Scalability of robot swarms when applied to maze solving

Mr. Thomas Campbell
Institute of Engineering
Murray State University
Murray, KY 42071
tcampbell12@murraystate.edu

Dr. James M. Hereford
Institute of Engineering
Murray State University
Murray, KY 42071
jhereford@murraystate.edu

Abstract— **Swarm robotics offers an intriguing approach to many real-world engineering problems. Such tasks that require covering a large search space or that involve working in potentially hazardous environments naturally lend themselves to robotic and/or swarm solutions. This research examines the effect of changing swarm size on the performance of several different maze-solving algorithms, including random movement, obstacle avoidance, and wall-following algorithms. The choice of algorithms was limited to ones that (a) do not require direct bot-to-bot communication, (b) use bots with a limited sensing range and (c) do not require the bots to know their position. Two different robots were used in the experimental trials: the HEXBUG Larva and the e-puck mobile robot. Trials were conducted with two, four, eight, and (with the Larva) sixteen bots. Results are presented for each of the trials as well as discussion on swarm performance as related to swarm size and bot intelligence. The results show that a small swarm that is more sophisticated or intelligent can perform as well as a larger swarm made up of less intelligent bots.**

Keywords— swarm robotics; scalability; maze solving; Markov model; e-puck

I. Introduction

A swarm approach to robotics is one that emphasizes many simple robots instead of a single complex robot. In swarm robotics, each robot is relatively simple and, potentially, inexpensive. Robot swarms have high fault tolerance, so they will perform well if some of the individual units malfunction or are destroyed. Swarms are also scalable, so the size of the swarm can be increased or decreased as needed.

This research looks into capabilities of a robot swarm when the individual bots have little or no processing capability. The less processing power in a swarm robot, then potentially the smaller, less power-intensive and cheaper it can be. However, is there a limit to what a swarm of "dumb" robots with little processing power can do? In other words, can a bunch of dumb, independent robots do anything constructive or do the individual robots need to have some minimum decision-making capability (i.e., intelligence)? Also, can a lack of processing capability be compensated for by increasing the number of bots in the swarm?

An important characteristic of swarms is their scalability. Scalability refers to how a swarm's behavior changes with changing numbers of bots. For example, a swarm using a particular algorithm may perform far better if the size of the swarm were to double or triple. A different swarm, however, may not be changed much at all if its size were to increase. The first swarm in this example would be considered very scalable, the second swarm being less so.

The issue of swarm scalability and performance arises in many different applications. For example, hazardous environments, such as collapsed buildings or chemical leaks, that are unsafe for humans naturally lead to robotic and/or swarm solutions [5]. In addition, the fault tolerant aspects of swarm systems are well suited for remote applications such as extra-planetary exploration where repairs cannot be made easily [8].

The question of swarm scalability is important as bot sizes decrease such as in the field of nanorobotics [4]. Potential applications for nanorobotics in medicine include targeted drug-delivery for cancer and monitoring of diabetes. In such plans, medical nanotechnology is expected to employ nanorobots injected into the patient to perform work at a cellular level [1]. Such nanorobots intended for use in medicine should be non-evolving at the bot level, since changing bot characteristics would lead to unpredictable behavior, reduce reliability and interfere with the medical mission. In addition, the individual nanobots would not have position information, though each bot could detect neighbor bots. The nanobots may or may not have communication capability and, most likely, the nanobots would have very little processing capability due to the small size and low power requirements. Thus, the nanobots would only be effective as part of a swarm configuration.

To investigate the issue of bot processing power and swarm scalability we did maze solving experiments with robot swarms. This paper presents results from those experiments that address the following two research objectives: (a) To determine how the performance of the swarm improves as more bots are added to the swarm, as well as the extent of that improvement; and (b) As the maze solving algorithm is made more complex (and, hopefully, improved), to determine how the maze-solving performance improves when compared to simpler algorithms. In other words, we are looking at how the scalability of the swarm improves as the intelligence or processing capability of the individual bots increases.

