

Shinerbots: Navigating Large-Scale Robot Swarms Inspired by Golden Shiner Fish

Enyu Luo · Ji Liu · Alexandra Bacula · Yuting Ng · Tamer Başar ·
Grace Xingxin Gao

Received: date / Accepted: date

Abstract We propose a collective robot swarm navigation algorithm inspired by the emergent navigation behavior of the Golden Shiner fish. The navigation algorithm combines two simple navigation operations. First, each agent in the swarm individually modulates its speed based on sensing its environment at its current location, but without acquiring location information and environmental gradients. Second, each agent moves toward its neighbors by sensing only their relative positions with respect to its current location.

To implement the algorithm, we designed and built a swarm of 85 robots, named “Shinerbots”. Each individual Shinerbot has a diameter of 40.6mm and a material cost of USD \$20. We used vibration motors for mobility, a photodiode for light intensity sensing, infrared reflection for neighbor proximity sensing, and a microcontroller for processing. To make large-scale swarm operations tractable, we also designed a swarm messaging system and a swarm charging plate.

We have demonstrated with simulation as well as with experiments using Shinerbots that our proposed algorithm successfully enables the robot swarm to collectively navigate towards a target region using minimal sensing and control.

Keywords Robot Swarm · Navigation · Bio-inspired Algorithms

Grace X. Gao
319 Coordinated Science Laboratory
1308 West Main St. Urbana, IL 61801
Tel.: +217-333-6360
E-mail: gracegao@illinois.edu

1 Introduction

Navigation has historically played an important role in many applications, such as piloting, environment monitoring, and search and rescue. Traditionally, entire navigation systems have been included on vehicles or vessels (Sturza, 1988). As robotics has progressed into these areas, navigation has developed into a fundamental problem of robotics and unmanned aerial vehicles (Koren and Borenstein, 1991). Over the past decade, the need for distributed information processing has arisen naturally in robotic and sensor networks due to limited communication capacity. This constraint on the flow of information precludes centralized information processing (Olfati-Saber et al., 2007; Dimakis et al., 2010). Compared with traditional centralized processing, distributed processing has advantages in fault tolerance, total cost, and the ability to accommodate various physical constraints. With these thoughts in mind, we aim to create a navigation system consisting of a group of low-cost robots which are able to perform swarm navigation in a distributed manner.

The merits of our proposed swarm navigation algorithm are the following:

- Minimal sensing: Each robot only senses (e.g. the light intensity) at its current spot, instead of the neighboring environment;
- Minimal communication: The robots do not exchange information with their neighbors nor other robots in the swarm;
- Simple and distributed algorithm;
- Scale is not a computational burden, but a key feature that enables collective navigation.

Existing robot swarm navigation platforms include MarXbot (Bonani et al., 2010), eSwarbot (Couceiro et al.,

2011), Pi-swarm (Hilder et al., 2014), e-Puck (Torrellas Socastro, 2013), Khepera (Mondada et al., 1999), Jasmine (Kernbach et al., 2013), Kilobot (Rubenstein et al., 2012) and Droplet (Farrow et al., 2014). Their sizes and costs are summarized in Table 1.

The larger swarm navigation platforms cost more and use wheels and/or tracks for mobility. The smaller swarm navigation platforms, which include Kilobot and Droplet, cost less and use vibration motors for mobility.

Recently, different types of distributed swarm navigation algorithms have been proposed. Notable among them are behavior-based control (Balch and Arkin, 1998), Lyapunov-based formation (Desai et al., 2001), stream functions (Waydo and Murray, 2003), nonlinear servomechanism (Gazi, 2005a), artificial potentials and sliding mode control (Gazi, 2005b), and potential functions and panel methods (Merheb et al., 2016). More recent advances in this research direction include distributed control of a swarm of heterogeneous robots (Prorok et al., 2015), cooperative surveillance via micro-aerial vehicles (Saska et al., 2016), and human-swarm interactions (Diaz-Mercado et al., 2017), adaptive foraging for robotic swarms (Castello et al., 2016), cooperative spatial coverage (Schroeder et al., 2017) and object transport (Alkilabi et al., 2017) in distributed multi-robot swarm systems.

Most existing swarm navigation systems and algorithms make use of sophisticated sensing and computational capabilities of the robots in the system. In particular, it is assumed that each robot can measure complicated environment parameters, such as the gradient. Even in biology, it was long-believed that fish could sense the light gradients in the environment. But, in a recent paper (Berdahl et al., 2013), it is observed that Golden Shiners, a type of schooling fish that naturally prefers darkness, neither sense the light gradients nor distinguish on a broad scale between darkness and light. In fact, each individual Golden Shiner senses only the light intensity at its current location, and either swims fast in bright spots or slows down in dark spots. Surprisingly, it is validated that with only this simple behavior from the individual fish, Golden Shiners are always able to navigate to shady areas by schooling in groups. Inspired by the emergent collective intelligence observed from Golden Shiners, this paper aims to establish and model a practical swarm navigation system. Accordingly named “Shinerbots”, our robots have been designed to individually mimic the sensing and movement strategy of Golden Shiners in order to perform similar navigation objectives.

The Golden Shiners idea has been partially applied to gradient climbing processes (Elor and Bruckstein, 2014) and distributed source seeking (Wu and Zhang,

2016). In (Elor and Bruckstein, 2014), a discrete-time algorithm is proposed for scalar-field gradient climbing. The algorithm relies on the assumption that every agent can sense the relative position of all other agents in the group, which is not feasible for large-scale robot swarms. In (Wu and Zhang, 2016), a continuous-time algorithm is presented for a group of mobile agents to collectively seek a source while keeping a desired formation. The algorithm requires that all agents agree on a specific direction decomposition of their velocities. Although such an agreement can be guaranteed by running a consensus algorithm, such an aforementioned process is not feasible for a large group of robots, as the consensus algorithm may require infinite time to converge. Our algorithm avoids the limitations in (Elor and Bruckstein, 2014) and (Wu and Zhang, 2016) by combining environmental and social factors in a process akin to consensus (see Section II).

The algorithm presented in this paper is also closely related to consensus (Jadbabaie et al., 2003), flocking (Olfati-Saber, 2006), and rendezvous (Lin et al., 2007) problems, in which each agent is required to keep a certain level of connectivity with other agents. The convergence of consensus and flocking algorithms is typically based on connectivity of the whole network. It is worth emphasizing that in our algorithms, individual robots are not required to maintain connectivity, and the desired convergence does not depend on the network connectivity, even though the whole group of robots may diverge into different subgroups from time to time.

The main contributions of this paper are:

- First, a simple but compelling continuous-time model for Shinerbots inspired by Golden Shiners;
- Second, analysis of the performance of the model validated by simulation;
- Third, creation of Shinerbots with small size (40.6mm) and low cost (USD \$20);
- Last, establishment of a Shinerbot swarm navigation platform demonstrated by experiments.

The material in this paper was partially presented in (Heng and Gao, 2014) and (Luo et al., 2016), but this paper presents a more comprehensive treatment of the work. Specifically, the paper provides analysis of the Shinerbot swarm navigation algorithm (Section 2), a simulation of the algorithm (Section 3), and more experiment results (Section 5.3), which were not included in (Heng and Gao, 2014) and (Luo et al., 2016).

The rest of the paper is organized as follows. Section 2 presents our Shinerbot swarm navigation algorithm with analysis, Section 3 validates the results and analysis in Section 2 via simulation, Section 4 describes our Shinerbot swarm navigation platform, Section 5

Table 1: Size and cost of existing robot swarm navigation platforms

Robot	Size	Cost
MarXbot (Bonani et al., 2010)	17cm diameter	
eSwarbot (Couceiro et al., 2011)	12.6cm diameter	
Pi-swarm (Hilder et al., 2014)	9cm diameter	part cost of USD\$100
e-Puck (Torrellas Socastro, 2013)	7cm diameter	
Khepera (Mondada et al., 1999)	7cm diameter	retail price of USD\$272
Jasmine (Kernbach et al., 2013)	2.7cm cube	part cost of USD\$112
Kilobot (Rubenstein et al., 2012)	33mm diameter	retail price of USD\$113
Droplet (Farrow et al., 2014)		part cost of USD\$50 without the shell

shows and discusses our experimental results, and Section 6 summarizes the paper.

2 Shinerbot Swarm Navigation Algorithm

In this section, we consider a simple mathematical model which explains the behavior of our Shinerbot swarm navigation algorithm.

