2018 Annual American Control Conference (ACC)
June 27–29, 2018. Wisconsin Center, Milwaukee, USA

Clustered Optimization of a Small-Scale Robot Swarm with Minimal On-Board Sensing*

Jinhong Qu, Xinlu (Hermione) Li, and Kenn R. Oldham, *Member, ASME*

Abstract— **This paper explores the ability of a swarm of small-scale robots to optimize robot locomotion speed if only extremely limited on-board sensing capability is granted for each individual robot. This is done to assess benefits of swarm optimization versus individual control. The optimization of simulated robot locomotion speed is evaluated under multiple classical and machine learning algorithms. Furthermore, a new algorithm, Clustered Particle Swarm Optimization (CPSO), is developed based on existing optimization and clustering algorithms. The algorithms are based on an experimentally-validated nominal dynamic model for small-scale robots, with environmental parameters extended over a wider range of uncertainty. The nonlinear relationship between multiple actuation parameters and robot performance, specifically robot speed in this work, encourages a numerical method to search for a set of actuation parameters that leads to a higher robot speed. Three algorithms are presented and compared, including Simulated Annealing (SA), Particle Swarm Optimization (PSO), and CPSO. The comparison between SA and PSO demonstrates the utility of optimizing robots in a swarm. By increasing the number of robots, the optimizing iteration number can be reduced. The comparison of simulation of PSO and CPSO shows a 23% improvement with minimal sensing assumed from the robots. The single sensing feature, selected manually using phase plots, is the average foot height at a certain actuation phase in each cycle.**

I. Introduction

Micro-fabrication technology has the potential to enable small-scale robots, several centimeters in length at most, to accomplish various missions, as for rescue, exploration, and defense, as a swarm of tens or even hundreds of individuals. Some previous researches focused on the sensing, control and communication within a small-scale robot swarm [1, 2] to achieve a better robot performance. However, when robots are scaled down to sub-centimeter dimensions, robot payloads limit their on-board sensing and control capability. Therefore, an open question is whether off-board sensing results are sufficient for control and optimization purpose. Using several optimization algorithms, this work explores this question for a very minimal addition of on-board sensing in a simple task of locomotion speed optimization.

Various micro-robot designs have been proposed and validated [3-7]. Among these designs, a centimeter-scale walking robot [8] is used as the sample platform to study the possible utility of swarm optimization with or without on-board sensing. This centimeter-scale hexapod robot (Fig. 1) integrates six lead zirconate titanate (PZT) strips in total to actuate each robot foot bi-directionally. The supporting structure of the robot is 3D printed with polylactic acid (PLA). Limited fabrication resolution causes variance in robot parameters such as beam properties of robot legs and resulting resonance modes. Different terrain conditions lead to even more difference between robots' motion. The dynamic model of this robot [8] is well developed and validated with the capability to predict the dynamic performance over a modest set of environmental (terrain) conditions, so the model can help evaluate the robot performance in a swarm instead of building tens of robots. It is possible to estimate robot velocity and leg motion by integrating the influences from piezoelectric actuation, rigid robot chassis, multiple-mode dynamics of the robot legs and nonlinear foot-terrain interaction. Based on this centimeter-scale robot model, another model has been modified to explain the dynamics of millimeter-scale robots [9]. Therefore, any findings and conclusions from this work have the potential to be further extended to even smaller robots.

Three optimization algorithms are applied to examine the importance of swarm optimization and on-board sensing: Simulated Annealing (SA), Particle Swarm Optimization (PSO), and clustered Particle Swarm Optimization (CPSO). Simulated annealing is a heuristic optimization algorithm that searches for the global optima by mimicking the annealing process [10, 11]. SA takes a small step in each iteration. This small step is accepted if it provides an improvement or is possibly accepted if no improvement. By tuning a few parameters in SA, a selected robot can search the actuation parameters associated with its global optimum performance.

