Intelligent Robotic IoT System (IRIS) Testbed

Jason A. Tran*, Pradipta Ghosh*, Yutong Gu*, Richard Kim*, Daniel D'Souza*,
Nora Ayanian†, and Bhaskar Krishnamachari*

***Abstract*— We present the Intelligent Robotic IoT System (IRIS), a modular, portable, scalable, and open-source testbed for robotic wireless network research. There are two key features that separate IRIS from most of the state-of-the-art multi-robot testbeds. (1) Portability: IRIS does not require a costly static global positioning system such as a VICON system nor time-intensive vision-based SLAM for its operation. Designed with an inexpensive Time Difference of Arrival (TDoA) localization system with centimeter level accuracy, the IRIS testbed can be deployed in an arbitrary uncontrolled environment in a matter of minutes. (2) Programmable Wireless Communication Stack: IRIS comes with a modular programmable low-power IEEE 802.15.4 radio and IPv6 network stack on each node. For the ease of administrative control and communication, we also developed a lightweight publish-subscribe overlay protocol called ROMANO that is used for bootstrapping the robots (also referred to as the IRISbots), collecting statistics, and direct control of individual robots, if needed. We detail the modular architecture of the IRIS testbed design along with the system implementation details and localization performance statistics.**

I. Introduction

Recent advancements in affordable technology and hardware miniaturization have materialized the goal of integrating robots into many aspects of human life. To this end, robot augmented wireless communication backbones or robotic wireless networks has become a cutting-edge field of research [1], [2]. Over the last few decades, researchers have developed a range of promising control frameworks, algorithms, and solutions to address a variety of challenges in robotic wireless networks such as connectivity maintenance [3] and robotic router formation [1]. Most of the advancements in these fields of research along with the advances in multi-robot systems research have been confined to the dimensions of theory and simulations due to the time consuming and complex proposition of translating theory to practice. To make the translation attainable, researchers have developed a range of open-source multi-robot testbeds and experimental facilities ([4]–[6]). However, most of these testbeds rely on a controlled facility with a costly, although accurate, global positioning system such as the camera based VICON system that restricts the breadth of real-world experiments. Real-world settings are critical for testing communication protocols, as they are heavily affected by the environment. Another approach utilizes camera-based SLAM to provide location, but that can be time-intensive to train and potentially prone to errors. Most of the state-of-the-art robotic testbeds also do not provide the necessary programmable communication stacks for low power wireless robotic network research. On the other hand, there exist a couple of Internet of Things (IoT) testbeds ([7], [8]) for wireless network research that has mobile components. However, they still lack flexible, controlled mobility for swarm robotic networks research. Thus, there is a demand for a modular, portable, and scalable multi-robot testbed with a programmable low power communication stack and high precision global localization feature that can be set up in arbitrary environments in a matter of minutes.

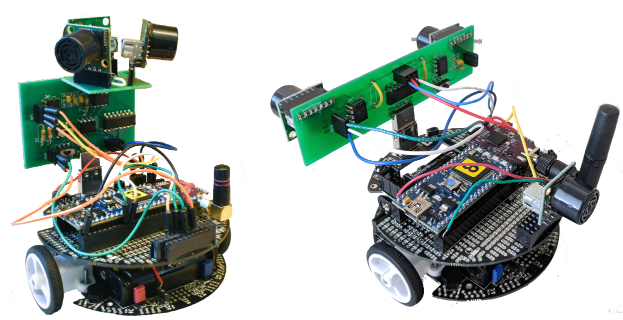


Fig. 1: IRISbot with (left) Omnidirectional TDoA (right) Differential TDoA

Our Contribution: We present the Intelligent Robotic IoT System (IRIS), an in-house modular, low power, scalable, and affordable multi-robot 2D wireless robotic network testbed with a relative/absolute localization system portable to random experimental environments without pre-deployed, costly localization systems (e.g. VICON). To support portability, the IRIS testbed uses a popular class of indoor localization techniques called Time-Difference of Arrival (TDoA) [9] localization that employs a combination of Radio Frequency (RF) devices and Ultrasound devices for centimeter level accuracy. Another key feature of the IRIS testbed is the inherent capability of low power IPv6 over IEEE 802.15.4 ad-hoc routing and peer to peer communication with a programmable communication stack to experiment with wireless robotic networking protocols. The system is also equipped with a new protocol called ROMANO (Robotic Overlay coMmunicAtioN prOtocol) which