2.1 Modeling Inspired by Golden Shiners

We first model Shinerbots as mobile autonomous agents constrained to move in a two-dimensional plane with a light intensity varying by location. For each point $x \in \mathbb{R}^2$ on the plane, we use $f(x)$ to denote the light intensity of the point. We assume that $f(x) \geq 0$ for all $x \in \mathbb{R}^2$ and $f(x) = 0$ if, and only if, $x = \mathbf{0}$, where $\mathbf{0}$ denotes the origin of the plane. Thus, the origin is the darkest spot in the plane. Throughout this section, we assume that the light intensity environment is a circular patch which is darkest at its center (i.e., the origin) which transitions to brighter levels proportionally to the distance a given point is from the center of the patch. To be more precise, let $\|x\|$ denote the Euclidean norm of $x \in \mathbb{R}^2$, or, equivalently, the distance from point x to the origin. Then, for any two points $x, y \in \mathbb{R}^2$, there holds $f(x) = f(y)$ if $\|x\| = \|y\|$, and $f(x) < f(y)$ if $\|x\| < \|y\|$. Such an environment is consistent with our experiment environment (see Section 5) and the environment in (Berdahl et al., 2013).

Consider a group of n agents, labeled 1 through n , moving in the plane. We use $[n]$ to denote the set $\{1, 2, \dots, n\}$. Each agent i can sense only those agents which are currently within the sensing range of agent i . We call those agents the *neighbors* of agent i and use $\mathcal{N}_i(t)$ to denote the set of the neighbors of agent i at time t . For simplicity, we assume that all agents have the same sensing radii. Thus, this notion of a neighbor induces a symmetric relation on the agent group. Neighbor relationships among the n agents may change

over time and can be conveniently described by a time-dependent, n -vertex, undirected graph.

Let $x_i(t) \in \mathbb{R}^2$ be the location of agent i in the plane at time t . Inspired by the collective phenomena observed in Golden Shiners as reported in (Berdahl et al., 2013), we model the dynamics of each agent i via the following differential equation

$$\dot{x}_i(t) = \text{env}(x_i(t)) \cdot \text{soc}\left(x_i(t), \{x_j(t)\}, j \in \mathcal{N}_i(t)\right),$$

in which the velocity of agent i is governed by an environment factor $\text{env}(x_i(t))$ and a social factor $\text{soc}(x_i(t), \{x_j(t)\}, j \in \mathcal{N}_i(t))$. It is worth emphasizing that the environment factor is real-valued and the social factor is vector-valued. The environment factor is a function of the light intensity at $x_i(t)$, which enables convergence to an optimal point, and the social factor is a function of neighbor proximity at $x_i(t)$, which enables collective intelligence and expediting of convergence. Specifically, we consider the following dynamics for each agent:

$$\dot{x}_i(t) = g(f(x_i(t))) \sum_{j \in \mathcal{N}_i(t)} (x_j(t) - x_i(t)), \quad i \in [n], \quad (1)$$

where $g(\cdot)$ is a real-valued function representing the light sensitivity of agent i . We assume that all agents have identical light sensitivity. From (1), the value of $g(f(x))$ affects the speed of an agent when it is located at x . It has been observed in (Berdahl et al., 2013) that Golden Shiners tend to swim slowly in dark regions and fast in bright regions. To incorporate this observation into our model, we impose the following assumption on the function g , which holds throughout this section.

Assumption 1 *The function $g : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ is a strictly increasing, bounded function such that $g(0) = 0$.*

The following lemma captures an important local behavior of a fully connected group of agents under Assumption 1.

Lemma 1 *Suppose that a group of agents contains at least two agents. Suppose that the gradient of the light*

intensity ∇f is constant and nonzero, and that the neighbor graph is a complete graph¹ for all time. Then, the center of mass of all agents in the group moves toward the direction of ∇f .

Proof: Suppose that there are $n \geq 2$ agents in the group. Since the neighbor graph is always a complete graph, it follows that $\mathcal{N}_i(t) = [n]$ for all $i \in [n]$ and t . Since all n agents in the group are identical, the center of mass, denoted by x_c , is located at the average location of the agents. Let u denote the unit vector in the direction of ∇f . Then,

$$\begin{aligned} \langle \dot{x}_c(t), u \rangle &= \left\langle \frac{1}{n} \sum_{i=1}^n \dot{x}_i(t), u \right\rangle \\ &= \left\langle \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n g(f(x_i(t))) (x_j(t) - x_i(t)), u \right\rangle \\ &= \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n \langle g(f(x_i(t))) (x_j(t) - x_i(t)), u \rangle \\ &= \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n g(f(x_i(t))) (\langle x_j(t), u \rangle - \langle x_i(t), u \rangle) \\ &= \frac{1}{2n} \sum_{i \neq j}^n \left\{ g(f(x_i(t))) (\langle x_j(t), u \rangle - \langle x_i(t), u \rangle) \right. \\ &\quad \left. + g(f(x_j(t))) (\langle x_i(t), u \rangle - \langle x_j(t), u \rangle) \right\} \\ &= \frac{1}{2n} \sum_{i \neq j}^n \left\{ (g(f(x_i(t))) - g(f(x_j(t)))) \right. \\ &\quad \left. \cdot \langle x_j(t) - x_i(t), u \rangle \right\} \\ &< 0 \end{aligned}$$

where $\langle \cdot \rangle$ denotes inner product, and the last inequality is a direct consequence of Assumption 1. ■

In the case when a group of agents are located in a small neighborhood, they will become neighbors of each other, and the gradient of the light intensity can then be approximated as constant. Lemma 1 implies that the center of mass of such a fully connected group of agents moves toward the direction of decreasing light intensity.

From (1), it can be seen that the movement of agents may change the neighbor relationships among the agents. Simple examples show that a connected group of agents may become disconnected, and vice versa. We call an agent *isolated* at time t if it is disconnected from all other agents (i.e., it cannot sense any other agent) at that time.

¹ A *complete graph* is a simple undirected graph in which every pair of distinct vertices is connected by an edge.

The following theorem describes a limiting behavior of the system without isolated agents.

Theorem 1 Suppose that all n agents adhere to the dynamics (1), and that no agent is isolated along the trajectory. Then, all n agents will asymptotically enter a small neighborhood around the origin, and their center of mass will asymptotically converge to the origin.²

Proof: Note that the dynamics (1) can be viewed as a continuous-time linear consensus process (Moreau, 2004)

$$\dot{x}_i(t) = \sum_{j \in \mathcal{N}_i(t)} a_{ij}(t) (x_j(t) - x_i(t)), \quad i \in [n], \quad (2)$$

in which $a_{ij}(t) = g(f(x_i(t)))$. Since each agent is not a point object, the agents cannot all rendezvous at the origin. Recall that the light intensity environment is a circular patch which is darkest at the origin and transitioned to the brightest light levels. It follows from Assumption 1 that there exists a constant β such that $a_{ij}(t) = g(f(x_i(t))) \geq \beta$ for all t and $i \in [n]$, except for the case of one agent located exactly at the origin. Thus, the system (2) is “cut-balanced” (Hendrickx and Tsitsiklis, 2013). By Theorem 1 in (Hendrickx and Tsitsiklis, 2013), if we ignore the size of each agent, the system (2) will asymptotically reach a steady state at which all agents are partitioned into one or more different subgroups, and each subgroup of agents rendezvous at the same point. This implies that, in the case when the size of each agent is taken into account, each subgroup of agents will enter a small neighborhood, and the neighbor graph will converge to one or more connected components, with each component being a complete graph. Since no agent is isolated along the trajectory, each subgroup contains at least two agents. In this case, the gradient of the light intensity within the small neighborhood of each subgroup can be approximated as constant. From Lemma 1, the center of mass of each subgroup of agents moves toward the direction of decreasing light intensity, which is toward the origin. Note that this argument will hold along the trajectory of each subgroup until its center of mass reaches the origin.³ Therefore, all subgroups will eventually enter in a small neighborhood around the origin, and the center of mass of all agents will converge to the origin. ■

Theorem 1 implies that all agents will finally gather together around the unique darkest point in the plane.

² The agents will not rendezvous at the same point because each agent represents a Shinerbot which is of size 40.6mm.

³ It is possible that two or more different subgroups become connected before reaching the origin, and thus constitute a new larger subgroup. The argument still applies to the new subgroup.