We varied several parameters during the maze-solving experiments to investigate what impacts swarm performance. We used swarms with two different robots (one programmable and one not); we used three different maze solving algorithms; and we used several different swarm sizes with each robot type. The bots used in the swarms have different sizes and different movement characteristics.

The paper is organized as follows. Section II gives some background information and describes the maze-solving algorithms used in the experiments. Section III describes the experiment and gives the experimental results and analysis, and Section IV gives the conclusions.

II. Research Plan

A. Overall Goals

The goal of this research is to investigate swarm behavior as (a) bot processing capabilities improve and (b) number of bots increases. One can imagine a graph with success as the vertical axis and processing capability or number of bots as the horizontal axis. Figure 1 shows an example plot where the multiple curves correspond to swarms made up of bots with different capabilities [7]. All of the curves start at the origin; if there are no bots then nothing can be accomplished so there is no success.

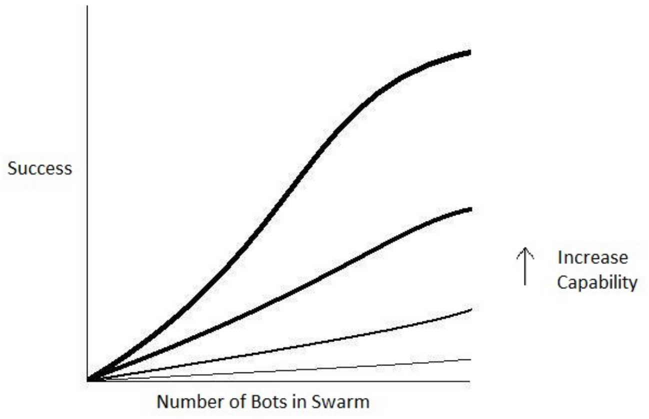

Fig. 1. Example plot showing how success of a robot swarm is presumed to increase as the number of bots in swarm increases and the capability of each bot increases.

We assume that success increases as the number of bots increases. We can envision scenarios when there are too many bots which leads to congestion and confusion, so the curve(s) will level off or even decline. However, knowing when success starts to decrease is valuable information in determining the correct swarm size. The curves in Fig. 1 also assume a nonlinear improvement in performance as bots are added to the swarm, which often occurs with swarm algorithms [6].

Clearly, success depends on the task being done by the swarm such as search, mapping, surveillance, or route planning. We chose maze solving as our task so in this paper success is defined as how fast (in terms of seconds) the bots exit the maze.

B. Specific Plan

To investigate the shape of the success vs number of bots curve, we did experiments with swarms with two different types of bots [3]. The first bot is the “dumb” robot. We considered two possible robots: the Hexbug Nano and the Hexbug Larva. The Nano has no intelligence other than it cannot move through obstacles. The motion of an individual Nano bot can be modified by adjusting its mechanical properties: adding weight, shortening the legs, adding weight to one side to change the balance. The larva (see Fig 2) is similar except it has an infrared (IR) sensor that detects obstacles, backs up, and turns away from them. We used the larva because it gives more flexibility, even though it costs more and displays a minute amount of intelligence so it’s not completely “dumb”. To get a baseline for what happens when no-intelligence robots interact, we disabled the IR sensor on the larva for some experiments.

Fig. 2. Photograph of Larva bot.

The second robot is the e-puck robot (see Fig. 3). The e-puck has a dsPIC microcontroller for programming and multiple sensors (infrared, accelerometer, sound, camera). We used the e-pucks to investigate what happens as we add intelligence to the individual bots in the swarm. Because of the price of the e-pucks we are limited in how big the swarm can be and thus limited in scalability investigations.