All authors are with the University of Southern California, Los Angeles, CA, USA.
*Ming Hsieh Department of Electrical Engineering
†Department of Computer Science
Email: {jasontra, pradiptg, yutonggu, richartk, dmdsouza, ayanian, bkrishna} @usc.edu.
This work was partially supported by the ARL Cooperative Agreement W911NF-17-2-0196.

is built on top of the well-known publish-subscribe MQTT-SN protocol [10]. ROMANO enables simple robot-to-robot communication and control, and also facilitates authentication, administration, and data collection to speed up swarm robotic experiments. Thus, the robots in the IRIS testbed, also referred to as the IRISbots, have two separate and independent communication features: (1) central server based publish-subscribe communication for standard coordinated robotic research and testbed administration, (2) low power peer-to-peer communication for wireless mobile ad-hoc networking. The IRISbot and the networking software modules have been built using both MBED-OS and RIOT-OS, two popular embedded, real-time operating systems for IoT, which allows for portability across the supported devices under both operating systems. All software and hardware designs along with relevant documentation can be found at *https://tiny.cc/iris-anrg*.

II. Related Works

The range of existing state-of-the-art testbeds and multi-robot systems can be divided into two verticals: multi-robot testbeds and Internet of Things testbeds.

A number of larger footprint multi-robot testbeds have been developed over the years [4] such as the HoTDeC [11] and the Mobile Emulab [5]. However, most of these testbeds consist of costly, large footprint robots that do not have the necessary hardware and software for low power wireless robotic networking protocol research. There also exists some small footprint robotic testbeds for swarm robotics research such as the Robotarium ([6], [12]) or the Kilobot testbed [13]. While these testbeds are relatively cheap and much smaller in size, they lack the necessary programmable communication stack for low power robotic wireless network research. Above all, most of these testbeds require a costly fixed global positioning system that rely on cameras such as the VICON system that are not portable to arbitrary settings. What separates IRIS from these systems is that the IRISbots are equipped with a programmable low power IEEE 802.15.4 compliant radio with relative and absolute localization capabilities without the need for a fixed, pre-deployed global localization system. This makes the IRIS testbed self-sustaining and highly portable to random experimental environments. Moreover, the size ($\approx 9.5cm$ in diameter) and cost ($\approx \$350$) of the IRISbots make the testbed much more scalable in indoor settings than the larger footprint robotic systems. For a detailed overview of the existing state-of-the-art multi-robot testbeds, interested readers are referred to [4], [14].

Along the second vertical of related works, there exists a range of testbeds for low power wireless sensor networks and IoT systems [7]. Some of the mentionable testbeds include the FIT IoT-Lab [8], Tutornet [15], and CONET [16]. For a more detailed overview of such testbeds, please refer to a survey paper like [7] and the references therein. The FIT IoT-Lab is the most recent and related testbed with thousands of static nodes and hundreds of mobile nodes located in multiple buildings across France for IoT communication and networks research. At the time of the preparation of this paper, access to the mobile nodes is limited to fixed trajectories to focus on collecting wireless networking statistics with respect to the numerous static nodes. The CONET testbed also has some mobile nodes. However, the distinguishing factor between these testbeds and IRIS is that IRIS is focused on communication and networking related research in the context of swarm robotics. To this end, IRIS is designed to enable the development of distributed control algorithms that integrate the control of mobility with communication goals.

III. The IRIS Architecture

In this section, we present an overview of the entire IRIS testbed architecture. Like any testbed, there is one central computer that acts as the IRIS server for management purposes. The main component of the IRIS server, as presented in Figure 2, is the management module which is used for bootstrapping and collecting experiment data using ROMANO. This device also acts as a border router for the robots to connect to the management module running in the IRIS server. Because the communication module uses IPv6, the robot nodes can connect to the internet. The server also has a controller module which can be used to control any subset of robot nodes via ROMANO if needed.