2.2 Improved Modeling with Added Randomness

If the assumptions of Theorem 1 do not all hold, some undesired behavior of the agents may occur, which is also observed in our experiments on Shinerbots. In this section, we will discuss two types of undesired behavior.

First, it is possible that some agent becomes isolated at some point. In this case, such an agent does not have any neighbor, and its velocity becomes 0 according to (1). Thus, it will become still unless some other agent moves into its sensing radii. The occurrence of isolated agents can be precluded by using some algorithm, such as those in (Olfati-Saber, 2006; Lin et al., 2007), to maintain connectivity among the agents. However, such algorithms requires high computational complexity.

Second, if the strict inequality $f(x) < f(y)$ does not hold for all $x, y \in \mathbb{R}^2$ such that $\|x\| < \|y\|$, the gradient of light intensity ∇f equals 0 in some small neighborhood. Then, from the proof of Lemma 1, when a subgroup of agents move into the neighborhood, its center of mass will not move, and thus the subgroup of agents will stay in the neighborhood. In this situation, we say that the agents are *stuck* in the neighborhood. Such scenarios are observed because the light intensity environment in our experiments contains some circular bands in which the light intensity is constant, and thus $\nabla f = 0$.

The above undesired behavior can be regarded as limitations of the dynamics (1). To get around these limitations, a stochastic term is added to the dynamics to reduce the possibility of isolated and stuck agents. Specifically, we consider the following modified dynamics:

$$\dot{x}_i(t) = g(f(x_i(t))) \left(\sum_{j \in \mathcal{N}_i(t)} (x_j(t) - x_i(t)) + Z_i(t) \right), \quad i \in [n], \quad (3)$$

where $Z_i(t)$ is an additive two-dimensional white Gaussian noise (i.e., $Z_i(t) = [Z_x(t) \ Z_y(t)]^\top$ with $Z_x(t)$ and $Z_y(t)$ being independent and of standard normal distribution).

From (3), it can be seen that the system has a unique equilibrium at which $x_i(t) = 0$ for all $i \in [n]$.

Lemma 2 *Suppose that a group of agents contains at least two agents. Suppose that the gradient of the light intensity ∇f is constant and nonzero, and that the neighbor graph is a complete graph for all time. Then, the center of mass of all agents in the group moves toward the direction of ∇f in expectation.*

Proof: Let x_c denote the center of mass and u denote the unit vector in the direction of ∇f . From the proof

of Lemma 1,

$$\begin{aligned} \langle \dot{x}_c(t), u \rangle &= \left\langle \frac{1}{n} \sum_{i=1}^n \dot{x}_i(t), u \right\rangle \\ &= \left\langle \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n g(f(x_i(t))) (x_j(t) - x_i(t)), u \right\rangle \\ &\quad + \left\langle \sum_{i=1}^n g(f(x_i(t))) Z_i(t), u \right\rangle \\ &= \frac{1}{2n} \sum_{i \neq j}^n \left\{ (g(f(x_i(t))) - g(f(x_j(t)))) \right. \\ &\quad \cdot \langle x_j(t) - x_i(t), u \rangle \Big\} \\ &\quad + \frac{1}{2n} \sum_{i=1}^n g(f(x_i(t))) \langle Z_i(t), u \rangle. \end{aligned}$$

In the last equality, the first summation is strictly less than 0 because of Assumption 1. Since $Z_i(t)$ is a zero-mean circularly symmetric Gaussian random vector, the expectation of the second summation equals zero. Thus, $\mathbb{E}[\langle \dot{x}_c(t), u \rangle] < 0$, which completes the proof. ■

Theorem 2 *Suppose that all n agents adhere to the dynamics (3), and that no agent is isolated along the trajectory. Then, in expectation, all n agents will asymptotically enter a small neighborhood around the origin, and their center of mass will asymptotically converge to the origin.*

Proof: The theorem can be proved using arguments similar to those in the proof of Theorem 1. ■

Compared with (1), the added randomness removes undesired equilibria. Since each agent cannot stay still at any point except for the origin, random movement of an isolated agent increases the possibility for it to sense other agents. Also, random movement of a group of stuck agents can help them move out of the zero gradient neighborhood.

3 Simulation Results

To illustrate the behavior of the swarm navigation algorithm (3), we present a simulation result in Fig. 1. The target is the darkest spot shown in red. At initial time $t = 0$, 100 agents are randomly distributed on the right side of the darkest spot. We use a green circle to track the center of mass of all agents by setting the center of circle as the darkest spot and the radius as the distance from the center of mass to the darkest spot. It can be seen from the figure that the nodes gradually diverge into three subgroups (see subfigures for $t = 15$ sec and $t = 30$ sec). Each subgroup of nodes then move

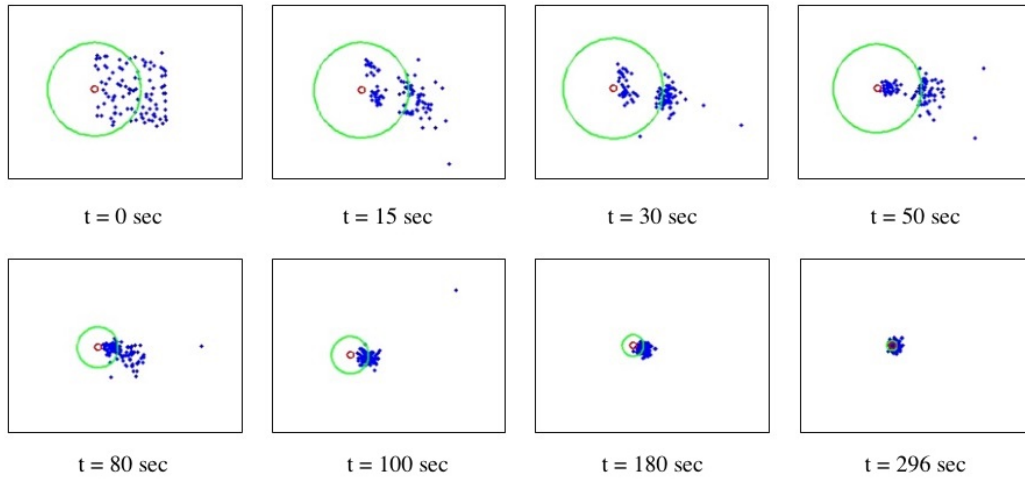


Fig. 1: Snapshots from a simulation video demonstrating the Shinerbot swarm navigation algorithm (3). The target destination is marked in red; and the radius of the green circle shows the distance from the center of mass of Shinerbots to the target. It is observed that several clusters are formed, and then merged into one cluster. At the end, the Shinerbots are able to collectively navigate to the target area.

toward the darkest spot. Along this process, two nodes become isolated (see subfigure $t = 50$ sec), one of which gets connected again with other nodes by its own random movement caused by the added randomness in the algorithm (see subfigure $t = 80$ sec). Finally, all agents, except for one, enter in a small neighborhood around the origin (see subfigures $t = 100$ sec, $t = 180$ sec, and $t = 296$ sec). Therefore, the simulation validates the analysis and result of the algorithm (see Theorem 2 and its proof).

4 Design and Build Shinerbot Platform

We design and build a robotic swarm system that will be able to perform the algorithm described in the previous section. Each robot must have the ability to move, detect distance and orientation of neighbors, and detect light intensity of its current location. Eighty-five individual robots were built to comprise the swarm. To simply building and management of such a swarm, factors including cost, size, and scalability are also taken into account as additional design requirements.

The design of a single Shinerbot consists of submodules for mobility, communications and neighbor sensing, ambient light sensing, LED indicator, and power. Each robot runs on a MSP430 microcontroller, which was chosen for its low power consumption. The nature of the swarm algorithm does not require much processing power. The block diagram of a single Shinerbot is shown in Fig. 2.