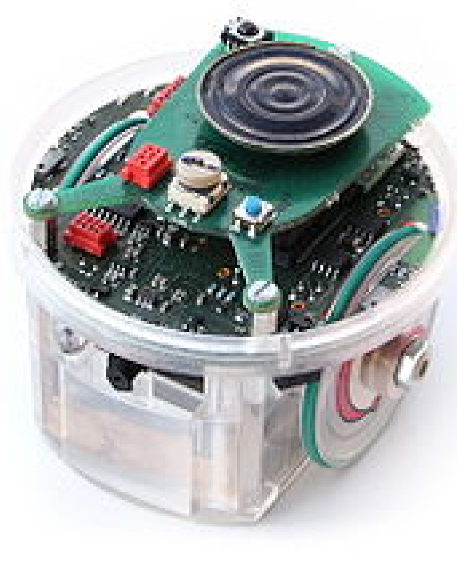

Fig. 3. Photograph of E-puck robot.

We set up a 2D perfect maze. A perfect maze has one entrance, one exit, no inaccessible regions and only one path between any two points. We performed test cases with the two types of robots. We monitored the number of bots that exit the maze versus time.

For the larva and e-puck bots, we used swarm sizes of 2, 4, and 8 bots. For each test case, we started the bots simultaneously (or at least close to simultaneously) at the maze entrance and tracked the time when bots leave the maze. Our hypothesis is that as the swarm size increases, then the time for the bots to exit the maze will be reduced. This hypothesis is based on the assumption that individual bot interactions will increase and thus lead to more bots finding the exit.

The e-puck and larva bots have different mechanical characteristics (see Table I). For the initial experiments we programmed the e-pucks to have the same processing capability as the larva. That is, the e-pucks were programmed to back up and turn when they encountered an obstacle. These experiments allowed us to compare whether bots with the same intelligence give comparable results. It also allowed us to correlate the performance of the e-pucks and larva. The e-puck, since it is programmable, was used to investigate the impact of increasing processing capability.

COMPARISON OF E-PUCK AND LARVA BOTS

Characteristic	E-puck	Larva
Size	7 cm diam 5 cm tall	10 cm long 3 cm wide 2 cm tall
Speed	10 cm/sec	≈ 14 cm/sec
Sensing	8 IR sensors	1 IR sensor
Programmability	Programmable via Bluetooth interface	Not programmable
Movement	Straight line motion, any turn possible	Moves primarily in large circles, only turns to the rights @ obstacles

C. Maze solving

The Larva and e-puck robots have limited sensing range and no direct bot-bot communication. For a single bot, it is analogous to a blind person trying to find their way out of a maze. Being blind means that the searching entity (person or bot) cannot identify where it is in the maze – no global information - and has no memory of what locations or cells it has already visited. Similarly, it cannot look far ahead to determine if the exit is "up there". The search entity can merely feel/sense objects directly in front of it or to its sides.

A blind person could follow one of several strategies to find the exit to a maze. For example, he could move forward with his hands out to ensure that he does not run into a wall; if he encounters an obstacle, then he could turn in a random direction or turn right by a random amount and continue moving forward. Another strategy is to follow a wall. Conceptually, the person holds out their (right) hand to the side and moves forward while keeping their hand on the wall. If the maze exit is not in the center (an assumption we make) then they will eventually find the exit. If the person becomes hysterical, they could possibly just move randomly - even in the presence of obstacles.

The blind-person-in-the-maze scenarios are analogous to the three maze solving algorithms we used with the robot swarm. We only considered algorithms that could be implemented by very simple bots and were easily scalable to large numbers. Thus the choice of algorithms was limited to ones that (a) do not require direct bot-to-bot communication, (b) use bots with a limited sensing range, and (d) do not require the bots to know their position.

