ISR TECHNICAL REPORT 2014-08

RSSI-Based Rendezvous on the Tiny Terrestrial Robotic Platform (TinyTeRP)

Han Beol Jang, Roberto Villalba, Derek Paley, and Sarah Bergbreiter

RSSI-Based Rendezvous on the Tiny Terrestrial Robotic Platform (TinyTeRP)*

Han Beol Jang[1], Roberto D. Villalba[2], Derek Paley[3], and Sarah Bergbreiter [4]

***Abstract*— The TinyTeRP is a centimeter-scale, modular wheeled robotic platform developed for the study of swarming or collective behavior. This paper presents the use of TinyTeRPs to implement collective recruitment and rendezvous to a fixed location using several RSSI-based gradient ascent algorithms. We also present a redesign of the wheel-based module with tank treads and a wider base, improving the robot's mobility over uneven terrain and overall robustness. Lastly, we present improvements to the open source C libraries that allow users to easily implement high-level functions and closed-loop control on the TinyTeRP.**

I. INTRODUCTION

The use of swarm intelligence in robotics systems is an approach that allows for the emergence of complex behaviors through the collaboration of large numbers of simple, inexpensive robots adhering to straightforward behavioral rules. Swarm robotics has demonstrated potential in areas where the number of robots required scale up or down over time, dangerous situations require cheap and expendable agents, large physical spaces have to be evaluated, and system redundancy is required in order to prevent catastrophic failure due to the removal of individual agents [1].

Sugawara and Watanabe demonstrated simple robotic swarming behavior in the context of food-foraging. By causing robots to be attracted to other robots that had found a food source, the robots were collectively able to converge on and consume a given food source faster than they would otherwise have done [2]. This positive feedback effect is characteristic of swarming algorithms, where the behavior of individual agents is improved by other agents that have already reached the desired state.

Studying swarming algorithms such as Sugawara and Watanabe's requires a large number of robots that are small, cost-effective, and highly mobile due to the space and budget constraints of most research facilities. Several centimeter and millimeter-scale platforms have been developed for swarming; the smallest so far is the 3×3 mm i-Swarm robotic platform, which uses an IR transmitter and receiver to sense the position of objects and other robots [3]. Another platform, Jasmine, is 3×3 cm and also uses six IR sensors for obstacle detection, communication, and crude shape characterization [4], [5], [6]. This paper presents the Tiny Terrestrial Robotics Platform (TinyTeRP), a small platform developed for the study and implementation of swarming and collective behavior. The TinyTeRP allows stacking of several different sensing boards atop the base board, and programs may use different code libraries and capabilities based on which boards are present.

*This work was sponsored by the National Science Foundation.

[1]Han Beol Jang is with the Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, USA infrared@mit.edu
[2]Roberto D. Villalba is with the Department of Computer Science, Cornell University, Ithaca, NY, USA rdv28@cornell.edu
[3]Derek Paley is with the Department of Aerospace Engineering, University of Maryland, College Park, MD, USA dpaley@umd.edu
[4]Sarah Bergbreiter is with the Department of Mechanical Engineering, University of Maryland, College Park, MD, USA sarahb@umd.edu

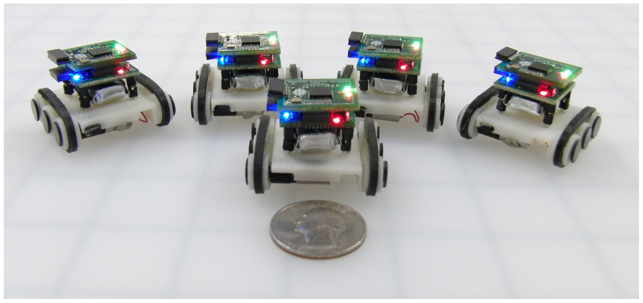


Fig. 1: A group of fourth generation TinyTeRPs.

The sensors currently available for the TinyTeRP include a radio with a received signal strength indicator (RSSI) on the baseboard, and an integrated inertial measurement unit (IMU) board with a three-axis accelerometer and three-axis gyroscope. Part of this paper is dedicated to the redesign of the TinyTeRP's chassis, as it was necessary to improve the robustness of the platform in order to test our rendezvous algorithms. Our design objectives with the TinyTeRP were to improve stability and maneuverability while maintaining a small profile. The redesigned tank-treaded version provided the necessary improvements, and was used for our experimental validations.