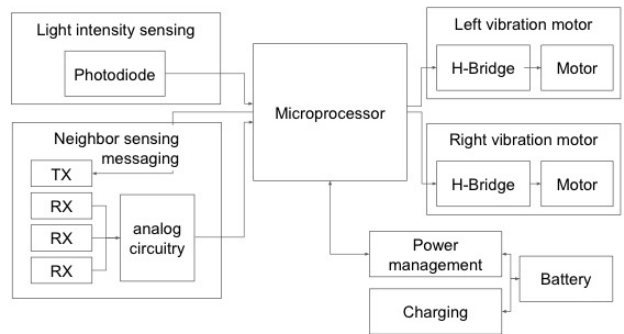


Fig. 2: Block diagram of a single Shinerbot showing various hardware components

4.1 Mobility

To achieve both lower cost and smaller form, the Shinerbots were designed to move using legs rather than wheels (Vartholomeos and Papadopoulos, 2006). Two Eccentric Rotating Mass (ERM) vibration motors are placed one on each side of the robot, which are driven with Pulse Width Modulated (PWM) signals via an H-bridge circuit, allowing both forward and backward movements. Rotating both motors in the same direction causes the robot to move either forward or backward, whereas rotating the motors in opposite directions causes the robot to rotate left or right. The challenge in this setup is that the speed of the robots are fixed, whereas the swarm algorithm needs the robots to have varying speeds. The direction of the robot can only be consistently con-

trolled at a certain power output to the motors due to the fact that the rotation of the eccentric mass in the motor has to be matched with the frequency at which the robot vibrates vertically. This allows the correct interaction of the forces from the eccentric mass and friction between the legs and the floor. However, this challenge can be overcome by duty cycling. Specifically, the robot will move and pause in varying time steps to achieve the desired average speed over time.

4.2 Neighbor Sensing

Shinerbots sense neighbors via infrared (IR) communications. Each robot transmits a packet at regular intervals, and neighboring robots will receive the packet and estimate the inter-robot distance using received signal strength indicator (RSSI). Using 3 receivers placed 120 degrees apart (see Fig. 3) on the perimeter of the robot, 3 distances are obtained which are used to calculate the neighbor's orientation. The transmitter is placed in the middle of the bottom of the robot. This allows the IR signal to bounce off the floor and spread in 360 degrees around the robot before reaching a receiver at the bottom of a neighboring robot, which is illustrated in Fig. 4.

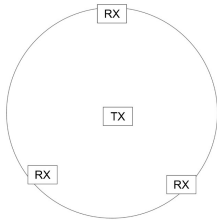


Fig. 3: Placement of 3 infrared (IR) receivers and 1 IR transmitter on the underside of the Shinerbot

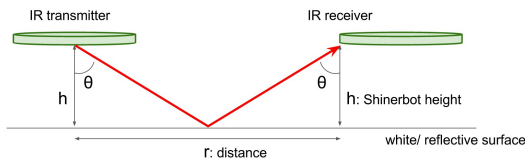


Fig. 4: Geometry that affects RSSI of the IR signal between two neighboring Shinerbots

To estimate distances from RSSI, the cosine function was used to model the gain of the transmitter and receiver due to departure and arrival angles. This was chosen based on the beam pattern plot obtained from

the datasheet shown in Fig. 5. The cosine model covers the factor that influence RSSI in the z direction, $1/r$ factors in the spread of the signal in the 2D plane, where r is the distance between the transmitter and a receiver of a neighboring robot, and constant k factors in all the gains combined from the transmit power, receiver gain, and Analog to Digital Converter (ADC) quantization range. Thus, the final theoretical RSSI is given by the following equation:

$$\begin{aligned} \text{RSSI} &= \frac{k}{r} \cdot (\cos \theta)^2 \\ &= \frac{k}{r} \cdot \frac{4h^2}{4h^2 + r^2} \end{aligned}$$

where r is the distance between two neighboring Shinerbots, h is the height of a Shinerbot, and θ is the angle between the signal path and the vertical. The coefficient k is a scaling factor selected for curve-fitting of RSSI measurements; however, this did not get accurate results due to manufacturing variations, as shown in Fig. 6.

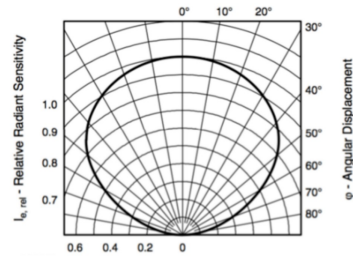


Fig. 5: IR beam pattern (Semiconductors, 2011)

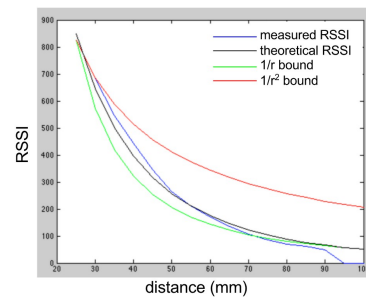


Fig. 6: RSSI measurements against distances

To get accurate results, a calibration based method is used to estimate distance from RSSI. A calibration routine will have to be run once after the robot is fully assembled. This routine consists of moving a transmitting robot away from the calibrating robot in steps of

5mm, and the procedure is repeated for all three receivers in one robot. Linear interpolation is then used to compute the distance in steps of 64 ADC units. Fig. 7 shows the measured RSSI in steps of 5mm distances, and also the estimated distance in steps of 64 RSSI ADC units. These distances are then stored as a lookup table in flash memory. When a robot needs to convert RSSI to distance, a simple bit shift and mask are performed to obtain the nearest two distances and a linear interpolation is operated to estimate the actual distance.

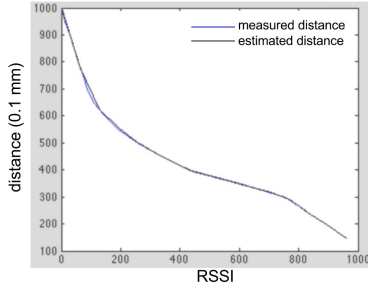


Fig. 7: Distance estimation from RSSI measurements

The three distances are then used to trilaterate the position of a neighboring robot given by equations below:

$$x = \frac{r_A^2 - r_B^2}{2l_3}$$

$$y = \frac{r_C^2 - r_D^2}{2(l_1 + l_2)} + \frac{l_1 - l_2}{2}$$

$$r_D^2 = \frac{r_A^2 + r_B^2 - \frac{l_3^2}{2}}{2}$$

where x and y are the distances to a neighboring Shinerbot in the x -direction and y -direction, respectively, r_A , r_B , r_C are distances from the IR transmitter (TX) to the IR receivers (RX_A , RX_B , RX_C), r_D is the distance from TX to the midpoint between RX_A and RX_B , and l_1, l_2, l_3 are geometrical lengths on the Shinerbot. The coordinate reference and the parameters are given in Fig. 8.

The actual transmissions are fixed length packets consisting of destination ID, RSSI byte, source ID, 5 data bytes, and a CRC. The RSSI byte is a fixed value of 0x55 which allows the RSSI module to switch ADC channels and record RSSI values for all three receivers. The physical signal uses the Infrared Data Association (IrDA) standard (Group et al., 1997). A Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA) algorithm is used to enable multiple robots to transmit in the same area. Although a generic communication

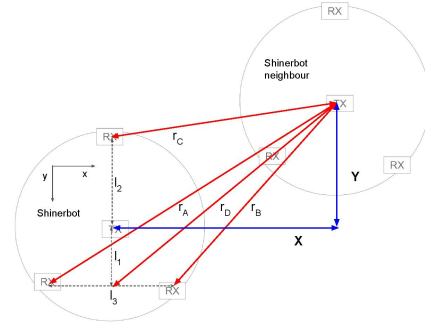


Fig. 8: Direction estimation using three IR distance measurements

system is implemented in Shinerbots, the data bytes are not used to communicate any intelligible information in regards to performing the swarm algorithm as presented in this paper.

4.3 Swarm Charging

Individually charging every robot via a power cable would be tedious, time consuming, and not scalable as the size of a swarm becomes large. In order to have a scalable charging system, we designed a charging pad that can charge 49 robots simultaneously. The charging pad is a copper-clad fiberglass plate, with alternating strips of copper as shown in Fig. 9. Each robot is charged via its legs, which are connected to a Li-ion charge management IC via H-bridges. This allows charging at 6V supplied at any polarity to the legs. The width and gaps of the alternating strips are designed such that each robot is guaranteed to have any 2 legs on an alternating strip when placed randomly. The copper is also tin coated to prevent oxidization. Status from the charge management IC is fed to the microcontroller so that the robot can go to sleep when it is placed on the charging plate. This prevents robots from vibrating and causing unreliable charging.

4.4 Other Hardware Submodules

The robot uses a photodiode for sensing ambient light. The photodiode is raised to the height of the battery so as not to be blocked by the side shadow of the battery. An RGB LED, powered by PWM signals from the microcontroller, is included as a general purpose indicator. There is also a separate smaller red LED for charging status indication.