The first algorithm is basic obstacle avoidance. Each bot moves forward until it encounters an obstacle (either a wall or another bot). When it encounters an obstacle, the bot turns a random amount away from the wall and moves forward again. (See flowchart in figure 4.) We implement this algorithm with both the larva and e-puck robots. The Larva bots were not programmable and could only turn right. The e-puck bots were programmed to turn either right or left at random.

```
Move forward;
While (not out of maze)
{
    if (obstacle in front)
        Turn randomly ( 90-180 degrees);
    else
        Move forward;
}
```

Fig. 4. Pseudo-code for obstacle-avoidance algorithm.

Another maze-solving algorithm is the wall-following algorithm (figure 5). In this algorithm, each individual bot travels parallel to a wall. The e-pucks in the swarm were programmed so that half followed the left wall and half followed the right wall. This was done to reduce a potential time bias when the exit is located closer to the entrance by following the right or left wall. Wall – following is computationally more complex than the obstacle-avoidance algorithm, so we expect it to lead to faster maze solving.

```
Move forward till obstacle detected;
While (not out of maze)
{
    if (obstacle in front)
        Turn left;
    else if (no obstacle on right)
        Turn right;
    else
        Move forward;
}
```

Fig. 5. Pseudo-code for wall-following algorithm.

We also implemented a random movement algorithm. This corresponds loosely to the hysterical blind person in the maze. We implemented the random movement algorithm with larva bots with no active sensors. Each bot would move randomly about the search space. Sometimes it would bump into an obstacle and get re-routed and sometimes it would get stuck in a corner. This algorithm gives us a performance baseline for comparison with the other algorithms as it represents a completely dumb (no intelligence) approach to maze solving.

III. Experimental methods and Results

A. Setup

For the trials, we constructed a simple maze with one entrance, one exit, and two dead ends (see figure 7). The width of the corridors was scaled appropriately according to the sensor range of either the e-pucks (20 cm width) or the larva (30 cm width). In all trials, all of the bots were assembled at the entrance of the maze and released simultaneously, with the entrance then being covered to prevent the bots escaping the maze. Times were recorded in seconds for each of the bots to find the exit. For certain algorithms, some bots would never find the exit, so after an allotted time, the trials were concluded even if bots remained in the maze.

Each algorithm on each kind of bot was run five times in the maze with each swarm size. The larva were available in a swarm as large as 16 bots, while the e-pucks were limited to swarms of eight. The tables of data that follow list the times for each bot in the swarm to leave the maze. Five trials were conducted for each test, and averages and standard deviations are provided.

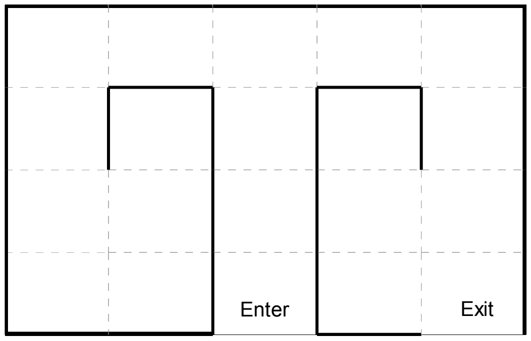

Fig. 7. Layout of maze used in experiments

B. Zero-Intelligence

As a control, we ran trials with the larva in groups of 2, 4, 8, and 16 with disabled IR sensors to make them "blind." This renders them completely unresponsive to obstacles, meaning that they will simply attempt to move forward no matter what. This lack of input or decision-making makes these trials a zero-intelligence point to which all other trials can be compared.

In these trials, less than half of the bots even made it out of the maze; in fact, in several trials, none of the bots reached the exit. This is due to the fact that most of the bots would get stuck in a corner almost immediately, and since they would not turn around, they would stay in said corner indefinitely. The results for the sixteen-bot swarm trials are given in Table II below, where an 'X' indicates that the bot never reached the maze's exit.

TABLE. II. Sixteen larva bots in Swarm. Random Movement. Time given in seconds. Fifth through sixteenth omitted as none reached the exit.

