a random turn and then transitions back into the "forward" state. If an agent detects the number of pings it received to be lower than α, then the agent detects that it is moving away from the swarm. In this case, the agent transitions into "coherence" state in which it takes a U-turn and transitions back into the "forward" state. The states of the algorithm explained till now will only keep the agents together and not produce any net movement towards the goal. The net movement towards the goal is caused by the "avoid" states. When the robot detects another robot within its "avoidance radius", it transitions into the avoid state in which the agent takes a turn away from the detected neighbor. All the illuminated robots sets its avoidance radius slightly higher than the shadowed robots, i.e, $avoid_{illum} > avoid_{shadow}$. This enables the illuminated agents to see the shadowed agents before the latter sees the former and move away from the shadowed agents. Thus, illuminated agents moving away from the shadowed agents (but towards the goal) and in effect produces a net movement of the swarm towards the goal.

2.2 The swarmFSTaxis Algorithm

In the swarmFSTaxis algorithm, additional to all capabilities described in Section 2.1, each agent is assumed to have internal timers and local directional communication. The goal can be occluded from an agent similar to the case of the the swarmtaxis scenario. If an agent is occluded from the goal, it is said to be "shadowed" and otherwise the agents are said to be "illuminated". The behavior of the agents differ depending on whether they are "illuminated" or "shadowed". Figures 1 and 2 shows the state machine of the swarmFSTaxis algorithm.

There are two types of behaviors in the swarmFSTaxis algorithm: the "ping" behavior and the "motion" behavior just as in the case of the FSTaxis algorithm [8]. The ping behavior describes the agent-to-agent communication during the execution of the algorithm. In ping behavior, as shown in Figure 1, the agents may assume three states: "pinging", "refractory" and "inactive". Initially, all agents are set to inactive state. In the inactive state, the agent monitors its receivers for incoming single bit local communication (pings). In the event of an incoming ping the agent broadcasts a ping for a certain duration t_{ping}. At the end of a ping, the agent enters the refractory state. During refractory time, t_{refrac}, the agent is insensitive to all incoming pings. At the end of the refractory time, the agent transitions back to inactive state. Apart from the above ping behavior, the "illuminated" agents have internal timers that are constantly counting down. When the timer counts down to zero, the agent broadcasts a ping. This means that an "illuminated" agent produces a ping either when the agent receives another ping or when its internal timer counts down to zero. The difference between the ping behaviors of shadowed and illuminated agents are illustrated in Figure 3.

In addition to the ping behavior described above, the agents also have a "motion" behavior. Unlike the ping behavior, the motion behavior is same for all agents regardless of whether they are "illuminated" or "shadowed". There are two kinds of motion behavior: "general motion behavior" and "avoid motion

behavior" as shown in Figure 2. In "General motion behavior", an agent at the event of an incoming ping, moves towards the incoming ping. In case there are multiple incoming pings, the agent moves towards the mean of the directions of all incoming pings.

During "avoid motion behavior", an agent move away from a detected neighbor. As in the case of the swarmtaxis algorithm, the swarmFSTaxis algorithm also implements dissimilar avoidance radii for "illuminated" and "shadowed" agents: $avoid_{illum}$ and $avoid_{shadow}$. The sensor range of the illuminated agents are set to a higher value which in effect, enables the agents to move away from an approaching shadowed agent. The pseudo-code for the swarmFSTaxis algorithm can be found in Algorithm 2.2.

The above explained behavior will result in "scroll waves" like in the case of slime mold [5] propagating through the swarm. Since the illuminated agents trigger pings when their internal timer counts down to zero, the waves will originate at the "illuminated" agents and propagate through the "shadowed" agents as they relay the pings. The "general motion behavior" ensures that the swarm stays together with the agents moving towards incoming ping while the "avoid motion behavior" ensures that the illuminated agents move away from the shadowed agents and in effect, move towards the goal.