Fig. 9: Shinerbots charging on the charging plate

4.5 PCB Design

A 40mm diameter circular PCB designed in KiCad constitutes the foundation of each Shinerbot. This 4-layer board has been designed such that all digital circuits are mounted on the top of the board and all analog circuits are mounted on the bottom of the board. All digital circuits are placed at the top of the board, while the analog circuits are placed at the bottom. We choose to separate analog and digital circuits by layers instead of by 2D location because the analog signals have to span across the board, given that the 3 infrared receivers are placed 120 degrees apart around the perimeter of the robot. In addition, analog and digital ground planes in the middle layers shield the signals from crosstalk.

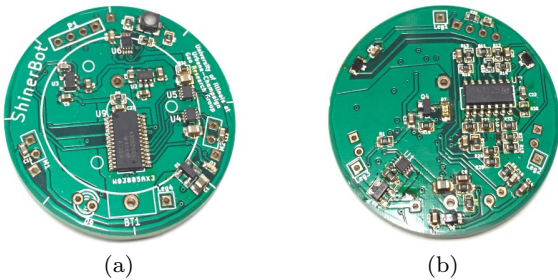


Fig. 10: Shinerbot PCB (a) top view; (b) bottom view.

4.6 Firmware Implementation

The firmware is structured in a layered approach as shown in Fig. 11. Starting with the MSP430 header

from Texas Instruments, `#define`'s are provided to all low-level registers in the microcontroller. I/O drivers provide an abstraction for reading and writing data to various hardware peripherals. The System Layer provides a higher level API that translates raw data from the sensors into sensible information, as well as some background tasks needed for the entire system. Lastly, the Application Layer is where the high-level behavior of the robot is programmed.

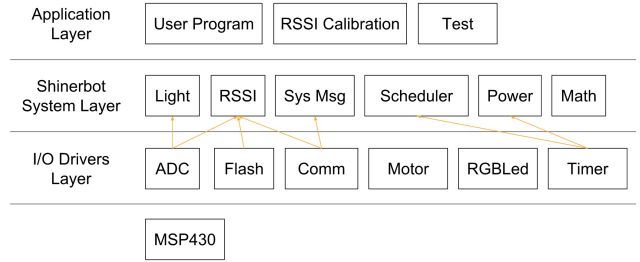


Fig. 11: Shinerbots firmware block diagram

There are three types of application code in the application layer. The User Program is the main loop that reads sensor information, and sends movement commands accordingly to perform the swarm navigation algorithm. The RSSI Calibration application is a routine that will read the raw Infrared RSSI values at expected distances and save calibration data to flash. This data is needed for computing neighbor distances, and only needs to be run once right after the robot is assembled. Test is an application that runs a fixed sequence of behaviors such as a color sequence on the LED, movement sequence, and responses to light and neighbors. The purpose of this application is to verify that the hardware assembly is done correctly and we get a fully functioning robot. At any one time, the robot is only programmed with one of the three applications due to RAM size constraints.

The I/O Drivers Layer consists of various drivers that map a peripheral command into a sequence of low-level register read and writes to achieve the desired function. Each driver has initialization, sleep, and wake routines. Initialization is done on boot. Sleep and wake routines are called when the robot is brought to a low power state, and back to a normal run state. Other driver functions are peripheral specific. ADC has channel selection, triggering, and reading values for single or a multi-channel sequence. Flash provides erase and write functions to specific segments. Comm transmits and receives packets. It translates packets into stream of bytes and vice versa, checks the checksum,

and performs the CSMA-CA algorithm. Both motor and RGBLED output desired power to respective motors or color channels via pulse-width modulation (PWM). Timer provides a 1ms systick to the system, as well as a delay API for the application layer to go to CPU sleep for a certain amount of time.

The System Layer provides high level functionality. Light provides a simple abstraction to ADC for reading current light intensity values. It also prevents ADC channel contention when RSSI needs to use the ADC as well. RSSI provides the coordinate of the neighbor of an associated incoming received packet. It does this by reading ADC values on 3 different channels, launching a background task to calculate 3 distances using the calibration data, and then computing the x-y coordinate of the neighbor. The Scheduler provides a simple multi-tasking environment. It allows system tasks to be done behind the scenes without the application layer's knowledge, and also brings the application layer's thread to sleep when Delays or other blocking system APIs are called. RSSI works closely with Comm. Comm triggers the RSSI background task on an incoming packet, and by the time a packet is fully received, the position values are ready. The position is then stored as part of the receive packet into the receiver queue, such that when the application program calls a read(), it gets both the data and position in an API call. This API provides generic robot-to-robot communication functionality. The swarm navigation algorithm presented in this paper does not use the data bytes in packets to communicate any intelligible information. Only the position of neighbors is used in this case. SysMsg is a library that will respond when an incoming packet is a system message. It also relays the same message to the neighboring robots, thus enabling control of the whole swarm. The messages include sleep, wake, run, and reset, and allows for simple control of starting and stopping the swarm experiment. The Power library performs the necessary sequence needed to bring the robot to a deep sleep mode, and to wake it back up. In deep sleep, most clocks and all peripherals are turned off. Only a lower power timer remains running. This allows the robot to get to a semi-awake mode where only the Comm peripheral is brought back up occasionally to listen for any wake system messages. The Math library provides a pseudo random number generator and a fast integer square root function.

4.7 System Messages

To control the swarm system, it would be impractical to have physical switches or buttons on each Shinerbot for turning them on or off. To achieve scalability in

control of the overall experiment, system messages are implemented. System messages are packets that send a specific command, which allows remote control of the whole swarm. There are 4 commands: Sleep, Wake, Run, and Reset. Sleep brings the robot to a deep sleep, or low power state, essentially turning off the Shinerbot. Wake brings the robot back up to a semi-low power state. Run resumes the user program on the robot. Reset causes a software reset, which essentially reboots the microcontroller. Upon receiving a system message, each robot will relay that message 3 times. This allows all the robots in the swarm to receive the same command. This is illustrated in Fig. 12.

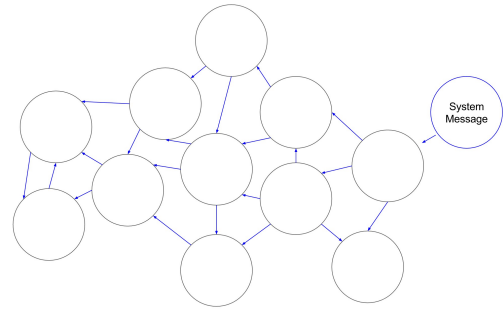


Fig. 12: Illustration of system messages being relayed throughout the swarm of Shinerbots

4.8 Power States

There are 4 levels of power states. Power State 0 (PS0) is where all subsystems of the robot are powered up, and the user program is running. Power State 1 (PS1) is where only the CPU sleeps while the rest of the peripherals are running. This occurs whenever the delay function is called. The delay function would put the user thread to sleep, and the scheduler would switch to the idle thread which essentially sleeps the CPU. Upon the corresponding timer interrupt, the scheduler would resume the user thread again. Power State 2 (PS2) is a semi-sleep state where the user thread, LEDs, motors are brought to sleep. The analog voltage supply, and communications system is left on to enable reception of messages. This allows the robot to respond to the wake system message. Power State 3 (PS3) is the deep sleep state where the robot is placed in the lowest power mode. Communications is put to sleep, the analog voltage powered off, the internal phase lock loops (PLL) and main clocks are turned off, and the timer is switched to a low frequency auxiliary clock.

As the Shinerbots are running, whenever a Sleep system message is received, it would transition to PS2, relay the Sleep system message 3 times, and transition to PS3. It would stay in deep sleep for 30 seconds, and wakes to PS2 for 400ms. This enables communications to listen to possible system messages. If nothing is heard, it would go back to PS3 and the cycle repeats. To wake the Shinerbots, a Wake system message has to be sent continuously over a period of more than 30 seconds. This ensures the reception of a Shinerbot when it is in PS3. As a Shinerbot receives the Wake message, it stays in PS2 and relays the Wake message over a duration of more than 30 seconds, so the rest of the Shinerbots would be awoken too. Because all robots are not synced in their deep sleep cycles, this stage would take some time. Once this awakening phase is done, a Run message can then be sent. Upon reception to the Run message, all Shinerbots relay that message 3 times, and transition to PS0. Since all Shinerbots are already in PS2, the response to the Run system message happens instantaneously unlike the Wake message. This power state transition sequence is illustrated in Fig. 13. The purpose of this design enables turning on and off of the robot remotely. Upon charging, the Shinerbot detects the charging state and transits to PS3 automatically without the need of system messages. This prevents unwanted movements on the charging plate. Upon removal from the charging plate, the charge management IC sends an interrupt which wakes the robot back up to PS0.