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Avg
First	15	17	27	17	18	18.8
2nd	19	139	39	18	X	53.8
3rd	47	X	52	18	X	39.0
4th	48	X	X	X	X	

C. Obstacle Avoidance

Both the e-pucks and the larva bots were run in the maze in groups of two, four, and eight using obstacle avoidance algorithms. Data is given for the exit times of both the e-pucks and the larva for each of the two, four, and eight bot swarms in the following tables (III – V).

TABLE. III. Two bots in swarm. Obstacle avoidance. Time for bots to exit maze (seconds)

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Avg	SD
epuck 1	43	28	32	159	58	64	54.4
epuck 2	148	>180	>180	>180	>180		
larva 1	46	96	20	56	16	46.8	32.3
larva 2	> 180	168	25	> 180	21		

TABLE. IV. Four bots in swarm. Obstacle avoidance. Time for bots to exit maze (seconds)

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Avg	SD
epuck 1	25	37	79	40	40	44.2	20.4
epuck 2	26	44	155	133	43	80.2	59.2
epuck 3	36	49	173	182	50	98	72.9
epuck 4	97	>180	188	>180	160		
larva 1	154	55	24	25	28	57.2	55.6
larva 2	220	79	34	56	41	86	76.9
larva 3	>240	79	54	93	88		
larva 4	>240	160	72	>180	148		

TABLE. V. EIGHT BOTS IN SWARM. OBSTACLE AVOIDANCE. TIME FOR BOTS TO EXIT MAZE (SECONDS)

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Avg	SD
epuck 1	29	20	46	21	37	30.6	11.0
epuck 2	36	26	71	47	49	45.8	16.8
epuck 3	42	90	75	85	70	72.4	18.7
epuck 4	56	95	90	151	97	97.8	34.1
epuck 5	97	133	110	183	227	150	54.1
epuck 6	200	>240	229	>240	>240		
epuck 7	>240	>240	>240	>240	>240		
epuck 8	>240	>240	>240	>240	>240		
larva 1	26	25	36	34	27	29.6	5.0
larva 2	38	58	37	76	40	49.8	17.0
larva 3	88	64	48	85	129	82.8	30.5
larva 4	140	69	59	91	135	98.8	37.2
larva 5	183	75	75	103	184	124	55.5
larva 6	213	110	122	117	186	149.6	46.7
larva 7	>240	175	205	162	>240		
larva 8	>240	>240	>240	>240	>240		

Comparing the e-puck and larva results for obstacle-avoidance side-by-side, we find that the results are remarkably similar, especially for the larger swarm sizes. Looking at the eight-bot swarms, the average time for the first e-puck to exit the maze is about 30.6 seconds, while the first larva finished in 29.6 seconds, only one second of difference. The time for half the e-pucks to finish was 97.8 seconds compared to 98.8 for the larva. Indeed, the results of the two swarms seem to correlate quite well with each other despite their differences in physical construction.

When looking at the performance of the swarm with regard to changing the size, we can see a few trends: As the size of the swarm increases, the time for the first bot to find the exit tends to decrease. Also, as swarm size increases, the average time for half of the bots to leave the maze tends to increase.

D. Wall-Following

The Wall-Following trials were conducted with the e-pucks in swarms of two, four, and eight. In each trial, half of the bots were following the right wall and half were followed the left. Tables VI through VIII show the exit times for the bots in each size swarm. We implemented the Wall-Following algorithm with the e-pucks only, since the larva bots do not have the sensors or processing capability to perform wall following.