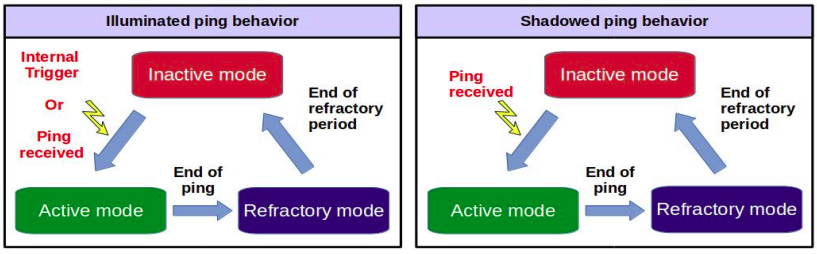

Fig. 1. A state transition diagram of the ping behavior of the swarmFSTaxis algorithm is shown in the figure. Illuminated and shadowed agents differ in ping behavior in that the shadowed agents only relay pings while the illuminated agents both produce and relay pings.

3 Methods and Results

The simulation environment used for testing the algorithms is Netlogo 4.3.1 [10]. In Netlogo, the simulation arena is divided into spatial units known as "patches" , the simulation time is measured in "ticks". The sensor radius, s_r, of the agents is measured in coordinate distances measured from the position of the agent of interest. It is taken to be 2.5 patches since this corresponds to a reasonable underwater local communication range using blue-light in project

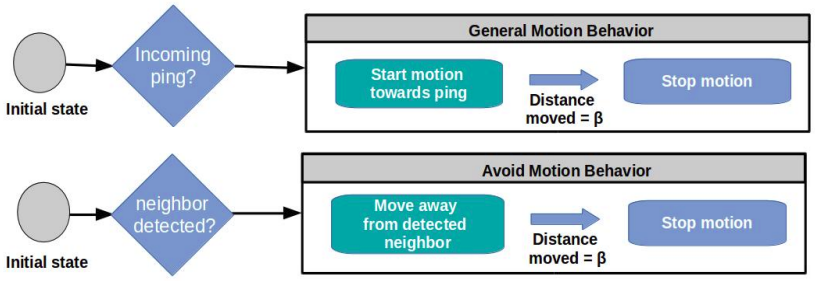

Fig. 2. A state transition diagram of the motion behavior of the swarmFSTaxis algorithm is shown in the figure. Illuminated and shadowed agents have the same motion behavior triggered by either an incoming ping or a neighbor.

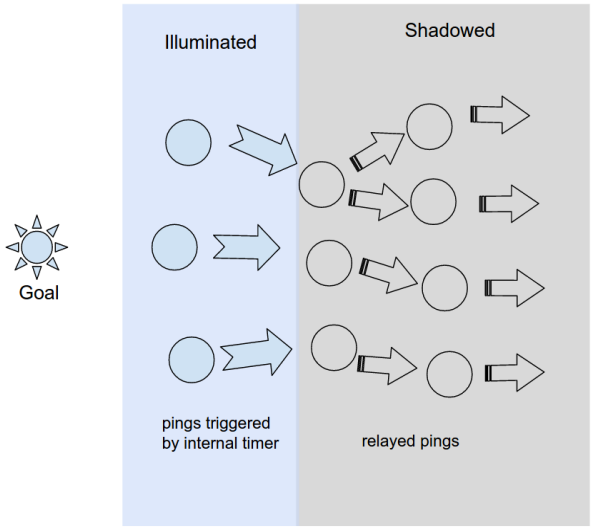

Fig. 3. The ping mechanism of the swarmFSTaxis algorithm is illustrated in the figure. The "illuminated" agents has a low internal timer value and hence will hijack the pinging frequency of the system by pinging frequently. The shadowed agents will keep relaying the pings produced by the illuminated agents.

Algorithm 1 The swarmFSTaxis algorithm

```
repeat
    for all agents do
        PING BEHAVIOR(t_ping, t_refrac, counter, main_counter)
        AVOID MOTION BEHAVIOR(move agent, heading of detected agent)
    end for
until forever
procedure PING BEHAVIOR(t_ping, t_refrac, counter, main_counter)
    set main_counter ← main_counter + 1
    if pingmode ! = "inactive" then
        set counter ← counter + 1
    end if
    if pingmode = "active" AND counter = t_ping then
        set pingmode ← "refractory"
    end if
    if pingmode = "refractory" AND counter = t_ping + t_refrac then
        set pingmode ← "inactive"
        set counter ← 0
    end if
    if pingmode = "inactive" then
        if any ping received? then
            set state ← active mode
            GENERAL MOTION BEHAVIOR(agentname)
            for i ← 1, no : ofpingsreceived do
                append list l ← heading of incoming ping
            end for
        end if
    end if
    if main_counter ≥ internal_timer_value then
        if is_illuminated? then
            set state ← active mode
        end if
        set main_counter ← 0
        set counter ← 0
        Create empty list, l
    end if
end procedure
procedure GENERAL MOTION BEHAVIOR(move agent)
    calculate h_mean of list, l
    set agent heading h_a ← h_mean
    move agent for distance β
end procedure
procedure AVOID MOTION BEHAVIOR(move agent, heading of detected agent)
    if agents in sensing range? then
        set agent heading h_a ← oppositetoagentinsensingrange"
        move agent for distance β
    end if
end procedure
```