Each of the IRISbots at minimum have four modules as illustrated in Figure 2. The localization module is used for *relative and absolute localization* of robots. It has a software API for each robot to switch between *anchor mode or client mode*. Anchor mode is used to make a node regularly beacon RF/ultrasound signals, and client mode is used when a node calculates its distance to any anchor node. In both cases, there is support for two different localization techniques: *Differential and Omni-directional*. This is further detailed in Section III-A. The controller module provides a skeleton for inserting movement controllers that are experiment specific. The management module on the IRISbot facilitates communication with the IRIS server. In addition, it facilitates intra-robot messaging and control, detailed further in Section III-B. The last but most important module is the communication module (detailed in Section III-C). It supports the management module by setting up wireless links and routing network packets across multiple hops if necessary. It also supports the localization module which relies on RF packets to perform TDoA localization.

A. Localization

One of the key features of the IRIS testbed is the unique localization method that does not require any costly and controlled facility setup and supports portability. To this end, we have developed hardware and software for efficient localization by using a well-known localization concept called Time Difference of Arrival (TDoA) based localization [9]. The main idea behind a TDoA system is to have an anchor node simultaneously transmit beacon signals of two *different* propagation speeds (RF and ultrasound in IRIS system). A receiver will receive the two signals at different time instances (say t_r and t_u, respectively) due to the speed

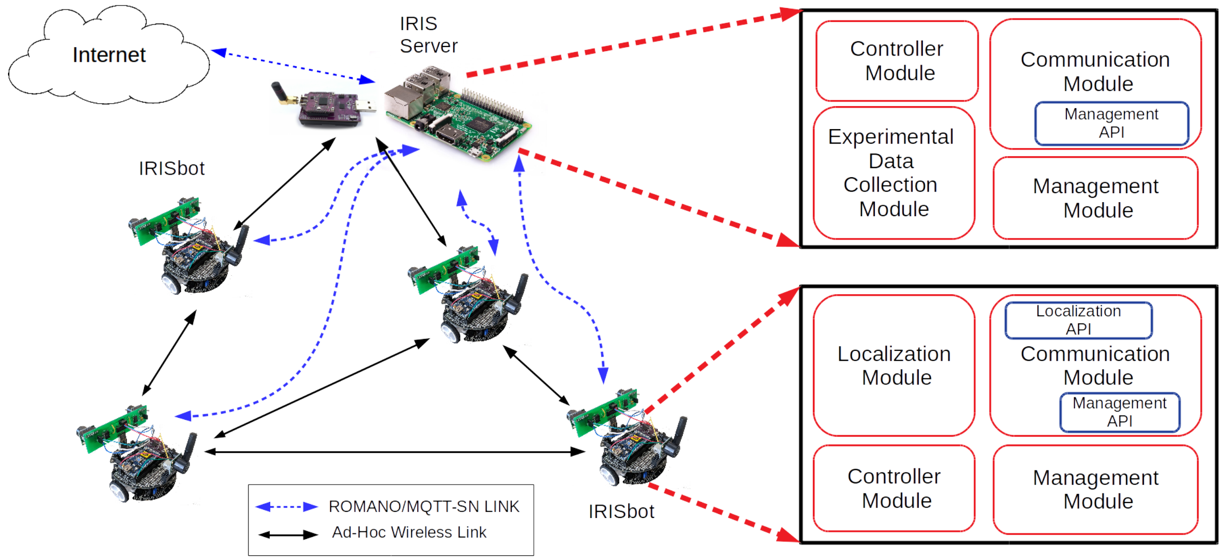


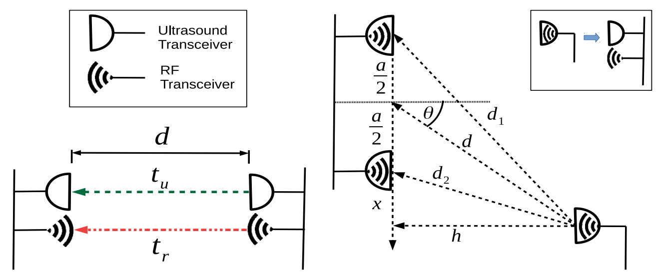
Fig. 2: Illustration of the IRIS Testbed Architecture