TABLE. VI. TWO BOTS IN SWARM. WALL-FOLLOWER. TIME FOR BOTS TO EXIT MAZE (SECONDS)

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Avg	SD
epuck 1	34	45	35	36	30	36.0	5.5
epuck 2	60	64	61	58	62	61.0	2.2

TABLE. VII. FOUR BOTS IN SWARM. WALL-FOLLOWER. TIME FOR BOTS TO EXIT MAZE (SECONDS)

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Avg	SD
epuck 1	35	32	35	40	31	34.6	3.5
epuck 2	36	42	47	42	34	40.2	5.2
epuck 3	64	66	65	65	57	63.4	3.6
epuck 4	104	74	68	79	61	77.2	16.4

TABLE. VIII. EIGHT BOTS IN SWARM. WALL-FOLLOWER. TIME FOR BOTS TO EXIT MAZE (SECONDS)

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Avg	SD
epuck 1	31	28	28	28	32	29.4	1.9
epuck 2	33	35	32	37	34	34.2	1.9
epuck 3	40	39	35	40	37	38.2	2.2
epuck 4	42	44	40	63	49	47.6	9.2
epuck 5	47	50	57	74	60	57.6	10.5
epuck 6	69	58	61	94	66	69.6	14.3
epuck 7	78	65	63	107	74	77.4	17.7
epuck 8	96	68	X	121	103	97.0	22.0

Compared to the other algorithms, the wall-follower is much more consistent. This is due to the fact that there is no random decision-making involved with this algorithm. Because of this, the e-pucks took roughly the same path in each trial, and so there is very little variation between runs. This can be seen in the standard deviations, some of which are less than 5.0.

E. Theoretical Comparison

To help address the research question of does performance improve as swarm size improves, we calculated the expected theoretical exit times for the bots assuming that there is no interaction among the bots in the swarm. The theoretical times are calculated by modeling the exit of the bots as a continuous-time Markov chain.

For the analysis, we track or count the number of bots that are in the maze. We modeled the number of bots in the maze by a birth/death Markov chain with an arrival or birth rate of λ and a departure or death rate of μ.

We assume that the bots start at the maze entrance and then implement the obstacle avoidance algorithm. Over time, bots will leave the maze which will be defined as a "death" and the number of bots in the maze will be reduced by one. There are no births (new bot arrivals) so this is a pure death process.

It can be shown [2] that the death process can be modeled by a Poisson process. Thus, the probability that there are M bots in the maze at t seconds is given by

$$p(n = M) = \frac{e^{-\mu t}(\mu t)^M}{M!}\ ; n \geq 0\ ; t \geq 0 \qquad (1)$$

Another way to think of the Poisson process is that the probability of one death within a short time segment h is given by

$$\mu h e^{-\lambda h} = \mu h + o(h) \qquad (2)$$

where

$$\lim_{h\to 0} \frac{o(h)}{h} = 0 \qquad (3)$$

In this theoretical model, the future state of the model (number of bots in the maze) is based only on the present state. For example, if there are three bots in the maze (state = 3), all that matters is that there are currently three bots and it does not matter how long it took to get to this state.

The time it takes for a birth/death process to reach a particular state for the first time is called the sojourn time or first passage time. Suppose that the number of bots in the maze is initially a; we want to find the time it takes for the bot size to become b. The first passage time is denoted by $T_{a,b}$:

$$T_{a,b} = T_{a,a-1} + T_{a-1,a-2} + T_{a-2,a-3} + \ldots + T_{b+1,b} \qquad (4)$$

The expected time to go from a to b is thus (5)

$$E\{T_{a,b}\} = E\{T_{a,a-1}\} + E\{T_{a-1,a-2}\} + E\{T_{a-2,a-3}\} + \ldots + \{T_{b,b=1}\}$$