Unlike the i-Swarm and the Jasmine robots, the TinyTeRP does not rely on IR sensors to communicate with and localize other robots. While the directionality of IR receivers provides reliable means of localizing other agents in indoor environments, the sensor is highly susceptible to lighting conditions and is less reliable as a means of communication between agents than a radio. This paper explores the potential for using RSSI in platforms equipped with radios in order to localize and rendezvous to a desired location or agent. We present experimental results for the following four algorithms: (1) Straight Line Ascent, (2) Gradient Estimation, (3) E. coli-inspired Gradient Ascent, and a (4) Random Turn algorithm that will be used for comparison.

The contributions of this paper are: (1) the design and fabrication of a swarming platform with a small footprint, low cost, and high maneuverability; (2) swarming algorithms for performing rendezvous through the use of a radio's RSSI; and (3) open-source software libraries that simplify the programming of the TinyTeRPs.

The rest of the paper is organized as follows: section II discusses the previous work that has been done on the TinyTeRPs, section III presents the new chassis design and open source support for the platform, section IV provides a background on the radio and characterizes the angular and spatial dependencies of RSSI, section V introduces the algorithms that were used and their implementation, section VI presents the experimental validation for each and the emergence of possitive feedback, and section VII summarises the work that has been presented.

II. PREVIOUS WORK ON THE TINYTERP

This section describes the electrical hardware, chassis, and existing closed-loop controller of the TinyTeRP version 3.0 [7], and presents the major issues addressed in the chassis redesign.

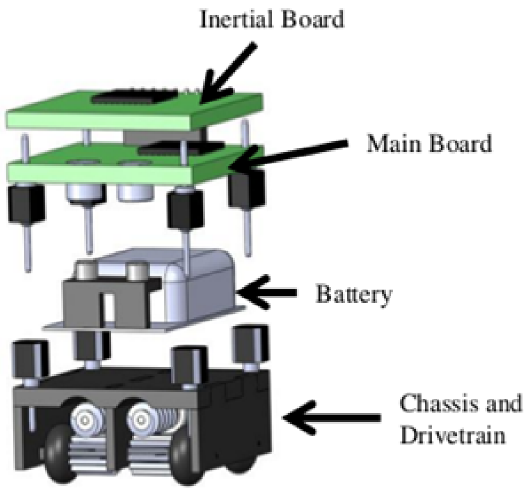


Fig. 2: A third generation TinyTeRP.

A. Hardware

The TinyTeRP base module contains a CC2533 microcontroller (Texas Instruments) with an integrated 2.4 GHz RF radio. The inertial sensing module contains an MSP430 microcontroller and a MPU 6000 for inertial sensing.

B. Chassis

The TinyTeRP v3.0 chassis was a rectangular 17 $\times$ 18 mm 3D-printed base sized to fit under the circuit board. While the form factor was small, light, and used little material, the weight of the boards caused the design to be top-heavy and susceptible to falling over when stopping suddenly or turning at high velocity. Furthermore, the drivetrain assembly required alignment of the worm and spur gear; imperfect alignment often resulted in jams or slipping.

C. Closed loop control for straight-line travel

Sabelhaus et. al (2013) demonstrated closed-loop control in the TeRPs using the yaw axis gyroscope readings from the IMU. However, the controller only allowed for straight-line motion, in which one wheel was given a fixed voltage and the voltage to the other wheel was varied until the yaw angular velocity was sensed to be at zero [8]. While this controller produced straight-line motion, it did not allow turning at a desired nonzero angular velocity.

III. PLATFORM IMPROVEMENTS

This section discusses the design choices for the new platform as well as the development of an open-source software library for the TinyTeRPs.

A. Chassis redesign

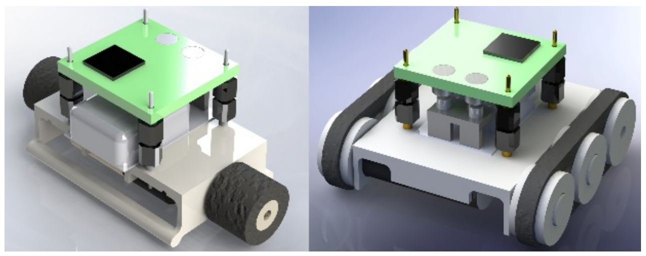


Fig. 3: The final wheeled and treaded prototypes.