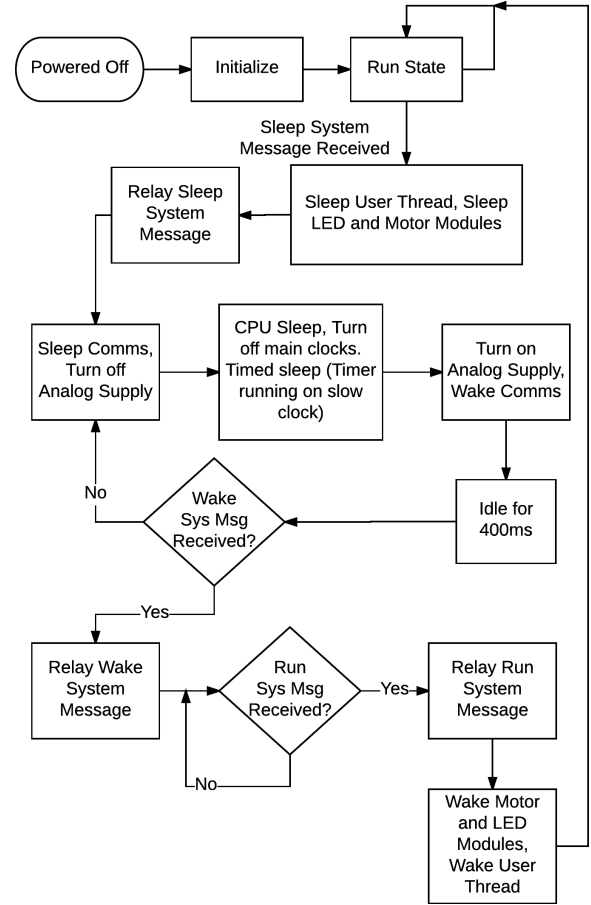


Fig. 13: Power state transition diagram in response to system messages

5 Experimental Results

We built each Shinerbot to have a compact form factor of 40.64 mm in diameter with a low part cost of USD \$20, as shown in Fig. 14. We performed an experiment using 85 Shinerbots. We created the experimental environment using a digital projector placed overhead, projecting a pattern onto a white horizontal surface, and call it the experiment platform. The pattern was a circular light gradient, bright in the center fading to dark at the edge. We initialized our experiment with Shinerbots placed in random positions on the experiment platform. Opposite to Golden Shiner fish, the target was the bright spot.

5.1 Intermediate Cluster Stage

In the experiments, the Shinerbots sometimes formed small clusters shortly after starting as discussed in Section 3 and seen experimentally in Fig. 15. In this experiment, at time $t = 1$ min, the Shinerbots formed

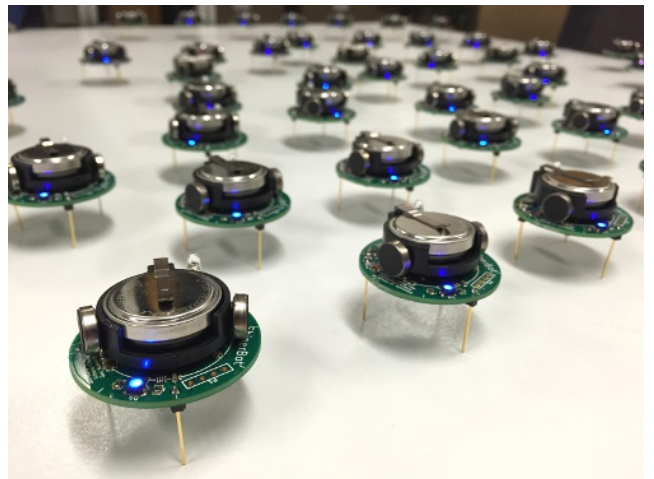


Fig. 14: Shinerbot: our bio-inspired collective robot swarm navigation platform.

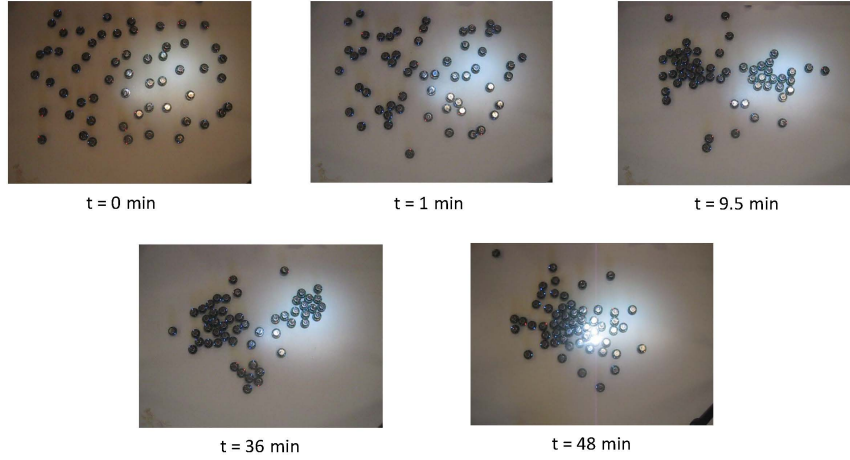


Fig. 15: In between initialization and convergence, the Shinerbots can form intermediate clusters.

small clusters of two to six Shinerbots; at time $t = 9.5$ min, they formed two larger clusters and a small cluster containing two Shinerbots; and at time $t = 36$ min, the Shinerbots formed three clusters, two of which are larger and the other is smaller. They then continue to converge to one group at time $t = 48$ min.

5.2 Social Swarm Navigation with Centered Light Source

The experiment shown in Fig. 16 had the light area centered among the Shinerbots, which emphasized the social aspect of the Shinerbots.

At time $t = 0$ min, the Shinerbots were initialized in random locations, distributed fairly evenly around the experimental platform. The light area was placed in the center of the experimental platform. By time $t = 1$ min, the Shinerbots have started to form small clusters and move closer to the center of the experimental platform. At time $t = 2.5$ min, the Shinerbots have formed one large cluster around the light area that includes most of the Shinerbots, and meanwhile there are a few small clusters still in the dark area. At time $t = 6$ min, the small clusters in the dark areas have broken up, and some of the Shinerbots joined the large cluster in the light area. The large cluster is then in the light area instead of around it. At time $t = 12$ min, the large cluster has become tighter and includes most of the Shinerbots. At time $t = 20$ min, the large cluster has conformed to the shape of the light area and includes most of the Shinerbots. For the remaining 5 minutes of the experiment, the large cluster stays in the light and a

few Shinerbots leave and join. The stable configuration shows the Shinerbots have converged.

5.3 Sensory and Social Swarm Navigation with Off-Centered Light Source

The experiment shown in Fig. 17 had the light gradient off-centered among the Shinerbots. The Shinerbots were distributed more heavily on the left side of the experimental platform, while the light gradient was on the right side. This emphasized the sensing aspect while also demonstrating the social aspect of the Shinerbots.

At time $t = 0$ min, the experiment begins. The light area was placed on the right side of the experimental platform. At time $t = 1$ min, the Shinerbots have formed small clusters in the dark areas of the experimental platform. At time $t = 3$ min, two main clusters have formed, with one near the light area and one in the dark area. There are still other small clusters of Shinerbots in the dark area as well as a few Shinerbots on their own. At time $t = 4.5$ min, the two main clusters have joined and formed one large cluster in the dark area. At time $t = 8$ min, more Shinerbots have joined the cluster in the dark area, but new small clusters have formed on the right side of the experimental platform. At time $t = 14$ min, the cluster in the dark area has tightened, and two smaller clusters in the light area have formed. At time $t = 18$ min, the two clusters in the light area merge, and the large cluster in the dark area remains tight. At time $t = 20$ min, there are two distinct clusters of Shinerbots, one in the dark area and one in the light area. At time $t = 26$ min, there is one

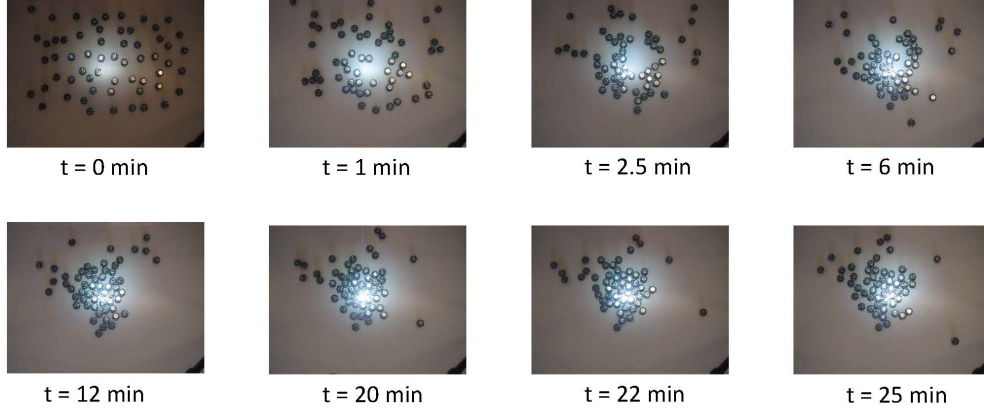


Fig. 16: Snapshots from an experiment video demonstrating successful autonomous social swarm navigation to a centered light area and achieving convergence among the majority of Shinerbots.

