Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Phototactic Movement of Battery-Powered and Self-Charging Robot Swarms

Ahmad Reza Cheraghi, Fabio Schlösser Vila, Kalman György Graffi

2020

Preprint:

This is an accepted article published in 5th Asia-Pacific Conference on Intelligent Robot Systems (ACIRS 2020). The final authenticated version is available online at: https://doi.org/10.1109/ACIRS49895.2020.9162628
Copyright 2020 IEEE

Phototactic Movement of Battery-Powered and Self-Charging Robot Swarms

Ahmad Reza Cheraghi
Technology of Social Networks
Heinrich Heine University
Düsseldorf, Germany
ahmad.cheraghi@hhu.de
https://tsn.hhu.de

Fabio Schloesser Vila
Technology of Social Networks
Heinrich Heine University
Düsseldorf, Germany
fabio.schloesser.vila@gmail.com
https://tsn.hhu.de

Kalman Graffi
Honda Research Institute Europe GmbH
Offenbach am Main, Germany
kalman.graffi@honda-ri.de
https://www.honda-ri.de

***Abstract*—In biology, phototaxing is one of the many forms of directed motions that organisms can exhibit. If stimulated by light, these organisms can move towards or away from the source. In this paper we present an algorithm that is reliably capable of moving a swarm of virtual robots away from a light source, all while the individual virtual robots are incapable of determining the direction and origin of the light source. The individual agents cannot generate phototactic behavior by themselves, instead this is realized through the overall, emergent behavior of the swarm. This paper explores an algorithm to produce phototactic movement in swarms of virtual robots. The algorithm is developed so that robots in a swarm of any size can execute it in a decentralized fashion, without the need for any globalized control or communication between them. We explore two different sets of virtual robots, one with self-charging and one with battery-powered characteristics. In a rigorous testing and evaluation we find that the swarms behavior and performance with regards to moving away from a light source varies strongly.**

***Keywords*—Robot, Swarms, Nanobots, Phototacting, Phototaxing, Moving**

I. Introduction

Phototaxing is a common biological phenomenon and just one of the many kinds of directed movements that organisms can exhibit. It describes the movement towards or away from a light source and is especially beneficial to organisms that photosynthesize. This process is a subcategory of biological taxis, which is an umbrella term for all stimulated directional movements that occur in nature. For instance, organisms can generate movement in response to stimuli like magnetic fields (magnetotaxis), temperature (thermotaxis) or electricity (electrotaxis). From bacteria, over plankton, to moths, a wide variety of species can exhibit this behavior and achieve it through various means. For example, trochophore larvae (zooplankton) use rudimentary photoreceptive cells that can register light emission. They can adjust themselves to face either towards or away from light, and, using this information, are able to swim towards or away from the source.[1]. Another example are jellyfish use photosensors as an anti-predatory measure, registering when another body casts shadows above them and possibly enabling them to evade the oncoming threat[2].

The phototaxing as described above is usually achieved by a single organism working on its own and through its own capabilities. The plankton does not require the existence of other plankton in order to generate phototactic movement, because it already possesses all the necessary abilities to achieve this by itself. In this thesis, however, we are working with agents that are individually incapable of producing phototactic behavior. These agents can only register light emission, but are unable to determine the direction of the source. Only in a swarm, they are able to collectively move away from a light source. We achieve this in a framework designed for swarm robotics simulation.

II. Related Work

Relatively few works have been published in the field of swarm phototaxing and even fewer are applicable to this thesis as they require different and more complex scenarios that we are not considering. This becomes apparent in the following paragraphs.

One paper achieves different types of taxes in nanomotors [3]. The individual small motors are placed inside a solution called "colloid" [4]. Within this liquid solution, they are able to move and to secrete chemicals, which can stimulate the other motors to move. Externally applied optical, chemical stimuli or magnetic fields cause the motors to show directed movement.

A different paper [5] deals with colloidal motors that respond to the presence of UV light and hydrogen peroxide by exhibiting autonomous and random movement, until they are subjected to a magnetic field. This field stimulates them to align themselves and to show directed movement.

None of these studies aim to achieve phototactic behavior in a "true" or basic sense. While the motors exhibit phototactic behavior, they first need to be placed in a colloidal solution with very specific properties and need to be subjected to multiple stimuli in order to achieve directed motion (UV light, hydrogen peroxide, individual chemical secretion and so on).