In order to address the mechanical issues with the previous generation of platform, we designed two prototypes, as shown in Fig. 3. Both of these prototypes make use of geared motors in order to eliminate the need for additional gears and the possibility of misalignment. In addition, both of the prototypes had a wider base in order to improve the stability of the platform. After evaluating both the platforms based on stability, ease of assembly, power consumption and mobility, we chose to use the treaded version. The chosen platform demonstrates drastic improvements in stability and maneuverability over uneven terrain, with the least susceptibility to human error during the assembly phase.

B. Software Upgrade

To provide current and future users with a simpler coding experience, we have developed open-source software libraries in C for programming the individual hardware components of the TinyTeRP, allowing programmers to code at a higher level of abstraction. The libraries provide support for initialization of all the individual components such as the radio and the motors. The radio and inter-module communication libraries allow robust transmission and reception of packets. The motor library provides support for automatic dead-band detection (range of motor voltages that are too low to cause movement), as well as adherence to a desired angular velocity.

Angular velocity control was achieved through the use of a proportional-integral-derivative (PID) closed-loop controller that uses the IMU gyroscope, and thus requires the inertial sensing board. Experimental results of the PID are presented in Fig. 4; it is evident that the robot is capable of converging to the desired velocities within less than two seconds even after a large initial jerk.

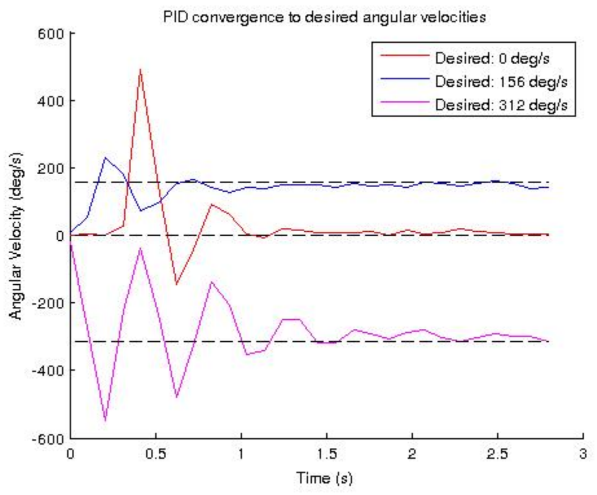


Fig. 4: Angular velocity data from the IMU in deg/s for three trials with PID controller, and desired angular velocities of 0, 156, and 312 deg/s.

IV. RADIO

This section will briefly describe the TinyTeRP radio and explore the angular and spatial dependence of RSSI. We find that nonuniformities in the RSSI signal due to angular dependence are significantly reduced with the presence of multiple radio beacons over which the signals are averaged. This signal improvement suggests the potential for enhanced behavior through the cooperation of agents in multi-robot RSSI-based rendezvous.

As previously stated, the CC2533 microcontroller comes with an on-board radio that is primarily used as a means of communication from robot to robot. However, the radio module on the chip also provides an RSSI with every radio packet received. This value measures the strength or gain of the incoming signal, and is usually given in decibels as referenced to one milliwatt (dBm); the dBm value is calculated as $RSSI_{dBm} = RSSI - 73$. The RSSI values in dBm range from -128 to 0, where 0 represents perfect transmission of the signal. The signal strength increases as the receiver gets closer to the transmitter, suggesting its use as a crude metric for distance. While the sensor provides steady readings in a non-changing environment, the signal strength is susceptible to a lot of noise from interference caused by environmental changes as well as changes in the robot's orientation. It should be noted that the rest of this paper does not convert the RSSI into dBm; instead we will use the raw values from the radio.