differences. The receiver uses the time difference of arrivals to calculate the distance (d) between it and the transmitter using $d = \Delta_v \cdot |t_r - t_u|$ where Δ_v is the difference in the speeds. This is illustrated in Fig. 3a. If the receiver can calculate the distance to at least three anchors not positioned in a line, it can trilaterate its position with respect to the anchors. Conversely, all anchors can also listen for beacons from a client robot and fuse the different TDoA values to trilaterate the position of the client robot. To prevent interference of TDoA beacons, anchors synchronize with one another using time division multiple access, a scheduling algorithm used in wireless networks.

Localization is a well-known and well-studied field of research in the context of robotics and wireless sensor networks [17]. The Global Positioning System (GPS) has become the de-facto standard, but its effectiveness breaks down indoors. For indoor positioning, there exists a range of solutions based on cameras [18], range finders [17], and radio frequency devices [19]. For a detailed survey of such indoor techniques, please refer to an indoor localization related survey [17]. While there exists different options, we choose a RF and ultrasound based approach due to its accuracy (e.g. TDoA achieves centimeter accuracy while received signal strength techniques are on the order of meters [19]), ease of use, cost, and low energy requirement.

While the TDoA localization concept has been around for a while, there are two inherent challenges in implementing a system: (1) ultrasound devices are directional devices which require transceiver pairs to be properly aligned and (2) typical TDoA systems cannot determine precise relative position (*i.e.*, both distance and relative angle) to an anchor without either physical movement or multiple anchors.

In our IRIS testbed, we address both of these issues with two separate solutions. To solve issue (1), we designed a PCB to simultaneously utilize three MaxSonar-AE0 ultrasonic transceivers to create an omnidirectional transceiver. We



(a) Basic TDoA (b) Differential TDoA

Fig. 3: Illustrations of the TDoA Localization Techniques

choose this model in the series because of its widest beam pattern. The ultrasound transceivers are placed at 120 degrees with respect to each other to cover all possible directions on a 2D plane. To simultaneously transmit the ultrasound beacons, all of the transceivers' enable pins are connected to a single GPIO pin of a microcontroller unit (MCU) *on the same board or system on chip with a 802.15.4 radio*. Being on the same board or chip is essential for accurately measuring TDoA values. For receiving, each of the analog envelope outputs of the transceivers are fed into a comparator circuit with a threshold voltage that is configurable in software via a digital to analog converter. The outputs of each comparator is inputted to an OR gate which outputs a single digital signal that represents the arrival of an ultrasonic ping (the line of sight component arrives first). The PCB can be seen attached to the top of a Pololu 3pi robot in Fig. 1(left). We refer to this scheme as the *Omnidirectional Localization* scheme.

To address issue (2), a MCU can poll the three comparator outputs in succession to determine the first sensor to receive the ping for a coarse estimate of angle. However, to be able to accurately estimate the angle arrival of the beacons, we have developed another PCB to process the ultrasound beacons received by two transceivers facing the same direction and spaced apart by a known distance a (illustrated in Figure 3b). Due to the spacing between the ultrasound transceivers, the time of arrival at both of the transceivers will be different, say, t_1 and t_2, respectively. Using the TDoA concept, we calculate the distances between the anchor nodes and the two separate ultrasound transceivers, say d_1 and d_2, respectively. We use the estimated d_1, d_2, and the known a to estimate the distance d and angle θ as follows. We refer to this as the *Differential Localization* Scheme.

$$\begin{aligned} d_2^2 &= h^2 + x^2 \\ d_1^2 &= h^2 + (a+x)^2 \\ x &= \frac{d_1^2 - d_2^2 - a^2}{2 \cdot a} \end{aligned} \implies \begin{aligned} d &= \sqrt{\frac{d_1^2}{2} + \frac{d_2^2}{2} - \frac{a^2}{4}} \\ \theta &= \arcsin \frac{\frac{a}{2} + x}{d} = \arcsin \frac{d_1^2 - d_2^2}{2 \cdot a \cdot d} \end{aligned} \tag{1}$$