Furthermore, these works barely belong in the field of swarm robotics. As it appears, the individual agents are well capable of exhibiting phototactic behavior on their own,

Colloid [4]: "any substance consisting of particles substantially larger than atoms or ordinary molecules but too small to be visible to the unaided eye[...]. Colloidal systems may exist as dispersions of one substance in another—for example, smoke particles in air—or as single materials, such as rubber or the membrane of a biological cell."

without requiring swarm interaction for a global movement pattern to emerge. We study the very basic concept of swarm phototaxing, without the need for all these external enablers. In our simulation, the agents do not require a liquid solution in order to move and are not required to emit any substances. Phototaxing under these more "bare-bones" circumstances has previously been achieved by Arizona State University and the Georgia Institute of Technology. In Phototactic Supersmarticles [6], the authors discuss means to make a group of physical robots move towards a light source, all while individually being incapable of determining the origin and direction of the source.

A further field of mobile, communicating device swarms is given through smartphone mesh networks. While this is not directly related to robot swarms, findings on communication options are interesting. In [7] it is investigated how to use Android devices for local, decentralized device-to-device communication. Through the findings in [8] it is shown how to prioritize the communication in order to increase the message delivery ratio independent of the message routing algorithm. Finally, ideas from [9] can be included to address security issues in the wireless device-to-device communication. In case that these robot swarms might communicate over cellular networks, the comparison of the available bandwidth using this technology is given in [10]. Nevertheless, while communication would bring a huge gain in the efficiency of the phototactic movement, we assume simpler robots without potential to communicate in this paper.

III. Problem Statement

We are addressing a very specific problem. A swarm of virtual robots is faced with a light source. Through the emergent behavior of the collective, these robots are to move away from the light source and over a finishing line. They achieve this while having very limited capabilities and without any sort of centralized control, all while being fully autonomous.

The solution and the individuals have the following limitations and features:

- The solution developed would be compatible with swarms of infinitely large group sizes, as the agents work autonomously.
- Two types of virtual agents: battery-powered and self-charging from light. All agents:
  - Operate using the same algorithm.
  - Are not able to communicate with each other.
  - Do not have any memory.
  - Cannot localize their position.
  - Can register light through a binary sensor, binary meaning that the information the agent receives is either 'light' or 'no light'.
  - Cannot determine the direction or origin of the light source.
  - Can scan their neighborhood. Up to a certain distance, they can see the other agents in their proximity.

Because the light sensor only provides binary information, this means that the individuals cannot measure the strength or quantity of light, thus they cannot possibly determine whether they are moving away from the source or towards it.

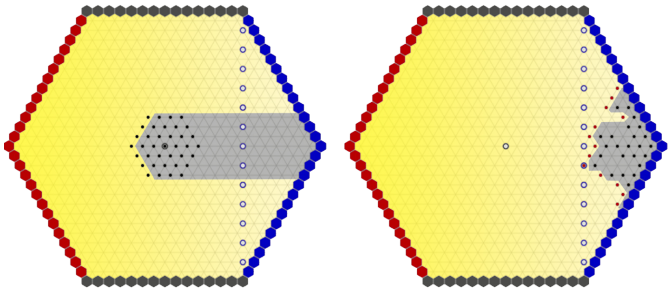


Figure 1: Example for a swarm phototaxing simulation

We will be working with two different sets of virtual robots, self-charging and battery-powered. The battery-powered robots are always operational, meaning they can move at any point, whenever they sense the light source. The self-charging robots need to be exposed to the light source for a longer amount of time before they are able to move.

We want to explore whether an algorithm can make a swarm of these virtual robots move away from a light source and whether there is any difference in behavior between self-charging and battery-powered robots.

In Figure 1, a visual example for the simulation is provided. Without delving too much into the specifics, virtual robots (represented as small dots) are placed onto a two-dimensional simulation space. The light is shone onto these robots from the red towards the blue blocks (West to East), which causes some robots to cast shadows onto other robots. Movement away from the light (which is to say, towards East) can be induced by having the robots that are sensing light move in a random direction. Overall, agents that sense light should become active (meaning they should move), while agents that are in shadow should be inactive, meaning that they should remain in place. In this example, the asymmetry in the agents' activity causes the swarm to exhibit a drift towards East, away from the light source. After a while, the individual robots all have crossed over the finishing line (as indicated by the blue circles).

Our solution requires the individual robots of the swarm to remain in close proximity to one another, as they cannot occlude each other from the light if they spread out too far. This can be seen on the right in Figure 1. None of the swarm's virtual robots become loose from the rest of the swarm and to achieve this we have to model a compression algorithm. This approach is further elaborated on in Chapter IV and the exact specifics of the simulation environment and evaluation methods are outlined in Chapter VI.