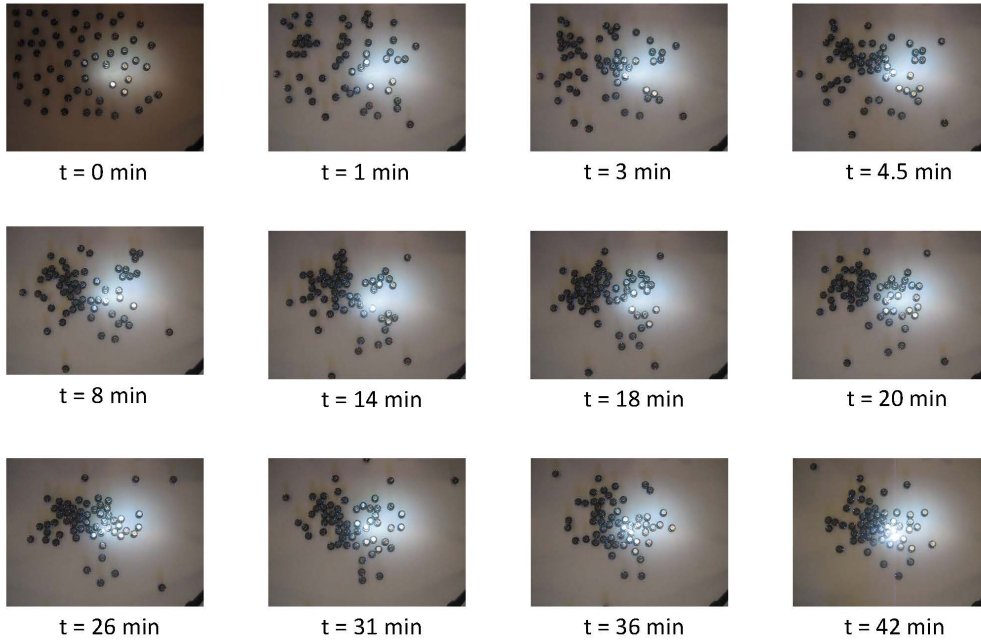


Fig. 17: Snapshots from an experiment video demonstrating successful autonomous sensory and social swarm navigation to an off-centered light area and achieving convergence among the majority of Shinerbots.

main cluster of the Shinerbots; however, only about half of them are in the light area, while the other half remain in the dark area. At time $t = 31$ min, the Shinerbots in the light area of the main cluster have spread out, while the Shinerbots in the dark area of the main cluster have tightened. At time $t = 36$ min, the main cluster

has tightened and now includes most of the Shinerbots. However, only half are in the light area. For the remaining 7 minutes of the experiment, the main cluster contains most of the Shinerbots and remains half in the light area and half in the dark area. The stable configuration shows the Shinerbots have converged.

In both experiments, a few Shinerbots remain isolated. This occurs at the beginning more frequently because the Shinerbots have not converged yet and the random motion can leave them alone. However, occasionally there are isolated Shinerbots once the rest have converged. These isolated Shinerbots usually do not move much and remain isolated due to the variability in manufacturing or a lower battery power. Since all the Shinerbots have the same motor output, a variance in manufacturing can cause one Shinerbot not to move as well as the others. If the experiment runs a long time, the battery power can also drain causing a similarly inhibited motion in the isolated Shinerbot. In both cases, the isolated Shinerbots have restricted mobility and do not converge with the other Shinerbots. This could be fixed in future experiments by calibrating each motor output specifically to each Shinerbot and compensating for battery drain in long experiments.

Some Shinerbots also get “lost”, which means that they have moved out of the experimental area. This occurs mainly at the beginning of the experiment, when the Shinerbots are not yet tightened around the center. Because of the heightened random motion at the beginning, a few Shinerbots on the edge may move off the experimental platform and are unable to return.

6 Conclusion

In this paper, we have designed and built a collective robot swarm navigation platform. The robots are named “Shinerbots” due to their inspiration drawn from the Golden Shiner fish. We have proposed a simple and compelling algorithm which mimics the sensing and navigation strategy of Golden Shiners. We have proved that leads to a desired swarm navigation objective under some technical assumptions. We have built 85 Shinerbots, each of size 40.6mm and of cost USD \$20, and designed our Shinerbot swarm navigation platform to implement our algorithm. We have also designed a swarm messaging system and a swarm charging plate for large-scale swarm operations. Our experiments have demonstrated that our Shinerbots collectively navigate toward to an optimal region using minimal sensing and control.

There are a number of opportunities for future work. First, it is of interest to see whether our model in Section II can be enriched to reflect real aspects of Shinerbots. Our current point mass model ignores the physical size of the Shinerbot. More complicated models, such as nonholonomic cart model, will be considered in future work. Second, the current algorithm and its analysis do not take collision avoidance into account. Performance could be enhanced by an algorithm which can avoid collision between Shinerbots, or by analyzing the effects of

collision on the swarm navigation. Third, it would be of interest to extend the functions of our Shinerbots for better control.

Acknowledgements The authors would like to thank Xinhui Fang, Kwok Bun Cheng for their help in soldering electronic components onto the PCBs and for assembling the Shinerbots; Katie Carroll for her help in calibrating the Shinerbots and Matthew Peretic for his help in editing the paper.

References

- Alkilabi, M. H. M., Narayan, A., and Tuci, E. (2017). Cooperative object transport with a swarm of e-puck robots: robustness and scalability of evolved collective strategies. *Swarm Intelligence*. <https://doi.org/10.1007/s11721-017-0135-8>.
- Balch, T. and Arkin, R. C. (1998). Behavior-based formation control for multi-robot teams. *IEEE Trans. Robot. Autom.*, 14(6):926–939.
- Berdahl, A., Torney, C. J., Ioannou, C. C., Faria, J. J., and Couzin, I. D. (2013). Emergent sensing of complex environments by mobile animal groups. *Science*, 339(6119):574–576.
- Bonani, M., Longchamp, V., Magnenat, S., Rétornaz, P., Burnier, D., Roulet, G., Vaussard, F., Bleuler, H., and Mondada, F. (2010). The MarXbot, a miniature mobile robot opening new perspectives for the collective-robotic research. In *Proceedings of 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-2010)*, pages 4187–4193. IEEE.
- Castello, E., Yamamoto, T., Libera, F. D., Liu, W., Winfield, A. F. T., Nakamura, Y., and Ishiguro, H. (2016). Adaptive foraging for simulated and real robotic swarms: the dynamical response threshold approach. *Swarm Intelligence*, 10(1):1–31.
- Couceiro, M. S., Figueiredo, C. M., Luz, J. M. A., Ferreira, N. M., and Rocha, R. P. (2011). A low-cost educational platform for swarm robotics. *International Journal of Robots, Education and Art*, 2(1):1–15.
- Desai, J. P., Ostrowski, J., and Kumar, V. (2001). Modeling and control of formations of nonholonomic mobile robots. *IEEE Trans. Robot. Autom.*, 17(6):905–908.
- Diaz-Mercado, Y., Lee, S. G., and Egerstedt, M. (2017). *HumanSwarm Interactions via Coverage of Time-Varying Densities*, chapter Trends in Control and Decision-Making for HumanRobot Collaboration Systems, pages 357–385.
- Dimakis, A. G., Kar, S., Moura, J. M. F., Rabbat, M. G., and Scaglione, A. (2010). Gossip algorithms