Particle Swarm Optimization [12, 13] optimizes the performance of a robot in a swarm with swarm and individual information, specifically the global optimum within the swarm and the global optima of individual robots. In practice, a swarm of microrobots will likely encounter different terrain conditions while their own performance will vary based on fabrication resolution, so each "particle", or microrobot, is not identical to the others. The diversity in the robot parameters hurts the performance of PSO because this algorithm is working based on the assumption of identical particles. Limited efforts have been done in clustered PSO [14] because the majority of PSO applications target the global optimum or the optimum of the overall swarm.

Clustered Particle Swarm Optimization considers the diversity of microrobot parameter and/or terrain perturbations; individuals experiencing apparently similar circumstances may to be clustered together to seek an improvement in optimizing time and performance. The clustering process depends on not only the robot performance but also on-board sensing features that indicate the similarity between robots' responses to their environment. With hierarchical clustering [15], the robots in a swarm are clustered into small groups by

*Research supported by NSF Foundation.

J. Qu, H. Li, and K.R. Oldham are with the University of Michigan, Ann Arbor, MI 48105 USA (734-353-2760; e-mail: jinhongq@umich.edu).

calculating the weighted distance between robots/groups. Each iteration combines a pair of closest robots/groups into one group. The distance is calculated with speed and sensing features of the robots.

The comparison between SA and PSO demonstrate the reduction of convergence time using swarm information; and the comparison between PSO and CPSO illustrates an improvement from the existence of on-board sensing in a minimal form.

This paper is organized as follows: Sec II introduces the centimeter-scale robot design and dynamic model. Sec III presents some preliminary results from classical algorithms and phase plots of robot motion. Sec IV explains the details of machine learning algorithms included in this work: their flow, derivation, and modification for small-scale robots. Sec V discusses the results when applying algorithms to the dynamic model of robots. Sec VI discusses future and provides concluding remarks.

II. Robot Design and Dynamic Model

A. Robot Architecture

The robot simulated [8] in this paper is a simple small-scale (<10 cm) walking robot with 3 pairs of compliant legs connected with a rigid body, as shown in Fig. 1. The robot frame is 3D printed with PLA. 6 bimorph PZT actuators (T219-A4Cl-103X) are epoxied to each PLA leg with intentional misalignment. The misalignment between PLA and PZT enables robot foot to generate coupled motion in two directions. 6 legs are actuated in a tripod gait. This motion overcomes the gravity of robot and friction with surface underneath, to move robot forward.

This architecture is inspired by a type of silicon-based millimeter-scale robot (2-5 mm in dimension) [6]. The millimeter-scale robot has two thin-film PZT actuators for each leg, to generate foot motion in two directions. Its chassis is made from silicon with parylene-C, again a polymer material, for elastic mechanisms. Accounting for scaling, both robot architectures perform similarly in a reasonable range of operating condition.

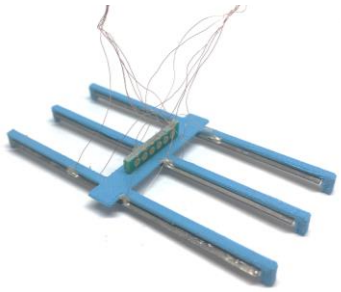

Figure 1 Photo of a 3D printed PLA hexapod robot integrating PZT actuators.

B. Dynamic Model

The dynamic model used in this paper includes robot body, leg and contact dynamics. This model is capable of replicating robot dynamics measured experimentally. Both robot forward motion and robot leg vertical motion from experiments and dynamic model are compared to validate this model.

The robot body is modeled as a rigid body with multiple degrees of freedom. Unrealistic side walking is excluded in this model, though this walking is observed in some other robot architectures.

Leg dynamics of the small-scale robot are explained with a simplified cantilever beam with complex boundary condition. The robot foot is modeled as a tip mass at the end of a cantilever beam, namely the robot leg. The mass of each robot leg is not negligible because the mass of PZT is comparable to other parts of robot foot. The actuation force from PZT actuators is modeled as distributed force along the beam (robot leg). The impact force calculated from contact dynamics is modeled as tip force. Based on this model, multiple resonance modes are calculated to determine the optimal operation frequency range and the distribution of external force into each mode.