IV. Methodology

In the following section, we introduce the phototaxing algorithm. This includes a light propagation, compression, and light sensing function. These functions are described in the following subsections.

A. Light propagation

Light emission is realized through specially prepared tiles. Before each execution of $phototaxing$ for the individual agents, the light emission is recalculated using the below $light_propagation$ algorithm.

Data: list of robots p_{list}, list of tiles $tiles_{list}$
for $tile : tiles_{list}$ **do**
  if *tile is a light emitter* **then**
    read position pos and light direction dir from $tile$'s memory;
    execute Algorithm 2(pos, dir, p_list);
  end
end

Algorithm 1: Light propagation algorithm (Part One)

Data: position pos, direction dir, list of robots p_{list}
from pos, calculate position in the direction dir, call that position pos_2;
if pos_2 *is vacant(no tile or robot) and not out of bounds* **then**
  execute Algorithm 2(pos_2, dir);
else if pos_2 *is occupied by a robot* $P \in p_{list}$ **then**
  notify P that it has registered light;

Algorithm 2: Light propagation algorithm (Part Two)

Algorithm 1 initiates the light propagation. It requires that all light emitting tiles have a memory entry $\{light_emission\ ,\ dir\}$, wherein dir denotes the direction in which the light is supposed to propagate. Algorithm 1 then selects the light emitter tiles that feature the key "$light_emission$" and for those, Algorithm 2 is initiated. From the coordinate of the respective tile, this algorithm recursively follows along the coordinates in direction dir. If it finds a robot on the coordinate, it writes a memory entry $\{light\ ,\ 1\}$ into the robot's memory. If it encounters a tile, it terminates (as this means that the light is blocked by the tile). Finally the algorithm terminates after a certain threshold of x and y coordinate values is reached as otherwise, the light could propagate into infinity or at least violate the boundaries of the simulation space.

B. Compression and graphs

Our solution depends on the virtual robots remaining globally connected to each other. We must ensure that the robots cannot split up into smaller groups or even individuals, so in a way, we employ a weak form of compression here.

Since all of our simulations commence with the swarm already being compressed into a perfect hexagon, we do not have to worry about achieving compression, we are only required to maintain it. In order to maintain this state for the entire swarm on a global level, we only have to ensure that local connectivity is preserved for each robot's move.

This is achieved by scanning and modeling the neighborhood of the moving robots as a graph and determining the connectivity of this graph. As we have previously discussed, the complexity and features of the agents should be kept at a minimum. Ideally, we want to only scan the 1-neighborhood.

However, this would severely limit the movement options for our robots. For example, if the blue robot in fig. 2 scans its 1-neighborhood, it has three options East, West, and Southwest for moving. But, as it appears, a move towards West or Southwest would break up the local connectivity of the neighborhood. This problem can be solved with a 2-radius