We have built the necessary software APIs for TDoA localization in RIOT-OS, an OS which supports many commercially available boards with 802.15.4 radios. *Note that, currently, both of our systems only work in 2D, but we are currently working on extending them to 3D.*

B. Overlay Management Infrastructure

In this section, we concisely present the Robotic Overlay coMmunicAtioN prOtocol (ROMANO), a novel *lightweight overlay networking protocol* for management and data collection in the IRIS testbed. It builds on top of the cutting edge MQTT-SN protocol [10], a UDP/IP based publish-subscribe protocol for low-power IoT devices. [1] As future work, we plan to implement a bridge to connect the MQTT-SN network to ROS.

In the IRIS testbed, each IRISbot is running a MQTT-SN client, and a broker is run in the IRIS server. The IRIS server's controller module runs a ROMANO server program that works in conjunction with the MQTT-SN/MQTT broker for IRIS management. Each of the IRISbots follows a standard sequence of operations for bootstrapping the ROMANO communication as follows.

- Connect to the MQTT-SN broker running in IRIS server with its IPv6 address as the device identifier.
- Subscribe to an unique topic called ROMANO ID which is same as the last 8 characters of the device IPv6 address.
- Publish the ROMANO-ID to a predefined management topic "init-info".
- Subscribe to another management topic "common", used for broadcast communication.

Our proposed ROMANO protocol in the context of the IRIS testbed can be described as follows.

- The ROMANO protocol and the MQTT-SN protocol are nested together in the application layer of the standard Internet model (see Fig. 4).
- The ROMANO protocol defines communication endpoints with the notion of topics. The subscriber of a topic is the receiver end whereas the publisher to a topic is the transmitter. The ROMANO server keeps track of all the available subscriptions and has the capability to publish to any of these topics.
- The ROMANO protocol uses the message types "MQTT Subscribe" and "MQTT Unsubscribe" to control subscriptions. In the IRIS testbed, the IRIS server uses this feature of ROMANO for bootstrapping or tearing down the communication endpoints by instructing the receivers to subscribe or unsubscribe to a particular topic.
- The ROMANO protocol provides another important message type called "MQTT Publish Request" which instructs the robots to publish certain types of data to certain topics (e.g. 'telemetry'). This feature is used by the experimental data collection module of the IRIS server.
- The ROMANO protocol also allows for direct control of the robots using the "Movement Control" type messages used by the controller module of the IRIS server.

[1] For a more detailed description of MQTT-SN, interested readers are referred to http://mqtt.org/new/wp-content/uploads/2009/06/MQTT-SN_spec_v1.2.pdf.

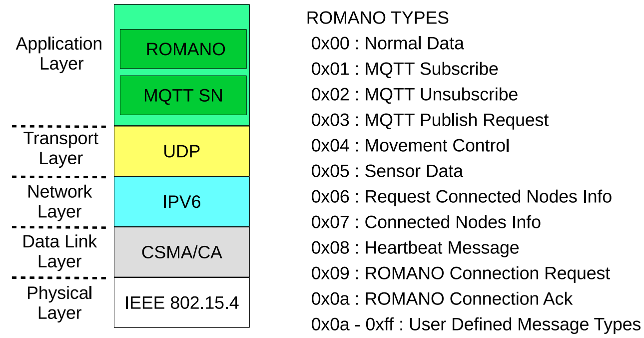
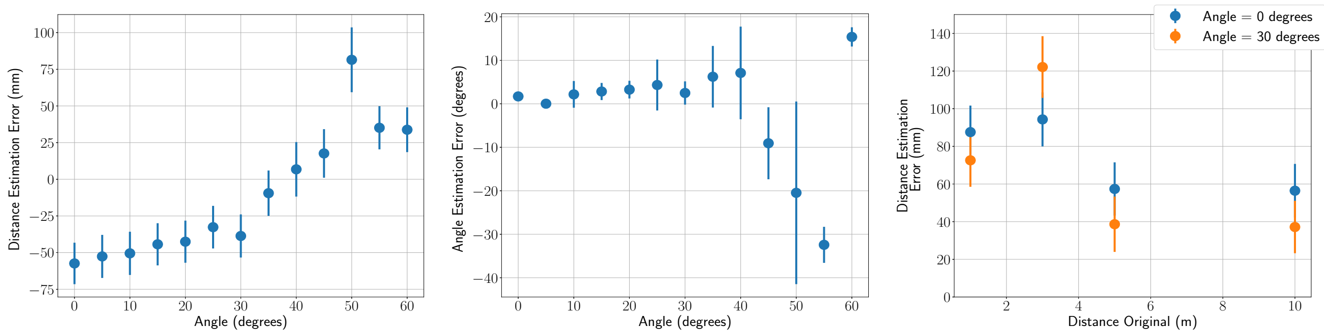