A. RSSI Characterization and Smoothing

Previous work on the TinyTeRP radio characterization indicated a dependence of the RSSI signal on the relative angular orientation of the sender and receiver. Further

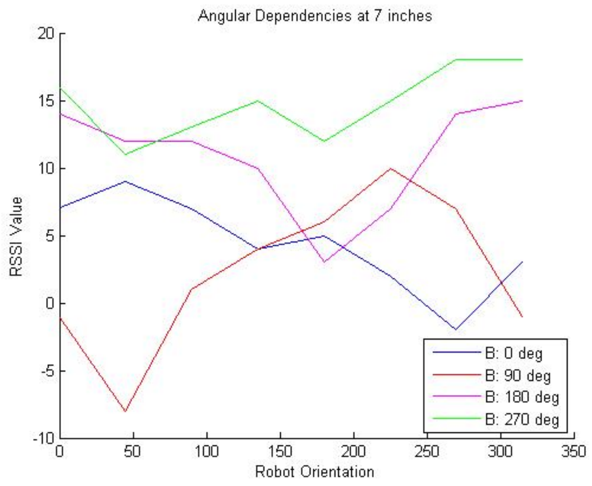


Fig. 5: RSSI at a fixed distance for various beacon and receiver orientations.

experiments reveal that the range of variation for a robot rotating in place can be as large as 17, as shown in Fig. 5. This amount is substantial compared to the variations in our readings due to actual movement about the testing environment, where RSSI values range from approximately -10 to 40.

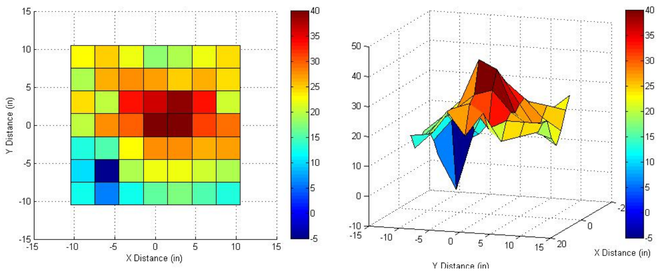
Fig. 6 further characterizes the variations in the signal over a two dimensional plane, with the beacon robot(s) broadcasting radio packets in the center. It is clear that RSSI is not only dependent on distance; were this the case, all of the values would be equivalent for a fixed distance, and RSSI would decrease uniformly with an increase in radial distance from the source. However, the decrease is not uniform and there are also unexpected peaks and valleys in the RSSI field.

The radio characterization experiment shown in Fig. 6b demonstrates that averaging out the values of several beacons clustered together can result in some smoothing of the RSSI field. As the number of transmitters increases, the surface becomes closer to monotonically decreasing (the ideal case). When this result is exploited by a rendezvous algorithm, a positive feedback effect emerges, as we will demonstrate in section VI. As the number of robots who have found the beacon (and are transmitting a signal) increases, so does the quality of RSSI as a tool for measuring the distance to the goal as well as the speed with which additional agents achieve rendezvous.

V. RENDEZVOUS ALGORITHMS

Here we will discuss the various algorithms that have been implemented and evaluated for performing RSSI-based gradient ascent. The goal of these algorithms is to drive the agents towards a beacon or group of beacons, which are also robots transiting radio data. All of these algorithms benefit from the improved RSSI data that results from multi-beacon averaging, which enforces positive feedback as robots who have found the beacon(s) also begin to transmit packets.

(a) RSSI field with one beacon at the center



(b) RSSI field with three beacons at the center

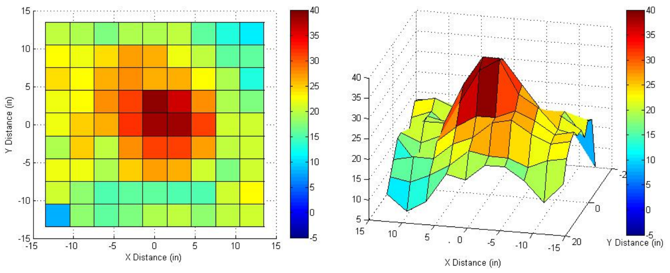


Fig. 6: Distribution of RSSI values over 2D plane with a) a single beacon robot and b) average values from four robots at the center. Fixed orientation of 0° between the beacon and receiving robot.