For a Poisson process, the probability density function (pdf) for the interevent times is given by an exponential distribution. For the pure death process the pdf is denoted by $f_i(t)$ which equals

$$f_i(t) = \mu_i e^{-\mu_i t}, \qquad (6)$$

where μ_i is the death rate while in state i. Since each interevent time, $T_{i,i-1}$ is an exponential process,

$$E\{ T_{i,i-1}\} = E\{ \mu_i e^{-\mu_i t}\} = 1/\mu_i. \qquad (7)$$

Thus, the expected time to go from state a to b is

$$E\{T_{a,b}\} = 1/\mu_a + 1/\mu_{a-1} + \ldots + 1/\mu_{b+1.} \qquad (8)$$

To repeat, we model the number of bots in the maze as a Markov chain and the change in the number of bots as a death process. We can thus calculate the passage time to go from, say, state 4 (4 bots in the maze) to state 2:

$$E\{T_{4,2}\} = 1/\mu_4 + 1/\mu_{3.} \qquad (9)$$

We now estimate the death rate for each state. One way to model the death rates is to assume that the death rate when there are 8 bots in the maze is the same as when there is only one bot in the maze. In this case, $\mu_i = \mu$ for all states. So to go from 4 -> 2 would be 2 μ and to go from 4 ->0 would be 4 μ. This leads to very high expectations for first passage times.

Instead, we assume that the death rate is proportional to population size:

$$\mu_i = \mu i. \qquad (10)$$

The expected transit time (passage time) becomes

$$E\{T_{a,b}\} = 1/\mu(1/a + 1/(a-1) + \ldots + 1/(b+1)), \qquad (11)$$

where μ is the death rate for one bot in the maze.

We estimated μ for both the epuck and Larva bots by measuring how long it takes one bot to exit the maze using the obstacle avoidance algorithm. We did 10 trials using one bot and calculated the average (e-puck average = 153 sec and larva average = 168 sec). Unlike other birth/death processes there is a physical limit to how fast the bots can leave the maze. For the e-puck, it would take 18 sec to exit the maze if it traveled at its maximum velocity and went straight to the exit. The fastest larva transit time is 14 sec.

To account for the minimum travel time, we included the travel time with the first transition time $T_{a,a-1}$ and then subtracted the minimum from the average to calculate the death rate for state 1. Thus, the death rate μ for the two bots is

E-puck: μ = 1/135 (1/seconds)

Larva: μ = 1/165 (1/seconds)

We compare the theoretical results based on our model with the actual experimental results from the obstacle avoidance algorithm in table 9 (larva bot) and 10 (e-puck bot). The model predicts the exit times for successive bots assuming that they are completely independent (no interaction among the bots in the swarm). In the tables, Bot1 refers to the time for the first bot to leave the maze, Bot2 refers to the time for the second bot, etc.

TABLE. IX. COMPARISON OF EXPERIMENTAL AND THEORETICAL TIMES FOR SUCCESSIVE BOTS TO LEAVE MAZE FOR OBSTACLE AVOIDANCE ALGORITHM (LARVA BOT)

Swarm size	**Time for bot to leave maze (sec)**					
8	Bot1	Bot2	Bot3	Bot4	Bot5	Bot6
Experimental	29.6	49.8	82.8	98.9	124	149.6
Markov model	34.6	58.2	85.7	119	152	193
4						
Experimental	57.2	86	111	>240		
Markov model	55.3	110	193	358		
2						
Experimental	46.8	>180				
Markov model	96.5	262				

TABLE. X. COMPARISON OF EXPERIMENTAL AND THEORETICAL TIMES FOR SUCCESSIVE BOTS TO LEAVE MAZE (EPUCK BOT)

Swarm size	**Time for bot to leave maze (sec)**					
8	Bot1	Bot2	Bot3	Bot4	Bot5	Bot6
Experimental	30.6	45.8	72.4	97.8	150	>240
Markov model	18.8	40.2	65.2	95.2	133	183
4						
Experimental	44.2	80.2	98	>240		
Markov model	37.5	87.5	163	313		
2						
Experimental	64	>180				
Markov model	75	225				

From comparing the theoretical and experimental results, we see that the swarm based results are uniformly less than the model results, implying that using a swarm does lead faster exit times. Even passive interactions among the bots (turn away before collisions, no two bots can be in the same place) leads to faster exit times than what would be predicted from bots acting independently.