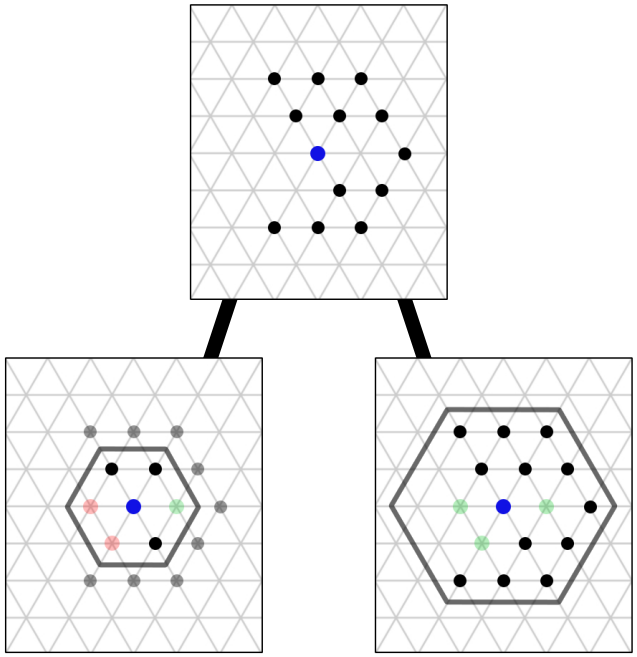


Figure 2: 1- vs. 2-neighborhood scanning

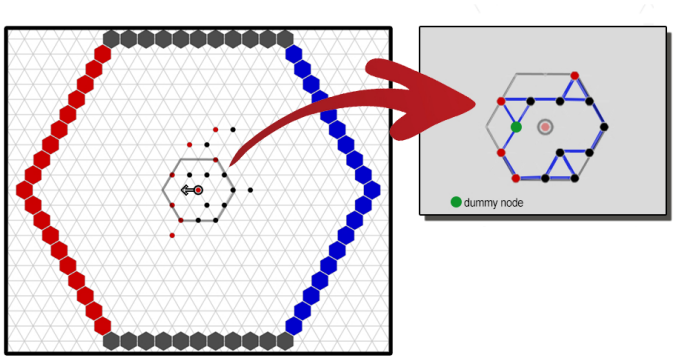


Figure 3: Graph creation for neighborhood of center robot

scan, however, the robot can accurately perceive this swarm as globally connected and make out that it would remain connected eve n after it moves towards West our Southwest.

In figure 3 we make use of 2-radius scanning to construct a neighborhood graph, which is generated as follows. The orange colored robot scans its 2-neighborhood. Next, a graph is constructed, in which all robots are wrapped inside graph nodes and adjacent neighbors are given connecting edges. Afterwards, the robot uses this graph for finding a position to move in which the connectivity between all the other robots will remain. To find this new position the robot insert in each free position a dummy node, which forms edges with its potential neighbors. Finally, a marking algorithm (Depth-First-Search) is run with the dummy node as starting point to check the chain connectivity. The robot will move to that position in that all nodes in the graph have been marked. In case of fig. 3 the robot can either move to East or West but not to Southwest. As a result, this compression algorithm will take care that the robots stay connected after each moving.

```
Data: List of robots p_list, List of tiles t_list
for robot P : p_list do
    light_propogation(p_list, t_list)
    if P has less than 6 neighbors then
        Let current_pos denote P′s current position;
        Choose a neighboring position from the vacant
          positions, call it adjacent_pos;
        if all robots in the 2-neighborhood of P are
          locally connected then
            if P senses light then
                Move P to adjacent_pos;
            else
                P remains at current_pos;
            end
        end
    end
end
```

Algorithm 3: Phototactic: battery-powered robots

```
Data: List of robots p_list, List of tiles t_list
light_propogation(p_list, t_list)
for robot P : p_list do
    if P has less than 6 neighbors then
        Let current_pos denote P′s current position;
        Choose a neighboring position from the vacant
          positions, call it adjacent_pos;
        if all robots in the 2-neighborhood of P are
          locally connected then
            if P senses light then
                Move P to adjacent_pos;
            else
                P remains at current_pos;
            end
        end
    end
end
```

Algorithm 4: Phototactic: self-charging robots

C. Light Recognition and Movement

Finally, we can present our solution towards generating phototactic behavior in a robot swarm. To recap, this algorithm:

- makes the robot choose one random adjacent position to move to.
- is executed by each robot individually, every round.
- requires no communication between the robots.
- only requires the individual robots to scan their 2-neighborhood.

The phototacting algorithm for the battery-powered and self-charging are shown in 3 and 4 The slightly difference between these two algorithm is that the by battery-powered algorithm checks each time with the $light_propogation$ function if all the robots are receiving light. By the self-charging this checking happens at the beginning and all the robots that are receiving light are allowed to move after they got charged.