1) Straight Line Ascent: In this algorithm the robot attempts to find the beacon(s) by driving in a straight line and turning whenever it detects that it is no longer moving toward the beacon(s). As the robot drives straight, it keeps track of the highest RSSI observed so far, $RSSI_{max}$. If at any time step the robot detects an RSSI value $RSSI_{current}$ that is lower than $RSSI_{max}$ by a noise threshold $thresh_t$, the robot will stop and rotate by a fixed angle $theta$ before continuing in a straight line. It is also important that once the robot makes a turn, $RSSI_{max}$ must be reset to the lowest possible value, in our case -128. The reset is necessary because it is possible that the robot will not get an RSSI as high as the previous $RSSI_{max}$ with its new orientation due to the angular dependence of RSSI. To further protect against noise, we also implemented a sliding window average of the last four RSSI values, which we cleared out upon making a turn.

2) Gradient Estimation: This algorithm causes the robot to drive in arcs of small radius, turning left or right depending on the value of the estimated RSSI gradient vector. The gradient vector at a given location is defined by the direction of greatest increase of the RSSI field, towards which the robot tries to turn. The robot is able to estimate the direction of greatest RSSI increase by fitting a plane onto the three-dimensional coordinates of its three most recent RSSI values in $X_b, Y_b, RSSI$, as shown in Fig. 7. Knowing its own trajectory, it is able to determine whether the direction of the gradient is to the right or left and turns in that direction.

The calculations for a left hand turn are as follows:

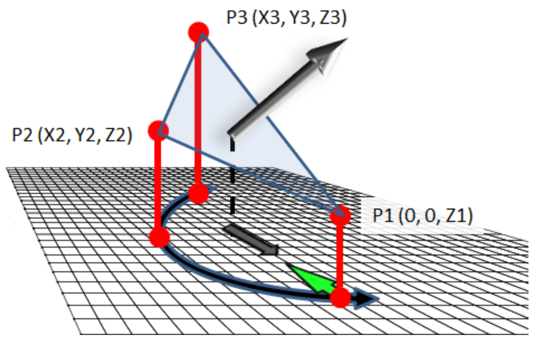


Fig. 7: This figure shows the robot's path as a blue arrow, the RSSI readings as red dots, and the direction of greatest RSSI increase as a green arrow.

$$\begin{aligned} \vec{V}_{12} &= P_2 - P_1 \\ \vec{V}_{13} &= P_3 - P_1 \\ \vec{N} &= \vec{V}_{13} \times \vec{V}_{12} \\ N_y &= -(V_{13_x} \cdot V_{12_z} - V_{13_z} \cdot V_{12_x}) \\ D &= -N_y \end{aligned}$$

The points P1, P2, P3 are all in the body-fixed frame of the robot, where the robot sits at (0, 0) and the X-axis is aligned with its orientation. We begin by calculating $\vec{V}_{12}$ and $\vec{V}_{13}$ which are then used to calculate the vector normal to the surface, $\vec{N}$. Since all of these vectors are in the body-fixed frame, we only need the Y component of $\vec{N}$, N_y, in order to choose the direction in which the robot should move. If D is positive the robot should turn left, and if it is negative the robot should turn right. As mentioned, the above calculations are for a robot making a left turn; if the robot is making a right turn then $D = N_y$, with the rest of the math unchanged.

This algorithm requires that the robot is able to keep track of its three most recent readings as well as its current turning direction, and to estimate the relative locations of the last three readings. If at any point the robot changes direction, its history of the two oldest RSSI values must be discarded and the robot must wait two more iterations before estimating the gradient again. Since our robots move in arcs of consistent size, the relative locations of the oldest two readings (the X and Y components of $\vec{V}_{13}$ and $\vec{V}_{12}$) are hard-coded and estimated by observing the robots path.

Note that if the three most recent RSSI readings are the same, N_y will have a magnitude of 0, giving the robot no information about the direction of the gradient. To prevent the robot from using gradient estimations that are less confident, we set a lower threshold for the magnitude of N_y below which the robot does not use the estimation, and continues moving in the same direction.