Fig. 4: (Left) The ROMANO Network Stack, (Right) ROMANO Data Types

For a complete overview of the different types of ROMANO messages and their formats, the interested reader is referred to our full paper on ROMANO [20].

C. Communication Module

The communication module of IRIS is mainly dedicated to peer to peer communication between the IRISbots via standard UDP, IP, or MAC layer packets. We use the OpenMote platform with a programmable low power 802.15.4 radio for the communication module in our IRISbots. By default, the communication module enables a tree based multihop routing protocol called RPL [21] for the routing layer, User Datagram Protocol (UDP) for transport layer, and Channel Sense Multiple Access (CSMA) based MAC layer using built-in RIOT-OS features. The communication module also includes two separate APIs, *the management and localization APIs*, to support the communication requirements of the management module and the localization module, respectively. The management API connects the ROMANO client to the MQTT-SN client process running on the OpenMote. The localization API is used to transmit/receive the TDoA beacons required for ranging. To keep all three types of communication



(a) Distance Estimation Error For Varying Angles (b) Angle Estimation Error For Varying Angles (c) Distance Estimation Error For Varying Distances

Fig. 5: TDoA Differential Localization Performance. Each test was performed at least 200 times.

separate (localization-related communication, management-related communication, and intrarobot communication), we use three different frequency channels of the IEEE 802.15.4 standard, all of which are non-overlapping. To support wireless robotic network related research, the communication module has the flexibility of implementing different protocols pertaining to different layers of a 802.15.4 compliant network stack.

IV. Implementation Details

With the main focus being low power, low bandwidth consuming wireless robotics research, we have carefully chosen a set of hardware to build our IRIS testbed, with some custom additions. The robot we use for IRIS is an off-the-shelf, commercially available robot from Pololu named 3pi that comes with an expansion board to accommodate an XBee form factor communication device and an mbed LPC1768 microcontroller board. For the XBee form factor device, we use another commercially available product called the OpenMote. For localization, we have designed custom PCBs for both of our localization techniques, omnidirectional and differential. For the ultrasound transceivers, we use the XL-MaxSonar-AE0, an off-the-self high precision sensor.

For programming of the OpenMote, we use the RIOT-OS, an open-source, real-time operating system for the Internet of Things. For programming the LPC1768, we use MBED-OS which is also a real-time operating system. The communication module on the IRISbot is implemented on the OpenMote while the rest of the modules are implemented on the mbed LPC1768 device. The reason for separating the modules across two MCUs is to reduce the software design complexity and maintain modularity since the implementation of our localization system, movement techniques, and radio communication are multi-threaded and have real-time constraints. For efficient, reliable inter-process communication between the OpenMote and the MBED, we wrote software for both operating systems that implements the well-known HDLC reliability protocol over UART [22]. The anchors used for localization are also IRISbots. *IRISbots can switch between anchor mode and client mode on demand.*

We use a standalone Raspberry Pi 3 with an OpenMote connected to it via UART over USB as the IRIS server. All the modules on the server are implemented in Python. The communication module of the server is implemented on the OpenMote. All software and hardware designs along with supporting documentation is open source and can be found at *https://tiny.cc/iris-anrg*.