subCULTon [7]. At the beginning of a typical run, the agents are distributed uniformly around a starting point and then the algorithm is executed. The center of mass of the swarm is used as a collective position estimate of the swarm. Once the center of mass of the swarm reaches the goal as shown in Figure 4(b), the run is terminated. During the entire run, the position of the center of mass of the swarm is tracked in order to produce a representative trajectory for the motion of the swarm as a whole in each run. The constants used for simulation are shown in Table 1.

Table 1. Table showing all constants used in the modified swarmtaxis algorithm.

Constants							
	t_{ping}	t_{refac}	s_r	β	swarm size	$avoid_{illum}$	$avoid_{shadow}$
Value	1	3	2.5	0.1	21	0.7	0.4
Units	ticks	ticks	p[1]	p[1]	-	p[1]	p[1]

In order to compare the swarmFSTaxis algorithm with the swarmtaxis algorithm, we have performed 100 runs of each each algorithm. To make the runs comparable, all runs were started from the same point, had the same swarm size, had the same parameters as shown in Table 1 and had the same goal. Also, the parameters used such as sensor ranges of illuminated and shadowed agents, range of directional communication and distance moved during motion behavior were kept the same for all 100 runs. Since the most intuitive comparison parameter is the time taken by each of these algorithms to converge to the goal, we have recorded and plotted this parameter in Figure 5 for the swarmFSTaxis algorithm and the swarmtaxis algorithm. Later, in Section 4, Figure 5 is discussed in detail.

4 Discussion and Conclusion

In the swarmtaxis algorithm [1], if an agent detects that it is connected to a number of agents lesser than the connectivity threshold α, the agent transitions into the "coherence" state in which it attempts to reconnect to the swarm by turning towards the swarm and away from the goal. The disadvantage of such an approach is that it makes the algorithm dependent on the success of communication of all the agents in the swarm. If a ping from some of the agents fail to be sent, transmitted or received, the algorithm executes sub-optimal transitions into the coherence state as it assumes that the swarm connectivity is withering. Evidently, this slows down the convergence of the swarmtaxis algorithm. In a later publication [11], the swarmtaxis algorithm was improved and made more resilient to such failures, however, the state transitions were still based on a poll based count. In contrast to this approach, the swarmFSTaxis algorithm

[1] unit p in Table 1 represents distance unit in Netlogo

(a) (b)

Fig. 4. The starting condition of a typical simulation run of the swarmFSTaxis is shown in Figure 4(a). Figure 4(b) shows a converged run. The green patch (occluded by the white circle in Figure 4(a)) shows the starting point, the white circle shows the center of mass of the swarm and the yellow patch shows the predefined goal. The white trace in Figure 4(b) shows the trajectory of the center of mass of the swarm. The yellow agents are the illuminated agents and the blue ones are the shadowed agents.

uses the direction of the incoming ping instead of a poll based count to keep the swarm coherent. In Figure 5, we see that that out of 100 runs of both algorithms, swarmFSTaxis algorithm is consistently faster than the swarmFSTaxis algorithm. From Figure 5, the mean of the number of iterations to convergence for the swarmFSTaxis and the swarmtaxis algorithms are 6597 and 9540 respectively. Therefore, the swarmFSTaxis algorithm has become about 30% faster than the swarmtaxis algorithm.

In the future, the communication mechanism inspired by slime mold and fireflies may have the potential to replace poll based counts that are common in engineered systems. Further research in this direction can ensure that the full capacity of such a communication mechanism is utilized. Apart from using merely the direction of incoming pings more statistical measures can also be developed according to the needs of the task to be accomplished.

Acknowledgments. This work was supported by EU-H2020 Project no. 640967, subCULTron, funded by the European Unions Horizon 2020 research and innovation programmer under grant agreement No 640967.