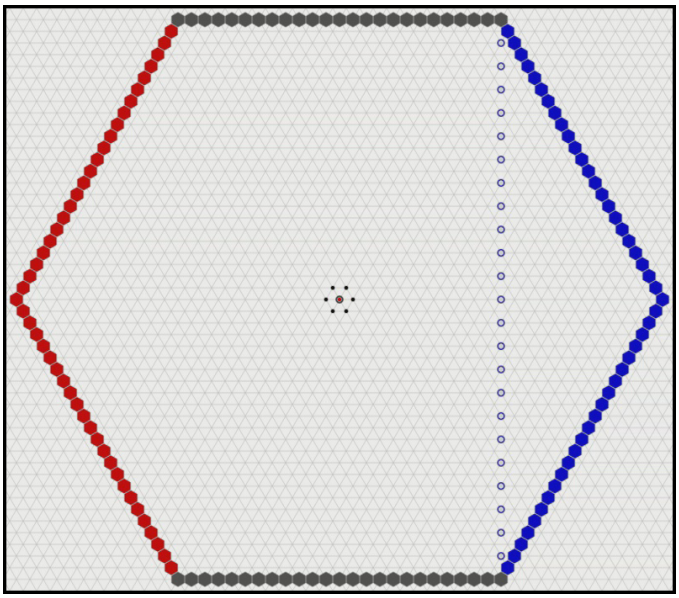


Figure 4: Test environment

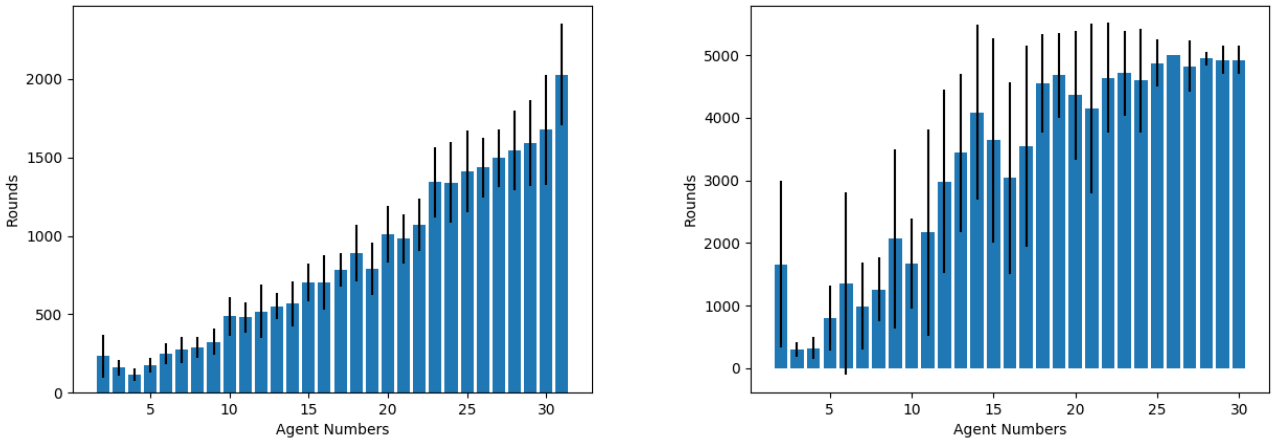
Table I: Success rate for self-charging and battery-powered agents

Self-Charging	Battery-Powered
100%	59.26%

V. Simulation Setup

To measure the differences in behavior between the two types of virtual robots we use the round based simulator Swarm-Sim [11]. This simulator was created at the "Technology of Social Networks Lab" of Heinrich Heine University in Düsseldorf/Germany. It is a 2D and 3D simulation core that provides interfaces for creating individual scenarios and solutions with the main focus of simulating swarms. This simulator is open source and can be downloaded from [12], the phototaxing algorithm is available at [13]. We place a swarm of virtual robots into a simulation space and lay down a finishing line opposite to the light source, which the swarm has to cross over entirely. For the simulation we use a testing environment, as shown in fig 4.

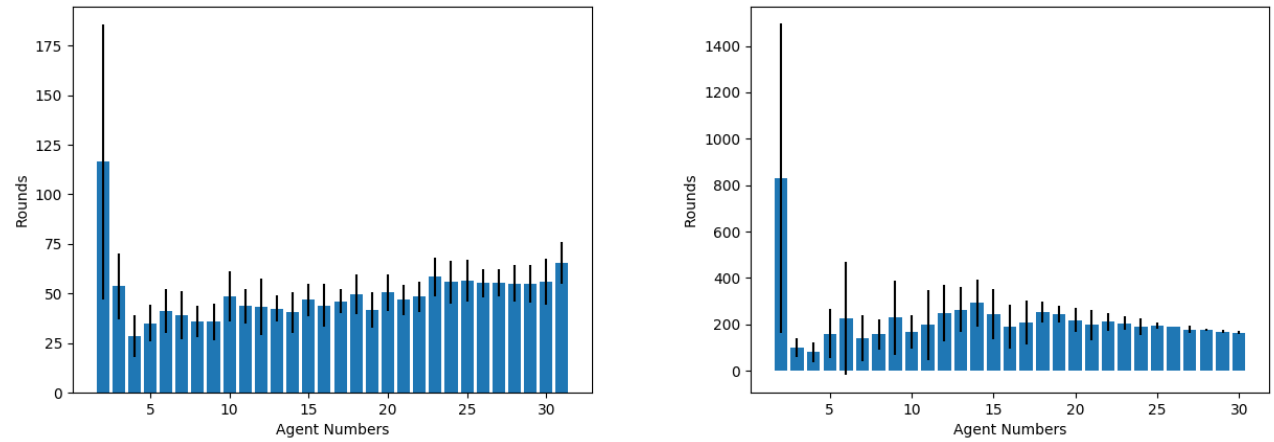
The standardized environment consists of a 24-radius hexagon. Light emitter tiles are placed along the Southwest and Northwest sides of the hexagon, which propagate light towards East. The locations that are placed along the coordinates with $x = 12$ are visual cues to identify the position of the finishing line. The blue tiles serve no other special purpose than to indicate the area that the robot swarm is supposed to move towards. In this environment, the robot swarm is always placed with its center on the $(0, 0)$ coordinate. The swarm-size starts from two and goes up to 30 agents. Since the agents are moving randomly, we use seeds to assure repeatability. For each scenario, we tested 10 seeds overall. The simulation will run for a maximum of 5000 rounds. If within this amount of rounds the entirety of agents of the swarm cross the finishing line, the simulation is marked as successful - otherwise as failed. For the evaluation we use the following five metrics.