Contact dynamics consider the foot-terrain interaction in two directions. A simple model with coefficient of friction and restitution is enough for centimeter scale, though more complex nonlinearities are observed with a millimeter-scale robot. Full details of the dynamics model are provided in [8] and [6].

III. Preliminary Results

A. Classical Approach

Some preliminary simulations are generated with the dynamic model to demonstrate common behavior. The robot speed trend with respect to frequency and duty cycle of a square wave input is shown in Fig. 2. From this simulation, the relationship between robot speed and actuation parameters does not support an analytical method for robot speed optimization, for reasons such as the impact events that occur at foot touchdown. Therefore, seeking for a numerical solution with suitable computational complexity becomes important for small-scale robot optimization.

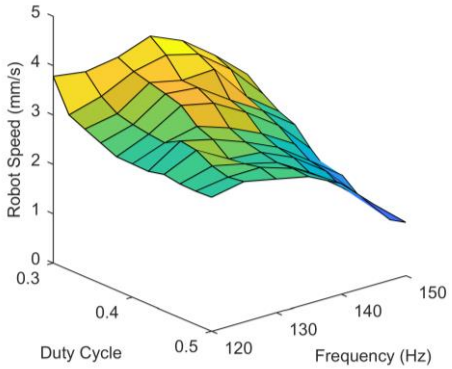

Figure 2. Simulated speed trend with respect to frequency and duty cycle of actuation signal. The relation between speed and actuation parameters tends to be non-derivative.

A sampling of intuitive algorithms has also been tried and are reported here. Fig. 3 shows the schema of an "Individual Algorithm", optimizing a swarm of several robots under random external perturbation. An external observer is assumed to be able to measure the speed of the robot locomotion, but to have no other information about on-robot dynamics. The Individual Algorithm acts to update the individual robot actuation parameters with the parameters and

the speed in previous iterations of control inputs, with the principle equation being:

$$d_i^{k+1} = d_i^k + \alpha * \frac{v_i^k - v_i^{k-1}}{d_i^k - d_i^{k-1}} + n_i^k \quad (1)$$

in which d_i^k and v_i^k denote the input parameter (d) and the speed (v) at the k-th iteration of the i-th robot, and α and n_i^k are the fixed scaled factor for perturbation and the randomized noise matrix, respectively, both empirically determined. The value of α and n_i^k are different for each input parameter; from testing, $\alpha = 10$ and $|n_i^k| \leq 0.1$ for frequency and $\alpha = 0.01$ and $|n_i^k| \leq 0.1\%$ for duty cycle.

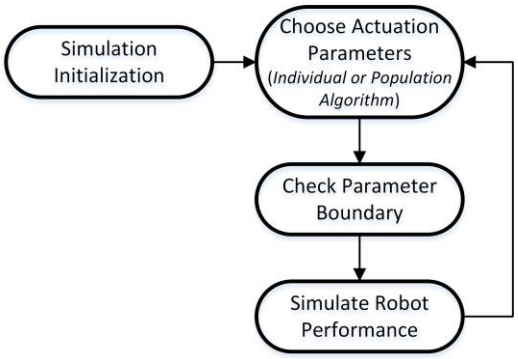

Figure 3 Schematic of the Individual and Population Algorithm for robot speed optimization

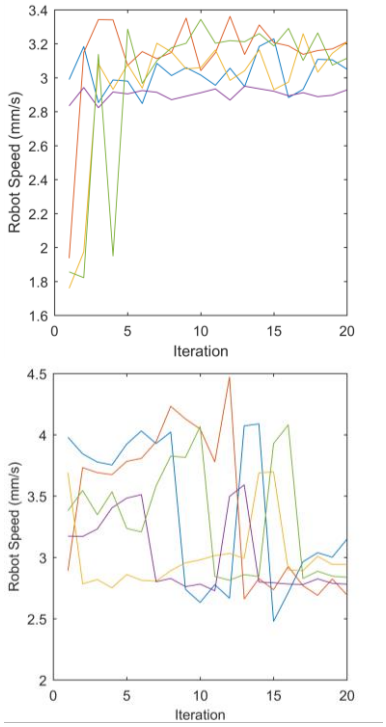

Figure 4 Result of speed optimization with single parameter (top) and two parameters (bottom) from the Individual Algorithm, where the five different trajectories represent five different robots