IV. CONCLUSIONS

When looking for a swarm solution to an engineering problem, one important question to arise is "how much?" Certainly, more bots in a swarm tend to increase the swarm's overall efficiency and/or speed, but for obvious reasons, one cannot have an unlimited number of bots to use. Also, at some point, adding more bots to the swarm can suffer from diminishing returns, such that, for example, doubling the swarm size from ten to twenty might increase performance by a big margin, but doubling again from 20 to 40 might "saturate" the workspace and end up not being worth the additional investment. So to question now becomes, "How many bots do I need so that my task is accomplished efficiently, without spending extra money on superfluous bots?" This number of bots we can call the "saturation point."

Looking at the present research, we can see that if the objective is for the swarm to find something in a large unknown search space, such as a maze, there is a natural limit to the swarm's effectiveness: an object cannot be found faster than the time it takes for a bot to travel to that object in a direct path from the starting location. From the obstacle-avoidance e-puck trials, we see that the fastest time that any bot ever left the maze was 20 seconds. With a swarm size of only two bots, the average time for the exit to first be found was 64 seconds, far from its potential. With four bots, the exit was found in only 44 seconds. With eight bots, 30 seconds. For 16 bots, the theoretical model predicts 26 seconds for the first bot to leave the maze which is not a significant improvement over the eight-bot swarm – only 13% - but would certainly be a significant difference in cost. Indeed, we seem near to the saturation point of our maze with a swarm of only eight.

If we look at the results for the wall-following algorithm, we see that saturation seems to happen with even less bots. Finding the exit is accomplished in an average of 29 seconds with eight bots in the swarm, but with a swarm of only two, the exit is still found in only 36 seconds, far better than the obstacle-avoidance's 64 seconds.

Interestingly, below the saturation point, the wall-following and obstacle-avoidance algorithms are, performance-wise, not alike at all, but at saturation, the average time for either group of bots to first find the exit is very similar. This indicates that a small swarm that is more sophisticated or intelligent can perform as well as a larger swarm made up of less intelligent bots. Essentially, a smarter swarm is able to "do more with less." Less bots in a swarm is preferable, as it means saving money and other resources.

ACKNOWLEDGMENT

The authors would like to thank the financial support of the Committee on Institutional Studies and Research and the Institute of Engineering at Murray State University. We would also like to thank Jeremiah Markham for help with the experiments.

REFERENCES

[1] G. Al-Hudhud, "On swarming medical nanobots", Inernational Journal of Bio-Science and Bio-Technology, Vol. 4, No. 1, pp. 75-88, March 2012.

[2] A. O. Allen, Probability, Statistics, and Queueing Theory with Computer Science Applications, Academic Press, 1978.

[3] T. Balch, "Hierarchic social entropy: An information theoretic measure of robot group diversity", Autonomous Robots, vol. 8, pp. 209-237, 2000.

[4] S. Hauert, S. N. Bhatia, "Mechanisms of cooperation in cancer nanomedicine: toward systems nanotechnology", Trends in Biotechnology, vol. 32, pp. 448 – 455, September 2014.

[5] J. M. Hereford, M.Siebold, S. Nichols, "Using the Particle Swarm Optimization algorithm for robotic search applications", 2007 Swarm Intelligence Symposium, Honolulu, HI, pp. 53 – 59, April 2007.

[6] J. M. Hereford, "BEECLUST swarm algorithm: analysis and implementation using a Markov chain model", International Journal Innovative Computing and Applications, vol. 5, No. 2, pp. 115-124, May 2013.

[7] S. Legg, M. Hutter, "Univeral intelligence: A definition of machine intelligence", Minds and Machines, vol. 17, pp. 391-444, December 2007.

[8] A. F. T. Winfield, J. Nembrini, "Safety in numbers: fault-tolerance in robot swarms", International Journal Modelling, Identification and Control, Vol. 1, No. 1, pp. 30-37, January 2006.