(a) Self-charging agents

(b) Battery-powered agents

Figure 5: Average rounds with standard deviation



(a) Self-charging agents

(b) Battery-powered agents

Figure 6: Average rounds per agent with standard deviation

A. Success Rate

Most importantly, we want to be able to tell whether the simulation actually completed within our desired time-frame. For all simulations, we set $round_{max} = 5000$ as the maximum amount of rounds. A test-run is deemed successful if, and only if, all robots of the swarm were able to cross the finishing line within $round_{max}$. For a simulation under a seed $seed = \{s \mid 1 \leq s \leq s\}$ and set of agents $A = \{a \mid 2 \leq a \leq 30\}$, we define $success_{(a,s)} \in \{0, 1\}$ with $a \in A$, $s \in S$, $success_{(a,s)} = 1$ if the simulation completed successfully, and $success_{(a,s)} = 0$ if it terminated unsuccessfully. If we average the success of multiple runs of each robot number with the 10 different seeds for a each tuple this is denoted as

$$success_a = \frac{\sum_{seed=1}^{10} success_{(a,s)}}{|seeds|}$$

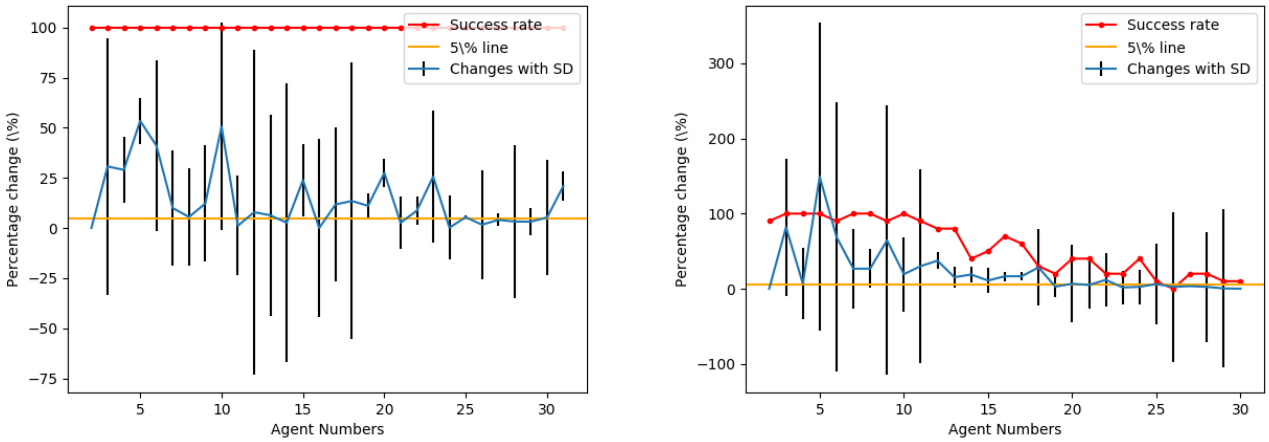
.

B. Average Rounds (AR)

When an agents receives light it is going to move in a randomly chosen direction. If we maintain the seed number, the results will be exactly the same. This assures repeatability. For that reason, for every set of agents 10 simulation runs were conducted with 10 different seeds. Based on that the average amount rounds is used represent the time that the group of agents needs to hit the goal line.