Fig. 4 shows the results of the Individual Algorithm on five randomly generated robots. Single-parameter updates and two-parameter updates are presented. The Individual Algorithm is not a stable optimization if multiple actuation parameters are included, requiring a more sophisticated multi-variable search than is presented here. Also, the robot speed is not stabilized at a high value compared to what will be seen using more coordinated optimization algorithms.

The second coordination method in the absence of on-robot data is then denoted as the "Population Algorithm." The schematic of the Population Algorithm is structured similarly to "Individual Algorithm", as demonstrated in Fig. 3. In this method, the information from all robots in the swarm has an influence on the single robot parameter update. Specifically, the average speed and actuation parameters of all robots are used in the update principle. The principle equation is derived to be:

$$d_i^{k+1} = d_i^k + \alpha * \frac{\bar{v}^k - v_i^k}{\bar{d}^k - d_i^k} + n_i^k \quad (2)$$

where $\bar{d}^k$ and $\bar{v}^k$ denote the average input parameter ($\bar{d}$) and the average speed ($\bar{v}$) of all robots at the k-th iteration. The value of α and n_i^k remain the same as discussed in the Individual Algorithm. Fig. 5 shows the result from performing the Population Algorithm of speed optimization. The speed of robots is slightly better after 20 iterations.

$$d_i^{k+1} = d_i^k + \alpha * \frac{\bar{v}^k - v_i^k}{\bar{d}^k - d_i^k} + n_i^k \quad (3)$$

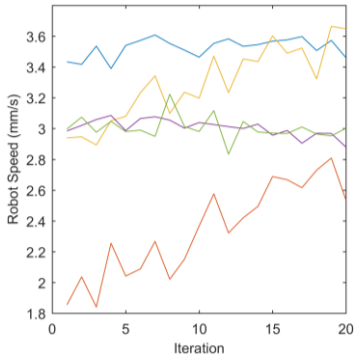

Figure 5 Result of speed optimization with two parameters from the Population Algorithm, where the five different trajectories represent five different robots .

B. *Features for Clustering*

Because of the limitations of two intuitive algorithms, machine learning algorithms are proposed, using data clustering based on simple measurements hypothesized from on-board robot sensing. One simplest piece of information to extract from a micro-robot is deformation of the actuator (i.e., vertical deformation of a robot leg in the simulated robots), as it requires no additional structures to be fabricated at the micro-scale level. Nonetheless, active sensing still incurs significant power or complexity costs, rising with sensing frequency[16, 17], so decision making with a small number of measurements per step is highly desirable.

Data clustering is a technique of combining similar data points together based on specific and distinctive features they share. Data clustering requires a candidate feature that contains three parameters: the measure itself (i.e. displacement or velocity), an angle from 0° to 360° reflecting its timing over one actuation cycle of robot leg motion, and the number of the leg it is associated with. One challenge is to select the feature for the data clustering algorithms. In this work, phase plots serve as a visual guide for selecting candidate features.