V. Experiments and Performance Evaluation

In this section, we present a preliminary evaluation of the performance of the localization module via a set of real world experiments. To illustrate the performance of the differential TDoA based localization, we performed the relative position estimation test for a range of orientations at least 200 times per angle at a distance of 3 meters. The results presented in Figs. 5a and 5b illustrate that for angles of up to 45 degrees the average distance errors are less than 5 cms and average angle errors are less than 10 degrees. For angles beyond 45 degrees, the performance deteriorates due to two possible errors: (1) the directivity of the ultrasound transceivers causes one of the two receivers to miss an ultrasonic ping which results in a failed estimation of orientation but successful estimation of the distance or (2) the transceivers receive a reflected component of the ultrasonic signal which causes an inaccurate angle estimation. To resolve this, the robots can rotate until a successful orientation estimation within +/- 45 degrees is acquired. Moreover, to analyze how the distance estimation performance changes with distance, we varied the distances between two nodes from 1m to 20m at two different angles (0 and 30 degrees). Our observation from the experiments, as illustrated in Figure 5c, shows that the performance of the system is reasonable up to 10m, after which performance gradually deteriorates due to the ultrasound signals occasionally falling below the threshold although RF beacons are detected. While this threshold can be adjusted, a 10 meter distance is practical for indoor swarm robotics experiments. Lastly, we evaluated the performance of trilateration using the omnidirectional configuration by localizing a single node. In this experiment, three omnidirectional IRISbots operating in anchor mode are placed at three corners of a $2m$ by $2m$ square to create a relative 2D plane: ($[0cm, 0cm]$, $[200cm, 0cm]$, $[0cm, 200cm]$). A fourth omnidirectional IRISbot placed inside the square operates in client mode and uses trilateration to determine its position at the following coordinates, 100 times each: [$0cm$, $50cm$], [$50cm$, $150cm$], [$100cm$, $100cm$], and [$200cm$, $100cm$]. The results presented in Table I illustrate that the trilateration based localization errors are less than 10cm.

TABLE I: Trilateration Error Statistics with Three Anchors Located at (0,0), (200,0), and (0,200), respectively (100 trials at each location).

Location (cm,cm)	$(50, 50)$	$(50, 150)$	$(100, 100)$	$(200, 100)$
Mean Error (cm)	4.029	7.069	2.803	6.828
Std Error (cm)	1.22	2.81	1.28	1.31

We have also developed and performed two example applications on the IRIS testbed to illustrate some of its features. In the first application, a group of three robots follow each other in a chain using a control loop and state machine over wireless adhoc links. The first robot (*i.e.*, the leader) moves in a random direction. It then stops moving, starts sending TDoA beacons, and instructs the second robot to localize using the TDoA beacons. The second robot then localizes and instructs the first robot to stop beaconing after finishing. It then attempts to move towards the leader robot while maintaining a one foot distance from the leader. Next, the second robot starts emitting TDoA beacons and instructs the third and final robot to localize. The third robot will repeat the same procedure as the second robot up to before emitting TDoA beacons. Because no robot is behind the last, the last robot instructs the leader to repeat the process from the beginning. The localization for this experiment is purely relative, *i.e.*, each robots localizes itself with respect to the robot in front via the differential TDoA system. Based on the estimated relative distances and angles, they move using typical PID controllers for dead reckoning with Pololu magnetic wheel encoders. This experiment is done to illustrate how the IRIS system can be self-sufficient without any external localization infrastructure or cameras.

The second example is similar to the trilateration experiment setup. In this example, the fourth node instead moves around inside the square and stops at various points to trilaterate. After determining its coordinates, it will switch to a different 802.15.4 channel to communicate with the IRIS server and report its estimates. This demo illustrates the accuracy and portability of our localization system. The videos of both experiments can be found at *https://tiny.cc/iris-anrg*.