C. Average Rounds Per Agent (ARPA)

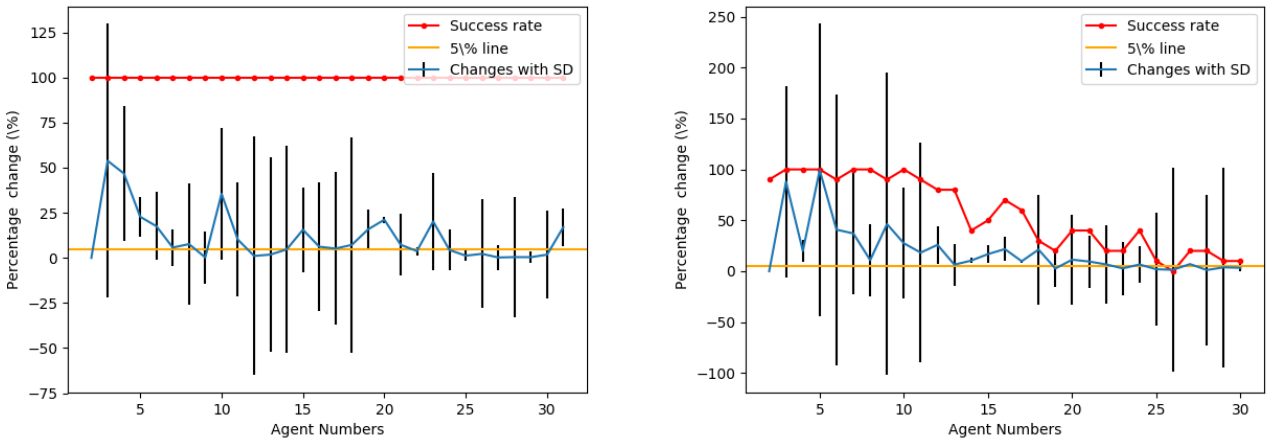
Another interesting metric is the average round per robot. This metric tells us how much rounds in average an agent needs to reach the finish line.



(a) Self-charging agents

(b) Battery-powered agents

Figure 7: Percentage change average rounds with standard deviation



(a) Self-charging agents

(b) Battery-powered agents

Figure 8: Percentage change average rounds per agent with standard deviation

D. Standard Deviation (SD)

Of course, an average estimate of the run-time is not enough for any practical application. While these statistics roughly tell us what to expect from the algorithms, they cannot tell us how far the algorithms can stray from those estimates. For instance, if an algorithm were to result in an average run-time of 1000 rounds, this could still mean that one run reached the goal in 200 rounds and another run in 5000 rounds. Thus, we use the standard deviation to tell how large the variation of the taken averages is.

E. Percentage Change Average Rounds(PCAR)

The normal average rounds are telling us how long each set of swarm needs to reach to finishing line. However, it does not tell us if there is a difference between the different swarm sizes. Therefore, the percentage changes average rounds metric is used, to tell us how much the difference average round in percentage is between the actual and previous swarm set.

F. Percentage Change Average Rounds Per Agent (PCARPA)

The percentage change average rounds per agent (PCARPA) is telling us how the changes between the average rounds of each agent differ between the actual and the previous agents set.

VI. Evaluation

After establishing our testing environment and evaluation metrics, we subject our virtual agents to a number of phototaxing simulations. We are gradually increasing the swarm size and measure how many rounds the agents require to traverse the finishing line. We conducted these tests for both the self-charging and battery-powered agents and repeat all scenarios a total of ten times in order to calculate an average results. We detail our findings below.

The results in Figure 5 show that, in all of our tests, the self-charging agents is able to complete the task given within

fewer rounds and with a smaller margin of error. Because we set the round limit at 5000, the battery-powered agents are sometimes not able to complete the task at all. Only 59.26% of tests for the battery-powered agents are complete, that is why the average rounds might appear to converge towards 5000. Generally, as the swarms grew in size, they require more round time to complete our simulation.

The data in Figures 7 and 8 suggests that for both agent variants, the percentage of change decreases for larger quantities. Disregarding the standard deviation, the percentage change for swarm sizes of 25 individuals and upwards is shown to remain below the five-percent line for both self-charging and battery-powered agents. However, once again the results for the battery-powered agents are likely affected by the round limit.

In Figure 6, we can observe that this average round time roughly scales along the amount of agents that we place in the simulation. The battery-powered agents even suggest that for more agents, the average rounds per agent decrease. However, this is likely due to our round limit. As they can't pass beyond 5000 rounds, the ARPA value will naturally decrease as more agents are added.

From this preliminary data, we find that self-charging agents are generally better at performing the phototaxing task. Especially the average round time per agent for swarm sizes of over 23 might suggest a linear relation between round time and swarm size for the self-charging testing subjects. Especially the results for smaller agent groups go to show that the asymmetry in movement decisions (between agents that sense light and those that do not) is responsible for producing the behavior that we have observed. Our testing data suggests that for more individual agents (which also means more agents in shadow), the ability of the swarm as a whole to achieve phototaxing is greatly increased. When more agents are in shadow, the implicit coordination of the swarm appears to improve.