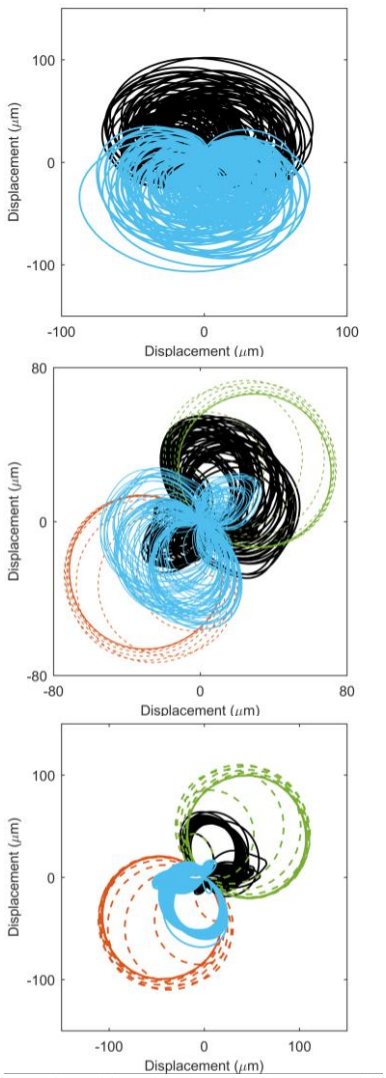

Figure 6. Phase plot of the robot foot displacement with respect to the body in the z-direction, when the robot lateral speed is slow (top), medium (middle) and fast (bottom); each number on the legend chart represent one leg on the robot with some leg motion overlapping each other on the graph.

Candidate sensing features were determined as follows. First, robot speed was divided into three manually defined categories: slow (< 2.00 mm/s), medium, and fast (> 5.00 mm/s). Then, an ideal candidate feature would produce distinctive phase plot patterns for each robot speed categories. It was found after numerous simulations that there was a rather regular correlation between the leg motion depicted on any phase plots and the robot speed: as the robot speed increases, the robot leg motion becomes less variable from cycle to cycle. Representative candidate features can be velocity and/or displacement of the foot. Some representative phase plots of foot displacement relative to the body are presented in Fig. 6.

IV. Algorithms

After a literature review of several machine learning algorithms, Simulated Annealing (SA) and Particle Swarm Optimization (PSO) were selected for proof-of-concept evaluation in this small-scale robot application. SA searches a global optimum, or a local optimum close enough to the global ones. This algorithm shows gives an indication of how much time a heuristic algorithm needs to search for a numerical solution. This will help to assess the utility of swarm optimization and additional sensor information. PSO is selected for its capability to optimize a swarm of objects. Other evolutionary algorithms can also be a potential candidate for robot application, with learning rate or evolution rate a possible concern.

A. Simulated Annealing (SA)

SA is adapted for small-scale robot application in this work as follows. This algorithm includes tunable starting and stop temperature (T_i and T_f), and cooling rate (δ_T). The principle is:

$$\begin{aligned}&Let\ d_i^k = d_i^0; T = T_i; k = 1\\&While\ T < T_f:\\&\quad d_i^k = random\ within\ range(d_i)\\&\quad if\ v(d_i^k) > v(d_i^{k-1})\ or\ P\left(v(d_i^k), v(d_i^{k-1})\right) > rand(1): k = k + 1;\\&\quad T = \delta_T T;\end{aligned}$$

in which k is the iteration number; $v(d_i^k)$ is the speed simulated from d_i^k, which is defined as in the Individual Algorithm; $rand(1)$is a random function returning a value within the range of 0 to 1. $P\left(v(d_i^k), v(d_i^{k-1})\right)$ is a probability function returning a value from 0 to 1. A conventional way to define this function is with the exponential function (*exp()*):

$$P\left(v(d_i^k), v(d_i^{k-1})\right) = exp\left(\left(v(d_i^k) - v(d_i^{k-1})\right)/T\right) \quad (4)$$

B. Particle Swarm Optimization (PSO)

PSO is also adapted for small-scale robot application as follows. The schematic for PSO is demonstrated in Fig. 7. The concept of this algorithm is to update each robot with its currently estimated optimum and the global optimum of all robots within its swarm in each step. The update equations are:

$$\begin{aligned}vd_i^{k+1} &= w_v vd_i^k + w_{sw} rand(1)\left(v^{sw} - v_i^k\right)\\&\quad + w_{sg} rand(1)\left(v_i - v_i^k\right)\\d_i^{k+1} &= d_i^k + vd_i^{k+1}\end{aligned} \quad (5)$$

in which w_{sw}, w_{sg}, and w_v are the weight of swarm global optimal speed, of single robot optimal speed, and parameter approaching speed, respectively; v^{sw} and v_i are the best speed of the swarm and *i*-th robot; vd_i^k is the approaching speed of parameter of the *i*-th robot at the *k*-th iteration.