References

1. Bjerknes, J.D., Winfield, A., Melhuish, C.: An analysis of emergent taxis in a wireless connected swarm of mobile robots. In: IEEE Swarm Intelligence Symposium. pp. 45–52. IEEE Press, Los Alamitos, CA (2007)
2. Hoff, N.R., Sagoff, A., Wood, R.J., Nagpal, R.: Two foraging algorithms for robot swarms using only local communication. In: Robotics and Biomimetics (ROBIO), 2010 IEEE International Conference on. pp. 123–130. IEEE (2010)

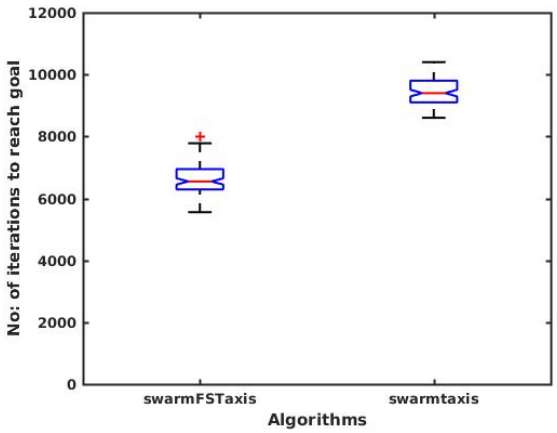

Fig. 5. The box plot shows the number of iterations each algorithm took to converge to the goal. The data from 100 runs of each algorithm is shown in the plot.

3. Moeslinger, C., Schmickl, T., Crailsheim, K.: Emergent flocking with low-end swarm robots. In: Dorigo, M., Birattari, M., Di Caro, G., Doursat, R., Engelbrecht, A., Floreano, D., Gambardella, L., Gro, R., Sahin, E., Sayama, H., Sttzle, T. (eds.) Swarm Intelligence, pp. 424–431. Lecture Notes in Computer Science, Springer Berlin / Heidelberg (2010), http://dx.doi.org/10.1007/978-3-642-15461-4_40
4. Schmickl, T., Crailsheim, K.: Trophallaxis within a robotic swarm: bio-inspired communication among robots in a swarm. Autonomous Robots 25(1-2), 171–188 (aug 2008), http://link.springer.com/10.1007/s10514-007-9073-4
5. Siegert, F., Weijer, C.J.: Three-dimensional scroll waves organize *Dictyostelium slugs*. PNAS 89(14), 6433–6437 (1992)
6. subCULTron: Submarine cultures perform long-term robotic exploration of unconventional environmental niches (2015), http://www.subcultron.eu/
7. Thenius, R., Moser, D., Cherian Varughese, J., Kernbach, S., Kuksin, I., Kernbach, O., Kuksina, E., Miškovi, N., Bogdan, S., Petrovi, T., Babi, A., Boyer, F., Lebastard, V., Bazeille, S., William Ferrari, G., Donati, E., Pelliccia, R., Romano, D., Jansen Van Vuuren, G., Stefanini, C., Morgantin, M., Campo, A., Schmickl, T.: subCULTron -Cultural Development as a Tool in Underwater Robotics Consortium for coordination of research activities concerning the Venice lagoon system. In: Artificial Life and Intelligent Agents. Springer (2016), in print
8. Varughese, J.C., Thenius, R., Wotawa, F., Schmickl, T.: Fstaxis algorithm: Bio-inspired emergent gradient taxis. In: Proceedings of the Fifteenth International Conference on the Synthesis and Simulation of Living Systems. MIT Press (2016)
9. Werger, B.B., Mataric, M.J.: Robotic "food" chains: Externalization of state and program for minimal-agent foraging. In: Proceedings, From Animals to Animats 4, Fourth International Conference on Simulation of Adaptive Behavior (SAB-96). pp. 625–634. MIT Press (1996)
10. Wilensky, U.: Netlogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL (1999), http://ccl.northwestern.edu/netlogo/

11. Winfield, A.F., Nembrini, J.: Emergent swarm morphology control of wireless networked mobile robots. In: Morphogenetic Engineering, pp. 239–271. Springer (2012)
12. Zahadat, P., Schmickl, T.: Division of labor in a swarm of autonomous underwater robots by improved partitioning social inhibition. Adaptive Behavior 24(2), 87–101 (2016)