- for distributed signal processing. *Proceedings of the IEEE*, 98(11):1847–1864.
- Elor, Y. and Bruckstein, A. M. (2014). “Robot Cloud” gradient climbing with point measurements. *Theoretical Computer Science*, 547:90–103.
- Farrow, N., Klingner, J., Reishus, D., and Correll, N. (2014). Miniature six-channel range and bearing system: algorithm, analysis and experimental validation. In *Proceedings of 2014 IEEE International Conference on Robotics and Automation (ICRA-2014)*, pages 6180–6185. IEEE.
- Gazi, V. (2005a). Formation control of a multi-agent system using nonlinear servomechanism. *Int. J. Control*, 78(8):554–565.
- Gazi, V. (2005b). Swarm aggregations using artificial potentials and sliding mode control. *IEEE Trans. Robot. Autom.*, 21(6):1208–1214.
- Group, I. . W. et al. (1997). Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. *IEEE*.
- Hendrickx, J. M. and Tsitsiklis, J. N. (2013). Convergence of type-symmetric and cut-balanced consensus seeking systems. *IEEE Trans. Autom. Control*, 58(1):214–218.
- Heng, L. and Gao, G. X. (2014). Navigating robot swarms using collective intelligence learned from Golden Shiner fish. In *Proc. Collective Intelligence Conference (CI2014)*.
- Hilder, J., Naylor, R., Rizih, A., Franks, D., and Timmis, J. (2014). *The pi swarm: A low-cost platform for swarm robotics research and education*, pages 151–162. Springer.
- Jadbabaie, A., Lin, J., and Morse, A. S. (2003). Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001.
- Kernbach, S., Häbe, D., Kernbach, O., Thenius, R., Radspieler, G., Kimura, T., and Schmickl, T. (2013). Adaptive collective decision-making in limited robot swarms without communication. *The International Journal of Robotics Research*, 32(1):35–55.
- Koren, Y. and Borenstein, J. (1991). Potential field methods and their inherent limitations for mobile robot navigation. In *Proc. IEEE International Conf. Robotics Automation*, pages 1398–1404.
- Lin, J., Morse, A. S., and Anderson, B. D. O. (2007). The multi-agent rendezvous problem. Part 1: the synchronous case. *SIAM J. Control Optim.*, 46(6):2096–2119.
- Luo, E., Fang, X. H., Ng, Y., and Gao, G. X. (2016). Shinerbot: Bio-inspired collective robot swarm navigation platform. In *Proc. Institute of Navigation GNSS+ conference (ION GNSS+ 2016)*.
- Merheb, A.-R., Gazi, V., and Sezer-Uzol, N. (2016). Implementation studies of robot swarm navigation using potential functions and panel methods. *IEEE Trans. Mechatronics*, 21(5):2556–2567.
- Mondada, F., Franzi, E., and Guignard, A. (1999). The Development of Khepera. In *Experiments with the Mini-Robot Khepera, Proceedings of the First International Khepera Workshop*, HNI-Verlagsschriftenreihe, Heinz Nixdorf Institut, pages 7–14.
- Moreau, L. (2004). Stability of continuous-time distributed consensus algorithms. In *Proc. 43rd IEEE Conf. Decision Control*, pages 3998–4003.
- Olfati-Saber, R. (2006). Flocking for multi-agent dynamic systems: algorithms and theory. *IEEE Trans. Autom. Control*, 51(3):401–420.
- Olfati-Saber, R., Fax, J. A., and Murray, R. M. (2007). Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233.
- Prorok, A., Hsieh, M. A., and Kumar, V. (2015). Fast redistribution of a swarm of heterogeneous robots. In *Proc. International Conf. Bio-inspired Information Communications Technologies*.
- Rubenstein, M., Ahler, C., and Nagpal, R. (2012). Kilo-bot: A low cost scalable robot system for collective behaviors. In *Proceedings of 2012 IEEE International Conference on Robotics and Automation (ICRA-2012)*, pages 3293–3298. IEEE.
- Saska, M., Vonasek, V., Chudoba, J., Thomas, J., Loianno, G., and Kumar, V. (2016). Swarm distribution and deployment for cooperative surveillance by micro-aerial vehicles. *J. Intelligent Robotic Systems*, 84(1):469–492.
- Schroeder, A., Ramakrishnan, S., Kumar, M., and Trease, B. (2017). Efficient spatial coverage by a robot swarm based on an ant foraging model and the Lévy distribution. *Swarm Intelligence*, 11(1):39–69.
- Semiconductors, V. (2011). TEMD7100X01 silicon PIN photodiode.
- Sturza, M. A. (1988). *Navigation System Integrity Monitoring Using Redundant Measurements*. Wiley Online Library.
- Torrellas Socastro, S. (2013). Cognitive stimulation through task-oriented telerobotics. *Master’s Thesis*.
- Vartholomeos, P. and Papadopoulos, E. (2006). Analysis, design and control of a planar micro-robot driven by two centripetal-force actuators. In *Proceedings of 2006 IEEE International Conference on Robotics and Automation (ICRA- 2006)*, pages

649–654. IEEE.

- Waydo, S. and Murray, M. (2003). Vehicle motion planning using stream functions. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 2484–2491.
- Wu, W. and Zhang, F. (2016). A speeding-up and slowing-down strategy for distributed source seeking with robustness analysis. *IEEE. Trans. Control Network Systems*, 3(3):231–240.



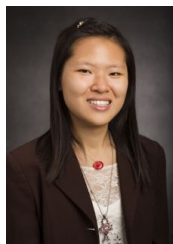
Enyu Luo is a Master's student in the Electrical and Computer Engineering Department at the University of Illinois at Urbana-Champaign. He received his B.S. degree in electrical engineering from the same university. His research interests include robotics and UAVs.



Ji Liu received the B.S. degree in information engineering from Shanghai Jiao Tong University, Shanghai, China, in 2006, and the Ph.D. degree in electrical engineering from Yale University, New Haven, CT, USA, in 2013. He is currently a Postdoctoral Research Associate at the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL, USA. His current research interests include distributed control and computation, multi-agent systems, social networks, epidemic networks, and power networks.



Alexandra Bacula is an undergrad in her junior year in Aerospace Engineering at UIUC. She has previously done undergraduate research work in the Aerospace Robotics and Controls Laboratory under Professor Soon Jo Chung. Recently, she has worked on the Shinerbots project under Professor Graec Gao and graduate student Enyu Luo. Alexandra has also participated in two of NASA's rocketry competitions, and was a summer intern at Jet Propulsion Laboratory working on fast motion planning.



Yuting Ng is a graduate student in the Aerospace Engineering Department at the University of Illinois at Urbana-Champaign. She received her B.S. degree in electrical engineering, graduating with university honors, from the same university. Her research interests are in advanced signal tracking, navigation, control, robotics, RADAR and UAVs.



Tamer Başar is with the University of Illinois at Urbana-Champaign (UIUC), where he holds the academic positions of Swanlund Endowed Chair; Center for Advanced Study Professor of Electrical and Computer Engineering; Research Professor at the Coordinated Science Laboratory; and

Research Professor at the Information Trust Institute. He is also the Director of the Center for Advanced Study. He received B.S.E.E. from Robert College, Istanbul, and M.S., M.Phil, and Ph.D. from Yale University. He is a member of the US National Academy of Engineering, the European Academy of Sciences, and Fellow of IEEE, IFAC and SIAM, and has served as president of IEEE CSS, ISDG, and AACC. He has received several awards and recognitions over the years, including the IEEE Control Systems Award, the highest awards of IEEE CSS, IFAC, AACC, and ISDG, and a number of international honorary doctorates and professorships. He has over 700 publications in systems, control, communications, and dynamic games, including books on non-cooperative dynamic game theory, robust control, network security, wireless and communication networks, and stochastic networked control. He was the Editor-in-Chief of *Automatica* between 2004 and 2014, and is currently editor of several book series. His current research interests include stochastic teams, games, and networks; security; and cyber-physical systems.



Grace Xingxin Gao received her B.S. degree in mechanical engineering and her M.S. degree in electrical engineering from Tsinghua University, Beijing, China in 2001 and 2003. She received her PhD degree in electrical engineering from Stanford University in 2008. From 2008 to 2012, she was a research

associate at Stanford University. Since 2012, she has been an assistant professor in the Aerospace Engineering Department at University of Illinois at Urbana-Champaign. Her research interests are systems, signals, control, and robotics. She is a senior member of IEEE and a member of ION.