C. Clustered Particle Swarm Optimization (CPSO)

CPSO, as shown in Fig. 8, is developed from standard PSO. After clustering robots into multiple groups, the group optima are also used to approach optimum of each robot. Within the first iteration, robots in a swarm are clustered into various groups with normalized speed ($v_{n,i}$) and their sensing feature value ($p_{n,i}$). Robots with similar performance are clustered by calculating the virtual distance ($l_{i,j}$) between each two robots (i and j). The virtual distance is calculated from:

$$l_{i,j} = \sqrt{w_{sp}\left(v_{n,i} - v_{n,j}\right)^2 + w_{ss}\left(p_{n,i} - p_{n,j}\right)^2} \quad (6)$$

in which w_{sp} and w_{ss} are the weight of normalized robot speed and normalized sensing feature value. The maximum number of robots in one group (max_g) is set as a constraint. The update equations are similar to the PSO, but further include the group optimum:

$$v_{d,i}^{k+1} = w_v v_{d,i}^k + w_{sw} rand(1)\left(v^{sw} - v_i^k\right) + w_{sg} rand(1)\left(v_i - v_i^k\right) + + w_g rand(1)\left(v_j - v_i^k\right)$$
$$d_i^{k+1} = d_i^k + v_{d,i}^{k+1} \quad (7)$$

in which most parameters are defined by the PSO techniques. Here, w_g is the weight of group optimal speed; v_j is the best speed of the *j*-th group that includes *i*-th robot; $v_{d,i}^k$ is the parametric speed of the *i*-th robot at the *k*-th step.

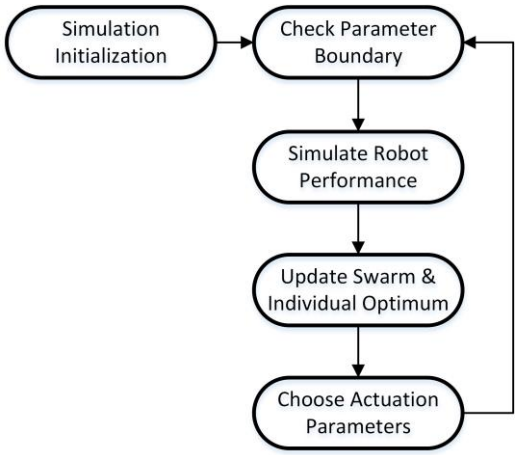

Figure 7 Schematic of the Particle Swarm Optimization for robot speed optimization

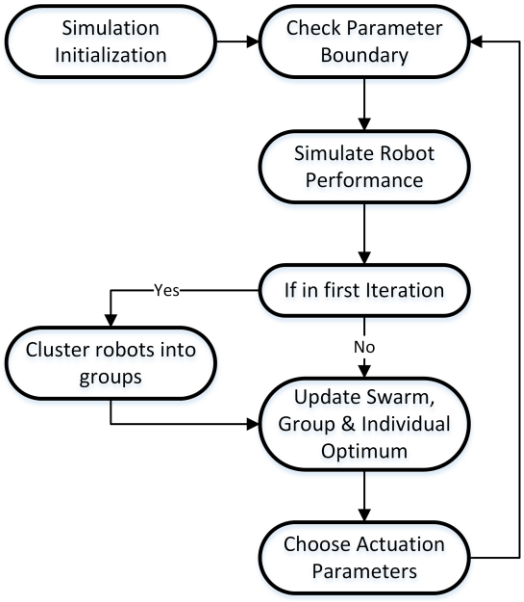

Figure 8 Schematic of the Clustered Particle Swarm Optimization for robot speed optimization

V. Results

The simulation of the three algorithms is generated after carefully selecting the algorithm parameters. Parameters in simulations, shown in TABLE I, are not all optimized, though they still assist to indicate the performance and potential of the algorithms. The results from SA and PSO illustrate the advantages and drawbacks of algorithms for a swarm of robots or a single robot within the swarm. The simulation with and without clustering is intended to examine the importance of on-board sensing, even being at a minimal level.