3) E. coli-Inspired Gradient Ascent: The third algorithm that we implement is a bio-inspired approach based on E. coli bacteria. The bacterium probabilistically decides to "tumble", turning in a random direction, or "run", moving in a straight line, in order to move up a concentration gradient towards a source of sugar. The E. coli bacterium tumbles with a higher probability when sensing a low concentration of sugar or a negative change in sugar concentration; thus it is more likely to change direction when far away from sugar or moving away from it, in hopes of finding a more favorable direction. If it is closer to the sugar or moving towards it, the bacterium is more likely to continue moving in that direction. In order to imitate this behavior on the TinyTeRP, we use a weighted combination of RSSI and change in RSSI over one time step to define the probability of moving straight or turning.

$$t = w_1(RSSI_{current} + a) + w_2 RSSI_{delta}$$

The value a shifts the range of the RSSI term so that it is centered around 0; w_1, w_2 are the weights for the current RSSI and change in RSSI, respectively. The value t is then scaled and shifted so that its range matches up with the range of our random number generator. At each iteration of the algorithm, t and a random number, $rand$, are calculated. If $rand$ is greater than t, the robot turns; if it is smaller, the robot moves in a straight line.

VI. Rendezvous Performance

This section includes a summary of the experimental results gathered from running our algorithms on the TinyTeRP. The experiments are performed on a flat (40×40) in. surface with walls along the boundaries. The beacon(s) are placed at the center with a randomized orientation at each trial. The other robots are placed at approximately 30 in. away with randomized orientations and positions. If any of the robots hit the wall, they move backward, turn, and continue running the algorithm. Robots are sent to search for the beacon and timed individually. Each test is run twenty or more times, and the average runtimes are presented in Fig. 8.

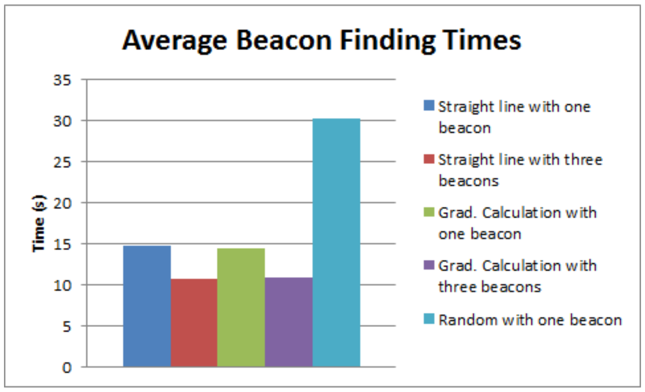


Fig. 8: Average runtimes. The E. coli runtime of 124.9s has been omitted for scaling purposes.

A. Control: Random Turning

A random turning algorithm is used as a control for experimental validation of our rendezvous algorithms. Randomly turning robots move in small arcs that turn left or right with a fifty percent chance of selecting either at each time step. Since no sensor input is involved (other than detecting when the beacon is reached), we did not test this algorithm with multiple beacons. The average runtime is 30.2 seconds, with a standard deviation of 16.6.

Although the theoretical runtime with no walls on the arena is unbounded, the algorithm performed fairly well due to the constrained size of the setup and the presence of walls. Random turning at a fixed interval allowed the robot to sweep significant portions of the space within a short amount of time.

B. Straight Line Ascent

The straight line ascent algorithm is the most successful overall. With a single beacon, the average runtime is 14.7 seconds with a standard deviation of 10.9. With three beacons, the average runtime is 10.6 with a standard deviation of 4.6, showing both faster and more consistent movement towards the goal. The success of this algorithm is likely due to consistency of orientation during execution. The straight movement largely removes the variations in RSSI shown in Fig. 5 and allows the RSSI to maintain a more constant field distribution.

It is also worth noting that this algorithm resulted in considerably fewer collisions with the boundary walls than others, suggesting that it would further outperform the others in the absence of an enclosed environment.

C. Gradient Estimation

The Gradient Estimation algorithm demonstrated performance similar to that of Straight Line Ascent. The average time for the single beacon experiment is 14.4 seconds with a standard deviation of 15.3, while the three-beacon experiment has an average of 10.8 seconds with a standard deviation of 8.1. Although the average runtimes are similar, the standard deviations indicate that Straight Line Ascent has more consistent performance.

The sensor characterization suggests that the angular dependence of the RSSI could be the cause for its less consistent performance. Since the algorithm requires the robot to continuously change its orientation, the calculation of the gradient becomes compromised by the large shifts in RSSI that result from the radio's angular dependence. If changes in the robot's orientation cause the RSSI to also change, the values shown in Fig. 6 should also shift; accounting for different shifts in the RSSI field in between time steps would be difficult, and gradient calculations must be robust to these shifts in order to be reliable.