VI. Conclusion

In this paper, we presented our IRIS testbed that is carefully designed for wireless robotic network research and portability to various deployment contexts. While the current system is a working version of a desired multi-robot networking testbed, there still remains a number of research questions that need to be answered in order to scale it up further. The beaconer nodes in our current design can either be event-driven or multiplexed via time division multiple access to manage intra-beaconer intereference. Both of the methods have their caveats which requires a careful design of an adaptive and collaborative beaconing approach. Also, adding support to remote programming of each of the devices in the IRISbots is also left as a future work. Lastly, we plan to identify and incorporate relevant security features into the IRIS testbed.

References

[1] R. K. Williams, A. Gasparri, and B. Krishnamachari, "Route swarm: Wireless network optimization through mobility," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014.

[2] Y. Mostofi, "Cooperative Wireless-Based Obstacle/Object Mapping and See-Through Capabilities in Robotic Networks," *IEEE Transactions on Mobile Computing*, January 2012.

[3] B. Ning, J. Jin, J. Zheng, and Y. W. Law, "Connectivity control and performance optimization in wireless robotic networks: Issues, approaches and a new framework," in *6th International Conference on Modelling, Identification & Control (ICMIC)*, 2014.

[4] A. Jiménez-González, J. R. Martinez-de Dios, and A. Ollero, "Testbeds for ubiquitous robotics: A survey," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1487–1501, 2013.

[5] D. Johnson *et al.*, "Mobile emulab: A robotic wireless and sensor network testbed," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2006.

[6] D. Pickem *et al.*, "The robotarium: A remotely accessible swarm robotics research testbed," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[7] A.-S. Tonneau, N. Mitton, and J. Vandaele, "How to choose an experimentation platform for wireless sensor networks? a survey on static and mobile wireless sensor network experimentation facilities," *Ad Hoc Networks*, vol. 30, pp. 115–127, 2015.

[8] C. Adjih *et al.*, "Fit iot-lab: A large scale open experimental iot testbed," in *IEEE 2nd World Forum on Internet of Things (WF-IoT)*, 2015.

[9] A. Savvides, C.-C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *ACM MOBICOM, 2001*.

[10] MQTT-SN, "http://mqtt.org/."

[11] A. Stubbs *et al.*, "Multivehicle systems control over networks: a hovercraft testbed for networked and decentralized control," *IEEE control systems*, vol. 26, no. 3, pp. 56–69, 2006.

[12] D. Pickem, M. Lee, and M. Egerstedt, "The gritsbot in its natural habitat-a multi-robot testbed," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[13] M. Rubenstein, C. Ahler, and R. Nagpal, "Kilobot: A low cost scalable robot system for collective behaviors," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[14] S. Gröber, M. Vetter, B. Eckert, and H.-J. Jodl, "Experimenting from a distance—remotely controlled laboratory (rcl)," *European Journal of Physics*, vol. 28, no. 3, p. S127, 2007.

[15] Tutornet, "http://anrg.usc.edu/www/tutornet/."

[16] J. R. Martinez-de Dios, A. Jimenez-Gonzalez, A. San Bernabe, and A. Ollero, *A Remote Integrated Testbed for Cooperating Objects*. Springer, 2014.

[17] I. Amundson and X. D. Koutsoukos, "A survey on localization for mobile wireless sensor networks," in *Mobile Entity Localization and Tracking in GPS-less Environnments*. Springer, 2009, pp. 235–254.

[18] R. Mautz and S. Tilch, "Survey of optical indoor positioning systems," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2011.

[19] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 37, no. 6, pp. 1067–1080, 2007.

[20] P. Ghosh, J. A. Tran, D. Dsouza, N. Ayanian, and B. Krishnamachari, "Romano: A novel overlay lightweight communication protocol for unified control and sensing of a network of robots," *arXiv preprint arXiv:1709.07555*, 2017.

[21] T. Winter, "Rpl: Ipv6 routing protocol for low-power and lossy networks," 2012.

[22] E. Gelenbe, J. Labetoulle, and G. Pujolle, "Performance evaluation of the hdlc protocol," *Computer Networks (1976)*, vol. 2, no. 4-5, pp. 409–415, 1978.