TABLE I. The optimization parameters, the ranges of actuation and terrain parameters selected for simulation

Algorithm	Parameters/Ranges	Value
SA	T_i	1
	T_f	0.01
	δ_T	0.9
PSO	w_{sw}	0.4
	w_{sg}	0.2
	w_v	0.1
CPSO	w_{sp}	1
	w_{ss}	1
	w_g	0.8
	max_g	10

A. Single vs. Swarm Optimization

Fig. 9 compares the simulation of one identical robot with single optimization or swarm optimization. The single robot optimization with SA runs 100 small steps, though only 20 steps are accepted due to the acceptance criteria. This increases the computation time, as well as testing complexity for real robots. Overall, a single robot optimization can approach a global optimum of robot actuation parameters, but a much longer time is traded off for this algorithm.

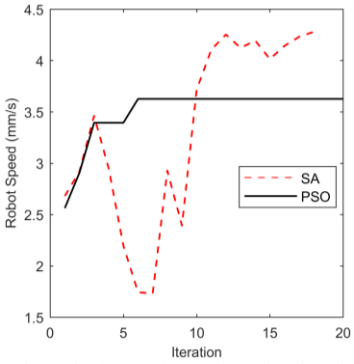

Figure 9 speed trend of one robot, using Simulated Annealing with 20 updates from 100 small steps and Particle Swarm Optimization within a swarm of 50 robots and 20 iterations.

PSO, as an example of swarm algorithms, generates stable results with around 6 iterations for a robot within a swarm of 50 robots. The steady-state result from this algorithm has smaller speed than that from SA. This difference may possibly be compensated in certain degrees with better parameters selection, because the parameters so far are selected within a reasonable range without further searching. Overall, swarm algorithm significantly reduces the time required to reach an equilibrium parameter selection, though at a cost of reduced average performance.

B. *Clustering with On-board Sensing*

To examine the utility of limited on-board sensing, comparison between PSO and CPSO identifies the robot performance difference caused by clustering robots with minimized on-board sensing. Fig. 10 compares the performance of PSO and CPSO with identical initial condition. A swarm of 50 robots is applied with PSO, the average speed of the robots is plotted against iteration up to 50 (Fig. 10). After 10 iterations, the average speed is almost stabilized around 3.55 mm/s. Converging speed is close to the performance of fewer robots in the last section.

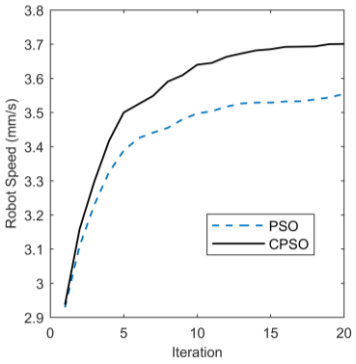

Figure 10. Average speed of 200 robots as a swarm, using Particle Swarm Optimization (PSO, blue dash line) and Clustered Particle Swarm Optimization (CPSO, black solid line);

CPSO optimizes the robot performance with hierarchical clustering. The clustering process only exists in the first iteration, with robot speed and robot sensing. After around 10 iterations, the robot speed optimized with CPSO is stabilized to 3.7 mm/s, turning out to be an improvement of 23% over PSO.

VI. Discussions and Conclusions

It appears that even minimal on-board sensing can help improve robot swarm performance based on the simulation results. Several conventional algorithms are compared in this work, along with a clustered optimization algorithm (CPSO), proposed in this work. PSO has a much better time-wise performance by trading off a much faster convergence rate for up to 30% reduction in average, as a proof-of-concepts for the relationship between individual optimization and swarm optimization. CPSO, with clustering, performs 6% better than the conventional PSO in this proof-of-concept scenario. Notably, CPSO improves the algorithm performance without trading off any time complexity. As observed from simulation, PSO performance gets worse if the diversity of robots increases, which emphasize the importance of applying clustering algorithm with on-board sensing features. In addition, the potential correlation between robot speed and selected sensing feature can weaken the urgency of external sensing. Also, the performance of swarm optimization, with and without clustering, can be further improved with the clustering algorithm. Worth attention here is that the parameters of swarm optimization algorithm, PSO and CPSO, are not optimized, so the algorithm performance may be further improved.