D. E. coli-Inspired Gradient Ascent

The E. coli inspired algorithm had an average runtime of 124.9 seconds to reach a single beacon, performing much worse than the random algorithm in the bounded test arena. The E. coli robot was unable to cover as much distance as the random robot before changing direction, and the probabilistic nature of decisions to turn introduced unfavorable movements in the overall behavior of the robots. The equations and parameters that we used were not effective in biasing the robot to move towards the beacon. Although it is possible that different equations for tumbling or better-tuned parameters could result in improved performance of this algorithm, it is not likely to result in fast convergence at the beacon(s).

E. Validation of Positive Feedback

Both of the algorithms that were tested with varying numbers of beacons demonstrated significant improvement in execution time and consistency when the RSSI was calculated as the average of all received RSSI. These results support the existence of positive feedback in RSSI-based rendezvous as robots that have reached the goal contribute to the signal quality. This effect should be scalable, and the overall quality of the data should further improve with larger swarm sizes.

VII. CONCLUSIONS

In this work, we present a new design for the TinyTeRP chassis, along with several algorithms for performing RSSI-based gradient ascent. The new platform has reduced susceptibility to human error during the build process, as well as improved stability and maneuverability. Along with the new design, we have developed the open-source libraries for the robot to facilitate easy implementation of new swarming and collaborative algorithms. Lastly, we have presented experimental evaluations of the performance of various rendezvous algorithms in an enclosed area, and compared the average runtimes to those of a random turning algorithm. We predict that RSSI-based ascent algorithms that move in straight lines are more appropriate for the TinyTeRP than ones that move along curved paths, due to the angular dependence of RSSI. Finally, we observe a positive feedback effect in a rendezvous scenario where robots that have found the beacon also transmit radio signals. The availability of multiple transmissions improves the quality of the signal sensed by other agents, resulting in more consistent behavior and shorter convergence times at the beacon.

ACKNOWLEDGMENT

We would like to thank the NSF REU, Dr. Sarah Bergbreiter, and Dr. Derek Paley for funding and support of this project. We would like to acknowledge Dr. Wereley's lab for use of the 3D printer, and all of the members of the Micro Robotics Lab and Collective Dynamics and Control Laboratory for their help and good company throughout the summer. We would especially like to thank graduate students Ivan Penskiy and Frank Lagor for their expertise and guidance.

REFERENCES

[1] E. ahin, "Swarm robotics: From sources of inspiration to domains of application," *Swarm robotics*, pp. 10–20, 2005. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-540-30552-1_2

[2] K. Sugawara and T. Watanabe, "Swarming robots-foraging behavior of simple multirobot system," *Intelligent Robots and Systems, ...*, no. October, pp. 2702–2707, 2002. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1041678

[3] P. Corradi, O. Scholz, T. Knoll, a. Menciassi, and P. Dario, "An optical system for communication and sensing in millimetre-sized swarming microrobots," *Journal of Micromechanics and Microengineering*, vol. 19, no. 1, p. 015022, Jan. 2009. [Online]. Available: http://stacks.iop.org/0960-1317/19/i=1/a=015022?key=crossref.f9ce27d95b96178d21a33eae4b4a8440

[4] S. Kornienko, O. Kornienko, and P. Levi, "Minimalistic approach towards communication and perception in microrobotic swarms," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 4005–4011. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1545594

[5] ——, "Collective AI : context awareness via communication," pp. 1464–1470, 2005.

[6] S. Kernbach, "Encoder-free odometric system for autonomous microrobots," *Mechatronics*, vol. 22, no. 6, pp. 870–880, Sept. 2012. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0957415812000785

[7] K. Gessler, A. Sabelhaus, and S. Bergbreiter, "ISR TECHNICAL REPORT 2013-12 REU : Improving Straight Line Travel in a Miniature Wheeled Robot," Tech. Rep., 2013.

[8] A. P. Sabelhaus, D. Mirsky, L. M. Hill, N. C. Martins, and S. Bergbreiter, "TinyTeRP : A Tiny Terrestrial Robotic Platform with Modular Sensing," Tech. Rep.