VII. Conclusion

During testing, we found that the self-charging robots performed much better with regards to AR and the time that it took them to traverse a specific distance. We evaluated the algorithm for both the battery-powered and self-charging robots and did so in constellations ranging from 2 robots up to 30 robots. Every one of these tests was conducted ten times, each one with a different randomizer seed to assure repeatability. In general, our self-charging virtual robots achieved their goal within much fewer rounds and with a considerably lower standard deviation.

Obviously, work on this topic is not nearing completion anytime soon. For us to make bolder predictions on the behavior of the swarms and the effectiveness of the algorithm for achieving phototaxing, the solution needs to be trialed on larger quantities of agents.

The algorithm has also not been challenged by any difficult terrain. Our simulation space offers a wide and open area over which the agents can freely move across. What would happen if we were to place tiles (which is to say, objects that block positions), within the swarm or along the path that leads away from the light source? If the obstacle is large enough, the swarm might get stuck along that structure and be rendered immobile.

Additionally, there are a multitude of options for modifying the agents beyond what we have tested here. This specific solution works with and only requires agents with a minimal set of abilities. If one were to add a memory of sorts to these agents, they might be able to 'remember' their overall trajectory of movement and thus interpolate the direction of the light source.

A potential improvement in the performance might be gained through adding the capability of communication to the robots. Through this, they could communicate which direction might be promising and coordinate their actions. We have investigated various swarm problems, such as coating, marking or ant optimization, that rely on local observations and local communications. Results using the communication ability are highly encouraging.

Acknowledgments

We would like to thank the Arab-German Young Academy of Sciences and Humanities for funding this research as well as our collaboration partner Dr. Ahmed Khalil for the valuable discussions.

References

[1] G. Jékely, J. Colombelli, H. Hausen, K. Guy, E. Stelzer, F. Nédélec, and D. Arendt, "Mechanism of phototaxis in marine zooplankton," *Nature*, vol. 456, no. 7220, p. 395, 2008.

[2] T. Katsuki and R. J. Greenspan, "Jellyfish nervous systems," *Current Biology*, vol. 23, no. 14, pp. R592–R594, 2013.

[3] A. Sen, M. Ibele, Y. Hong, and D. Velegol, "Chemo and phototactic nano/microbots," *Faraday Discuss.*, vol. 143, pp. 15–27, 2009. [Online]. Available: http://dx.doi.org/10.1039/B900971J

[4] "Encyclopædia britannica," Mar 2019. [Online]. Available: https://www.britannica.com/science/colloid

[5] N. Chaturvedi, Y. Hong, A. Sen, and D. Velegol, "Magnetic enhancement of phototaxing catalytic motors," *Langmuir*, vol. 26, no. 9, pp. 6308–6313, 2010.

[6] S. Cannon, J. J. Daymude, W. Savoie, R. Warkentin, S. Li, D. I. Goldman, D. Randall, and A. W. Richa, "Phototactic supersmarticles," *arXiv preprint arXiv:1711.01327*, 2017.

[7] A. Ippisch and K. Graffi, "Infrastructure Mode Based Opportunistic Networks on Android Devices," in *Proceedings of the IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, 2017, pp. 454–461.

[8] S. Sati, C. Probst, and K. Graffi, "Analysis of Buffer Management Policies for Opportunistic Networks," in *Proceedings of the 25th International Conference on Computer Communication and Networks (ICCCN)*, 2016, pp. 1–8.

[9] P. S. Mogre, K. Graffi, M. Hollick, and R. Steinmetz, "A security framework for wireless mesh networks," *Wireless Communications and Mobile Computing*, vol. 11, no. 3, pp. 371–391, 2011.

[10] N. Goebel, R. Bialon, M. Mauve, and K. Graffi, "Coupled simulation of mobile cellular networks, road traffic and v2x applications using traces," in *2016 IEEE International Conference on Communications (ICC)*, 2016, pp. 1–7.

[11] A. R. Cheraghi, K. Actun, S. Shahzad, and K. Graffi, "Swarm-sim: A 2d & 3d simulation core for swarm agents," in *3rd Int. Conf. of Intelligent Robotic and Control Engineering (IRCE 20)*, 2020.

[12] A. R. Cheraghi and U. Duesseldorf. (2020) Swarm simulator. [Online]. Available: https://gitlab.cs.uni-duesseldorf.de/cheraghi/swarm-sim

[13] A. R. Cheraghi and F. Schloesser Vila. (2020) Phototaxing algorithm. [Online]. Available: https://gitlab.cs.uni-duesseldorf.de/cheraghi/swarm-sim/-/tree/phototaxing