Some future works have the potential to improve the optimization algorithms. Instead of clustering at the first iteration only, further clustering can be executed every few iterations of optimization. This modification is related to clustering performance by repeating the process. The minimized sensing capability is observed to have an influence on generating robot actuation parameters for better robot performance. However, it is still unclear that how much improvement can be reached by adding more sensing capability.

Acknowledgment

The authors thank Dr. Yi Chen for assistance with machine learning algorithms.

References

[1] H. Woern, M. Szymanski, and J. Seyfried, "The I-SWARM project," in *Robot and Human Interactive Communication, 2006. The 15th IEEE International Symposium on*, 2006, pp. 492-496.

[2] M. Dorigo and M. Birattari, "Swarm intelligence," *Scholarpedia,* vol. 2, p. 1462, 2007.

[3] S. Bergbreiter and K. S. Pister, "Design of an autonomous jumping microrobot," in *Robotics and Automation, 2007 IEEE International Conference on*, 2007, pp. 447-453.

[4] R. S. Pierre, D. Vogtmann, and S. Bergbreiter, "Model-Based Insights on the Design of a Hexapod Magnetic Walker," in *Experimental Robotics*, 2016, pp. 715-727.

[5] R. J. Wood, "The first takeoff of a biologically inspired at-scale robotic insect," *Robotics, IEEE Transactions on,* vol. 24, pp. 341-347, 2008.

[6] J. Choi, M. Shin, R. Q. Rudy, C. Kao, J. S. Pulskamp, R. G. Polcawich, *et al.*, "Thin-film piezoelectric and high-aspect ratio polymer leg mechanisms for millimeter-scale robotics," *International Journal of Intelligent Robotics and Applications,* vol. 1, pp. 180-194, 2017.

[7] S. Hollar, A. Flynn, C. Bellew, and K. Pister, "Solar powered 10 mg silicon robot," in *Micro Electro Mechanical Systems, 2003. MEMS-03 Kyoto. IEEE The Sixteenth Annual International Conference on*, 2003, pp. 706-711.

[8] J. Qu, C. B. Teeple, and K. R. Oldham, "Modeling Legged Microrobot Locomotion Based on Contact Dynamics and Vibration in Multiple Modes and Axes," *Journal of Vibration and Acoustics,* vol. 139, p. 031013, 2017.

[9] J. Qu, J. Choi, and K. Oldham, "Dynamic Structural and Contact Modeling for a Silicon Hexapod Microrobot," *Journal of Mechanisms and Robotics,* 2017.

[10] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *science,* vol. 220, pp. 671-680, 1983.

[11] P. J. Van Laarhoven and E. H. Aarts, "Simulated annealing," in *Simulated annealing: Theory and applications*, ed: Springer, 1987, pp. 7-15.

[12] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, 1998, pp. 69-73.

[13] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of machine learning*, ed: Springer, 2011, pp. 760-766.

[14] J. Kennedy, "Stereotyping: Improving particle swarm performance with cluster analysis," in *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, 2000, pp. 1507-1512.

[15] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika,* vol. 32, pp. 241-254, 1967.

[16] B. Edamana, B. Hahn, J. S. Pulskamp, R. G. Polcawich, and K. Oldham, "Modeling and Optimal Low-Power On–Off Control of Thin-Film Piezoelectric Rotational Actuators," *IEEE/ASME Transactions on mechatronics,* vol. 16, pp. 884-896, 2011.

[17] B. Hahn and K. R. Oldham, "A Model-Free ON–OFF Iterative Adaptive Controller Based on Stochastic Approximation," *IEEE Transactions on Control Systems Technology,* vol. 20, pp. 196-204, 2012.